CitectSCADA

v7.20

CitectSCADA User Guide

October 2010

Legal Notice

DISCLAIMER

Schneider Electric (Australia) Pty. Ltd. makes no representations or warranties with respect to this manual and, to the maximum extent permitted by law, expressly limits its liability for breach of any warranty that may be implied to the replacement of this manual with another. Further, Schneider Electric (Australia) Pty. Ltd. reserves the right to revise this publication at any time without incurring an obligation to notify any person of the revision.

COPYRIGHT

© Copyright 2010 Schneider Electric (Australia) Pty. Ltd. All rights reserved.

TRADEMARKS

Schneider Electric (Australia) Pty. Ltd. has made every effort to supply trademark information about company names, products and services mentioned in this manual.

Citect, CitectHMI, and CitectSCADA are registered trademarks of Schneider Electric (Australia) Pty. Ltd.

IBM, IBM PC and IBM PC AT are registered trademarks of International Business Machines Corporation.

MS-DOS, Windows, Windows NT, Microsoft, and Excel are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

DigiBoard, PC/Xi and Com/Xi are trademarks of Digi International Inc.

Novell, Netware and Netware Lite are either registered trademarks or trademarks of Novell, Inc. in the United States and other countries..

dBASE is a trademark of dataBased Intelligence, Inc.

All other brands and products referenced in this document are acknowledged to be the trademarks or registered trademarks of their respective holders.

GENERAL NOTICE

Some product names used in this manual are used for identification purposes only and may be trademarks of their respective companies.

October 2010 edition for CitectSCADA Version v7.20

Manual Revision Version v7.20.

Contact Schneider Electric (Australia) Pty. Ltd. today at www.Citect.com/citectscada

Contents

Chapter: 18 Defining and Drawing Graphics Pages 433

Chapter: 34 Working with Multiple Monitors 847

Chapter: 35 Using OPC Server DA2.0 849

Chapter: 36 Compiling and Running a Project 865

Getting Started

This section contains information on CitectSCADA v7.20 and describes the following:

Safety Information

Hazard categories and special symbols

The following symbols and special messages may appear in this manual or on the product to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

A lightning bolt or ANSI man symbol in a "Danger" or "Warning" safety label on the product indicates an electrical hazard which, as indicated below, can or will result in personal injury if the instructions are not followed.

The exclamation point symbol in a safety message in a manual indicates potential personal injury hazards. Obey all safety messages introduced by this symbol to avoid possible injury or death.

Symbol	Name
⚡	Lightning Bolt
	ANSI man
⚠	Exclamation Point

⚠ DANGER

DANGER indicates an imminently hazardous situation, which, if not avoided, will result in death or serious injury.

⚠ WARNING

WARNING indicates a potentially hazardous situation, which, if not avoided, can result in death or serious injury.

⚠ CAUTION

CAUTION indicates a potentially hazardous situation which, if not avoided, can result in minor or moderate injury.

CAUTION

CAUTION used without the safety alert symbol, indicates a potentially hazardous situation which, if not avoided, can result in property damage.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric (Australia) Pty. Ltd. for any consequences arising out of the use of this material.

Before You Begin

CitectSCADA is a Supervisory Control and Data Acquisition (SCADA) solution. It facilitates the creation of software to manage and monitor industrial systems and processes. Due to CitectSCADA's central role in controlling systems and processes, you must appropriately design, commission, and test your CitectSCADA project before implementing it in an operational setting. Observe the following:

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use CitectSCADA or other SCADA software as a replacement for PLC-based control programs. SCADA software is not designed for direct, high-speed system control.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.*
- Each implementation of a control system created using CitectSCADA must be individually and thoroughly tested for proper operation before being placed into service.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

* For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control".

Chapter: 1 Getting Technical Support

There are various support options to help you get the most from this product.

- If you have questions about using CitectSCADA, consult the extensive online Help to answer your questions. You can use the **Contents** list to find the section you're interested in, enter an item into the **Index**, or enter an item using the **Search** tab.
- If you seek more technical information than is provided in the online Help, check the knowledge base.

If you cannot find the information you need, you can obtain technical support and Training. Consulting services are also available upon request.

See Also

Technical Support
Contact information

CitectSCADA Technical Support

Please note the following about technical support for your CitectSCADA product.

Technical Support

Citect Maintenance and Support Agreements are available for purchase. To purchase a Maintenance and Support Agreement, price of which is determined by the list price of your system, you need to contact your local Schneider Electric (Australia) Pty. Ltd. representative.

Schneider Electric (Australia) Pty. Ltd. offers a range of Support services; for further information visit http://www.citect.com/support. Upon receipt of your order for a Maintenance and Support Agreement, you will receive a site number and a list of dongle serial numbers, which you need to quote when you contact Support. A site or dongle serial number enables the Customer Support team to provide quality service product by logging and tracking your Customer Service Requests (CSRs).

Before Calling Customer Support

Fill out the online CitectSCADA CitectHMI Support Request form, found at the web site http://www.citect.com/support.

If you are unsure of which contact details to use, email: support@citect.com.

Training

Various training facilities are also available. Contact your local Citect distributor for more information.

See Also

Contact information

Contact information

For contact information in your region, consult the support web site at:

http://www.Citect.com/support

Chapter: 2 What's New in CitectSCADA v7.x

This section describes new CitectSCADA features and enhancements for v7.20. For the purposes of continuity, this section also describes the features that were added for the CitectSCADA v7.0 and v7.10 releases.

Introduced in v7.0

Introduced in v7.10

Introduced in v7.20

For details on how to upgrade an existing project to run in v7.20, refer to Upgrading to V7.20.

What's New in CitectSCADA v7.0

CitectSCADA v7.0 incorporates the following new features.

Introduced in v7.0:

- The Migration Tool
- Clustering
- Local Variables
- Publish Alarm Property
- Memory Mode
- Client-side Online Changes
- Publisher-Subscriber Model
- Dual Network Support
- Project-Based Network Configuration

For changes to:

- Citect.ini parameters, see Citect.ini Parameters in Version 7.0.
- Cicode functions, see Cicode Functions in Version 7.0.
- CtAPI functions, see CtAPI Functions in Version 7.0.
- Kernel commands, see Kernel Commands in Version 7.0.

For details on how to configure an existing project to run in v7.20, refer to Upgrading.

See Also

What's New in CitectSCADA v7.x

The Migration Tool

The Migration Tool is a separate application which has to be manually run after the automatic upgrade has been executed, and initiated by you after you have prepared the project for final migration. This tool will accommodate the important changes in project functionality which are incorporated in version 7.0.

See Also

Migration Tool

Clustering

Clustering allows you to group different sets of the runtime components within a single project, allowing multiple independent systems to be monitored and controlled.

There are countless variations in how a clustered system can be configured. The most appropriate configuration will depend on the requirements for the solution to be deployed and the environment in which it is being deployed. For more information see Typical system scenarios.

See Also

Included projects
Implementing Clustering

Local Variables

Local variables allow you to store data in memory when you start your runtime system. They are created each time the system starts, and therefore do not retain their values when you shut down.

Local variables are useful when you need each process to have a separate copy of the data. Each process has its own copy of each local variable configured in the project, and the values in a local variable are available only to the process that wrote them.

See Also

Configuring Local Variables

Publish Alarm Property

Alarm devices were defined as devices with their Protocol field set to "Alarm". The function of these devices are now configured on an Alarm Server by setting the "Publish Alarm Properties" property to True.

See Also

Alarm Server Definitions

Memory Mode

I/O Devices can now be configured to run in memory mode:

- An I/O Device running in memory mode is created in memory and its values stored in memory at runtime.
- Devices using memory mode are not connected to any hardware, and write their values to a cache. Memory mode is useful when you are configuring a system for the first time, as you can design and test your system before connecting a physical I/O Device.

> **Note:** Memory mode replaces Memory I/O Devices, which are no longer supported. Devices configured as Memory I/O Devices will be converted to local variables during the upgrade to v7.20.

See Also

Using Memory Mode

Client-side Online Changes

The following live changes can now be made to the project without restarting clients:

- I/O Devices (restart the I/O Server)
- Tags (restart the I/O Server)
- Alarms (restart the Alarm Server)
- Trends (restart the Trends Server)
- Reports (restart the Reports Server)
- Accumulators (restart the Reports Server)

Clients only require that graphics, code and communications configurations are deployed to them. Other configuration information is deployed to the appropriate server.

Project changes are still deployed as before

- Manually
- Run/Backup Run/Copy
- FTP (IDC)
- HTTP (Web Client)

See Also

Improved Client Side Online Changes

Publisher-Subscriber Model

CitectSCADA now uses a Publisher-Subscriber data acquisition model. Client computers subscribe to configured tags and receive notification when the tag values change. Cicode functions can also be triggered by the change of a tag, removing the need to poll, and improving the efficiency of the system.

See Also

TagSubscribe
TagUnsubscribe

Dual Network Support

Previous CitectSCADA versions have been able to support redundant networks via NetBIOS. From version 7.0 users can only use TCP/IP for network configuration and can specify multiple IP addresses for each server, providing native support for network redundancy.

Project-Based Network Configuration

In version 7.0, the project topology is embedded in the project, and network configuration can be performed from within the Project Editor. Servers and their IP addresses are set up in the **Network Addresses** dialog in the Project Editor.

This means that physical computers in the system can easily be changed. As long as the IP address or computer name of the new machine is the same as the one being replaced, the new computer will be able to immediately take the same role.

Citect.ini Parameters in version 7.0

The following sections detail the changes made to Citect.ini parameters in Citect-SCADAversion 7.0:

- New Parameters
- Obsolete Parameters

New Parameters

The following parameters are new in version 7.0. For a complete list of the system parameters, refer to the Parameters help file.

Alarm Parameters:

[Alarm.ClusterName.ServerName]Clusters	Sets which clusters this Alarm Server process connects to at startup
[Alarm.ClusterName.ServerName]CPU	Sets the CPU that the Alarm Server process is assigned to
[Alarm.ClusterName.ServerName]Events	The list of events that this Alarm Server process enables
[Alarm.ClusterName.ServerName]ShutdownCode	Determines the Cicode function to run when Alarm Server process shuts down
[Alarm.ClusterName.ServerName]StartupCode	Determines the Cicode function to run when Alarm Server process starts up

Note: The default alarm property write behavior was to write the new value to DBF/RDB. This has changed in version 7.0 onwards. Refer to the parameter [Alarm]UseConfigLimits.

Backup Parameters:

[Backup]SaveiniFiles	Determines whether the "Save ini files" checkbox is checked by default during Backup.

Client Parameters:

[Client]Clusters	Selects which clusters the client is connected to at startup.
[Client]ComputerRole	Specifies the role of the computer
[Client]Events	Sets the events to be enabled on the client.
[Client]ForceClient	Starts only the client process, and connects to a configured Citect servers using a network connection.

[Client]FullLicense	Specifies that a Control Client will use a full license
[Client]ShutdownCode	Determines the Cicode function to run when DisplayClient component shuts down.
[Client]StartupCode	Determines the Cicode function to run when DisplayClient component starts up.
[Client]WaitForConnectAtStartup	Specifies that connection to a server will wait until it can establish a connection to the server processes before starting up.

CtCicode Parameters:

[CtCicode]FastFormat	Controls whether fast formatting is used in the Cicode Editor

CtEdit Parameters:

[CtEdit]Config	The directory where the CitectSCADA configuration files such as citect.ini are located.
[CtEdit]Logs	The directory where the CitectSCADA log files are located.

Dial Parameters:

[Dial]MissedScheduleTolerance	Specifies how many consecutive scheduled dial attempts can be missed before the cache becomes stale

Driver Parameters:

[<*DriverName*>]OverriderOSProtection	Determines whether to override the protection mechanism built-in to the I/O Server for drivers that may not be compatible with Windows Vista.

General Parameters:

[General]Multiprocess	Determines whether CitectSCADA runs as a multi-process or single-process application.

IOServer Parameters:

[IOServer.ClusterName.ServerName]Clusters	Sets the clusters that the I/O Server process will connect to at startup
[IOServer.ClusterName.ServerName]CPU	Sets the CPU that the I/O Server process is assigned to

[IOServer.ClusterName.ServerName]Events	The list of events that this I/O Server process enables
[IOServer.ClusterName.ServerName]ShutdownCode	Determines the Cicode function to run when I/O Server process shuts down
[IOServer.ClusterName.ServerName]StartupCode	Determines the Cicode function to run when I/O Server process starts up

Report Parameters:

[Report.ClusterName.ServerName]Clusters	Sets the clusters this Reports Server process will connect to at startup
[Report.ClusterName.ServerName]CPU	Sets the CPU that this Reports Server process is assigned to
[Report.ClusterName.ServerName]Events	The list of events that this Reports Server process enables
[Report.ClusterName.ServerName]ShutdownCode	Determines the Cicode function to run when this Reports Server process shuts down
[Report.ClusterName.ServerName]StartupCode	Determines the Cicode function to run when this Reports Server process starts up

Trend Parameters:

[Trend.ClusterName.ServerName]Clusters	Sets the clusters this Trends Server process will connect to at startup
[Trend.ClusterName.ServerName]CPU	Sets the CPU that the Trends Server process is assigned to
[Trend.ClusterName.ServerName]Events	The list of events that this Trends Server process enables
[Trend.ClusterName.ServerName]ShutdownCode	Determines the Cicode function to run when Trends Server process shuts down
[Trend.ClusterName.ServerName]StartupCode	Determines the Cicode function to run when Trends Server process starts up

Obsolete Parameters

The following parameters are no longer supported in version 7.0:

Alarm Parameters:

[Alarm]CPU	Sets the CPU that the Alarm Server component is assigned to
[Alarm]Primary	Determines if this Alarm Server is the Primary Alarm Server
[Alarm]Process	Sets the CitectSCADA process the Alarm Server component is assigned to
[Alarm]Server	Determines whether this computer is an Alarm Server

Client Parameters:

[Client]Display	Sets the CitectSCADA computer as a Control Client
[Client]Manager	Sets the CitectSCADA computer as a View-only Client
[Client]Primary	The name of the primary CitectSCADA server
[Client]Process	Sets the CitectSCADA process the Control Client component is assigned to
[Client]Standby	The name of the standby CitectSCADA server

Code Parameters:

[Code]AlarmShutdown	Determines the Cicode function to run when Alarm Server component shuts down
[Code]AlarmStartup	Determines the Cicode function to run when Alarm Server component starts up
[Code]AutoReRead	Controls whether the ReRead() function is automatically called
[Code]I-OServerShutdown	Determines the Cicode function to run when I/O Server component shuts down
[Code]IOServerStartup	Determines the Cicode function to run when I/O Server component starts up
[Code]Process	Sets the process a code component is assigned to

[Code]ReportShutdown	Determines the Cicode function to run when Reports Server component shuts down
[Code]ReportStartup	Determines the Cicode function to run when Reports Server component starts up
[Code]Shutdown	Determines the Cicode function to run when Control Client component shuts down
[Code]Startup	Determines the Cicode function to run when Control Client component starts up
[Code]TrendShutdown	Determines the Cicode function to run when Trends Server component shuts down
[Code]TrendStartup	Determines the Cicode function to run when Trends Server component starts up

DNS Parameters:

[DNS]<Server name>	Determines the IP address (or fully qualified host name) of the primary I/O Server

Event Parameters:

[Event]Alarm	The classes of events to be enabled by the Alarm Server
[Event]IOServer	The classes of events to be enabled by the I/O Server
[Event]Name	The classes of events assigned to the Name entry to be enabled
[Event]Report	The classes of events to be enabled by the Reports Server
[Event]Trend	The classes of events to be enabled by the Trends Server

General Parameters:

[General]BadOptimise	Determines whether certain strings are replaced with labels on compile
[General]CitectRunningCheck	Checks if a project is currently running on the local machine when a compile is triggered

IOServer Parameters:

[IOServer]BlockWrites	Determines whether CitectSCADA will try to block optimize writes to I/O Devices. The IOserver will not block writes
[IOServer]CPU	Sets the CPU that the I/O Server component is assigned to
[IOServer]Name	The name of the default I/O Server
[IOServer]Process	Sets the CitectSCADA process the I/O Server component is assigned to
[IOServer]SaveBackup	This parameter has been superseded by SaveNetwork
[IOServer]Server	Determines whether this computer is an I/O Server

LAN Parameters:

[LAN]Bridge	Determines the bridge level
[LAN]CancelOnClose	For users with Novell NetBIOS emulator issues
[LAN]Disable	Enables/disables CitectSCADA from the LAN
[LAN]GroupName	Determines whether CitectSCADA uses the group name 'CITECT STATION50' or the computer name specified by the [Lan]Node parameter.
[LAN]KillPiggyBackAck	Controls whether CitectSCADA will try to optimize network protocols which support piggyback ACK
[LAN]LanA	Defines the protocol stack that CitectSCADA uses for NetBIOS communication
[LAN]NetBIOS	Enables/disables NetBIOS
[LAN]NetTrace	Determines whether the NetBIOS window is enabled on startup
[LAN]NetTraceBuff	Sets the number of trace buffers.
[LAN]NetTraceErr	Enables the error mode of the NetBIOS window
[LAN]NetTraceLog	Enables the logging mode of the NetBIOS window
[LAN]Poll	Puts CitectSCADA LAN communications into polled mode

[LAN]RemoteTimeOut	The timeout period for remote I/O Device write requests from a Control Client to the I/O Server
[LAN]Retry	The number of times to retry establishing communications after a timeout - before an alert message is generated
[LAN]SendTimeOut	The timeout to send a network packet across the network
[LAN]SesRecBuf	The number of receive NetBIOS Control Blocks (NCBs) that CitectSCADA uses for every session
[LAN]SesSendBuf	The number of send NetBIOS control blocks that CitectSCADA uses for every session
[LAN]TimeOut	The timeout to send a network packet across the network

Proxi Parameters:

[Proxi]<I/O Server name>	Defines a list of proxy server associations

Report Parameters:

[Report]Primary	Determines if this Reports Server is the Primary Reports Server
[Report]Process	Sets the CitectSCADA process the Reports Server component is assigned to
[Report]Server	Determines whether this computer is a Reports Server

Server Parameters:

[Server]Name	The name of the CitectSCADA server

Trend Parameters:

[Trend]BlockByIODevice	Verifies that I/O problems causing gaps for a trend tag do not cause gaps for every trend tag
[Trend]CPU	Sets the CPU that the Trends Server component is assigned to
[Trend]Process	Sets the CitectSCADA process the Trends Server component is assigned to

[Trend]Redundancy	Enables/disables trend redundancy action
[Trend]Server	Determines whether this computer is a Trends Server
[Trend]StaggerRequestSubgroups	Reduces network traffic by spacing out trend sample requests

Cicode Functions in v7.0

Some Cicode functions have been introduced, modified, deprecated or removed. The following sections detail the changes made to these functions:

New Functions

Miscellaneous Functions

AccControl	Controls accumulators for example motor run hours.
AccumBrowseClose	Closes an accumulator browse session.
AccumBrowseFirst	Gets the oldest accumulator entry.
AccumBrowseGetField	Gets the field indicated by the cursor position in the browse session.
AccumBrowseNext	Gets the next accumulator entry in the browse session.
AccumBrowseNumRecords	Returns the number of records in the current browse session.
AccumBrowseOpen	Opens an accumulator browse session.
AccumBrowsePrev	Gets the previous accumulator entry in the browse session.
ProcessIsClient	Determines if the currently executing process contains a Client component
ProcessIsServer	Determines if the currently executing process contains a particular server component.
ServiceGetList	Gets information about services running in the component

	calling this function.

Alarm Functions:

AlarmDspLast	Displays the latest unacknowledged alarms.
AlmSummaryAck	Acknowledges the alarm at the current cursor position in an active data browse session.
AlmSummaryClear	Clears the alarm at the current cursor position in an active data browse session.
AlmSummaryClose	Closes an alarm summary browse session.
AlmSummaryCommit	Commits the alarm summary record to the alarm summary device.
AlmSummaryDelete	Deletes alarm summary entries from the browse session.
AlmSummaryDeleteAll	Deletes alarm summary entries from the browse session.
AlmSummaryDisable	Disables the alarm at the current cursor position in an active data browse session.
AlmSummaryEnable	Enables the alarm at the current cursor position in an active data browse session.
AlmSummaryFirst	Gets the oldest alarm summary entry.
AlmSummaryGetField	Gets the field indicated by the cursor position in the browse session.
AlmSummaryLast	Places the data browse cursor at the latest summary record from the last cluster of the available browsing cluster list.
AlmSummaryNext	Gets the next alarm summary entry in the browse session.
AlmSummaryOpen	Opens an alarm summary browse session.
AlmSummaryPrev	Gets the previous alarm summary entry in the browse session.
AlmSummarySetFieldValue	Sets the value of the field indicated by the cursor position in the browse session.
AlmTagsAck	Acknowledges the alarm tag at the current cursor position in an active data browse session.

AlmTagsClear	Clears the alarm tag at the current cursor position in an active data browse session.
AlmTagsDisable	Disables the alarm tag at the current cursor position in an active data browse session.
AlmTagsEnable	Enables the alarm tag at the current cursor position in an active data browse session.
AlmTagsFirst	Gets the oldest alarm tags entry.
AlmTagsGetField	Gets the field indicated by the cursor position in the browse session.
AlmTagsNext	Gets the next alarm tags entry in the browse session.
AlmTagsNumRecords	Returns the number of records in the current browse session.
AlmTagsOpen	Opens an alarm tags browse session.
AlmTagsPrev	Gets the previous alarm tags entry in the browse session.

Super Genie Functions

AssGetProperty	Gets association information about the current Super Genie from the datasource
AssGetScale	Gets scale information about the associations of the current Super Genie from the datasource
AssInfoEx	Replaces the AssInfo function and supports online changes.

Cluster Functions

ClusterActivate	Allows the user to activate an inactive cluster.
ClusterDeactivate	Allows the user to deactivate an active cluster.
ClusterFirst	Allows the user to retrieve the first configured cluster in the project.
ClusterIsActive	Allows the user to determine if a cluster is active.
ClusterNext	Allows the user to retrieve the next configured cluster in the project.

ClusterServerTypes	Allows the user to determine which servers are defined for a given cluster.
ClusterStatus	Allows the user to determine the connection status from the client to a server on a cluster.
ClusterSwapActive	Allows the user to deactivate an active cluster at the same time as activating an inactive cluster.

I/O Device Functions

SubscriptionAddCallback	Adds a callback function to a tag subscription.
SubscriptionGetAttribute	Reads an attribute value of a tag subscription.
SubscriptionRemoveCallback	Removes a callback function from a tag subscription
TagGetProperty	Gets a property for a variable tag from the datasource.
TagGetScale	Gets the value of a tag at a specified scale from the datasource
TagSubscribe	Subscribes a tag for periodic monitoring and event handling.
TagUnsubscribe	Unsubscribes a tag for periodic monitoring and event handling.

Tag Functions

TagInfoEx	Supports online changes.
TagWriteEventQue	Opens the tag write event queue.

Task Functions

TaskCluster	Gets the name of the cluster context in which the current task is executing

Trend Functions

TrnBrowseClose	Closes a trend browse session.

TrnBrowseFirst	Gets the oldest trend entry.
TrnBrowseGetField	Gets the field indicated by the cursor position in the browse session.
TrnBrowseNext	Gets the next trend entry in the browse session.
TrnBrowseNumRecords	Returns the number of records in the current browse session.
TrnBrowseOpen	Opens a trend browse session.
TrnBrowsePrev	Gets the previous trend entry in the browse session.
TrnGetCluster	Gets the name of the cluster the trend graph is associated with.
TrnGetPenComment	Gets the comment of a trend pen.

Report Functions

RepGetCluster	Retrieves the name of the cluster the report is running on.

ModifiedFunctions

Alarm Functions

AlarmAck	Acknowledges alarms.
AlarmAckRec	Acknowledges alarms by record number
AlarmActive	Determines if any alarms are active in the user's area.
AlarmClear	Clears acknowledged, inactive alarms from the active alarm list.
AlarmClearRec	Clear an alarm by its record number
AlarmDelete	Deletes alarm summary entries.
AlarmDisable	Disables alarms.
AlarmDisableRec	Disables alarms by record number
AlarmDsp	Displays alarms.

AlarmDspLast	Displays the latest unacknowledged alarms.
AlarmEnable	Enables alarms.
AlarmEnableRec	Enables alarms by record number
AlarmFirstTagRec	Searches for the first occurrence of an alarm tag, name, and description
AlarmGetDelayRec	Gets the delay setting for an alarm via the alarm record number
AlarmGetFieldRec	Gets alarm field data from the alarm record number
Alarm-GetThresholdRec	Gets the thresholds of analog alarms by the alarm record number
AlarmNextTagRec	Searches for the next occurrence of an alarm tag, name, and description.
Alarm-NotifyVarChange	Activates a time-stamped digital or time-stamped analog alarm
AlarmSumAppend	Appends a new blank record to the alarm summary.
AlarmSumCommit	Commits the alarm summary record to the alarm summary device.
AlarmSumDelete	Deletes alarm summary entries.
AlarmSumFind	Finds an alarm summary index for an alarm record and alarm on time.
AlarmSumFirst	Gets the oldest alarm summary entry.
AlarmSumGet	Gets field information from an alarm summary entry.
AlarmSumLast	Gets the latest alarm summary entry.
AlarmSumNext	Gets the next alarm summary entry.
AlarmSumPrev	Gets the previous alarm summary entry.
AlarmSumSet	Sets field information in an alarm summary entry.
AlarmSumSplit	Duplicates an alarm summary entry.

AlarmSumType	Retrieves a value that indicates a specified alarm's type.

I/O Device Functions

DriverInfo	Provides information about the driver for a particular I/O Device.
IODeviceControl	Provides control of individual I/O Devices.
IODeviceInfo	Gets information on an I/O Device.

Miscellaneous Functions

AccControl	Controls accumulators for example motor run hours.
ServerInfo	Gets client and server information.
ServerInfoEx	Gets client and server information from a specified process in a multiprocessor environment.
Shutdown	Ends CitectSCADA's operation.

Report Functions

RepGetControl	Gets report control information.
Report	Runs a report.
RepSetControl	Sets report control information.

SPC Functions

SPCAlarms	Returns the status of the specified SPC alarm.
SPCProcessXRSGet	Gets the process mean, range and standard deviation overrides.
SPCProcessXRSSet	Sets the process mean, range and standard deviation overrides.
SPCSpecLimitGet	Gets the specification limits (USL and LSL) for the specified tag.

SPCSpecLimitSet	Sets the specification limits (USL and LSL) for the specified tag.
SPCSubgroupSizeGet	Gets the size of a subgroup for the specified SPC tag.
SPCSubgroupSizeSet	Sets the subgroup size for the specified SPC tag.

Super Genie Functions

Ass	Associates a variable tag with a Super Genie.
AssPage	Associates up to eight variable tags with a Super Genie and displays the Super Genie in the current window.
AssPopUp	Associates up to eight variable tags with a Super Genie and displays the Super Genie in a popup window.
AssTag	Associates a variable tag with the current Super Genie. The association will be created for the current Super Genie only, and will only come into effect after you re-display the Super Genie.
AssVarTags	Associates up to eight variable tags with a Super Genie. This association is only made for the next Super Genie you display (either in the current window or in a new window). You can use this function repeatedly to associate more than 8 variable tags to a Super Genie.
AssWin	Associates up to eight variable tags with a Super Genie, and displays the Super Genie in a new window.

Tag Functions

TagGetProperty	This function reads a property of a variable tag from the datasource
TagGetScale	Gets the value of a tag at a specified scale from the datasource
TagRamp	This function will increment a Tag by the amount defined by iPercentInc
TagRead	Reads a variable from the I/O Device
TagScaleStr	Gets the value of a tag at a specified scale
TagWrite	Writes to an I/O Device variable by specifying the variable tag.

Task Functions

MsgOpen	Opens a message session with a CitectSCADA server or client.

Trend Functions

TrendDsp-CursorTag	Displays the tag name of the current pen.
TrnAddHistory	Restores an old history file to the trend system.
TrnDelHistory	Deletes an old history file from the trend system.
TrnEventSetTable	Sets trend data from a table, for a specified trend tag.
TrnE-ventSetTableMS	Sets event trend data and time data (including milliseconds) for a specified trend tag.
TrnFlush	Flushes the trend to disk.
TrnGetDefScale	Gets the default engineering zero and full scales of a trend tag.
TrnGetPen	Gets the trend tag of a pen.
TrnGetTable	Stores trend data in an array.
TrnInfo	Gets the configured values of a trend tag.
TrnNew	Creates a new trend at run time.
TrnSelect	Sets up a page for a trend.

Window Functions

WinCopy	Copies the active window to the Windows clipboard.
WinFile	Writes the active window to a file.
WinNewAt	Opens a new display window at a specified location, with a selected page displayed.
WinPrint	Prints the active window.

Osolete Functions

Cluster Functions

ClusterGetName	Returns the names of the primary and standby cluster servers.
ClusterSetName	Connects to a specific cluster server.

Display Functions

DspCol	Displays a color at an AN.

Task Functions

ReRead	Causes CitectSCADA to re-read the I/O Device data associated with the current Cicode task. Tags are now subscribed at the start of a function and updated tag values are sent to the subscribing function. Tag subscriptions are made at the update rate of: ● the graphics page if called from a page ● the default subscription rate as determined by [Code]TimeData in the Parameters online help if called from Cicode (default 250 ms) ● the update rate requested of a task created using TaskNewEx ● the update rate requested of a subscription created using TagSubscribe Verify that the subscription update rate matches the requirements of your function. After removing ReRead from looping code you may need to extend the period of the Sleep function. This is to replace the pause ReRead created while it read the tag values.

CtAPI Functions in v7.0

The following section details the changes made to CtAPI functions in CitectSCADA v7.0.

Obsolete Functions

Previously available "Point" related functions are now no longer available, and if used will detect and return an error indicating that they are not supported. In order to obtain the same result as was previously invoked by those function, replace them with the Tag based equivalent using the appropriate Tag arguments and conditions.

The "point" functions that are no longer available are listed below along with their replacement functions:

Function	Replacement
ctPointGetProperty	ctTagGetProperty
ctPointNew	ctListNew or ctTagWrite and ctTagRead
ctPointRead	ctTagRead or ctListRead with ctListData
ctPointWrite	ctTagWrite or ctListWrite

If you are using the point functions on single tags, use the ctTagRead, ctTagWrite functions instead. If you are building up multiple tags into one point, use ctListNew and add tags to the list through ctListAdd. Then use ctListWrite, ctListRead and ctListData to write and read from the tags.

The following functions are not relevant to tag based operations. They are obsolete and there is no replacement function.

- ctPointBitShift
- ctPointClose
- ctPointCopy
- ctPointDataSize
- ctPointToStr
- ctStrToPoint
- ctTagToPoint

What's New in CitectSCADA v7.10

CitectSCADA v7.10 incorporates the following new features.

Introduced in v7.10:

- Windows® Integrated Security
- CitectSCADA Security Enhancements
- Multi-Signature Support
- Edit DBF Files in Microsoft® Excel
- Enhanced Driver Installation
- New Font Selection for Graphics Button
- Microsoft® Windows Vista™ Support
- New Location for Configuration and Project Files

- New Alarm Field Enhancements
- New Time Synchronization Service

For changes to:

- Citect.ini parameters, see Citect.ini Parameters in Version 7.10.
- Cicode functions, see Cicode Functions in Version 7.10.

See Also

What's new in CitectSCADA v7.x

CitectSCADA Security Enhancements

The CitectSCADA v7.20 release includes changes that are designed to reduce the security exposure of the product from external threats via the network. The product features that have been affected are detailed below. Please review the list to understand what effect they may have on your system with regards to the upgrade and design process.

Managing surface area

A set of new configuration parameters have been added to provide control over the CitectSCADA network interfaces. These parameters help you protect your system by allowing control over unused features of the product. The following services can be enabled / disabled: DDE, Remote CTAPI, ODBC, OLEDB and FTP.These services are disabled by default.

User login necessary for control actions

A user is now necessary to be configured and logged in to CitectSCADA to allow the display process to perform a tag write (control) action. Design CitectSCADA projects to avoid Cicode task that perform tag writes that are not issued by a user.

We advise that projects be configured to take advantage of the change to provide increased system security protection. If your system has existing network security protection in place and does not require the additional security protection, it can be turned off using the following parameters to avoid the impact of the changes:

Parameter for the client/display node: See [LAN] SecureLogin in the Parameters help file for more information.

Parameter for the server node: See [LAN] AllowLegacyConnections in the Parameters help file for more information. (As part of CitectSCADA7.20 this parameter was made obsolete)

These parameters may be necessary during an upgrade process when there is a mix of old and new version CitectSCADA nodes in a running system.

See Also

System Parameters

Windows Integrated Security

In CitectSCADA you now have the ability to incorporate CitectSCADA users and security options with the standard Windows security system. Of course you can still use the CitectSCADA native security if you prefer to define users in the project and logon to CitectSCADA runtime.

Using the integrated Windows security feature, the Windows user can logon to CitectSCADA runtime with runtime privileges configured within the project.

See Also

Using Windows Security

Multi-Signature Support

CitectSCADA now provides the facility for up to four users to approve an action or tag write operation using the new Cicode functions MultiSignatureForm and MultiSignatureTagWrite.

Two further Cicode functions, VerifyPrivilegeForm and VerifyPrivilegeTagWrite, enable you to restrict access to a specific action or tag write for a user with a specific set of privileges.

Edit .dbf Files in Microsoft® Excel

CitectSCADA allows you to edit and save .dbf files (tables) used in CitectSCADA by opening them in Microsoft® Office Excel®.

Microsoft Office Excel® 2007 does not allow you to save files in .dbf format though you may open and edit them using the File > Open command. In order to overcome this limitation CitectSCADA includes an Add-In for Microsoft Excel called ProjectDBFAddIn. When this Add-In is loaded into Excel, it allows you to browse, open, edit and save CitectSCADA .dbf files in the correct format.

See Also

Using Microsoft Excel to Edit .dbf Tables

Enhanced Driver Installation

The installation of CitectSCADA prior to v7.10 installed the communication drivers automatically with the product. From v7.10, the installation of these drivers is performed at the final stage of the product installation using a separate installation process. This installation process allows you to select individual drivers that you want to install, specific to your system and its I/O Devices.

There are certain drivers that the product installation will install that are necessary for CitectSCADA to function correctly. These will be installed automatically as in previous releases.

If you are using the Microsoft® Windows Vista™ operating system verify that any drivers which you select to install are identified as being compatible with that operating system. If you select any driver that is not yet identified as being compatible, or is specifically identified as not compatible, the installation process will provide an alert to that effect, and will allow you to deselect the driver prior to continuing with the installation.

Note: If you choose to ignore any alert, the driver will be installed but the driver may not operate correctly.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

- Do not ignore alerts during driver installation.
- If alerts are preventing the installation of a driver, contact Technical Support of this product.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

The communication driver installation can also be invoked individually at any time after the product installation to install additional drivers.

New Font Selection for Graphics Button

In previous releases of CitectSCADA, you were not able to change the properties of text such as font, size, style on buttons in the Graphics Editor. This inability to configure the button text properties led to graphics with text from different source objects having different font settings on the same page, which appears aesthetically untidy and inconsistent on the runtime displays.

From v7.10, the text displayed on a button object can be configured in the same manner as other CitectSCADAtext objects within the Graphics Editor and the automation interface. This will allow you to present a more polished and consistent GUI to meet individual project runtime presentation requirements.

When migrating from a previous release, button object text properties are preserved and converted to the new button object text properties with the appropriate default property values automatically placed in the new configuration such as Font=Arial, Size=12, Alignment=centre, style=regular, etc.

See Also

Button Properties - Appearance

Microsoft Windows Vista Support

CitectSCADA V7.10 has achieved the Microsoft® "Works with Windows Vista™" certification. However, merely meeting the requirements of this certification was not sufficient to make CitectSCADA fully functional on Vista. A number of other changes were necessary to achieve satisfactory functionality on the Vista operating system.

Version 7.10 also satisfies many of the requirements of the "Certified for Windows Vista™" certification, and by having this level of qualification we are confident that our users will find minimal differences when running the product on the Vista operating system compared to previous operating systems.

New Locations for Configuration and Project Files

Due to security changes in Windows Vista, some modifications to the location of configuration and user files used by CitectSCADA have been made.

When installed on Windows XP or earlier, configuration and project files are by default stored in the Documents and Settings/All Users/Application Data/Citect/CitectSCADA 7.xx/ folder. When installed on Windows Vista, configuration and project files are by default stored in the ProgramData/Citect/CitectSCADA 7.xx/ folder. Install locations are as follows:

File type	Platform	Install Path
Configuration files such as the citect.ini file	Pre-Vista	Documents and Settings/All Users/Application Data/Citect/CitectSCADA 7.xx/Config
	Vista	ProgramData/Citect/CitectSCADA 7.xx/Config

File type	Platform	Install Path
User directory	Pre-Vista	Documents and Settings/All Users/Application Data/Citect/CitectSCADA 7.xx/User
	Vista	ProgramData/Citect/CitectSCADA 7.xx/User
Data directory	Pre-Vista	Documents and Settings/All Users/Application Data/Citect/CitectSCADA 7.xx/Data
	Vista	ProgramData/Citect/CitectSCADA 7.xx/Data
Log files All log files produced by drivers are written to a sub-folder called 'Drivers'.	Pre-Vista	Documents and Settings/All Users/Application Data/Citect/CitectSCADA 7.xx/Logs
	Vista	ProgramData/Citect/CitectSCADA 7.xx/Logs

Alarm Field Enhancements

There are four enhancements to alarm fields:

- Runtime writes to custom alarm fields
- Alarm summary field changes
- Alarm display field changes
- Alarm paging

Runtime writes to custom alarm fields

It is now possible to write to the eight custom alarm fields during runtime. In previous releases these fields could really only be used for alarm filtering.

Alarm summary field changes

Alarm Summary Fields can now be used to format an alarm display or alarm log device. In addition any Alarm Display Field can be used in your alarm summary, apart from State.

New Alarm Summary Fields

Field Name	Description
{SumType,n}	Type of alarm summary (similar to alarm display "Type").

Alarm display field changes

Now any alarm display field can be used for any type of alarm. Where not applicable for a particular alarm type, zero or an empty string will be displayed.

New Alarm Display Fields

Field Name	Description
{TagEx,n}	Alarm Tag with Cluster Name prefix
{AlarmType,n}	Alarm type (string), not localized.
{TypeNum,n}	Alarm type number (use AlarmType to get string value instead). V
{AlmComment,n}	The text entered into the Comment field of the alarm properties dialog.
{Cluster,n}	Cluster Name
{CUSTOM1,n} {CUSTOM2,n} {CUSTOM3,n} {CUSTOM4,n} {CUSTOM5,n} {CUSTOM6,n} {CUSTOM7,n} {CUSTOM8,n}	Alarm custom fields as configured.
{LocalTimeDate,n}	Alarm date and time.
{Paging,n}	Indicates whether the alarm has to be paged.
{PagingGroup, n}	Indicates the paging group to which the alarm belongs.

Alarm paging

The CitectSCADA alarm facility constantly monitors equipment data and alerts operators of any equipment error or alarm condition. When an alarm is triggered it is displayed on the standard alarm display page. The operator has to be continuously sitting in front of an HMI monitoring the system. CitectSCADA v7.20 provides the facility to link alarms with a remote paging system for operators.

Two Alarm Properties have been added to enable CitectSCADA to interface with any third-party paging system. The Paging property is a flag to indicate that the alarm is going to be paged, the PagingGroup property is a freeform text field indicating the sequence of people to notify in the event the alarm occurred.

See your third-party paging system documentation for information on how to interface with CitectSCADA.

New Time Synchronization Service

In order to maintain time synchronization CitectSCADA now installs a Windows service called TimeSyncService, which runs under the built-in LocalSystem account. This replaces the existing time synchronization server which is not compatible with Windows Vista. This purpose of this service is to maintain the time on the local computer against one or more time sources.

A Time synchronization utility is provided by CitectSCADA to assist you to configure time synchronization, and control the service as part of your administration environment. This utility requires administrator rights as it configures and controls a windows service. When run on Windows Vista with User Access Control (UAC) on, you will be prompted to elevate to an administrator. When run on earlier operating systems, the utility will exit after displaying an error if the current user is not an administrator on the local machine.

See Also

Time Synchronization

New Equipment Database Functions and Forms

An equipment database form is provided by the CitectSCADA Project Editor to assist in the configuration of system equipment. In addition, several Cicode functions have been added to allow you to programmatically load, search and edit the database at runtime.

See Also

Using Equipment

Equipment Database Functions

Citect.ini Parameters in v7.10

This topic lists the parameters that have been added or changed in version 7.10 of CitectSCADA.

It includes:

- New parameters
- Obsolete parameters

New Parameters

The following parameters are new in version 7.10 . For an entire list of the system parameters, refer to theParameters documentation.

Alarm Parameters:

[Alarm]ArgyleTagValueTimeout	Defines the length of time that the alarm server will wait for argyle tag values to become available (without error) before starting to scan for argyle alarms.

Code Parameters:

[Code]HaltOnInvalidTagData	When enabled will cause the cicode to halt when any tag read returns invalid data.

Client Parameters:

[Client]AutoLoginMode	Set to enable auto login. Users can select one of seven modes.

CtApi Parameters:

[CtAPI]AllowLegacyConnections	When enabled current version of CTAPI server can accept connections from previous versions of CTAPI client.
[CtAPI]AllowLegacyServices	When enabled the Citect Web Service and the Citect OLEDB Provider can connect to the CTAPI server.

DDE Parameters:

[DDE]AllowCicode	Allows Cicode to be run on the Citect server via the DDE Execute command.
[DDE]AllowWrites	Allows tag writes to the Citect server via the DDE Poke command.

Kernel Parameters:

[Kernel]ErrorBuffers	The total number of error buffers available for logging to the syslog.dat file.

Lan Parameters:

[LAN]AllowLegacyConnections	Set to allow previous versions of client to connect to the server.
[LAN]AnonymousLoginName	The name of the default identifier to allow 'view-only' data access for a client process to the SCADA server(s).
[LAN]SecureLogin	When set to 0 security measures are disabled and the system acts as it did in versions prior to 7.10.
[LAN]ServerLoginEnabled	Set to disable default server login.
[LAN]ServerLoginName	The name of the default identifier to allow data access for a server process to another SCADA server process(es).

ODBC Parameters:

[ODBC]Server	Set to enable ODBC connections.

Page Parameters:

[Page]AllowHScroll	Defines the default behavior for horizontal scrolling.
[Page]AllowHScrollBar	Defines the default behavior when displaying horizontal scroll bars.
[Page]AllowVScroll	Defines the default behavior for vertical scrolling.
[Page]AllowVScrollBar	Defines the default behavior when displaying vertical scroll bars.

Obsolete Parameters

Win Parameters:

[Win]CtrlEsc	Determines whether the Windows key command [Ctrl]+[Esc] can be used in the runtime system (to display the start menu).

Com Parameters:

[Com]StartTimeout	Determines the period to wait for I/O Devices to come online before displaying any data.

Cicode Functions in v7.10

Some Cicode functions have been introduced, modified, deprecated or removed. The following sections detail the changes made to these functions:

New Functions

Security Functions

FormSecurePassword	Adds both a password prompt and edit field to the current form.
MultiSignatureForm	Displays a form that allows up for 4 users to have their credentials verified in order to approve an operation.
MultiSignatureTagWrite	Displays a form that allows up for 4 users to have their credentials verified in order to approve a write of a specific value to a specific tag.
UserLogin	Logs an operator into the CitectSCADA system using a secure password string.
UserVerify	Uses the authentication functionality in the user login system
VerifyPrivilegeForm	Displays a form that allows a single user to enter their credentials.
VerifyPrivilegeTagWrite	Displays a form that allows any single user to enter their credentials in order to approve a write of a specific value to a specific tag.

Miscellaneous Functions

KernelQueueLength	Obtains the number of rows in a queue.
KernelTableInfo	Provides a consistent method of accessing items within Kernel Table.
KernelTableItemCount	Obtains the number of rows in a Kernel Table
ProcessRestart	Restarts the current process in which Cicode is running.
ServerRestart	Restart any alarm, report, trend or I/O server from any Cicode node in system, without affecting other server processes running on same machine.

Tag Functions

TagRDBReload	Works in conjunction with the TagInfo function. Reloads the variable tag database so when TagInfo is called it picks up online changes to the tag database.

Windows Functions

WinStyle	Switches on and off scrolling and scrollbar features for existing windows.

Modified Functions

None

Obsolete Functions

Window Functions

WndGetProfile	Gets the value of a WIN.INI parameter. If called it will return 0.
WndPutProfile	Updates a parameter in WIN.INI. If called it will return 0.

Time and Date Functions

TimeSet	Sets the new system time

What's New in CitectSCADA v7.20

CitectSCADA v7.20 incorporates the following new features.

Introduced in v7.20:

- Control SCADA Client Connections
- Dynamically Optimized Writes
- Environment Variable changes
- Graphics Enhancements
- Improved Client Side Online Changes
- Improved Installation Process
- Improved CitectSCADA Security
- Multi-process Support in Demo Mode
- New Example Project

- New Web-based Help
- OPC Factory Server (OFS) Driver
- Pelco Camera Support
- Performance Improvements
- Persisted I/O Memory Mode
- Post Compile Commands
- Server Side Online Changes
- Microsoft® Windows 7 Support
- Supportability Enhancements
- Tab Menu Templates
- Tag Extensions
- Time Scheduler

For changes to:

- Citect.ini parameters, see Citect.ini Parameters in Version 7.20.
- Cicode functions, see Cicode Functions in Version 7.20.
- CtAPI functions, see CtAPI Functions in Version 7.20.

For details on how to configure an existing project to run in v7.20, refer to Upgrading to V7.20.

See Also

What's New in CitectSCADA v7.x

Control SCADA Client Connections

Two Citect.ini parameters determine how a client will behave if it is unable to maintain a connection with a primary Alarms, Reports or Trends server. Each server type has access to these parameters [Type.ClusterName.ServerName]Priority and [Type.ClusterName.ServerName]DisableConnection, where Type is the relevant server type (Report, Trend or Alarm).

See Also

Connectivity Parameters

Dynamically Optimized Writes

Following the move to the new Publish-Subscribe infrastructure with Version 7.0, a number of customers were adversely affected by a change in the way the product behaves in respect to Block Writes. This change was generic to the driver interface and specific issues have been raised with in regard to HITACHI, MODBUS and OPC, however other drivers may also have been affected.

In Version 7.20 changes have been made to the way that writes are performed at the I/O Server in order to restore the pre-V7.0 behavior.

These changes will result in a similar level of blocking as occurred in previous versions. It does not confirm that writes will be blocked, but it is more than likely that they will be if they are initiated close enough together.

This will also allow use of the re-instated Citect.ini parameter [IOServer]BlockWrites in order to choose whether to use the Block Writes functionality.

See Also

[IOServer]BlockWrites in the Parameters on line help

Environment variables in 7.20

When you define a Super Genie, you are creating a Super Genie template, similar to a page template. When a Genie calls the Super Genie, this template is used to create a new Super Genie page. Environment variables saved with the template are now propagated to the Super Genie page. When subsequent changes are made to the environment variables of the Super Genie template, the environment variables of the Super Genie page are also updated.

If you modify a Super Genie template when the project is running in the background, you need to perform an Update Pages to see the changes in the runtime project. If a runtime page (created from a Super Genie template) is displayed when the change is made, it will not be updated until you exit then re-display it.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

When upgrading from a previous version of CitectSCADA to v7.20 existing Super Genie template environment variables will override Super Genie page environment variables. Any manual updates you made to Super Genie page environment variables prior to the upgrade will be lost.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Graphic Enhancements

Enhancements have been made to how you can configure graphic pages and the objects you place on the page. These enhancements can be used in the creation and implementation of Genies and Super Genies.

In addition, wide screen formats are now natively supported including 16:10 and 16:9 aspect ratios.

See Also

Metadata

Improved Installation Process

The installation process of CitectSCADA has been improved to simplify the operation and guide the user through the installation by use of Installation Profiles and the creation of default component selections. Whilst still allowing flexibility for the experienced user, the complexity and multiple installation paths and options have been greatly reduced. The installer has been enhanced to allow the installation of a runtime-only version of the product. This allows the runtime environment to be installed without the configuration tools of the CitectSCADAIntegrated Environment. The Runtime Only installation provides not only a smaller installation footprint but also the ability to set up workstations which do not allow project configuration. This automatically improves the security of the system configuration.

Note: You can also install the CitectSCADA runtime from a single installation file. This file is on the installation DVD. This allows installation of the software to computers which only need the runtime. The file can be copied to a network location for remote installation.

Improved CitectSCADA Security

Security enhancements have been implemented in this release to address known security issues from previous versions and to reduce the possibility of malicious attack. These security enhancements include, improved inter-operability through the introduction of new INI parameters, trusted network authentication, and the addition of assigning roles to CitectSCADA runtime users, as you currently do for Windows users.

Note: The [Privilege]Shutdown parameter is now used to specify the necessary privilege level of a user to perform a shutdown operation triggered by clicking the Close button of the project (default privilege level is 8).

Multi-process Support in Demo Mode

If using a demonstration version of CitectSCADA you can now design and run your system in multi-process mode (with TCP\IP turned off). This new default for v7.2 will assist you in finding and fixing design flaws in your project , which may have previously gone unnoticed due to only being able to run and test your system in single process mode.

See Also

Demo Mode

New Example Project

The Example Project has been updated to demonstrate the new tab menu templates that are available with CitectSCADA v7.20.

The project includes a "What's New?" menu to introduce some of the new features offered in v7.20. This menu links to pages that demonstrate:

- the use of tag extensions and tag properties on graphics pages
- server monitoring and the ability to implement online changes for alarm and trend servers
- multi-monitor support
- Instant Trending using the Process Analyst

The new content complements pages drawn in from the existing Example Project and CSV_Example Project, which are now superseded.

To view the new Example Project, select and run it from Citect Explorer. For more information, use the help button included in the project on the main navigation panel.

Web-based Help

The wide spread use of the Internet to distribute information has now enabled Schneider Electric (Australia) Pty. Ltd. to introduce that technology, to bring its users a new rich and expanded format to deliver its help information.

The traditional on-line help, now referred to in CitectSCADA as PC-based Help, is still available to provide immediate information which is related to the task at hand. If you are a registered Technical Support customer you have the option of accessing the new Web-based Help to provide the very latest information, which will include updates for help topics to include the improvements to the software application introduced by Service Packs.

Updates will also be included from enhancement to the help topics brought about by user comments and general internal developments by the Schneider Electric (Australia) Pty. Ltd. Technical Publications team.

You will also be able to "rate" individual help topics with a star rating, and add your own comments to any help topics. Your comments and topic ratings will be retained in our database and we will endeavor to incorporate your contributions in our ongoing development of the quality of our documentation.

The future development of Web-based help will also include links to technical documentation in the form of:

- Technical Papers
- White Papers
- Knowledge Base content
- Education and Training material

The Web-based help is planned for release by the end of December 2010. So, please check on its availability periodically with following the link www.citect.com/webhelp

OFSOPC Driver

The release of CitectSCADA v7.20 coincides with the availability of the OFSOPC Driver for Schneider Electric's OPC Factory Server (OFS).

OFS Factory Server is a foundation component for communication with Schneider Electric PLCs. The OFSOPC Driver allows CitectSCADA to tightly integrate with OFS Factory Server, minimizing the amount of configuration necessary for an end-to-end Schneider Electric system.

You can install the OFSOPC Driver and its supporting documentation via the Driver Selection page of the CitectSCADA v7.20 installer.

Pelco Camera Support

This feature adds a button to the Graphics Builder toolbox, which will allow two of the Pelco Camera ActiveX controls to be easily added to a graphics page. This control provides an ActiveX component that will connect to Pelco IP cameras with configurable bandwidth usage for slow network connections and auto-resizes video to fit the ActiveX control size.

The two ActiveX controls supported are:

Video Streaming - Fully Resizable, multiple bandwidth levels, MPEG4 Video, returns the camera name and model.

Camera Control PTZ (Pan, Tilt and Zoom) - Communicates with DVRs and IP cameras. Featuring pan zoom and tilt, iris, focus, presets, patterns and adjustable speed.

The button is only enabled if you have installed the Pelco camera software. Refer to the Pelco camera software documentation for further information.

For details on configuring and using the Pelco Camera ActiveX control, refer to the documentation installed with the Pelco product.

See Also

Using Objects

Performance Enhancements

The architecture of CitectSCADA Version 7.20 includes a new threading model that offers significant performance improvements. The new Platform Task Framework (PTF) defines an explicit threading environment for each subsystem, providing a standard protocol for work to be created and passed between them.

The performance improvements have been implemented in a way that retains existing functionality.There is no changes to the configuration or operation of a system, just performance benefits and improved stability.

Persisted I/O Memory Mode

With the release of V7.20, a feature called persistence can be applied to I/O devices. When implemented, the tag cache for an I/O device is written to an XML file at a set period, allowing the last available data to be reloaded following a shutdown. Persistence is enabled using the **Persist** field in the extended section of the I/O Devices Properties dialog.

A persisted memory I/O device is recommended as an improved alternative to a disk I/O device, as synchronization is supported if a server becomes unavailable for a period of time.

> **Note:** It is recommended that data assigned to disk I/O devices be migrated to the new persisted memory I/0 mode. See the topic Persisted I/O Memory Mode for information on how to migrate a disk I/O device to a persisted memory I/O device.

See Also

Using Persisted I/O Memory Mode

Post Compile Commands

After a project has compiled successfully you can execute an optional command, script or batch file. This offers useful functionality if you have tasks that could be automated after a successful compile. This provides an expansion point for you to add your own script or command to perform additional tasks. You can also launch an optional command, script or batch file to execute after an unsuccessful compile.

See Also

Post Compilation

Improved Client Side Online Changes

Prior to V7.20, you could not load a page if the Cicode library has changed without restarting the display client. In V7.20, you can now reload a page when the Cicode library has changed. This will result in the page being reloaded and any Cicode on that page will use the memory version of the Cicode library, not the latest compiled version. This may cause a hardware alarm to be raised.

See Also

Client Side Online Changes

Server Side Online Changes

To improve the ability to change configurations on a live system without having to restart the servers, CitectSCADA now provides the facility to reload server configurations during runtime either programmatically or using the Runtime Manager.

See Also

Server Side Online Changes

Microsoft Windows 7 Support

CitectSCADAv7.20 also supports the Microsoft Windows 7 and Microsoft Server 2008 R2 operating systems. The changes to CitectSCADA undertaken in the 7.1 release to support Windows Vista significantly reduced the changes that were necessary to support Windows 7 and Server 2008 R2. Previous Vista users will experience no functional differences in CitectSCADAwhen migrating to Windows 7. However if you migrate to Windows 7 from Widows XP be aware of the functional differences with CitectSCADAbetween XP and Vista, as described in New Locations for Configuration and Project Files.

Supportability Enhancements

Supportability Enhancements have been added to CitectSCADA to provide easier access to the diagnostics functionality of the product. Although the enhancements were primarily introduced to assist Technical Support personnel with system analysis, they have resulted in many benefits to the end user. These include:

- Timestamp harmonization across log files.
- Additional [Debug] parameters to support category and severity filtering (see Citect.ini Parameters in Version 7.20).
- Support for online logging adjustments using the new SetLogging() and GetLogging() Cicode functions.
- A set of parameters that can be modified while online due to periodic or an on-demand read of the citect.ini file during runtime.

Additionally, the home page of the Computer Setup Editor now includes a link to the Logging Parameters page, which provides comprehensive instructions for the configuration of logging.

See Also

Configuring logging

Adjusting logging during runtime

New Tab Menu Templates

To improve the user interface of projects and integrate the look and feel with the latest Windows® systems, CitectSCADA now features an optional ribbon style menu system. Main menu items can be represented as tabs along a menu bar, below which subsidiary items are displayed in a ribbon. New projects have the new Tab_Style_Include templates already available to them.

Starter Project

In addition to the new Tab Style templates, it is now possible to create a compilable starter project that contains a basic level of built-in functions such as viewing alarms and trends.

See Also

Tab Menu Templates

Creating a New Tab Style Project

Tag Extensions

With the addition of Tag Extensions in CitectSCADA v7.20, the variable tag can now represent data as a collection of elements, and each of these elements can contain a collection of items.

Each element provides access to a view of the data value for the tag. Each variable tag can be used on its own or by referencing a particular element. The tag and each element have items that can be referenced to access the following information:

.v : The value, which will access the data value of the tag or element.

.vt : The value timestamp, which will access the timestamp of when the value last changed.

.q : The quality, which will access the quality of the value , either GOOD, UNCERTAIN or BAD. The Quality variable can access further detail using the Cicode Quality functions.

.qt : The quality timestamp, which will access the timestamp of when the quality last changed.

.t : The timestamp, which will access the timestamp of when the tag or element was last updated.

See Also

Tag extensions

Time Scheduler

The Time Scheduler is a calendar based programming tool that allows you to manipulate tag values within a CitectSCADA project. It can be used to create a sequence of automatically executed commands, delivering a valuable scheduling tool for applications. If you choose the Time Scheduler follow the on screen instruction.

For details on configuring and using the Time Scheduler, refer to the documentation installed with the Time Scheduler product.

Citect.ini Parameters in v7.20

This topic lists the parameters that have been added or changed in version 7.20 of CitectSCADA.

It includes:

- New parameters
- Modified parameters
- Re-installed parameters
- Obsolete parameters

New Parameters

The following parameters are new in version 7.20 . For an entirelist of the system parameters, refer to the Parameters documentation.

Alarm Parameters

[Alarm.ClusterName.ServerName]DisableConnection	Specifies if a client will not connect to a server.
[Alarm.ClusterName.ServerName]Priority	Specifies the client priority for the server connection.
[Alarm]ReloadBackOffTime	Back-off time configured to control the pace of the reload on an alarm server.

Client Parameters

[Client]AutoLoginClearPassword	When set to 1 the cache is cleared of any client login credentials for consistency with the [Server]AutoLoginClearPassword ini parameter.
[Client]DisableDisplay	Sets whether to allow the client process to run in the back-

	ground without a visible window.
[Client]EvictTimeout	Sets the amount of time a tag reference is cached before it is evicted.
[Client]PartOfTrustedNetwork	Tells a Client process to attempt to authenticate using the stored server password. It is automatically set by the Computer Setup Wizard.
[Client] StalenessPeriod	Number of seconds to use for tag staleness period.
[Client]StalenessPeriodTolerance	Staleness period tolerance

CtAPI Parameters

[CtAPI]RoundToFormat	Indicates to the user if values rounded to format.

CtDraw.RSC Parameters

[CtDraw.RSC]-AllowEditSuperGeniePage	When set enables the user to choose whether or not to open and edit a Super Genie page.

CtEdit Parameters

[CtEdit]CompileSuccessfulCommand	Indicates to the compiler an optional command, script or batch file to execute after a successful compile.
[CtEdit]CompileUnsuccessfulCommand	Indicates to the compiler an optional command, script or batch file to execute after an unsuccessful compile.
[CtEdit]Starter	Specifies the directory where the starter projects are located.

Debug Parameters

[Debug]ArchiveFiles	Archives log files once the size specified by [Debug]MaximumFileSize is reached.
[Debug]CategoryFilter	Allows you to filter logging messages by component category.
[Debug]CategoryFilterMode	Enables logging of categories declared by the [Debug]CategoryFilter value.
[Debug]EnableLogging	Enables or disables the logging mechanism.
[Debug]MaximumFileSize	Sets the maximum size for a log file.
[Debug]Priority	Allows you to filter logging messages according to their priority.
[Debug]SeverityFilter	Allows you to filter logging messages according to their severity.

[Debug]SeverityFilterMode	Enables logging of severities declared by the [Debug]SeverityFilter value.

General Parameters

[General]MiniumlUpdateRate	Specifies the time period (sec) at which a DataSource will send tag update value notifications to the subscription clients.
[General]StalenessPeriod	Specifies the time period (sec) after which a tag value is considered to be "stale" if it was not updated during this period.

IOServer Parameters

[IOServer]EnableEventQueue	Enables the event queue.
[IOServer]MaxEventsDrop	Sets the number of events that are dropped when too many are queued.
[IOServer]MaxEventsQueued	Sets the total number of events that can be queued.
[IOServer]MaxTimeInQueueMs	Sets the total time for which an event can be queued.

LAN Parameters

[LAN]AllowRemoteReload	Enables remote reloading of servers from a client.
[LAN]ClientRetryTime	Sets the length of time between connection attempts by a client.
[LAN]EarliestLegacyVersion	Specify the minimum legacy version from which the current version will accept connections.
[LAN]HighWaterMark	The number of messages waiting to be sent on a particular network connection at which the high water mark event will occur.
[LAN]KeepAliveInterval	Sets the length of time between two keep alive transmissions by the client.
[LAN]KeepAliveTime	Sets the length of time between two keep alive transmissions in idle conditions.
[LAN]ListenerRetryTime	Sets the length of time a server waits between attempts to listen for a client connection.
[LAN]LowWaterMark	After the high water mark has been reached on a particular network connection, the low water mark represents the number of messages waiting to be sent at which we will resume normal operations.
[LAN]NoSocketDelay	Switches off the delay on a socket caused by the use of the Nagle algorithm.

[LAN]ReadOnlyLegacyConnections	When set to 1 v7.10 clients can only communicate in read-only mode.This parameter overrides 'EarliestLegacyVersion' .

Multi-Monitor Parameters (CSV Include project)

[MultiMonitor]DisableAutoStart	Disables the new multi-monitor functionality.

Page Parameters

[Page]AddDefaultMenu	Determines whether to add the default menu items to the tabbed menu bar.
[Page]BadDitheringColor	Sets the dithering color for graphics elements which are dithered if the value quality is "bad".
[Page]BadDitheringDensity	Sets the dithering density for graphics elements which are dithered if the value quality is "bad".
[Page]BadText	Text Objects can be displayed as #COM type errors, or as the text overlaid with a dithered pattern if the 'display value' expression has "bad" quality.
[Page]BadTextBackgroundColor	Sets the background color for numeric / text graphics objects to indicate "bad" quality.
[Page]EnableQualityToolTip	Set by default it controls the quality tooltip
[Page]ErrorDitheringColor	Sets the dithering color for graphics elements which are dithered if an internal error occurs.
[Page]ErrorDitheringDensity	Sets the dithering density for graphics elements which are dithered if an internal error occurs.
[Page]ErrorTextBackgroundColor	Sets the background color for numeric / text graphics objects to indicate an internal error.
[Page]IgnoreValueQuality	Defines the value quality handling by graphics pages.
[Page]OverrideDitheringColor	Sets the dithering color for graphics elements which are dithered if their values are override ("forced").

[Page]OverrideDitheringDensity	Sets the dithering density for graphics elements which are dithered if an internal error occurs.
[Page]OverrideTextBackgroundColor	Sets the background color for numeric / text graphics objects to indicate that the value presented on the objects is override (“forced”).
[Page]ShowBadText	Text Objects can be displayed as #BAD text, or as the text overlaid with a dithered pattern if the "display value" expression has “bad” quality.
[Page]ShowErrorText	Text Objects can be displayed as #COM type errors, or as the text overlaid with a dithered pattern if the ‘display value’ expression has “uncertain” quality.
[Page]ShowUncertainText	Text Objects can be displayed as #UNC text, or as the text overlaid with a dithered pattern if the "display value" expression has “uncertain” quality.
[Page]Splash	Specify the name of the Splash Screen page.
[Page]SplashTimeout	Time in milliseconds for the Splash Screen to display.
[Page]SplashWinName	Specify the label of the Splash Window for use with the Cicode function WinNumber().
[Page]StartupDelay	Milliseconds between when Splash Screen and Start Screen are displayed.
[Page]StartupHeight	Height of the Start Page on main display monitor.
[Page]StartupMode	Mode of Start Page on main display monitor.
[Page]StartupWidth	Width of the Start Page on main display monitor.
[Page]StartupWinName	Specify the label of the Start Window for use with the Cicode function WinNumber().
[Page]StartupX	X coordinate of Start Page on main display monitor.
[Page]StartupY	Y coordinate of Start Page on main display monitor.

[Page]UncertainDitheringColor	Sets the dithering color for graphics elements which are dithered if the value quality is "uncertain".
[Page]UncertainDitheringDensity	Sets the dithering density for graphics elements which are dithered if the value quality is "uncertain".
[Page]UncertainText	Text Objects can be displayed as #COM type errors, or as the text overlaid with a dithered pattern if the 'display value' expression has "uncertain" quality.
[Page]UncertainTextBackgroundColor	Sets the background color for numeric / text graphics objects to indicate "uncertain" quality.
[Page]WaitForValidData	Specifies whether the animation system will attempt to wait for valid data from subscriptions necessary to draw a graphics page before it is animated.

Report Parameters

[Alarm.ClusterName.ServerName]DisableConnection	Specifies if a client will not connect to a server.
[Alarm.ClusterName.ServerName]Priority	Specifies the client priority for the server connection.

Runtime Manager Parameters

[RuntimeManager]AllowReload	Enables or disables the reload option in the Runtime Manager menu.

Security Parameters

[Security]DisableDEP	Set to turn off DEP protection for the CitectSCADA runtime.

Server Parameters

[Server]AutoLoginMode	Determines the auto login method the server will use when establishing connections to other servers.

Trend Parameters

[Trend]AcquisitionTimeout	Sets a timeout to stop a trend server infinitely acquiring a valid data sample from an I/O device.

[Trend.ClusterName.ServerName]DisableConnection	Specifies if a client should not connect to a server.
[Trend.ClusterName.ServerName]Priority	Specifies the client priority for the server connection.
[Trend]ReloadBackOffTime	Back-off time configured to control the pace of the reload on an Trend server.

Modified Parameters

CtEdit Parameters

[CtEdit]Copy	Supports runtime changes, it enables you to switch the SCADA node to use a new runtime configuration by pointing to a new location.

Re-instated Parameters

IOServer Parameters

[IOServer]BlockWrites	Determines whetherCitectSCADA will try to block optimize writes to I/O devices.

Obsolete Parameters

AnmCursor Parameters

[AnmCursor]Colour	Replaced with [AnmCursor]Color. Sets the color of the cursor.

General Parameters

[General]TagAssMode	Validates the tag name in the Ass Function. Refer to [General]TagDB instead.

LAN Parameters

[LAN]AllowLegacyConnections	Set to allow previous versions of client to connect to the server.
[LAN]ServerLoginEnabled	Set to disable default server login.

Page Parameters

[Page]BackgroundColour	Replaced with [Page]BackgroundColor. Specifies the color used to fill in the background when a page is smaller than the minimum width of a window.

[Page]ComBreak	Determines whether an error status is displayed on the screen if a communication error occurs.
[Page]ComBreakText	Determines the display of text objects if a communication error occurs that affects the text.
[Page]DynamicComBreakColour	Replaced with [Page]DynamicComBreakColor. Sets the color of the ComBreak dithering.
[Page]DynamicComBreakDensity	Sets the density of the ComBreak.

Time Parameters:

[Time]Deadband	The deadband time checked by the Time Server before it adjusts the time on the client(s).
[Time]Disable	Enables/disables the processing of time messages from the Time Server.
[Time]Name	Enables the time synchronization functionality.
[Time]PollTime	The period that the Time Server uses to synchronize other CitectSCADA computers on the network.
[Time]RTsync	Determines whether the Time Server will synchronize with the hardware clock.
[Time]Server	Determines whether this computer is a Time Server.

Trend Parameters

[Trend]CursorColour	Replaced with [Trend]CursorColor. Allows the cursor color to be specified.

CtAPI Functions in v7.20

The following section details the changes made to CtAPI functions in CitectSCADA v7.20.

New Functions

ctListItem	Gets the tag element item data.
ctTagReadEx	Performs the same as ctTagRead, but with an additional new argument

Modified Functions

ctTagRead	Reads the current value from the given I/O device variable tag element value.
ctTagWrite	Writes the given value to the I/O device variable tag elements which have read/write access.
ctTagWriteEx	Asynchronously writes the given value to the I/O device variable tag element value for the tag elements which have read/write access.
ctListAdd	Adds a tag element value to the list.
ctListAddEx	Adds a tag element value to the list.

Obsolete Functions

None

Cicode Functions in v7.20

Some Cicode functions have been introduced, modified, deprecated or removed. The following sections detail the changes made to these functions:

New Functions

Alarm Functions

AlarmCatGetFormat	Returns the display format string of the specified alarm category.
AlarmDspClusterAdd	Adds a cluster to a client's alarm list.
AlarmDspClusterInUse	Determines if a cluster is included in a client's alarm list.
AlarmDspClusterRemove	Removes a cluster from a client's alarm list.

Display Functions

DspAnGetMetadata	Retrieves the field value of the specified metadata entry.
DspAnGetMetadataAt	Retrieves metadata information at the specified index.
DspAnSetMetadata	Non-blocking function, that sets the value of the specified metadata entry.
DspAnSetMetadataAt	Sets the value of a metadata entry.
DspPopupConfigMenu	Displays the contents of a menu node as a pop-up (context) menu, and run the command associated with the selected menu item.

Format Functions

FmtGetFieldCount	Retrieves the number of fields in a format object.

FmtGetFieldName	Retrieves the name of a particular field in a format object.
FmtGetFieldWidth	Retrieves the width of a particular field in a format object.

Menu Functions

MenuGetChild	Returns the handle to the child node with the specified name.
MenuGetFirstChild	Returns the handle to the first child of a menu node.
MenuGetGenericNode	Returns the root node of the default menu tree.
MenuGetNextChild	Returns the next node that shares the same parent.
MenuGetPageNode	Returns the Base menu node of a specific page.
MenuGetParent	Returns the parent node of the menu item.
MenuGetPrevChild	Returns the previous node that shares the same parent.
MenuGetWindowNode	Returns the handle of the root menu node for a given window.
MenuNodeAddChild	Dynamically adds a new item to the menu at runtime.
MenuNodeGetProperty	Return the item value of the specified menu node.
MenuNodeHasCommand	Checks whether the menu node has a valid cicode command associated with it.
MenuNodeIsDisabled	Checks whether the menu node is disabled by evaluating its DisabledWhen cicode expression.
MenuNodeIsHidden	Checks whether the menu node is hidden by evaluating its HiddenWhen cicode expression.
MenuNodeRemove	Remove the menu node from the menu tree.
MenuNodeRunCommand	Run the associated command for a menu node.
MenuNodeSetDisabledWhen	Set the DisabledWhen expression for a newly added node.
MenuNodeSetHiddenWhen	Set the HiddenWhen expression for a newly added node.
MenuNodeSetProperty	Set the item value of the specified menu node.
MenuReload	Reload base Menu Configuration from the compiled database.

Miscellaneous Functions

GetLogging	Gets the current value for one or more logging parameters.
SetLogging	Adjusts logging parameters while online.
ProductInfo	Returns information about the CitectSCADA product.
ProjectInfo	Returns information about a particular project, which is identified by a project enumerated number.

Page Functions

PageBack	Displays the previously displayed page in the Window.
PageForward	PageForward() restores the previously displayed page in the window following a PageBack command.
PageHistoryDspMenu	Displays a pop-up menu which lists the page history of current window.
PageHistoryEmpty	Returns whether page history of the current window is empty.
PageHome	Displays the predefined home page in the window.
PagePeekCurrent	Return the index in the page stack for the current page.
PageProcessAnalyst	Displays a Process Analyst page (in the same window) preloaded with the pre-defined Process Analyst View (PAV) file.
PageProcessAnalystPens	Displays a Process Analyst page (in the same window) preloaded with the pre-defined Process Analyst View (PAV) file and specified trend or variable tags.
PageRecall	Displays the page at a specified depth in the stack of previously displayed pages.
PageTask	Used for running preliminary Cicode before displaying a page in a window.
PageTransformCoords	Converts Page coordinates to absolute screen coordinates.

Process Analyst Functions

ProcessAnalystLoadFile	Loads the specified PAV file to a Process Analyst object, which is identified by parameter ObjName.
ProcessAnalystPopup	Displays a Process Analyst page (in the same window) preloaded with the pre-defined Process Analyst View (PAV) file and specified trend or variable tags.
ProcessAnalystSelect	Allows a set of pens to be selected before displaying the PA page.
ProcessAnalystSetPen	Allows a new pen to be added to a PA display.
ProcessAnalystWin	Displays a Process Analyst page (in a new window) preloaded with the pre-defined Process Analyst View (PAV) file.

Quality Functions

QualityCreate	Creates a quality value based on the quality fields provided.
QualityGetPart	Extracts a requested part of the Quality value from the QUALITY variable.
QualityIsBad	Returns a value indicating whether the quality is bad.
QualityIsGood	Returns a value indicating whether the quality is good.
QualityIsUncertain	Returns a value indicating whether the quality is uncertain.
QualitySetPart	Sets a Quality part’s value to the QUALITY variable.

QualityToStr	Returns a textual representation of the CitectSCADA quality.
QualityIsOverride	Returns a value indicating whether the tag is in Override Mode.
QualityIsControlInhibit	Returns a value indicating whether the tag is in Control inhibit mode.
VariableQuality	Extracts the quality from a given variable.

Server Functions

ServerBrowseClose	This function terminates an active data browse session and cleans up resources associated with the session.
ServerBrowseFirst	This function places the data browse cursor at the first record.
ServerBrowseGetField	This function retrieves the value of the specified field from the record the data browse cursor is currently referencing.
ServerBrowseNext	This function moves the data browse cursor forward one record.
ServerBrowseNumRecords	This function returns the number of records that match the filter criteria.
ServerBrowseOpen	This function initiates a new browse session and returns a handle to the new session that can be used in subsequent data browse function calls.
ServerBrowsePrev	This function moves the data browse cursor back one record.
ServerGetProperty	This function returns information about a specified server and can be called from any client.
ServerReload	This function reloads the server specified by cluster and server name.
ServerIsOnline	This function checks if the given server can be contacted by the client for giving the online/offline status of the server.

String Functions

StrCalcWidth	Retrieves the pixel width of a string using a particular font.
StrTruncFont	Returns the truncated string using a particular font (specified by name) or the specified number of characters.
StrTruncFontHnd	Returns the truncated string using a particular font (specified by font number) or the specified number of characters.

Super Genie Functions

AssMetadata	Performs Super Genie associations using the "Name" and "Value" fields.
AssMetadataPage	Uses the metadata information from the current animation point for the page associations for a new Super Genie page, and displays the new Super Genie in the current page.
AssMetadataPopup	Uses the metadata information from the current animation point for the associations for a new Super Genie page, and displays the new Super Genie in a new pop up window.

AssMetadataWin	Uses the metadata information from the current animation point for the associations for a new Super Genie page, and displays the new Super Genie in a new window.

Tag Functions

SubscriptionGetInfo	Reads the specified text information about a subscribed tag.
SubscriptionGetQuality	Reads quality of a subscribed tag.
SubscriptionGetTag	Reads a value, quality and timestamps of a subscribed tag.
SubscriptionGetTimestamp	Reads the specified timestamp of a subscribed tag.
SubscriptionGetValue	Reads a value of a subscribed tag.
TagSetOverrideBad	Sets a quality Override element for a specified tag to Bad Non Specific.
TagSetOverrideGood	Sets a quality Override element for a specified tag to Good Non Specific.
TagSetOverrideUncertain	Sets a quality Override element for a specified tag to Uncertain Non Specific.
TagSetOverrideQuality	Sets a quality of Override element for a specified tag.

Task Functions

TaskCall	Calls a Cicode function by specifying the function name and providing an arguments string.

Timestamp Functions

TimestampToStr	Converts a TIMESTAMP variable into a string.
TimestampDifference	Returns a difference between two TIMESTAMP variables as a number of milliseconds.
TimestampCreate	Returns a timestamp variable created from the parts.
TimestampFormat	Format a TIMESTAMP variable into a string.
TimestampGetPart	Returns one part (year, month, day, etc) of the timestamp variable.
TimestampToTimeInt	Converts a TIMETSTAMP variable into a time INTEGER which is represented as a number of seconds since 01/01/1970.
TimeIntTo Timestamp	Converts a time INTEGER which is represented as a number of seconds since 01/01/1970 to a TIMETSTAMP
TimestampCurrent	Returns the current system date and time as a TIMESTAMP variable.
TimestampAdd	Adds time (in milliseconds) to a TIMESTAMP variable.
TimestampSub	Subtracts time (in milliseconds) from a TIMESTAMP variable.
VariableTimestamp	Extract the TIMESTAMP from a given variable.

Window Functions

MultiMonitorStart	Displays a CitectSCADA window on each of the configured monitors when a display client starts up.
WinSetName	Associates a name with a particular window by its window number.
WndMonitorInfo	Returns information about a particular monitor.

Modified Functions

Accumulator Functions

AccumBrowseOpen	Opens an accumulator browse session.

Alarm Functions

AlarmDsp	Displays alarms.
AlarmDspLast	Displays the latest, unacknowledged alarms.
AlmSummaryOpen	Opens an alarm summary browse session.
AlmTagsOpen	Opens an alarm tags browse session.

Display Functions

DspStr	Displays a string at an AN.
DspText	Displays text at an AN.

Format Functions

FmtOpen	Creates a format template.

Miscellaneous Functions

Shutdown	EndsCitectSCADA operation.

Page Functions

PageGetInt	Gets a local page-based integer.
PageGetStr	Gets a local page-based string.
PageInfo	Gets information about the current page.
PagePeekLast	Gets any page on the PageLast stack.
PageSetInt	Stores a local page-based integer.
PagesetStr	Stores a local page-based string.

Security Functions

Login	Logs an operator into the CitectSCADA system. Not available when logged

	in as Windows user.

Super Genie Functions

The following functions were updated to accept string identifiers for substitution parameters.

Ass	Associates a variable tag with a Super Genie.
AssGetProperty	Retrieves association information about the current Super Genie from the datasource.
AssGetScale	Gets scale information about the associations of the current Super Genie from the datasource (that is scale information about a variable tag that has been substituted into the Super Genie)
AssInfo	Gets association information about the current Super Genie (that is information about a variable tag that has been substituted into the Super Genie).
AssInfoEx	Retrieves association information about the current Super Genie (that is information about a variable tag that has been substituted into the Super Genie).
AssScaleStr	Gets scale information about the associations of the current Super Genie (that is scale information about a variable tag that has been substituted into the Super Genie).

Tag Functions

SubscriptionGetAttribute	Reads an attribute value of a tag subscription.
TagRead	Reads the value of a particular tag element.
TagWrite	Writes a tag element value for the tag elements which have read/write access.
TagSubscribe	Subscribes to a particular tag element.

Window Functions

WinNumber	Gets the window number of the active CitectSCADA window.
WndInfo	Gets the Windows system metrics information.

Reinstated Functions

Following functions have been reinstated for 7.20.

Time and Date Functions

TimeSet	Sets the new system time

Kernel Commands in Version 7.0

The following Kernel commands are obsolete in CitectSCADA from Version 7.0:

- Kernel Alarm
- Kernel Trend
- Kernel Report
- Kernel IOServer
- Kernel Client
- Probe
- NetBIOS
- PageNetstat

Chapter: 3 Upgrading to CitectSCADA v7.20

To upgrade an existing project to v7.20 from v6.x, perform each of the following procedures.

You do not need to carry out these procedures if you are upgrading from v7.0 or v7.10 to v7.20.

- Upgrade CTAPI Applications
  Verify that CTAPI applications are upgraded before upgrading and running any CitectSCADA V7.x projects.
- Configure I/O Devices
  Before upgrading, verify that I/O Devices are configured as necessary to run in the project.
- Run the Citect Installer
  The installer will lead you through a number of steps until the installation is complete.
- Launch CitectSCADA
  An automatic upgrade of your projects will occur when you initially start CitectSCADA
- Run the Migration Tool
  The automatic update that occurs when you initially launch CitectSCADA does not fully upgrade your projects, as such it needs to be followed by running the Migration Tool.
- Define Clusters
  Clusters can now be defined. The project needs to be configured to use at least one cluster.
- Configure Network Addresses
  The network addresses and ports of the computers to be used as servers are now defined in the project.
- Configure Servers
  The Alarm, Report, Trend, and I/O Servers are now defined in the project.
- Configure Tags to Use Clustering
  Alarms, reports, trends, SPC tags, and accumulators can now be configured to run in a specific cluster.

- Configuring Multiple Monitor Support
  You may need to check whether a custom configuration operates correctly in a multiple-monitor environment.
- Compile the Project
  Once you have configured your project, compile it and verify that there are no errors.
- Run the Computer Setup Wizard
  Run the Computer Setup Wizard for each computer running the project. At each stage of the Wizard, configure the appropriate settings for that computer.

> **Note:** If you are running version 5.5, verify that you upgrade your projects to version 6.x before upgrading to v7.x.

See Also
What's new in V7.x

Upgrade CTAPI Applications

To set up CitectSCADA v7.20 communications with CTAPI applications in your system (including CitectSCADA Reports and Ampla), verify that these products have been upgraded to their latest versions before running any upgraded CitectSCADAv7.20 projects.

Similarly, if you are using any custom CTAPI applications, upgrade CitectSCADA on the computers where these applications are installed before upgrading any other CitectSCADA computers.

See Also
Configure I/O Devices
"CtAPI Functions" in the CitectSCADA Technical Reference

Configure I/O Devices

The upgrade process verifies that the functionality of your project is upgraded to version 7.x. To facilitate this, parts of your project configuration may change during the upgrade. It is therefore important to verify that a project is configured the way you want it to run before you upgrade.

In particular, memory I/O Devices defined in your project (specifying MEMORY as the port) will be configured in the upgraded project as local variables, since memory I/O Devices are no longer supported. Local variables offer the same functionality as memory I/O Devices without the need for further configuration.

However, this also means if you have I/O Devices temporarily defined as memory I/O Devices for testing or simulation, they will be incorrectly configured as local variables by the upgrade process. Verify that you configure these devices as you require them to run before upgrading to version 7.x.

⚠ **WARNING**

UPGRADE ALTERS COMMUNICATIONS CONFIGURATIONS

After upgrading, confirm and adjust the configuration of all I/O devices in your project.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Before attempting to configure your I/O devices for the changes caused by the upgrade process, first read the information in Configuring Local Variables and Using Memory Mode. This provides details about local variables and the other I/O Device options that replace memory I/O Devices, allowing you to select and configure your project appropriately.

Note: Alarm devices with their Protocol property set to "Alarm" are no longer used and will be removed by the Migration Tool. All Alarm Servers will now publish Alarm Properties.

The reconfiguration will take place when you run the Migration Tool. For detailed information on this tool, refer to Migration Tool.

See Also

Run the Citect Installer
Configuring Local Variables
Using Memory Mode

Run the Citect Installer

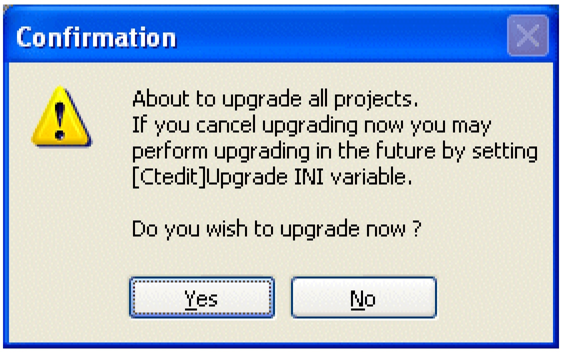
To begin the upgrade, run the CitectSCADAv7.20 installer. The installer will lead you through a number of steps until the installation is complete.

Note: Uninstall any existing version 6.x or version 7.0 before installing v7.20, as CitectSCADA does not support different versions running side-by-side. Additionally, to use the v7.20 Example and CSV_Example projects, it is recommended that you delete the existing Example and CSV_Example projects using Citect Explorer before starting the installation.

Launch CitectSCADA

An automatic upgrade of your projects will occur when you initially start CitectSCADA.

1. To launch CitectSCADA, click **Start | All Programs | Citect | CitectSCADA 7.10 | CitectSCADA Explorer**. The following message will display:



Confirmation

About to upgrade all projects.
If you cancel upgrading now you may perform upgrading in the future by setting [Ctedit]Upgrade INI variable.

Do you wish to upgrade now ?

2. Click **Yes** to confirm the upgrade.

See Also

Define Clusters

Migration Tool

Migration Tool

The automatic update that occurs when you initially launch CitectSCADAv7.20 does not fully upgrade your projects, and needs to be followed by the use of the Migration Tool (if migrating from v6.x this is particularly noteworthy). The automatic update is a passive action which updates the database field definition for any database that has been changed between the two versions and copies new files that are necessary in v7.20. Prior to the automatic upgrade proceeding you are given the option of canceling the upgrade. The upgrade can be invoked at a later time by setting the [CtEdit]Upgrade parameter to 1 (True) in the Citect.ini file.

After the automatic update has completed then prepare your projects prior to running the Migration Tool.

The Migration Tool is a separate application which has to be manually run after the automatic upgrade has been executed, and initiated by you after you have prepared the project for final migration. This tool will accommodate the important changes in project functionality that are incorporated in v7.0 and v7.20.

It is important that you prepare your existing projects for a successful upgrade using this tool.

Some of the features introduced in v7.20 of CitectSCADA require changes in the project data from version 6.x

⚠ WARNING

UPGRADE ALTERS COMMUNICATIONS CONFIGURATIONS

After upgrading, confirm and adjust the configuration of all I/O devices in your project.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

See Also

Memory devices
Alarm devices
Included projects
Using the Migration Tool

Memory Devices

In previous versions of CitectSCADA an I/O Device could be defined as a memory device by setting the port value to "Memory". This was generally done for one of the following purposes:

- To provide for future devices that were not currently connected to the system, but their points needed to be configured at this stage of project.
- For virtual devices where there was no corresponding physical I/O Device and you needed data storage with the entire functionality normally associated with I/O variables such as alarms.
- To act as a variable which was local to the process being used in place of Cicode global variables.

You can still use I/O Devices for future or virtual devices in version 7.0, but manually set the Port parameter to an unused value other than Memory, and set the Memory property of the device to True to indicate that it is an offline in-memory device before running the Migration Tool.

You need to review your project to identify which memory I/O Devices are local variable holders and which ones need to be changed to non-memory so that the Migration tool does not convert their variables.

The Migration Tool will set any I/O Device's port which is identified as a Memory device to the new Local Variable, and the original device record will be deleted.

See Also

Configure I/O Devices
Alarm Devices

Converting Memory Variables

Alarm Devices

In previous versions of CitectSCADA Alarm devices were defined as devices with their Protocol property set to "Alarm". In version 7.0 the function of configuring such a device is now replaced by setting the Publish Alarm Properties property to True on the Alarm Server.

Alarm devices with their Protocol property set to "Alarm" will be deleted from I/O Devices table by the Migration Tool.

See Also

Alarm Server Definitions

The Migration tool can delete memory and alarm device records. If you want to delete the devices at a later time, deselect the "Remove obsolete Memory and Alarm Devices" option.

> **Note:** Alarm devices with their Protocol property set to "Alarm" are no longer used and will be removed by the Migration Tool. All Alarm Servers will now publish Alarm Properties.

See Also

Converting Memory Variables

Converting Memory Variables

A memory variable is a variable with its I/O Device Port property set to either "Memory" or "MEM_PLC".

If there are multiple I/O Devices with the same name, possibly on different I/O Servers, then the device would not be considered as a memory device regardless of its port value. In other words the Migration tool will not process the variables for memory devices with duplicate names.

See Also

Inserting new local variables
Deleting Variable Tags

Inserting new local variables

When the Migration Tool runs, a local variable record will be inserted for each identified memory variable, and the variable data will be copied into the new local variable.

Local variables have fewer fields than variables; the following table shows the mapping from variable to local variable when copying their data.

Variable Tag parameter or constant value	Local variable parameter
Variable Tag name	Name
Data Type	Date Type
(Empty)	Array Size
Eng. Zero Scale	Zero Scale
Eng. Full Scale	Full Scale
Comment	Comment

With the exception of the Array Size, which has been introduced in version 7.0 exclusively for local variables, every field receives it's value from the same or similar field.

See Also

Deleting Variable Tags

Deleting Variable Tags

Once the Migration Tool has created the local variable records it will insert those variable tag records that have been converted in the previous step, and delete the original variable tag.

If an error is detected during the insertion of the local variables, the deletion of the variable tags will not be performed. If this occurs it is possible to have two records with same name and data, one in the local variable (the newly inserted record) and one in the variable tags (the original record that has not been deleted). You need to delete either of the variables manually, or restore the backed up project after removing the cause of the error then run the Migration Tool again.

See Also

Default Scale
Deleting Obsolete I/O Devices

Default Scale

The Scale properties in both variable tags and local variables are optional. If a Scale value is not specified the default value is indicated by a parameter in the Citect.ini file. The parameter name is "DefaultSliderScale" under the [General] section in the Citect.ini file. The default values for Scale is 0-32000, unless the default slider scale is true in which case the default value depends on the type for example Integer, String etcetera.

The Migration tool will read this parameter and if it is not set, or set to false, then it will explicitly set any empty Scale property to a value in to the range of 0 to 32000. This will be done even if either of the Zero Scale or Full Scale parameters has a value, in which case the empty Scale parameter will receive the default value.

If the DefaultSliderScale in the Citect.ini file set to True, the Scale parameters will not be populated with a default value if they are empty, rather they will be interpreted at run-time.

Deleting Obsolete I/O Devices

Deleting obsolete I/O Devices is an optional step in the Migration Tool and will be performed after the memory variables are converted. If the delete option is chosen, then obsolete Memory devices and Alarm devices will be deleted as the final step of the Migration Tool operation.

See Also

Included Projects

Included Projects

Each project may contain multiple included projects. Additionally any included project may contain its own included project so creating a cascading project.

The Migration Tool needs to process the original project and included projects in a single step. The reason for this is that variables can be defined in one project that refer to I/O Devices defined in another included project.

The Migration Tool performs this procedure sequentially on the "master" project then each included project.

In the case where two master projects share the same project as an included project, it is important that you do not select the "Remove obsolete Memory and Alarm devices" check box when you process a project that contains shared included projects. This is because the removal is performed at the conclusion of the migration process on each master and included projects sequentially. This could cause the deletion of an I/O Device in the first master project which is referenced by a tag in a shared included project which is processed in a later step.

If two separate "master" projects contain the same included project, run the Migration Tool on each "master" project without selecting to delete obsolete devices.

⚠ WARNING

UPGRADE ALTERS COMMUNICATIONS CONFIGURATIONS

After upgrading, confirm and adjust the configuration of all I/O devices in your project.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

To remove obsolete devices it is recommended that once the Migration Tool has completed successfully (without the check box being selected), run it a second time with the check box selected. This will safely remove the devices since every tag conversion were completed in the first pass of the Migration Tool.

Creation of roles for existing users

When upgrading an existing project using the migration tool, a new role will be created (if needed) for every existing user. The new role will have the same security settings that were defined for that user and be given a generic name such as Role_1, Role_2 etc. During the upgrade process, if a role exists with the same security settings as the user, then the existing role will be assigned to the user being upgraded. For example; If Role_1 exists and matches the security settings of the upgraded user then that user will be assigned Role_1 also.

If you do not want to migrate users from an existing project deselect the option "Create Roles from User security information" from the migration tool dialog before running it.

See Also

Incremental compilation

Using the Migration Tool

Note: Before you use the Migration Tool is strongly recommended that you familiarize yourself with the process that it performs, and the preparatory steps that you need to carry out with your existing projects as described under Migration Tool.

To run the Migration Tool:

1. Backup the projects that you intend to migrate.
2. From the Citect Explorer or Citect Editor menu bar select Tools | Migration Tool to display the Migration Tool dialog.
3. Either accept the project displayed in the edit box, or browse for the project that you wish to upgrade. To migrate this project as well as any included projects within it, select the option 'Migrate Included Projects'.

4. Select the Remove obsolete Memory and Alarm devices check box if you wish to delete these devices after successful migration.

   > **Note:** Do not select this check box if the project contains any included projects which are shared with more than one master project when you run the tool for the first time on such projects. Run the tool a second time using this option if the migration is successful after it is run the first time if you want to delete the devices.

5. Leave as selected the option 'Create Roles from User security information' if you wish to migrate the users database from an existing project
6. Check the option 'Copy XP_Style menu into Tab_Style Menu' to convert legacy menu entries to the format necessary for the new menu configuration system. By default this option is unchecked, to avoid potential compile errors that may occur after migration if the legacy menu.dbf contains functions which have been removed.
7. Check 'Migrate Included Projects' to migrate the included projects that are in the project you previously selected.

   > **Note:** If 'Copy XP Syle menu into Tab_Style Menu' and 'Migrate Included Projects' are both selected when the migration tool runs the following message will be displayed "Copying menus of included projects may lead to conflicts. Any conflicts will need to be manually corrected". To avoid this from occurring it is recommended you run the migration tool twice. In the first instance just select the option 'Copy XP_Style menu into Tab_Style Menu', and in the second instance just select the option 'Migrate Included Projects'.

8. Click Migrate to begin the migration process, or click Close to exit without performing the migration.
9. The migration process will begin and display a progress dialog indicating the stage of the conversion and the name of the project being migrated. If you wish to cancel the migration at this point click the Abort button.

   > **Note:** Aborting a migration will stop the migration process, and any changes already completed will not be rolled back. You will have to restore your project from the backup created in the first step.

10. When the migration process is concluded a confirmation dialog box will display indicating the number of variables converted and the number of I/O Devices deleted (if device deletion was selected at the start of migration)
11. Click the Close button to close the dialog.

Define Clusters

Even if you do not intend to use clusters in your project, you need to define at least one. Tags and servers will default to run in the defined cluster.

1. In the Project Editor, select **Servers** | **Clusters**. The Cluster dialog box displays:
2. In the **ClusterName** field, enter the name of the cluster. The name needs to be unique to the project and not contain spaces.
3. In the **Comment** field, enter any useful comment. This property is optional and is not used at runtime.
4. Click **Add**.

See Also

Configure Network Addresses

Configure Network Addresses

In the Project Editor, configure the network address of each machine to be used as a server in the project.

1. In the Project Editor, select **Servers** | **Network Addresses**. The Network Addresses dialog box displays:
2. In the **Name** field, enter a name for the network address being configured. The name needs to be unique to the project and not contain spaces.
3. In the **Address** field,enter the IP address or computer name of the machine being configured.
4. In the **Comment** field, enter any useful comment. This property is optional and is not used at runtime.
5. Click **Add**.

See Also

Configure Servers

Configure Servers

All Primary and Standby Alarms, Reports, Trends, and I/O Servers are now defined using the Project Editor. This involves specifying a network address and a cluster for each server.

Default Ports

Each server has a unique default port assigned to it. This default port may only be used with that type of server. Attempting to use a default port on another type of server will result in a compilation error of:

"Invalid port number (2073-2082,20222,21) are reserved"

The following table lists the default port numbers and their associated server type.

Default Port	Legacy Port	Server Type	Server Role
21	N/A	FTP Server IDC	Page downloads for IDC Internet Display Server/Client communications
2073	N/A	CTAPI	CTAPI Communications
2074	N/A	Client	Cicode Debugging
2084	2075	Reports Server	Reports Server communications
2080	2076	Alarm Server	Alarm Server communications
2085	2077	Trends Server	Trends Server communications
2078	N/A	I/O Server	Legacy I/O Communications
2080	2076	Alarm Server	Alarm Properties Connector
2082	2078 (Client server communication only)	I/O Server	Publish Subscribe I/O Server Communications
20222	N/A	ODBC	ODBC Server

For details on configuring each type of server refer to:

- Alarm Server Definitions
- Reports Server Definitions
- Trends Server Definitions
- I/O Server Definitions

Configure Tags to Use Clustering

If you have defined multiple clusters, you can configure alarms, reports, trends, SPC tags, and accumulators to run on particular clusters. Using the Project Editor, you can now specify the appropriate cluster for each tag.

- Configure Alarm Tags to use Clustering
- Configure Report Tags to use Clustering
- Configure Trend Tags to use Clustering
- Configure SPC Tags to use Clustering
- Configure Accumulators to use Clustering

Configure Alarm Tags to use Clustering

In the Project Editor, select **Alarms**, and then the type of Alarm you are configuring. The dialog box for the chosen alarm type displays:

For each alarm, in the **Cluster Name** field, select the name of the cluster that will run the alarm. If there is only one cluster defined in the project, you can leave this field blank. The alarms will default to the defined cluster.

If the project has multiple clusters, and you do not select a cluster name in this dialog, then CitectSCADA considers the alarm to run on defined clusters.

Using the **F2** key in the Alarm Server Definitions dialog, open the extended properties page and set the Publish Alarm Properties to "True" if you want the alarm properties to be published and be viewed as normal variable tags, and have the Alarm Server listen as if it were an I/O connector.

Click **Replace** to save the changes.

See Also

Configure Report Tags to use Clustering

Configure Report Tags to use Clustering

In the Project Editor, select **System | Reports.** The Reports dialog box will display:

For each report, in the **Cluster Name** field, select the cluster that will run the report. If there is only one cluster defined in the project, you can leave this field blank. The reports will default to the defined cluster.

If the project has multiple clusters, and you do not select a cluster in this dialog, then CitectSCADA considers the report to run on defined clusters.

Click **Replace** to save the changes.

See Also

Configure Trend Tags to use Clustering

Configure Trend Tags to use Clustering

In the Project Editor, select **Tags | Trend Tags.** The Trend Tags dialog box will display:

For each trend, in the **Cluster Name** field, select the cluster that will run the trend. If there is only one cluster defined in the project, you can leave this field blank. The trends will default to the defined cluster.

If the project has multiple clusters, and you do not select a cluster in this dialog, Citect-SCADA considers the trend to run on defined clusters.

Click **Replace** to save the changes.

See Also

Configure SPC Tags to use Clustering

Configure SPC Tags to use Clustering

In the Project Editor, select **Tags | SPC Tags.** The SPC Tags dialog box will display:

For each SPC tag, in the **Cluster Name** field, select the cluster that will run the SPC tag. If there is only one cluster defined in the project, you can leave this field blank. The SPC tags will default to the defined cluster.

If the project has multiple clusters, and you do not select a cluster in this dialog, Citect-SCADA considers the tag to run on defined clusters.

Click **Replace** to save the changes.

See Also

Configure Accumulators to use Clustering

Configure Accumulators to use Clustering

In the Project Editor, select **System | Accumulators.** The Accumulators dialog box will display:

For each accumulator, in the **Cluster Name** field, select the cluster that will run the accumulator. If the project has only one cluster defined, you can leave this field blank. The accumulator will default to the defined cluster.

If the project has multiple clusters, and you do not select a cluster in this dialog, Citect-SCADA considers the accumulator to run on defined clusters.

Click **Replace** to save the changes.

See Also

Compile the Project

Configuring Multiple Monitor Support

Multiple monitors are now supported in the core product instead of within a set of templates. This means you can display an existing project across multiple monitors by adjusting a set of multi-monitor parameters.

While the standard built-in functions work seamlessly in a multi-monitor environment, you may need to check whether a custom configuration functions correctly in multiple windows simultaneously.

See Also

Working with Multiple Monitors
MultiMonitors Parameters

Compile the Project

Once you have configured your project, compile it and verify that there are no errors.

At this stage you may want to reconfigure some of your Cicode to support online changes. Particularly the AssInfo and TagInfo functions that will be deprecated in future versions of the software. In most cases they can be replaced with the functions AssInfoEx and TagInfoEx.

Run Computer Setup Wizard

Run the Computer Setup Wizard for each computer running the project. At each stage of the Wizard, configure the appropriate settings for that computer.

> **Note**:If updating projects created using previous versions of CitectSCADA, check that the Computer Role Setup Page has the correct process mode selected.

See Also

Running the Computer Setup Wizard

Troubleshooting

Carefully consider the following results when upgrading to CitectSCADAv7.20:

- Compiler Errors
- Upgrading a Project that uses Distributed Servers

Compiler Errors

Before you configure your project to run in version 7.0, compiling the project will generate a number of compiler errors. These may include messages concerning deprecated and deleted functions, as well as the detected error "No Clusters defined". This detected error will be resolved once you define a cluster in the project.

See Also

Troubleshooting

Upgrading a Project that uses Distributed Servers

If you have implemented clustering in Version 6.x using Distributed Servers, a Global Include Project, and Cluster Projects, configure your project to use clustering in version 7.0.

The existing structure of the Global Display Project and Cluster Projects can remain the same (that is, the Global Display Project includes each of the Cluster Projects).

The following points describe a recommended project structure for clustering in version 7.0:

- Include a separate communications project in the Global Display Project. In this project, it is recommended that you define only the Network Addresses, Clusters, and Servers.

> **Note:** Defining a separate communications project means that when the Global Display Project compiles, it has the communications information without needing to load the data from the Cluster Projects.

- In each Cluster Project, specify the appropriate cluster for alarms, trends, reports, SPC tags, and accumulators.
- You may need to modify the buttons and pages in the Global Display Project, particularly if they are using Cluster functions which have been modified or deprecated.

Chapter: 4 About CitectSCADA

CitectSCADA is a Supervisory Control and Data Acquisition (SCADA) solution that is used to manage and monitor processes in manufacturing, primary production, utilities delivery and facilities management.

The graphics, controls, configuration data and programming associated with a CitectSCADA installation is configured and implemented through projects. A project acts as a digital representation of your production facility that is deployed in tandem with your plant infrastructure, allowing the entire system to be monitored and controlled in real-time.

Dynamic-point count licensing

In CitectSCADA 6.10 or earlier, multiple tags that share the same I/O device address will be counted as a single I/O point. This is calculated at compilation time and the compiler will report the number I/O points used in a project and is referred to as the Static Point Count. Any other tags used in super genies, read/written to via CTAPI, ODBC, DDE, TagRead() and TagWrite() are counted at runtime and are referred to the Dynamic Point Count. Thus, the total I/O point count will be equal to Dynamic Count plus Static Count, which is calculated at runtime.

However, from CitectSCADA 7.0 the compiler does not generate any static point count any more. CitectSCADA counts I/O device addresses dynamically at runtime. This includes tags used by alarms, trends, reports, events, pages, in Super Genies, use of the TagRead() and TagWrite() Cicode functions, or read or written to using DDE, ODBC, or the CTAPI. A particular variable tag is only counted towards your point count the first time it is requested. That is, even though you may have configured a certain tag on a particular page in your project, unless you navigate to that page and request the data, the variable tag will not be counted towards your point count.

In addition to this, there have been a number of other changes that have been made to the licensing structure from CitectSCADA 7.0. These are listed below:

- I/O point count is now tag based not address based. For example, two tags that use the same PLC address will be counted twice. If two trend tags use the same variable tag, it will be counted once. The same applies to alarms.
- For the multi-process mode, each server component will accumulate its own point count. The server component point count is the count added up from each server

component -if two server components use the same tags, say alarm and trend, the tags will be counted twice when the point count gets totaled.

- For the multi-process mode, the client component will also accumulate its own point count including super genie and CTAPI tags.
- For the multi-process mode, the machine point count will be the point count on the client component or the point count added up from each server component, whichever is bigger. For example, if the total point count for each server component is 100, and the client component point count including CTAPI and super genies is 95, the kernel "General" window will show 100. If the client component point count reaches 120 later and the server component point count still remains 100, the kernel "General" window will show 120.
- Reading properties of a tag with TagGetProperty() will cause that tag to be included in the point count, even if the value is not read.
- Writing to local variables or disk IO variable tags via OPC etc will also increase the point count. For example, if you use an OPC client to write to a local variable, each local variable will be counted once, the first time it is used.

Configuring a CitectSCADA project

Initially you use CitectSCADA's configuration environment to identify and address the devices and datasources included in a project by tagging the variables associated with each.

You can then use templates to design graphics pages that reference these tags, creating an interface your staff can use to view and control the system.

With these graphics pages, you can:

- use animations to display the operating status and performance of a plant
- provide operators with centralized or local control of production equipment using keyboard commands and graphical tools
- develop a multi-layered security system that controls user access according to functional groups or geographical areas
- implement historical and millisecond trending of tag data in a graphical format.

A powerful scripting language is also included to enable customized, programmable functionality.

Deploying CitectSCADA

The project is then deployed across a client-server network architecture. The servers are used to manage communication with plant equipment and collate production data, while the clients provide the interface for operators and managers to assess and interact with the system.

This architecture allows the flexibility to adapt CitectSCADA to any production scenario, with support for scalability, server clustering, and system redundancy.

Running a project

When a project is eventually compiled and implemented in runtime, your production staff can visually monitor the system, initiate production processes and respond to alarm conditions.

Historical and trend data can also be collated and distributed to assess operational performance metrics such as production volume, efficiency, and maintenance requirements.

Chapter: 5 Tools

CitectSCADA's architecture can be divided into three distinct areas of functionality:

- Configuration
- Runtime
- Drivers

Configuration involves the tasks necessary to prepare and build a project, while runtime is the implementation of a project in a live production environment.

Drivers enable communication with devices via a number of communication protocols. The driver defines the specific project settings necessary for CitectSCADA to communicate with a particular device.

When considering the tools included with CitectSCADA, it is easiest to look at their roles in either configuration or runtime.

See Also

Configuration Tools
Runtime Tools
Drivers

Configuration Tools

The following tools enable you to configure a project and its components, and set up computers to use CitectSCADA:

	CitectSCADA Explorer	The application used to create and manage your projects. It displays a list of projects, and provides direct access to the components of each. You can use Explorer to rename, back up, restore or delete a project. See Administering Projects
	CitectSCADA Project Editor	The application used to create and manage the configuration information for your project, including tags, alarms, system components, and communications components See Components of a project

	CitectSCADA Graphics Builder	The application used to design, create, and edit the graphics components of a project, including templates, graphics objects, symbols, genies, and Super Genies See Defining and Drawing Graphics Pages
	Computer Setup Editor	A utility for editing configuration files and generating reports to compare and analyze files See Computer Setup Editor online help
	Computer Setup Wizard	A wizard that allows you to customize a computer's setup and define its role and function See Running the Computer Setup Wizard

Runtime Tools

The following tools enable you to run, monitor, and control projects during runtime:.

	CitectSCADA Web Client	Displays a live CitectSCADA project within a web browser See CitectSCADA Web Client
	Internet Display Client	A computer used to run a CitectSCADA project over the Internet from a remote location. See Running Your System Over the Internet
	Process Analyst	An Active X control that allows you to compare and analyze historical and real-time trend and alarm data during runtime. See "Configuring the Process Analyst" in the Process Analyst User Guide

	CitectSCADA Runtime Manager	An application used to manage and control the CPU configuration of the project, and the running state of each component See "Launching Runtime Manager" in the Runtime Manager online help

Drivers

CitectSCADA can communicate with an array of I/O Devices, including PLCs (Programmable Logic Controllers), loop controllers, and distributed control systems (DCS).

The I/O Devices may be local (directly connected to an I/O Server) or remote (connected to CitectSCADA via an intermediate communications means like a phone line).

Drivers enable communication with devices via a number of communication protocols (including Ethernet, TCP/IP, and Serial). The driver defines the specific project settings necessary for CitectSCADA to communicate with a particular device. This includes information about:

- Boards
- Ports
- Devices
- Tag addressing

For detailed information on drivers and how to use them in your system, see Communicating with I/O Devices

Chapter: 6 Components of a project

The components you can incorporate in a project are logically divided across the following categories:

- Graphics components
- Tags
- Alarms
- System components
- Communications components
- I/OServer components
- Cicode / CitectVBA

These categories are represented in Citect Explorer through the set of folders associated with each project.

As you build a project, the components you include are listed in the relevant project folder. Selecting an item from one of these folders launches the selected component in the tool necessary to edit its properties.

See Also

Graphics components
Tags
Alarms
System components
Communications components
I/OServer components
Cicode / CitectVBA

Graphics components

The graphical components of a project represent the content used to create the screens presented on clients. They include:

	Pages	The basis for the screen layouts See Defining and Drawing Graphics Pages
	Templates	A collection of page layouts used to standardize display screens

		See Using Page Templates
	Symbols	Graphics objects stored in a library for reuse See Using symbols
	Genies	Objects that group multiple graphical and functional elements for easy duplication See Understanding Genies
	Super Genies	Genies that can pass device-specific information at runtime. See Using Super Genies

As you create project pages in Graphics Builder, the included components are added to the relevant subdirectory in the current project's Graphics folder.

Tags

Tags are used to identify the end points in the infrastructure you are using CitectSCADA to monitor and control. The name you give to a tag becomes a label for a register address, allowing it to be intuitively applied across graphics pages and in alarm notifications.

Three tag types are included in a project's Tags folder in Citect Explorer:

	Variable tags	used to label register addresses See Tagging Process Variables
	Trend tags	used to label tags for data trending See Trending Data
	SPC tags	used to label tags according to Statistical Process Control principles See SPC Tags

Selecting one of these tag types in Citect Explorer calls up the associated configuration dialog in Project Editor.

Alarms

Alarms are used to identify conditions in a system that require attention. CitectSCADA supports seven different alarm types:

⚠	Digital See Digital Alarms
⚠	Analog See Analog Alarms
⚠	Time-stamped See Time-stamped Alarms
⚠	Advanced See Advanced Alarms
⚠	Multi-Digital See Multi-digital Alarms
⚠	Time-stamped Digital See Time-stamped Digital Alarms
⚠	Time-stamped Analog See Time-stamped Analog Alarms

You can also use alarm categories within your project to help identify and manage alarms.

System components

The system components of a project allow you to customize, manage, and track your run-time system. They include:

⌨	Keyboard key	A meaningful name assigned to a keyboard key See Keyboard Keys

	Keyboard commands	Key sequences with associated instructions See System Keyboard Commands
	Reports	Customized presentation of runtime data and special conditions See Reporting Information
	Events	Commands that execute in response to specific runtime triggers, such as a Cicode expression or variable tag. When the event trigger is true, the command will execute See Configuring Events
	Accumulators	Variable tags tracking continuous runtime data. Data can be monitored and displayed by animating or trending the variable tags See Using Accumulators
	Devices	Components that can transfer high-level data to other components such as RTF files, ASCII files, and printers See Configuring Devices
	Users	User profiles to restrict and grant access to the runtime system See Adding User Records
	Groups	Groups of system areas used to simplify the management of user profiles See Defining Areas
	Label	System-wide substitutions for commonly used commands and expressions See Using Labels
	Fonts	Fonts for displaying alarms and objects See Using System Fonts
	Parameters	Built-in operating settings for fine tuning the runtime system See "Citect.ini File Parameters" in the CitectSCADA Technical Reference
	Included projects	Predefined projects with database records automatically included in user projects

See Including projects

Communications components

The communications components of a project are the configured representation of the communications hardware in your system. They include:

	Board	Hardware enabling various types of communication with I/O Devices See Boards Properties
	Port	The physical connection between the board and the I/O Device See Ports Properties
	Modem	Hardware used to connect CitectSCADA to a dial-up remote I/O Device. See To set up a modem in CitectSCADA:
	I/O Devices	An item of equipment that communicates with plant-floor control or monitoring equipment See I/O Devices Properties
	I/O Device address	The unique address of an I/O Device CitectSCADA is communicating with

I/O Server components

The server components of a project are the configured representation of the server computers in your system. They include:

	Cluster	Logical groups of servers running across several physical machines See Implementing Clustering
	Network addresses	The IP addresses or machine names of the primary and standby servers See Network Address Definitions

	Alarms servers	Servers that monitor alarms and display them on the appropriate client(s)
	Reports servers	Servers that control the processing of reports
	Trends Servers	Servers that control the accumulation and logging of trend information
	I/O Servers	Dedicated communications servers that exchange data between I/O Devices and clients

Cicode / CitectVBA

CitectSCADA offers two programming languages with which you can control and manipulate CitectSCADA components:

	Cicode	A structured programming language designed for use in CitectSCADA to monitor and control equipment. See Introducing Cicode.
	CitectVBA	A Visual Basic for Applications (VBA) and VBScript-compatible Basic scripting language. See Introducing CitectVBA.

Chapter: 7 Typical system scenarios

The scenarios described in this chapter demonstrate how CitectSCADA can be used to support typical processes found in primary production, utilities delivery, and manufacturing.

In reality, a project will incorporate a combination of the scenarios described here, with a high degree of customization and scalability. However, these examples have been simplified to demonstrate how CitectSCADA can be configured and deployed to meet the specific requirements of a production system.

Standalone system

Every component of a system runs on a single computer. See Standalone system.

Distributed I/O system

CitectSCADA is used to monitor and manage distributed devices that are each connected to remote I/O Servers. See Distributed I/O system.

Redundant server system

One or more of the servers associated with a system are duplicated and defined as primary and standby units, allowing the system to keep running in the event one of the servers becomes inoperative. See Redundant server system.

Client-server system

The servers and clients associated with a system are independently distributed across a number of computers on a network, offering greater accessibility and performance benefits. See Client-Server system

Redundant and distributed control system

Remote or geographically separate sections of a production system have fully operational sub-systems in place that are monitored and controlled locally. If such a sub-system becomes partially or wholly inoperative in a manner preventing local control, this arrangement allows remote Control Clients to take control of the affected sub-system. See Redundant and distributed control system.

Cluster controlled system

A production system is organized into discrete areas being monitored by operators within each area. However, there is also a level of control that supervises every area of the system. See Clustered control system.

Load sharing system

The system splits the load of an otherwise stressed system across multiple machines, better utilizing the available infrastructure. See Load sharing system.

See Also

Cluster Connections Configuration

Standalone system

A standalone installation of CitectSCADA runs every server and client component of a system on a single computer. These include:

- I/O Server
- Alarm Server
- Trends Server
- Reports Server
- Control Client

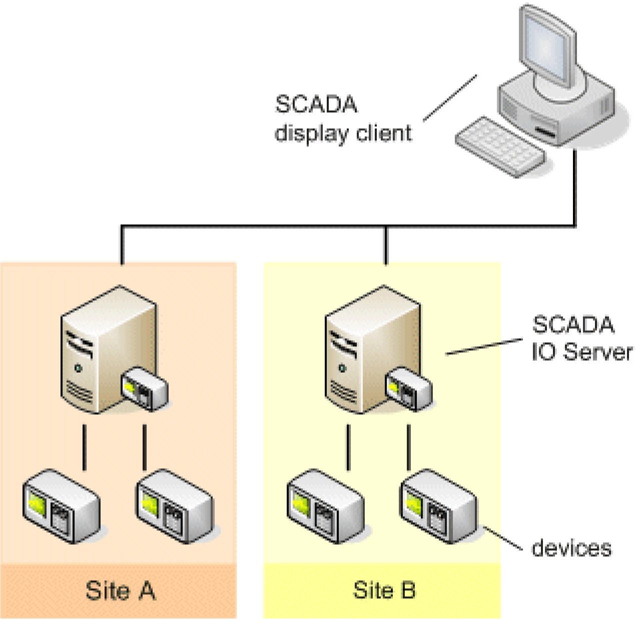
This allows CitectSCADA to be run as a small, self contained system.

> **Note:** You can run the server and client components of a standalone system as a single-process or multi-process system. It is recommended that a single- process setup only be used as a short term solution for your control system, or to run demonstrations and test projects. Adding redundancy to your system will make it more reliable and more efficient.

Distributed I/O system

This scenario demonstrates a method of connecting CitectSCADA to a number of devices that are distributed across several sites over a wide geographical area.

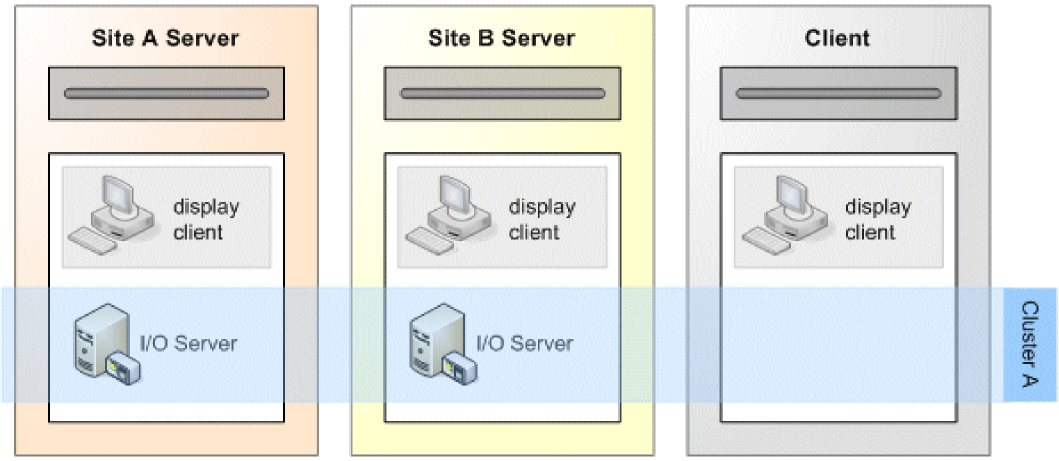
Instead of attempting to connect devices directly via a remote connection, an I/O Server is placed at each site, enabling communication to be managed within the system.



This model is also useful in plants that contain devices with a serial port or limited communications capabilities. By placing I/O Servers on the factory floor to interface with these devices, you can optimize communications on slow or low-bandwidth networks and improve overall performance.

Despite the geographical distribution of I/O Servers across many sites, this type of system can be configured as a single cluster system, as a cluster is able to support many I/O Servers.

The diagram below demonstrates how to approach the deployment of this type of system across the server machines using a single cluster.

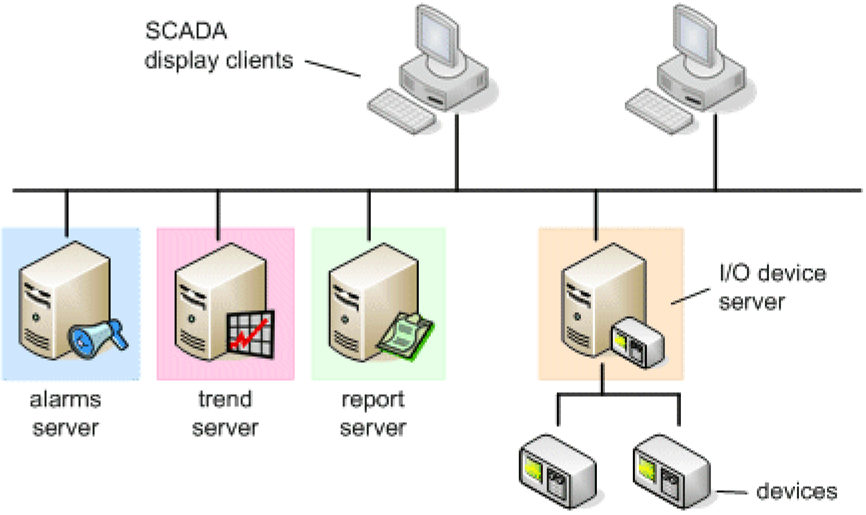


A second cluster will only become necessary if your project requirements call for more than one redundant pair of alarms, trends or reports servers.

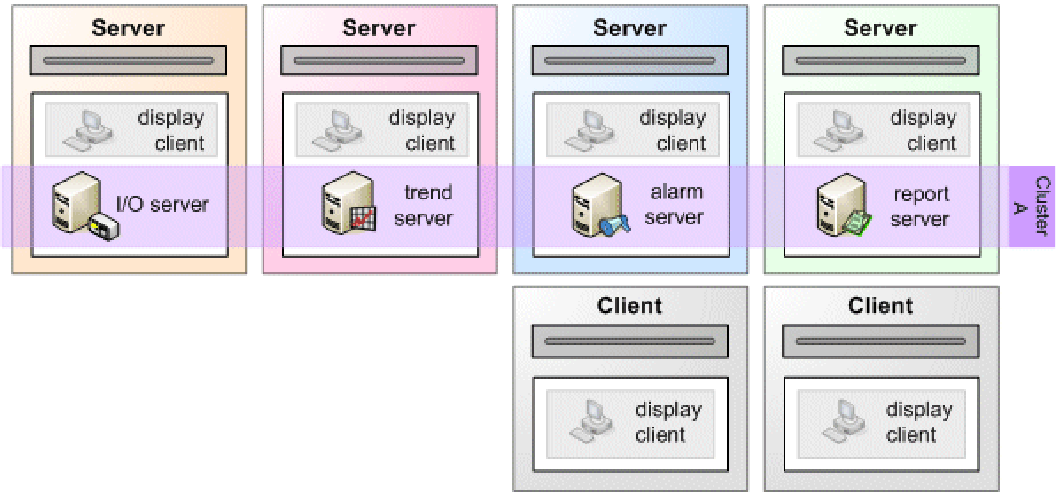
Client-Server system

CitectSCADA's client-server architecture allows the components of a system to be distributed across a number of computers on a LAN, creating a system that offers geographical flexibility and performance benefits.

Each component is simply identified within the project by an address, allowing the location and hardware requirements for each to be considered independently.



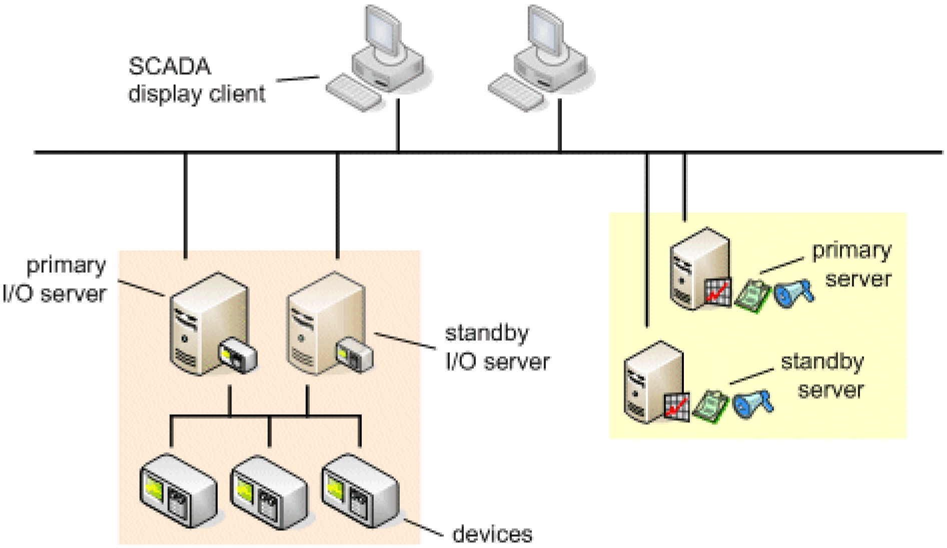
The diagram below demonstrates how this example can still be configured within a single cluster.



Each server also acts as a Control Client across the system architecture.

Redundant server system

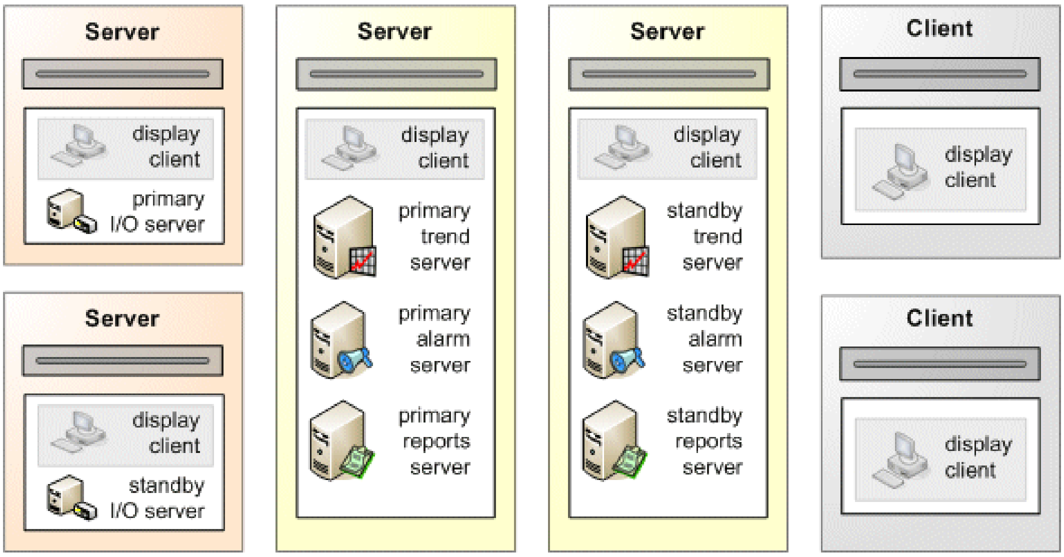
The ability to define primary and standby servers within a project allows hardware redundancy to be built into your system infrastructure. This helps prevent situations where an error on one server results in the overall system becoming inoperative. Systems of this type are especially beneficial when service continuity and/or secure data collection are important.



In the case of I/O Server redundancy, a standby server is maintained in parallel to the primary server. If a hardware error is detected, the standby server can assume control of device communication with minimal interruption to the system. You can also use redundant I/O Servers to split the processing load.

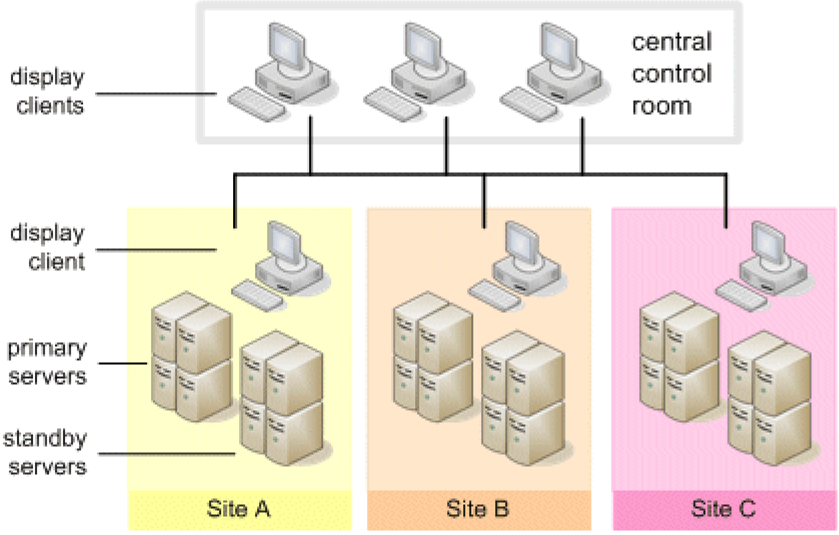
Alarm, report and Trends Servers can also be implemented as redundant servers. This improves the likelihood that clients will continue to have access to data from a standby server in the case a primary server becomes inoperative. CitectSCADA maintains identical data on both servers.

In the diagram below, the primary and standby I/O Servers are deployed independently, while the alarms, trends and reports servers are run as separate processes on common primary and standby computers. In this case, the entire system can be configured as a single cluster.



Clustered control system

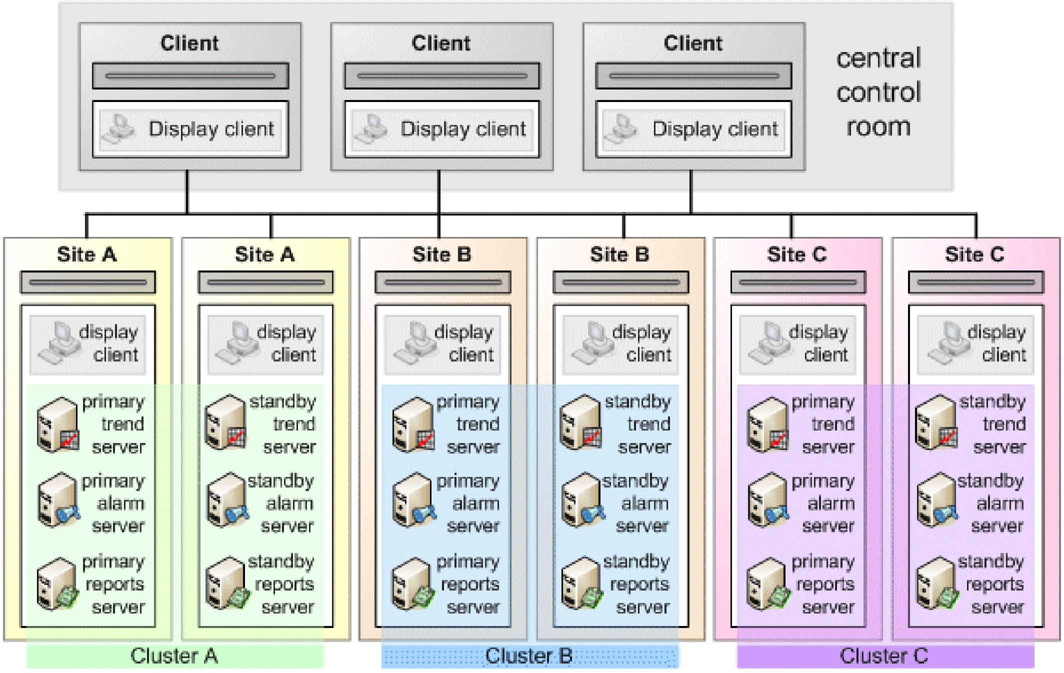
In this scenario, the system is organized into discrete sites being controlled by local operators, and supported by local redundant servers. At the same time, there is a level of management that requires sites across the system to be monitored simultaneously from a central control room.



Each site is represented in the project with a separate cluster, grouping its primary and standby servers. Clients at each site are only interested in the local cluster, whereas clients at the central control room are able to view every cluster.

The deployment of a control room scenario is fairly straightforward, as each site can be addressed independently within its own cluster. The control room itself only needs Control Clients.

The deployment of servers could be mapped out as follows:



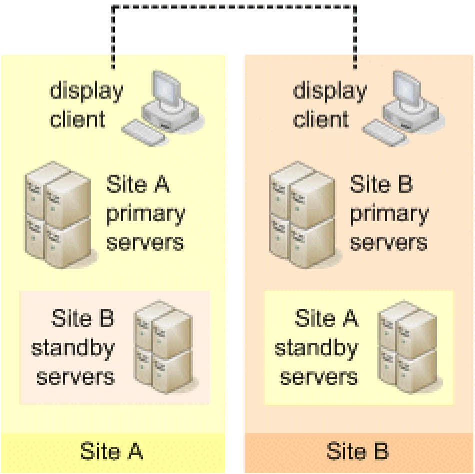
CitectSCADA's support for dynamic clustering means each site can be monitored and controlled from the central control room if necessary. For example, if an operator at a particular site only works during regular business hours, then the monitoring can be switched to the central control room after hours.

Redundant and distributed control system

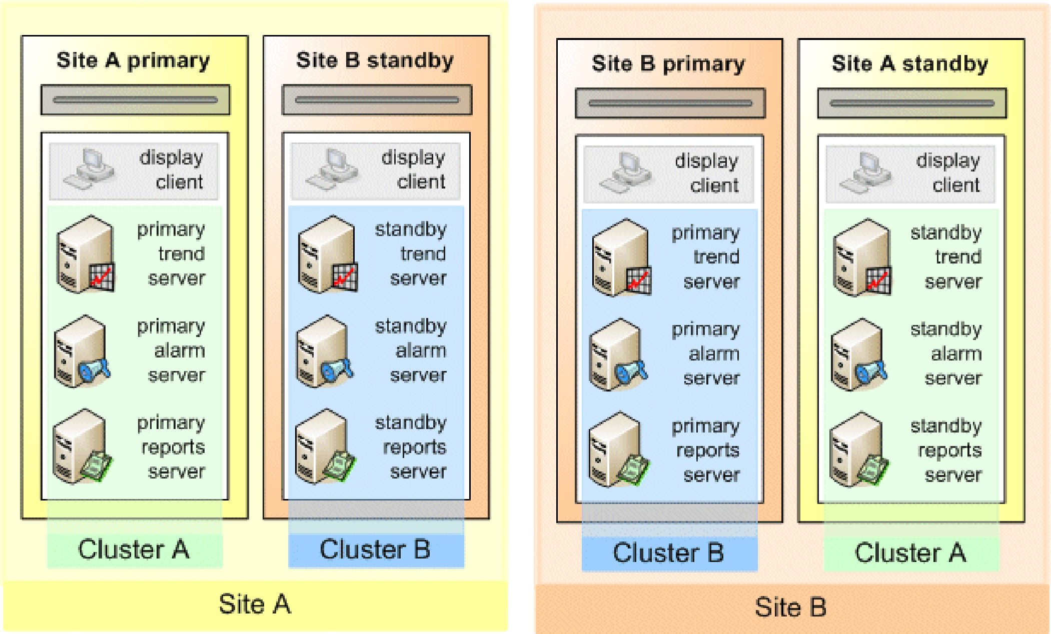
In this scenario, a project represents a number of locally operated sites each containing its own set of servers and clients. For example, a number of pumping stations across a water distribution system, or multiple production lines in a manufacturing facility. However, there is a requirement for monitoring to continue in the event the system at one of the sites becomes inoperative.

This is achieved by distributing the primary and standby servers across the different sites, or by placing the standby servers in a central location.

Clustering is used to define the role of the different servers at each site, which can be viewed in a common project running on every client. This means Site A can be monitored from Site B, and vice versa, if a system becomes inoperative at one of the sites.



The example above would require the creation of two clusters, so that the project can include two sets of primary and standby servers. The clusters represent the redundant pairs of servers, and would be deployed across the two sites as follows:



The clusters offer the benefit of keeping a logical structure to the project during configuration, despite the unusual distribution of redundant server pairs.

Load sharing system

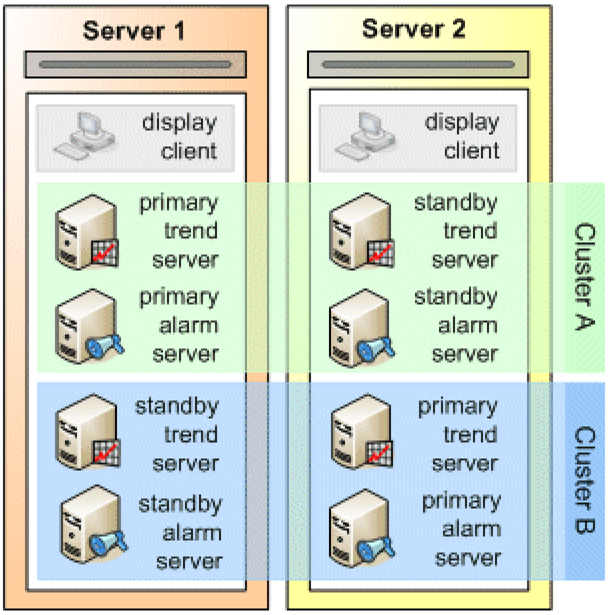
Load sharing of system components across different computers and CPUs means the work load of a potentially stressed system can be split across multiple machines, better utilizing the available infrastructure.

For example, managing alarms can draw heavily on a CPU's performance, while trending data can use a lot of disk space. By assigning your Trends and Alarm Servers to different processes on a shared computer, an Alarm Server can be used as a standby Trends Server, making practical use of idle disk space.

This approach can used to improve network performance, data access times, and general system stability.

If you introduce clustering, you have the flexibility to run multiple servers of the same type on a single computer. As long as a client has access to every cluster configured in a project, it doesn't matter if a set of servers is distributed across a number of clusters.

In the diagram below, two servers have been configured to act as standby units for each other, supporting two sets of redundant Trends and Alarm Servers.



Both machines have an even balance of Trends and Alarm Servers, making effective use of the CPU and disk space. By distributing the servers across two clusters, the servers are also able to act as redundant units to each other. This has reduced the necessary number of computers from a maximum of eight down to just two.

Using CitectSCADA

This section contains information on using CitectSCADA and describes the following:

Chapter: 8 Planning a Project

This chapter describes the planning phase of a CitectSCADA system.

A planned approach to the design and configuration of your system allows you to make optimal use of the product's features and performance capabilities, and helps you meet the requirements of your production facility. It also helps avoid unnecessary rework during the configuration of your project.

It is important to consider the following when planning a system:

1. The Physical Layout of a Plant
2. Operational Requirements
3. Project Design
4. Building Your Project
5. Setting up Your Computers

The Physical Layout of a Plant

You need to consider the physical layout of the plant where you would like to implement CitectSCADA.

This information will help determine the architecture of your project, and many of its operational requirements. It also allows you to assess the available equipment, to determine if any additions or modifications are necessary.

Consider the following items when examining the physical layout of a plant:

Geography

The physical layout of your facility, including whether the plant is spread across multiple geographical locations or specific areas of functionality, such as a number of production lines running in parallel.

Machinery

The equipment (machines, physical connections, and devices) in your plant that will be monitored and controlled by your system.

Existing computer hardware

The computers that currently exist within your facility to support CitectSCADA's client-server architecture. The specifications and limitations of the existing equipment will have to be considered to determine if the operational requirements of your system can be supported.

Network configuration

The current configuration of the network that will support your system and its communication with plant equipment. This will include the protocols used, the performance capabilities of the system, and security.

See Also

Operational Requirements

Operational Requirements

By developing a set of operational requirements for your project, you'll define a comprehensive list of needs and objectives that your system needs to support to effectively monitor and control production.

The things consider to determine the operational requirements include:

Architecture
Security
Reliability
Monitoring
Data collection

Architecture

Production Processes

The operating processes within your production facility need to be considered to determine how they can be logically represented and supported within your project. If the processes are dependent on each other, you also need to consider how the interaction between them will be managed, particularly if unexpected circumstances occur.

Security

When planning a project you need to consider who will be using the system, and which parts of a project they will need to have access to. To effectively do this you will need to understand how roles, privileges and areas work together to enable you to develop a secure CitectSCADA system.

See Also

Users and Areas

Reliability

The nature of your production processes will determine the importance of system reliability. Consider issues such as:

- The need for uninterrupted operation
- the impact and cost of down-time
- the need to collect and protect system data
- the severity of alarm conditions.

This will help determine if your project needs to include redundancy, the type of redundancy necessary, and the most appropriate way to implement it.

For more information, refer to Redundancy.

Monitoring

System monitoring is a key function of a SCADA system and needs to be considered in terms of the necessary interaction between personnel and production processes.

You need to consider if the delivery of data is time-critical. For example, alarm conditions need to be presented in real time, trend data may be delivered with a slight delay, while maintenance data can be accumulated and viewed as necessary.

The system may also need to be monitored at different levels, from machinery operators to control room personnel managing plant-wide processes. For each level of monitoring, consider the data that needs to be presented, and the specific performance and diagnostic conditions that need to be flagged.

Data collection

Consider the kind of data you need to collect from the production process, and how it will be used. Depending on your requirements, CitectSCADA can collect:

- Production data
- Purchasing requirements
- Batch processing statistics
- Equipment status and performance data
- Maintenance scheduling information
- Process performance data
- Dynamic visual analysis data

Carefully consider an assessment of the likely amount of accumulated data, as it will significantly impact on your computer hardware and network performance requirements.

For more information, refer to Logging and Trending Data.

See Also
Project Design

Project Design

Once you have developed a clear set of operational requirements, you need to plan how to design your project to best meet these requirements. When designing your project, consider the following issues:

Naming Standards
Page Templates
Genies and Super Genies
Clustering
Included projects
Redundancy

Naming Standards

By adopting naming standards, you can configure project components with meaningful names that convey useful information, such as the location or type of component. The standard that you use depends on the type of information that will be useful to system operators. A naming standard helps promote consistency throughout the project, making it easier to quickly identify components, and reducing duplication and user training. Naming standards may be useful for devices, variable tags, reports, graphics objects, and pages.

Reserved names

Some names such as IO_Server, Report_Server, Alarm_Server, Trend_Server, and Client are reserved for use in the include project. Using one of these names for your server will result in a compilation error. It is better to use names in your project that are meaningful to your installation.

Page Templates

Page templates are predefined page layouts that you can use to build the display screens (graphics pages) for your project. Templates allow you to create new pages quickly, and allow your runtime system to have a consistent look and feel. They can incorporate standard navigational and support tools that are common to every page. CitectSCADA includes a number of standard templates, and you can also design new templates to suit the requirements of your system.

See Also

Using Page Templates

Genies and Super Genies

Genies are like object templates that you can place on graphics pages to help simplify the configuration of many similar devices. They group functional and graphical elements, and present device information using configurable string substitutions as placeholders for specific tags or expressions. The information is then presented during runtime.

Super Genies are genies that can be passed device-specific information at runtime. Super Genies are useful for dynamic controls, like a popup switch that can be used to control many devices.

See Also

Using Genies and Super Genies

Clustering

Clustering allows you to group independent sets of CitectSCADA's server components within a single project, allowing multiple systems to be monitored and controlled simultaneously.

The most appropriate configuration will depend on the **requirements** for the solution to be deployed and the **environment** in which it is being deployed.

Some typical clustering configurations include:

- Standalone system
- Distributed I/O system

- Client-Server system
- Redundant server system
- Clustered control system
- Redundant and distributed control system
- Load sharing system

CitectSCADA's implementation of clustering allows for the flexible deployment of graphics pages that can access data from different clusters dynamically. A page can be allocated a cluster context when it is called, and any elements on that page will be assigned the same cluster, unless they have a cluster explicitly specified. See About cluster context.

See Also

Typical system scenarios
Rules of Clustering

About cluster context

Many CitectSCADA items require a cluster to be specified in order for them to work correctly. This ClusterName can either be explicitly specified when the item is called or displayed, or an item can use the default cluster context of the calling process or page. If your system has only one cluster, then no cluster name needs to be specified. See Cluster context rules.

Cluster context, therefore, is the default cluster used for tag resolution and execution of expressions and Cicode functions. In the case of tags, a cluster is explicitly supplied by prefixing the tag name with the cluster name. For built-in Cicode functions ClusterName is usually an optional parameter.

Server processes (Alarm, Trend, Report) have their default cluster context set to their own cluster, so that, for example, Alarm definitions that contain variable tags without clusters explicitly supplied will attempt to resolve those tags to their own cluster. Cicode tasks started from server code will inherit the servers cluster unless one is explicitly given.

Graphics pages have no cluster context by default. A page's cluster can be supplied statically in the page appearance properties, can be inherited from a previous context, or can be applied dynamically using an optional parameter in the Cicode function that displayed the page. Any Cicode tasks started from a page will inherit the cluster context of that page.

See Also

Cluster context rules

Cluster context rules

1. Single cluster systems always use that cluster and do not require the use of cluster prefixes.
2. Variable Tags referenced in an Alarms, Reports or Trends Server context are implicitly resolved to that cluster unless explicitly stated otherwise using a cluster prefix such as *ClusterName.TagName*.
3. Cicode called in an Alarms, Reports or Trends Server context runs with the server's cluster context as default.
4. Server to Server connections (for example, SPC Alarms) are within the cluster by default.
5. Client pages can have a cluster context associated with them.
6. Client pages can be configured with a cluster context or inherit the cluster from the previous (or parent) page.
7. Client pages can have the configured cluster context overridden dynamically at runtime.
8. Client pages have no default cluster and do not inherit context in multi-cluster systems by default.
9. The TaskNew Cicode function inherits the caller's (either page or task) cluster context unless explicitly overridden.

See Also

About cluster context

Included projects

If you have a large production environment, you can simplify the configuration and management of your system by designing your project as a collection of smaller "included" projects.

Included projects can operate independently, however, they share resources and operate interdependently during runtime. This means you can create and test projects representing functional or physical sections of a plant, and then gradually bring them online. Ongoing maintenance can then be managed with a minimal impact on production.

For more information, refer to Including projects.

Redundancy

Redundancy can be implemented at different levels of your system, depending on the reliability requirements of your project. The following types of redundancy are available:

Device Redundancy

Multiple data paths to a device can be configured within CitectSCADA. Therefore, if the primary path becomes unavailable, data can still be monitored over the secondary path.

Server Redundancy

Primary and Standby Alarms, Reports, and Trends Servers can be configured so that if a primary server becomes unavailable to process a client's request, the request can be channelled to a standby server for processing.

LAN Redundancy

To avoid service interruptions when the primary network isn't operating, a redundant LAN can be implemented that will provide an alternative path to a server if necessary.

See Also

Building Redundancy Into Your System

Building Your Project

Once you have determined your project requirements and design, you can begin implementing the design in CitectSCADA.

The following topics will assist you to identify the project components and options that you need to implement.

Projects
Setting up I/O Device Communication
Graphics components
Alarms
Data Collection
Users and Areas
System Components

Projects

You will first need to create a new project, and familiarize yourself with tasks like storing, including, and archiving it.

Before running the project, the process of compiling it will alert you to any errors in the configuration.

See Also

Building Your CitectSCADA Project
Administering Projects
Compiling the Project

Setting up I/O Device Communication

CitectSCADA can be configured to communicate with I/O Devices from a number of different vendors. To establish communications with an I/O Device, you will need to perform the following steps:

- Install the relevant device driver
- Configure the hardware and software necessary by the device
- Set up a test project to test the communications channel
- Configure a variable tag for each data point on the I/O Device that you want to communicate with.
- Configure the device in the project, either manually or using the Communications Express Wizard.

See Also

Using the Communications Express Wizard
Communicating with I/O Devices
Tagging Process Variables

Graphics Components

Graphics components are the means through which operators view and interact with the runtime system. Graphics pages can be designed to provide operators at different system areas or levels with relevant monitoring and control options.

To create graphics components that meet your operational requirements, be familiar with how to create graphics pages, use page templates, and configure graphical objects like Genies and Super Genies.

See Also

Defining and Drawing Graphics Pages

Using Genies and Super Genies
Using Objects
Defining Common Object Properties.
Understanding Object Types

Alarms

The CitectSCADA alarm system monitors your production processes and alerts operators to unexpected events that may require attention.

There are two types of alarms that you may need to configure:

- **Hardware alarms** - alert you to inoperative or partially operative equipment
- **Configured alarms** - allowyou to specify relevant alarm conditions for your facility (for example, the value of a variable tag monitoring the level, temperature, or status of a specific piece of equipment). There are seven types of configured alarms, depending on the type of alarm condition you need to set up.

To help operators process alarms, you can create graphics pages that provide alarm information (such as the action an operator needs to perform to correct the situation).

See Also

Configuring and Processing Alarms
Configured alarms
Formatting an Alarm Display

Data Collection

Data collection in CitectSCADA incorporates two main aspects:

- **Trends -** The trends system allows you to collect and monitor plant data. Depending on your requirements, data can be collected on a periodic basis, or when a specific event occurs. The data can then be saved to disk for analysis or displayed on a graph or report. To use trends in your system, you will need to be familiar with how to configure trend tags and display trend data in a graph or report.
- **Reports** - Reports provide information on the status of your plant and processes. You can configure reports with the following information so that they meet your operational requirements:
  - **Period/Trigger**: Reports can be run on a request basis, periodically, or when a specific event occurs.
  - **Report format**: You can use a text editor to create a file that specifies how a report is displayed.
  - **Report output**: Reports can be output to a file, device, or displayed on a graphics page.

See Also

Logging and Trending Data
Reporting Information

Users and Areas

You can design security for your system which incorporates both of the following features:

- **Users** - User accounts allow you to restrict access to your runtime system. Every user needs to log in to the system with a user name and password to gain access. User accounts can be set up for individuals or for groups of users.
- **Areas** - Areas allow you to define geographical or functional boundaries in your system.You can then control both the access users have to different parts of the project, and the tasks they can perform.

See Also

Using Security

System Components

CitectSCADA includes the following system components, which provide further options for monitoring, control, and user interaction:

- **Commands and Controls** - Configurable keyboard commands and slider controls allow operators to interact with the runtime system.
- **Events** - Events (such as variable tags or expressions) can be configured that trigger a specific action, like a command.
- **Accumulators** - Accumulators track incremental runtime data. The data is stored as variable tags in an I/O Device, and updated regularly while the trigger is active.
- **Statistical Process Control** - SPC allows to you to track quality by collecting and interpreting process variables associated with a product.
- **Labels** - System wide substitutions can be configured for commonly executed commands and expressions.
- **Devices** - High-level CitectSCADA data (including reports and logs) can be transferred to other system elements such as printers, databases, or files.
- **Remote Access** - A project can be accessed remotely or wirelessly in the following ways:
  - **CitectSCADA Web Client** - The CitectSCADA Web Client allows you to view a live project within a Web browser.
  - **Internet Display Client** - An Internet Display Client can be used to run a runtime-only version of a project over the Internet from a remote location.

See Also

Defining Commands and Controls
Configuring Events
Using Accumulators
Understanding Statistical Process Control
Using Labels
Using Equipment
Using Devices
CitectSCADA Web Client
Running Your System Over the Internet
Exchanging Data with Other Applications

Setting up Your Computers

Once you have built your project, you need to configure each computer in your system. The configuration information is stored on each computer in a Citect.ini file. The information includes:

- The role the computer has in the Citect network
- The project being run
- The CPU Configuration
- The Citect Events enabled for each component
- The Cicode run for each component on startup
- The cluster configuration
- The security settings applied

The Computer Setup Wizard displays a series of pages where you can configure these settings. The selected options are written to the Citect.ini file. run the wizard on each computer as the final step before running your project.

See Also

Running the Computer Setup Wizard

Setting up CitectSCADA as an OPC data source

CitectSCADA OPC Server allows you to access data available in the CitectSCADA runtime environment (for example from PLCs and databases) through any OPC Client application (v1.0 or v2.0).

When CitectSCADA is used to monitor and control a plant, data from PLCs is collected and displayed in the runtime environment. OPC Clients can access device and tag information through the interface to the OPC Server, which in turn interacts with the CtAPI interface to the Runtime. For details on how to configure the OPC Server, refer to Using OPC Server DA2.0

Chapter: 9 Administering Projects

CitectSCADA is a project-based application. This section of the help looks at the administrative tasks associated with creating, storing and maintaining your projects. It includes:

- Managing your projects
- Archiving projects
- Including projects
- Working with the Project Editor
- Using Find and Replace in a project

Managing your projects

This section of the help explains how Citect Explorer is used to manage the administration of your projects. It includes the following topics:

- Creating a project
- Editing the properties of an existing project
- Copying projects
- Printing project details
- Deleting a project
- Linking projects
- Time Synchronization

Creating a project

There are two ways to create a new project:

- Use a pre-defined starter project
- Create a project from scratch

To base a project on an existing starter project:

1. Start Citect Explorer.
2. Choose **New Project** from the **File** menu, or click the **New Project** button.
3. Type a name for your project and choose a location for the files. This is mandatory.

4. Enter a **Description**, and the **Location** where the new project files are stored.
5. Click the **Create project based on starter project** checkbox.
6. Choose the project on which you want to base your new project.
7. Click **OK**.

The starter project will contain pages, roles and other features that will help you quickly get started with you project. Exact features will vary based on the template you base your project on. For example, the Tab_Style starter project will contain:

- A cluster named "Cluster1".
- A role named "Administrators" which is linked to the "BUILTIN\Administrators" Windows group and have global privilege of 8.
- Pages of Alarm, Summary, Disabled, Hardware, ProcessAnalyst and !ProcessAnalystPopup based on the relevant templates found in the Tab_Style_Include project.

The newly created project will be immediately compilable, and will contain a basic level of built-in functions such as viewing alarms and trends.

To create a project from scratch

To make it easier to configure a project from scratch, follow these steps:

1. Start Citect Explorer.
2. Choose **New Project** from the **File** menu, or click the **New Project** button.
3. Type a name for your project and choose a location for the files. This is mandatory.
4. Enter a **Description**, and the **Location** where the new project files are stored.
5. Select a **Template style** and **Template resolution** to set the appearance of the graphics pages.
6. Click **OK**.

If creating a project based on the tab style templates, don't include pages based on templates that use a different style, including the earlier CSV_Include project. Doing so might affect functionality.

See Also

New Project dialog
Creating a New Tab Template Project

New Project dialog

This dialog box lets you create a new project. To create a new project, enter a value in the **Name** field (the other field entries are optional), then click **OK**.

Once created, project properties can be viewed and edited using the Project Properties dialog, which contains the items described below.

Name

A unique name for the project. The project name is restricted to 64 characters. It can contain any characters other than characters in the Windows file naming rules "*| \{}:<>?/;'

Since the project name is a unique identifier, CitectSCADA does not permit you to create or restore a project with the same name.

Description

A description of the project. This field is useful for giving an explanation of the role of the project. You are urged to complete this field.

Location

The directory path where the project files are stored. As the Name field is entered, the directory is automatically generated in the Location field. You can override this by manually entering the location or clicking **Browse**.

Create project based on starter project

Select this option if you want to create a project based on the built-in starter projects. Choose the project style from the **Project** drop down list that is displayed when this option is selected.

You can create custom starter projects by placing *.ctz backup files in the <User>/<Data>/Starter folder (where <User>/<Data> is the directory you chose during installation). The [CtEdit]Starter parameter can be used to change this default path.

[Page defaults] Template style

The style (appearance) of the graphics pages in the runtime system. The style you select is the default style for any new pages you add to the project. You can change the style of existing pages and templates using the Page Properties, accessed through the Graphics Builder.

Most users prefer the Standard style. You can view the pre-defined styles by looking in the Include project under Graphics, Templates.

[Page defaults] Template resolution

The default screen resolution of the standard graphics pages (such as alarms pages and standard trend pages):

Screen Type	Screen Width (pixels)	Screen Height (pixels)
VGA	640	480
SVGA	800	600
XGA	1024	768

Screen Type	Screen Width (pixels)	Screen Height (pixels)
SXGA	1280	1024
WUXGA	1920	1200
User	****	****

[Page defaults] Show template title bar

Determines whether to display the Windows title bar (at the top of each graphics page). The title bar contains the title of the window, maximize, minimize and close buttons (at the right hand end of the title bar), and the control menu button (at the left hand end of the title bar).

To display a page in fullscreen (without a title bar), the size of the page needs to be the same size as the display (or larger). If the page is smaller than the display, the title bar still displays, even if fullscreen mode is enabled. Standard templates styles are available for both page sizes.

[Page defaults] Background color

The background color that will be displayed in newly created graphics pages.

Editing the properties of an existing project

To edit the properties of an existing project:

1. Open Citect Explorer.
2. Select a project from the list.
3. Click the **Properties** button, or select **Project Properties** from the **File** menu.
4. Edit the properties in the Project Properties dialog.
5. Click **OK** to save your changes, or **Cancel** to abort.

Properties

Projects have Project General Properties and Project Page Properties.

Project General Properties

(General) Name

The name of the project. This name is identical to the name that was used when the project was created. The project name is restricted to 64 characters. It can contain any characters other than the semi-colon (;) or single quote ('). Since the project name is a unique identifier, CitectSCADA will not permit you to create or restore a project with the same name. Maximum length is 64 characters.

(General) Status

The status of the project. This can be either **COMPILED** or **UNCOMPILED**.

(General) Location

The directory path where the project files are stored. This field cannot be edited.

(General) Description

A description of the project. This field is useful for giving an explanation of the role of the project. You are urged to complete this field. Maximum length is 255 characters.

(General) Major revision

CitectSCADA sets this property to one (1) when the project is first created. You can use this field to track major changes to the project. You can use an incremental revision history (for example 1, 2, 3, . . . or A, B, C, . . .). Maximum length is 4 characters.

(General) Minor revision

CitectSCADA sets this property to zero (0) when the project is first created. You can use this field in conjunction with the Major Revision to track your project's development. Maximum length is 4 characters.

(General) Date and Time

CitectSCADA will initially set these fields to the date and times at when the project was created. These fields are useful when used in conjunction with the Revision fields. Maximum length is 20 characters each.

(General) Project ID

A unique number for the project. The project number can be between 1 and 1022.

If you enter an ID that has already been used for another project, CitectSCADAwill detect this when it compiles the project if the projects are part of the same include structure.

The project number is part of the unique identifier (object ID (OID)) used by OPC drivers when reading from and writing to tags.

If you do not specify a project number, CitectSCADA will automatically generate one the next time you select this project in the Citect Explorer, or the next time you compile. Maximum length is 4 characters.

Note: If you enter 0, your project ID is automatically set after closing the project's

"Properties" page.

(General) Read-only

Specifies that no changes can be made to the project. If an attempt is made to modify the project with this option selected, a message will prompt the user to disable the option before continuing.

Note: If you change any properties, you need to click **OK** to save the changes to the project.

Project Page Properties

(Page Defaults) [Template] Resolution

The default screen resolution of the standard graphics pages (such as alarms pages and standard trend pages):

Screen Type	Screen Width (pixels)	Screen Height (pixels)
VGA	640	480
SVGA	800	600
XGA	1024	768
SXGA	1280	1024
User	****	****

Note: You can override this default for your own pages at the time when you create them or any time afterward.

(Page Defaults) [Template] Style

The style (appearance) of the graphics pages in the runtime system. The style you select is the default style for any new pages you add to the project. You can change the style of existing pages and templates using the Page Properties, accessed through the Graphics Builder.

Most users prefer the Standard style. You can view the pre-defined styles by looking in the Include project under **Graphics | Templates**.

Note: You can override this default for your own pages at the time when you create them, or any time afterward.

(Page Defaults) [Template] Show title bar

Determines whether the Windows title bar displays (at the top of each graphics page). The title bar contains the title of the window, maximize, minimize and close buttons (at the right hand end of the title bar), and the control menu button (at the left hand end of the title bar).

To display a page in full screen (without a title bar), the size of the page needs to be the same size as the display (or larger). If the page is smaller than the display, the title bar still displays, even if full screen mode is enabled. Standard templates styles are available for both page sizes.

Note: You can override this default for your own pages at the time when you create them, or any time afterward.

(Page Defaults) Background color

The color that will display in the background of new graphics pages.

Copying projects

You can copy the contents of one project into an existing or a new project.

To copy a project:

1. Open Citect Explorer.
2. Select the **Copy** icon, or select **Copy Project To** from the **File** menu.
3. In the Copy Project dialog box, select the source project from the drop-down list under **Project name**.
4. Select an existing destination project to copy to or select a new project.
5. Click **OK** to copy the project, or click **Cancel**.

See Also

Copy Project dialog

Copy Project dialog

This dialog box lets you copy the contents from one project into another. To copy a project, specify the source [From] and destination [To] projects, then click **OK**.

[From] Project name

The name of the source project being copied. If more than one project exists, you can choose a project name from the drop-down list.

[To] (Existing or New) project

You can copy to either an Existing or a New project name and location.

- **Existing Project**: The source project is written over (replaces) an existing project location under an existing project name.
- **New Project**: The source project is copied to the new location under a new project name. A new project needs to be given a new name not currently being used, and which complies with the naming requirements as detailed below.

[To] Name

The name of the destination project being copied to.

When copying to an existing project, you need to choose a project name from the existing project names drop-down list.

When copying to a new project, you need to create a new and unique name for the project. The project name is restricted to 64 characters, and can contain any characters other than the semi-colon (;) or single quote ('). Since the project name is a unique identifier, CitectSCADA will not permit you to create or copy to a project with an existing same name.

After the new project is created, you can change the Name through the Project Properties.

When copying to an existing project location, you can choose to delete the existing contents of the destination project, including subdirectories, before the source project is copied, by checking both the *Clear location before copying*, and the *Clear subdirectories* check boxes. This removes many files that may be left behind to interfere with the copied project. If you do not clear the project location before copying, only common files in the destination project are overwritten.

[To] Clear location before copying

Specifies to delete the contents of the existing destination project before copying the source project to the destination location. This removes many files that may be left behind to interfere with the copied project.

[To] Clear subdirectories

Specifies to delete the contents of the sub directories of the existing destination project before copying the source project to the destination location. This removes many files that may be left behind to interfere with the copied project.

Location

The directory path where the destination project files are stored. As the *Name* field is entered, the directory is automatically generated in the *Location* field. You might override this by manually entering the location or clicking **Browse**.

Check that the project names and location are correct in the confirmation dialog box. Click **Yes** to copy the project, or **No** to cancel.

Printing project details

You can print configuration elements (database records, pages, Cicode files, etc.) in the current project. CitectSCADA prints to the Windows default printer.

To print project database details:

1. Open the Citect Project Editor
2. Select **Print** from the **File** menu.
3. Use the **Print selection list** to choose the elements you want to print.
4. Click **OK** to start printing, or **Cancel** to abort.

Before printing your database, print a small portion to test the results. You can change the default font, font size, and page size by choosing **Options** from the **Tools** menu. For other print options, refer to your Windows documentation.

See Also

Print (project details) dialog

Print (project details) dialog

This dialog box allows you to print the configuration elements (database records, graphic pages, Cicode files, etc.) in the current project. Click **OK** to print the selection, or **Cancel** to abort printing.

[Print selection]

Lists the elements in the project that can be printed. To select (or deselect) an element for printing, click the check box; a checkmark indicates it will be printed.

Click **Select All** to select every item in the list, or **Deselect All** to clear your selections.

[Options] Graphics pages included in print selection

Specifies a particular page to print. Use the drop-down list to select a single page from the project. Choose the **<All pages>** entry to print the pages in the project.

[Options] Group printouts by graphics page

Print the objects database information with the related page. If this option is not set, then the objects database information is printed as continuous lists, with just a page reference.

You can only print the contents of the current project. Included projects will not be printed. You can specify the print font, font size, and page size in the **Options** for the Project Editor (in the **Tools** menu).

Deleting a project

To delete an existing project:

1. Open Citect Explorer.
2. Select a project from the list.
3. From the **File** menu, select **Delete Project**.
4. A message box asks you if you want to proceed. Click **Yes** to delete the project, or click **No** to cancel.

You cannot delete a project that is currently open or any installed project. You also cannot delete the Include project that is supplied with CitectSCADA.

> **Note:** You cannot recover a deleted project that hasn't been backed up.

See Also

Linking projects

Linking projects

CitectSCADA installations on different computers over the same network can share the same project. After a project has been created on one computer, other computers on the same network can link to the same project, but only if the project location is on a shared or network drive. Once linked, the remote project is visible in the local Citect Explorer, and can be edited and compiled over the network. Only one version of a project ever exists, and this version has to be kept on the computer it was created upon.

> **Note:** Linking to a project provides the developer with normal access and control to the project, even though it might be on a remote machine over the network.

> **Note:** It is possible to delete a linked project, even though it might be on a remote machine over the network. unlink a project rather than delete it over the network.

Linked projects will not be included into the compile of any other project unless they have specifically been Included into that project from within Project Editor.
For details, see Including projects.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Restart the client process if the hardware alarm "Cicode library timestamp differs" is raised after a page is opened.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Note: A hardware alarm of "Cicode library timestamp differs" will be raised if the Cicode library used by a page has a different timestamp from the one in memory. The timestamps will be different if the project has been fully recompiled, the project has been incrementally recompiled after the page has been modified, or if the project has been incrementally recompiled after any Cicode has been modified.

To link to a project:

1. Open the Citect Explorer.
2. Click the **Add Link** button, or select **Add Project Link** from the **File** menu.
3. Use the Select Project Directory dialog to choose a project location**.**
4. Click **OK** to link the project, or click **Cancel**.

If the new project has the same name as an existing one, you are prompted to change it before proceeding. Edit the properties in the Project Properties dialog.

To remove a link to a project:

1. Open the Citect Explorer.
2. Select a project from the list.
3. Click the **Remove Link** button, or select **Remove Project Link** from the **File** menu.
4. You are prompted if you want to proceed. Click **Yes** to remove the link, or **No** to cancel.

See Also

Including projects, Improved Client Side Online Changes

Time Synchronization

Previous versions of CitectSCADA employed a message-based time Synchronization server to verify clocks on computers running a CitectSCADA project maintained time synchronization. To support CitectSCADA running under standard user rights with User Access Control (UAC) switched on in Windows Vista, our existing Time Server functionality needed to be replaced.

Note: This has also made the Cicode function TimeSet obsolete, and any usage of it is recommended to be removed from your existing code.

In order to maintain time synchronization CitectSCADAv7.20 now installs a Windows service called TimeSyncService, which runs under the built-in LocalSystem account. The purpose of this service is to maintain the time on the local computer against one or more time sources. A time source is a computer on which the time service is running.

A Time synchronization utility is provided by CitectSCADA to assist you to configure time synchronization, and control the service as part of your administration environment. The dialog stores and reads settings in the TimeSyncConfig.xml file, which is installed in the CitectSCADA Config directory by default. See New Locations for Configuration and Project Files for information about configuration file locations. Using the configuration utility, you can specify an alternative path to the config file, such as a network share. This can be useful where you have multiple computers using the same configuration data and to change any setting you only need to change it on one machine.

To display the Time Synchronization dialog, open Citect Explorer and from the Tools menu select the Time Synchronization menu item.

See Also

Time Synchronization Dialog

Time Synchronization Dialog

The fields available on the Time synchronization dialog are described in the following table.

Field	Description
Current status	Displays the status of the TimeSync Windows service, as displayed in service properties under computer management. You may click the Start Service button if the service is stopped, or Stop Service service if it is running. If the service is identified as being disabled, the button is also disabled. To enable the service use the Windows administrative tools as either automatic or manual startup.
Startup type	Identifies if the service is started manually, or automatically. If the service is disabled, use the Windows administrative tools to enable the service as either automatic or manual startup.
TCP/IP Port	The network port the service will use to listen for connections from clients.

Field	Description
Last synchronization	Displays the value of the LastSyncTime registry setting. This is the Local time at which the last successful Synchronization occurred.
Current local time	Displays the current time on the local computer, updating every 1 second.
Log information events	Controls whether the service writes events of type 'Information' to the event log. The default is unchecked so that only alerts (called "warnings" in the software) and errors are recorded.
Keep this computer's time synchronized	Select this check box to enable the computer to be a time client. This allows you to enter the poll time and list of time servers against which to synchronize.
Synchronize every	Enter a number in hours between 1 and 168 (inclusive) to specify frequency that synchronization needs to occur. The default value is 24.
Synchronise Now	Click to synchronize immediately
Synchronize with first available	Displays a list of computers, and the current time on those computers if available. The display is updated every 1 second
Add button	Displays a dialog for you to enter the name of a server to add.
Remove	Select a computer from the list above, and click "Remove" to remove it from the list

Note: When you add a time source to the list, the current time on that machine will be displayed, provided the service is running on that remote machine and listening on the same port number. If "Not available" then the service is not running, or is running and using a different port number, or that port number is being blocked by a firewall. The column in the list box is provided as a diagnostics function to ensure that the machine names entered can be synchronized against. The time displayed in this box is an approximate only.

See Also

Time Synchronization

Archiving projects

Once you have configured your system, back up (or archive) the project. This will avoid the loss of any configuration data in the event your primary storage becomes inoperative or inaccessible.

Note: When you are developing a project, adopt a regular backup strategy. Before performing a backup, verify that you have refreshed any linked tags in your project.

CitectSCADA lets you back up a project to a local drive (hard drive), network location, or removable media (floppy drive, memory stick).

This section of the help includes information on the following archiving tasks:

- Backing up a project
- Backing up INI files
- Configuring a backup with password encryption
- Running a backup from the command line
- Restoring a project

Backing up a project

The CitectSCADA Backup program archives files using a standard compression routine, producing PKZip® v2.04g compatible files. The default extension for CitectSCADA backup files is .CTZ, though any extension (including .ZIP) can be used. This means you can also use the PKZip utility to extract files from a compressed CitectSCADA backup.

Note: Files produced with this backup program cannot be restored by product versions earlier than 5.10.

To back up a project:

1. Open Citect Explorer.
2. Click the **Backup** button, or select **Tools | Backup**. The Backup Project dialog box displays:
3. In the **Name** field, select the name of the project to back up.
4. In the **Backup file** field, enter the path to the backup file location, including the file name. You can either type the path in directly or use the **Browse** button.
   The backup file name defaults to <project>.CTZ. If the extension is omitted then .CTZ is used.
   When you back up a project to a floppy disk, the backup program will ask you if you wish to delete the files on the floppy disk before starting the backup.
   If the destination drive is configured as A: or B: and is detected as removable, you will have the option to delete any existing files on the disk.
5. Under **Options**, select the necessary options from the following list:
   - **Use compression:** You can use data compression when you are backing up a project to save space.

- **Save compiled:** By default, CitectSCADA backs up the project in **uncompiled** mode. If you select this option, CitectSCADA backs up both the **compiled** and **uncompiled** projects, resulting in a larger backup file.
- **Save sub-directories:** If you select this option, CitectSCADA also backs up data in any sub-directories within the project directory. The directory structure is maintained in the backup, and you can choose to restore the sub-directories when restoring the project. For example, if you wish to back up your Process Analyst Views, save them in a sub-directory of the project and select this option. When you restore the project, you will have the option to also restore the Process Analyst Views directory.
- **Use encryption:** As an added security measure, you can back up your project in an encrypted format. If you select this option, CitectSCADA requests a password. CitectSCADA writes the project to disk in a format that encodes the password along with the protected project. The project can only be restored if the password is entered.
- **Save configuration files:** Select this option to back up *.ini files from the Config folder. This will also backup the TimeSyncConfig.xml file used to store the time synchronization settings configured in the Time Synchronization utility.

6. Click **OK**.

See Also

Backing up INI files

Backing up INI files

By default, when you select the 'Save configuration files' option, *.ini files from the Config folder are backed up.

If you are using a custom INI file (for example 'abc.ini') and it is placed in the Config folder, it also will be backed up. If you are using a custom INI file that is stored in a subdirectory of the project select the 'Save sub-directories' option to back it up as well.

Note: You can define a non-default INI file for CitectSCADA by passing a parameter through to the Project Explorer from the Project Explorer Properties dialog box on the Shortcut tab. See Using an Alternative INI File for further information on how to do this.

If you run the backup program from the command line, and you specify an INI file as a parameter, the specified INI file will be backed up instead of Citect.ini.

See Also

Configuring a backup with password encryption

Configuring a backup with password encryption

When you select the "Use Encryption" backup option, CitectSCADA writes the project to disk in a format that encodes the password. The project can only be restored when the password is entered.

To use encryption:

1. Select the "Use encryption" option on the Backup Project dialog box.
2. Click **OK**. The Backup/Restore-Encryption dialog displays:
3. In the **Enter Password** field, enter your password. Asterisks will display in place of the characters.
4. In the **Re-Enter Password** field, re-enter your password. CitectSCADA checks that you have typed the same password both times.
5. Click **OK**. The project will be backed up.

See Also

Running a backup from the command line

Running a backup from the command line

You can execute the CitectSCADA backup program from the command line to back up and restore files other than CitectSCADA projects.

From version 5, the backup program is called CtBack32.exe. For older versions, it is called CtBackup.exe. By default, it is installed in the CitectSCADA project 'Bin' folder.

The CitectSCADA Backup program archives files using a standard compression routine, producing PKZip v2.04g compatible files. The default extension for CitectSCADA backup files is .CTZ, though any extension (including .ZIP) can be used. This means you can also use PKZip to extract files from a compressed CitectSCADA backup if you prefer.

When you execute a backup from the command line, if you specify an INI file as a parameter, it will be backed up instead of the default citect.ini file.

The backup program reads the citect.ini file for any parameters set using the [BACKUP] category. These settings (if any, and their defaults if not) are over-ridden by any values passed as command line options.

The table below describes the backup command line options.

Option	Description
-d<name>	database name

-m<ext>	include extension
-x<ext>	exclude extension
-e	encrypt with password
-p<password>	encrypt/decrypt password
-s[+/-]	recurse subdirectories
-f<level>	format level, 0 only format if necessary, 2 always format disk. [obsolete since version 3.xx, 4.xx]
-u[+/-]	save uncompiled, use **-u-** to save compiled
-g[+/-]	show configure dialog
-c[+/-]	compress files
-b<path>	path to backup from
-r<path>	path to restore to
-i<filename>	ini file name
-f1	use old file format (truncates long filenames to 8.3)
-a	run in auto mode (**Note:** Every necessary input needs to be in command line or INI file.)

Examples

- To back up (in version 3) c:\data use the following command:

```
CTBACKUP -g- -bc:\data
```

- To restore the above data use (in version 5):

```
CTBACK32 -g -rc:\data
```

- To backup a CitectSCADA database, for example, to backup demo use:

```
CTBACK32 -dDEMO -b -u- -c+ -d-
```

Ctbackup also uses the following parameters in the CITECT.INI file:

```
[BACKUP]
Database= ! database to backup or restore
BackupPath= ! file to backup to, for example c:\temp\example.ctz.

DrivePath= ! path to backup to or restore from.

FilePath= ! file path, used in not a database
BackupFile= ! file name on backup disk, default CTBACKUP.

Password= ! encryption password
Drive=0/1/2 ! 0=other, 1=A, 2=B
DiskSize=0/1 ! low density=0, high density=1
Encrypt=0/1 ! encrypt backup
FormatLevel= ! format level.

Configure=0/1 ! display configure dialog
Compress=0/1 ! compress backup
Overwrite=0/1 ! overwrite
SaveCompiled=0/1 ! save compiled
Recurse=0/1 ! recurse sub directories
DeleteAll=0/1 ! delete all before restore
SaveIniFiles=0/1! determines whether save ini files is checked
Operation=0/1 ! 0=backup, 1=restore
Include= ! include list
Exclude= ! exclude list, default DBK,_CI
CompiledFiles= ! compiled files, default RDB
FileFormat=0/1 ! 1= use old format (truncates long filenames to 8.3)
```

Restoring a project

You can restore backed up and archived projects using the Restore Project program. This program allows you to overwrite any current project with a backed up version, or restore a backed up project as a new project.

Note: Be careful when restoring files as every file in the destination and sub-directories will be deleted before restoring. If you accidentally set your restore path to the root directory of the drive, the program will delete your entire disk drive.

CAUTION

HARD DISK DRIVE ERASURE

Do not set the Restore Project path to the root directory of your drive (usually c:\).

Failure to follow these instructions can result in equipment damage.

To restore a project:

1. Open Citect Explorer.
2. Click the **Restore** button

or select **Tools** | **Restore**. The Restore Project dialog box will display.

3. In the **Backup file** field, enter the name of the project to restore.
4. Under **To**, select `Current project', to overwrite a project with the backed up one, or `New project' to restore a backed up project as a new one.
5. In the **Name** field, enter a name for the restored project.
6. In the **Location** field, enter the location of the project to restore, including the file name. You can either type in the path directly, or use the **Browse** button.
7. Under **Options**, select `Configuration files' to restore backed up INI files, and the TimeSyncConfig.xml file used to store the time synchronization settings configured in the Time Synchronization utility.
8. If you backed up the sub-directories under the project, the directories will be listed under `Select sub-directories to restore'. You can choose to restore every or no sub-directories, or you can select specific sub-directories to restore.
9. Click **OK**.

See Also

Archiving projects

Including projects

With large systems, it might be more convenient to develop the application using a series of smaller projects, instead of one large project. For example, you could use a separate project for each section of the plant, or for each main process. This way, you can develop and test each of the smaller projects before including them in the main project.

CitectSCADA projects will not be included into the compile of any other project unless they have specifically been included into that project from within the Citect Project Editor.

Note: If a project exists remotely on the same network as the local installation and it is on a shared or network drive, it can be linked to the local Citect Explorer. This is different to including a project. Linking makes a project visible in the local Citect Explorer. Once linked, it can be selected as the current project for editing over the network.

Any linked project (visible in Citect Explorer) can be included within a local project, and is subsequently included in the compile of the local Project.

Be careful not to confuse include files with included projects:

- **Include Files** contain CitectSCADA commands and/or expressions and are used as substitutions in a CitectSCADA command or expression property field.
- **Included Projects** are separate and (usually smaller) projects that can be included in another CitectSCADA project so that they appear together as one project.

Each CitectSCADA system is supplied with a number of include projects. These projects contains pre-defined database records.

Recommended implementation structures

There are many ways of implementing included projects. However, there are a few preferred rules for locating projects so that servers and clients function correctly on deployment. These are listed in the table below:

		Deployment	
Recommendation	**Development Computer**	**Server / Display Client**	**WebClient**
Good	c:\user\ProjectMain	d:\run\ProjectMain (<-RunPath)	\temp\citect\deployname\ProjectMain
	c:\user\ProjectInclude	d:\run\ProjectInclude	\temp\citect\deployname\ProjectInclude
	c:\user\Include	d:\run\Include	\temp\citect\deployname\Include
OK*	c:\user\Dev\ProjectMain	d:\runA\ProjectMain (<-RunPath)	\temp\citect\deployname\ProjectMain
	c:\user\Dev\ProjectInclude	d:\runA\ProjectInclude	\temp\citect\deployname\ProjectInclude
	c:\user\Include	c:\user\Include*	\temp\citect\deployname\Include
OK*	c:\user\Dev\ProjectMain	d:\run\ProjectMain (<-RunPath)	\temp\citect\deployname\ProjectMain

Recommendation	Development Computer	Deployment Server / Display Client	WebClient
	c:\user\Includes\ProjectInclude	c:\user\Includes\ProjectInclude*	\temp\citect\deployname\ProjectInclude
	c:\user\Includes\Include	c:\user\Includes\Include*	\temp\citect\deployname\Include

* For these implementations, the client/server machine needs to already have project contents at the c:\user\Include location and the implementations won't work with the RUN/COPY features.

Including a project in the current project

To include another project (in the current project):

1. Open the Citect Explorer.
2. Select a Project from the list.
3. Select the **System** icon and then **Included Projects**.
4. Complete the Included Projects dialog that is displayed.
5. Click **Add** to append a record you have created, or **Replace** if you have modified a record.

> **Note:** Do not define circular references. That is, if project A includes project B, do not include project A in project B. This will exit without completing at compile time with a "Cannot open file" error. Instead, create another project and include both A and B into this.

See Also

Included Projects dialog

CitectSCADA's included projects

Included Projects dialog

This dialog box lets you include another project in the current project. With large systems, develop the application using a series of smaller projects instead of one large project.

You can include up to 240 projects. (You have to set `[CtEdit]DBFiles` to 310 in order to enable this limit.) Every record in each project are globally accessible (i.e., a record defined in one project can be used in another).

> **Note:** Each system automatically has an include project, which contains predefined database records and graphics libraries.

Project Name

The name of the project to include in this project (64 characters maximum).

Comment

Any useful comment (48 characters maximum).

Included projects

Each CitectSCADA installation is supplied with three predefined include projects, designed to help you develop your project faster. They are:

- the **Include** project - a template project with trending and alarm pages.
- the **CSV_Include** project - a Windows XP-styled set of templates with common toolbars and advanced visualization tools.
- the **CSV_Instant Trend** project - created to support the CSV_Include project's instant trending feature.

These projects contain pre-defined database records and graphics libraries that can be used as the foundation for the content within your own project.

> **Note:** Do not modify the include project for use as a runtime project. It will not compile successfully, and be set aside for use as a template for new projects. CitectSCADA upgrades install a new version of the CSV_Include project, which will overwrite any changes you make to the project when this happens.

The include projects are hidden from the project tree in Citect Explorer by default.

To show/hide a CitectSCADA Include project:

1. Open the Citect Explorer.
2. Select **Show Include Project** from the **View** menu.

See Also

Introducing CSV_Include

Working with the Project Editor

The Project Editor is the primary tool used to configure the variable addressing, communications and system components of a project. This section looks at the components incorporated into the Project Editor to support this process.

- Setting the Project Editor options
- Paste Tag dialog box
- Paste Function dialog box
- Find User Function dialog box
- Using Find and Replace in a project

Setting the Project Editor options

The Project Editor offers the option to change the way the project configuration environment operates.

To set the Project Editor options:

1. Launch the Project Editor
2. Select **Options** from the **Tools** menu.
3. Make the necessary option adjustments.
4. Click **OK**.

See Also

Project Editor Options dialog

Project Editor Options dialog

This dialog box allows you to adjust the functionality of the Project Editor.

Show deleted

Enables the display of deleted records in the databases. When enabled, a check box at the bottom of the database form indicates if a record is deleted.

Incremental compile

Enables the incremental compilation of the project.

Extended forms

Enables the display of extended database forms. You can also use the F2 key on the keyboard to display extended forms.

Inform record changed

Enables the "Record has been changed" message window to appear when you add (or change) data in a database form and then try to close the form - before you add or replace the record.

> **Note:** If you disable this option, you will lose data if you change a database record

> and forget to add or replace the record.

Disable user functions search

When you use a combo box to select a function (for a command or expression field), a list of built-in Cicode functions and user-written functions displays. If you disable user functions, only the built-in functions are displayed in the list.

Confirm on project packing

Enables the "Packing databases may take a long time" message window to appear when packing a database.

Auto open error form

Automatically displays the Compile Errors form if an error is detected when the project is compiled.

Compile enquiry message

Enables the "Do you want to compile?" message window to appear when the project has been modified and Run is selected from the File menu. Normally, CitectSCADA compiles the project automatically (if the project has been modified) when Run is selected.

Compile successful message

Enables the "Compilation Successful" message window to appear when the project has been compiled.

Prompt on tag not exist

Enables the "Variable tag not found. Do you wish to create this tag?" message window to appear when a variable tag is specified that does not exist in the database. With the message window enabled, you can create new variable tags as they are necessary.

Prepare for Web deployment

Automatically runs the Web Deployment Preparation tool every time you compile a project. Please be aware that this dramatically increases the amount of time taken for each compile, particularly for large projects.

Log deprecated warnings during compile

If you select this option, the compiler will generate an alert message to identify any deprecated elements it detects in a project, that is any functions, parameters, or Kernel commands that are no longer supported.

By unchecking this option, the alert messages are still included in the displayed alert count, but they are not added to the error log.

Info popup time

The delay (in seconds) from the beginning of a database search until a search information window displays. The search information window displays the number of the traced records and allows you to cancel the search. You can cancel the search by selecting the Cancel button in the information window.

Cicode Editor

The text editor that is used for editing Cicode function libraries and report format files. You need to enter the name of the executable file in this field. The default editor is the Cicode Editor (ctcicode.exe) supplied with CitectSCADA.

Report Editor

The editor that is used for editing Report Format Files. You need to enter the name of the executable file in this field. The default editor is Write (write.exe). If you are using Rich Text Format (RTF) reports, verify that your editor is RTF capable.

Print page size

The number of lines (1 to 66) printed on each page when printing database records.

Print font point

The font size used when printing database records.

Print font name

The name of the font used when printing database records.

Maximum list box items

The maximum number of records that are displayed in drop-down combo boxes.

Warn about unused tags during full compile

Enables the generation of alert entries for unused tags that are not used directly in a project. The alert entries are included in the Project Editor's Compile Errors form when a full compile is run. By default this option is not selected.

> **Note:** For this option Alert entries are generated only for a full compile, not an incremental compile.

Log "tag not defined" warnings during compile

If you select this option, the compiler will generate a `tag not defined' alert in the error log for any tags detected that are not defined in the variable database.

As CitectSCADAv7.20 now allows you to include undefined tags on your graphic pages, this alert may be redundant and impractical. By unchecking this option, the alerts are still included in the displayed count, but they are not added to the error log.

Paste Tag dialog box

If you need to insert a variable tag into a tag or expression field, you can use the Paste Tag dialog box.

To insert a variable tag into a tag or expression field:

1. Select the location you wish to insert a tag in to such as an expression field in a form.
2. Select **Paste Tag** from the **Edit** menu to display the Insert Tag dialog box.
3. Select the tag name, and click **OK** or click **Cancel**.

The tag will be inserted in the tag or expression field at the location of the cursor.

Paste Function dialog box

If you need to insert a function into a tag or expression field, you can use the Insert Function dialog box.

To insert a function into a tag or expression field:

1. Select the location you wish to insert a function in to such as an expression field in a form.
2. Select **Paste Function** from the **Edit** menu to display the Insert Function dialog box.
3. Select the function name, and click **OK** or click **Cancel**.

To insert the function with its arguments included, select the **Insert arguments** box.

The function is inserted in the current field at the location of the cursor.

> **Note:** If the total length of the function and its parameters is greater than 254 characters, it won't appear in this dialog box. Instead, the message "Text Too Big" is displayed.

Find User Function dialog box

If you need to locate a Cicode function in your Cicode source files, you can use the Find User Function dialog box.

To locate a function within a Cicode source file:

1. Select **Find User Function** from the **Edit** menu.
2. Enter the function name (or part of the function name) and click **OK** or click **Cancel**.

A list of functions that match your search criteria will be displayed.

Note: If you leave the Find field empty and click OK, a full list of functions appear in the list.

To edit the Cicode file that contains the function:

1. Select the function name from the list that appears when searching for the function (see above) and click **Edit** or click **Cancel**.

The file containing the selected function will be opened in the Cicode Editor.

Using Find and Replace in a project

You can use the Find and Replace dialog to locate specified text in your projects. You can perform global text replaces in your projects, as well as export search results.

This is explained in the following topics:

- The Find and Replace dialog
- Specifying search coverage
- Using the results list
- Removing results
- Exporting results
- Jumping to a result (Go To)
- Replacing results
- Find and Replace alert messages

There is also a topic on Troubleshooting Searches to help determine if a search has been correctly configured when unexpected results are returned.

The Find and Replace dialog

You open the Find and Replace dialog box from either the:

- **Project Editor**: You can find and replace text strings in your projects and included projects.
- **Graphics Builder**: You can find and replace text within a single graphics page, template, or Genie (including fields of the Metadata and Associations tabs [except the "In Use" field]).

You can configure your search coverage, view your results, replace results, or open a search result for more information.

To display the Find and Replace dialog box:

- From the Project Editor or Graphics Builder, click **Edit | Find** or **Edit | Replace**. The dialog box appears with either the **Find** tab or **Replace** tab selected, depending on which command you selected.

To search text:

1. On the **Edit** menu in the Project Editor or Graphics Builder, click **Find**.
2. In the **Find** box, type the text string you want to search for. The search is not case-sensitive, so it doesn't matter whether you enter lower- or uppercase letters.
   You can enter an entire string or a portion of the string you want to find. For example, typing BIT will return any string containing BIT, such as BIT_1, BITE, HABIT, HABITS, and so on. You cannot enter wildcard characters, but you can include special characters, as well as spaces if you want.
3. Specify your search coverage using the **Look in** and **Search options** lists.
4. Click **Find**. Search results appear in the results list when the search completes. The status text under the results list indicates the progress of the search.

   Note: When you start a search, the **Find** button changes to a **Stop** button you can use to exit the search. If you stop a search, a partial list of the results is displayed.

To replace text:

1. On the Edit menu in the Project Editor or Graphics Builder, click **Replace**.
2. In the **Find** box, type the text you want to search for.
3. In the **Replace with** box, enter the replacement text.
4. Specify your search coverage.
5. Click **Find**.
6. View the search results.
7. Make your replacements using **Replace** or **Replace All**.

Specifying search coverage

You specify search coverage using the **Look in** and **Search options** lists to determine which items in your projects you want to search for.

When you choose a **Look in** option, the **Search options** change. Each time you select a **Look in** option, associated **Search conditions** are selected by default. The **Look in** options work with the **Search options** as follows.

- Selecting **Current Project** or **Current Project and Include Projects** displays the options described below**.**

  - **General**: Searches configuration databases associated with a project as well as included projects (if that option is selected).
  - **Graphics**: Performs an "express search" for graphics pages only (the graphics page does not have to be currently open). *This search will not find text in symbols, Genies, or templates if they have not been used on a page*. If you don't find the text you want, try the more comprehensive graphics search by opening a page, Genie, symbol, or template and selecting **Current Graphic** as the **Look in** option (see below).
  - **Code Files**: Searches Cicode/CitectVBA files in the current project (and included projects, if that option is selected).
  - **Reports**: Searches report files within the project folder and included projects (if that option is selected).
- Selecting **Current Graphic** makes available the options described below. (This search is a more extensive search than that performed by the **Current Project:Graphics** search described above, and will search graphics documents that are currently open.)
  - **Inside Genies and Symbols** includes graphics objects contained within a Genie or symbol.
  - **Inside Templates** includes objects contained within a page template.
- Selecting **Current Form**: Searches the current form.

Using the results list

As matches are found they are listed the results list. The results list shows an overview of items that match the string entered in the search.

The results list can display a maximum of 200 results per page, sorted by project and then item (you cannot change the sort order). The results list contains the following columns:

Column	Description
Project	The name of the project in which the found text occurs.
Item	Depends on the type of document in which the item occurs. If the document type is a: **Database** - User-friendly name of the database. **Page** - Name of the page. **Cicode/VBA** - Name and path of the Cicode/VBA file. **Report** - Name of the report.
Field	Identifies that portion of the document/database in which the found item occurs in. For example, if the found item appears in a database, this refers to the column name in the database. Be aware that the search covers both expression/command as well as

Column	Description
	numeric properties.
Location	Shows the specific record number, AN, or line number on which the found item occurs within the document/database.
Context	An example of the context in which the found item occurs within the project. For example, if the document type is a: **Database** - a search result of BIT* might have a context of BIT_!. **Page** - BIT* might have a context of Toggle(BIT_!) **Cicode/VBA** - UserName might have a context of FUNCTION GetUserName() **Report** - PUMP* might have a context of @(Pump A)

If the number of results returned exceeds 200 items, use the **First**, **Previous**, **Next**, and **Last** buttons to navigate your results in groups of 200 results.

You can toggle between the Find and Replace functionality without losing the search results, but if you close the Results page, your search results are lost.

> **Note:** You can resize list columns by moving your mouse cursor onto the separator between the list columns. When the mouse cursor changes shape to a black bar with arrows, drag the column to the new size. You can also double-click the vertical bar between fields to resize that field to fit the widest item.

Removing results

You can remove a search result from the Results window. Results that are removed are not included in exports or in replacement operations. Removing a result does not delete it, but merely removes it from the Results window.

To remove a result:

- With the result you want to remove highlighted, click **Remove**. The result is removed from the Results window.

Exporting results

You can export search results in a tab-delimited format to a specified location. Results are exported in the format

```
<Project> <Item> <Field> <Location> <Context>
```

If the Results window contains more than 200 results, every result is exported, not just the ones currently displayed. If you remove an item from the results list, it will not be exported. (For details on removing results, see Removing results.)

If you export an item that has a context, the context string is stripped of tabs and new line characters.

Results exported are in Unicode format. Because of this, two leading characters and two trailing characters are added to the file, but in most cases will remain hidden. When exporting results, use Excel 2000 and later, which support the Unicode format.

To export results:

1. With the search results you want to export listed in the Results window, click **Export**.
2. Specify the location in the dialog box and then click **Save**. If the file already exists, you're given the option to overwrite the file. Status text under the results list indicates the progress of the export.

> **Note:** If you want to stop the export, click **Stop**. You cannot perform a partial export, so clicking **Stop** cancels the export entirely.

Jumping to a result (Go To)

You can jump to an individual result to see where the result occurs. Depending on the type of document that contains the search result, the following occurs:

- **Database**: The form opens in the Project Editor and the text string is highlighted.
- **Cicode/VBA**: The document opens within the Cicode Editor and the text string is highlighted.
- **Graphic**: The page opens in the Graphics Builder and the Properties dialog box appears for the object that contains the text string. The property containing the text string is displayed. If the string occurs on or inside a Genie, the Genie form also appears.
- **Report**: The configured report editor opens and displays the report file, but the text string is not highlighted.

To jump to a result:

- With the search result you want to jump to highlighted in the Results window, click **Go To**. The document or form containing the occurrence opens.

See Also

Replacing results

Replacing results

You can replace single results or multiple results with the replacement text string you specified. You can also test a single result before replacing it. Depending on the type of document that contains the search result, the following occurs when a replacement is made:

- **Database**: The result is replaced with the replacement text and the database record updated. The form containing the search result is not opened; to see the location of the search result before or after the replacement is made, use the Go To command.
- **Cicode/VBA**: The Cicode file containing the matched text loads (if it is not loaded already), the replacement is made, and the file saved.
- **Graphic**: The page opens in the Graphics Builder (if it is not already) and the replacement made. If the page is open and contains unsaved changes, you're instructed to save or discard the changes before making the replacement. If there are multiple changes to be made to the same graphics page, the page remains open until every change has been made.
- **Report**: The found text is replaced with the replacement text and the file is saved.

> **Note:** Replacements cannot be undone once performed. take care to check your replacements before making them, especially when working with multiple replacements.

To test a result:

1. With the result you want to test highlighted, click **Test**. A dialog box appears showing the result of the text replace.
2. Click **Accept** to accept the text replacement, or click **Cancel**.

To replace a single result:

- With the result you want to replace highlighted, click **Replace**. The replacement is made and the result removed from the Results window. The next result in the list is then selected.

To replace multiple results:

1. With the search results you want to replace listed in the Results window, click **Replace All**. A confirmation dialog appears.

2. The replacements are made and removed from the Results window. (Replacements that are not made remain in the results list. This will occur if, for example, you try to replace a property that is read-only.)

> **Note:** Clicking **Stop** during this process does not undo any replacements already made.

When attempting to make a replacement, you might encounter an alert message that alerts you of project-related issues be aware of before making a replacement. For details, see Find and Replace alert messages.

Find and Replace alert messages

Find and Replace will display one of the following errors if it cannot replace a text string:

- File in use
- Replaced text truncated
- Original text not found
- Replaced text out of range
- Replacement text not numeric
- Field is read-only
- Undetermined error

File in use

This alert message appears if the database or file that is necessary for writing to has become unavailable. This may be the case if the database/file is being used by a third-party application.

Do one of the following:

- Click **Try Again** (Default) to repeat the operation on the database/file.
- Click **Ignore** to skip the operation on this file.
- Click **Ignore All** button to skip any operations on files that are currently in use; this option causes this message not to reappear.

Replaced text truncated

This alert message appears if performing the text replacement in a DBF field exceeds the field width limits. For example, if an Engineering Units field containing % is replaced with "%LONGTEXTSTRING", a "replaced text truncated" alert message appears because this field has a max width of 8. The replacement text would therefore be "%LONGTEX".

This alert message does not occur if searching the current graphics page.

Do one of the following:

- Click **Yes** to commit the truncated text to the database. Usually this will generate a compilation alert message when the project is compiled.
- Click **Yes to All** to commit every change to fields regardless if truncation exists without displaying the alert again.
- Click **No** (default) to leave the text as is.
- Click **No to All** button to leave truncated fields as is.

Original text not found

This alert message appears when attempting to replace an item on the current graphics page when the animation could not be found or the text could not be found. This would occur if the animation was deleted, or if the text found in an animation's field was changed after the find but before replacing the item.

In the example below, the Fill Level Maximum contained a value of 23, and the search text was 23. Before replacing the record, it was changed to 66.

Do one of the following:

- Click **Ignore** to skip this operation, leave the entry in the list, and move on to the next replacement if it exists.
- Clicking **Ignore All** acts like the **Ignore** button, except that it skips any not found errors that occur during this replacement.
- Click **Stop** to stop the replacement at the current record.

Replaced text out of range

This alert message appears when carrying out a replacement on the current graphics page in two different circumstances:

1. The field is text and is too long for the allowable field width.
2. The field is a numeric field, and would be out of the allowable range for a replacement value.

In the example below, the Fill Level Maximum allows a range or 0-100, and the value was 23 and is being replaced with 101, which would be out of range.

Do one of the following:

- Click **Ignore** to skip this operation, leave the entry in the list, and move on to the next replacement if one exists.
- Clicking **Ignore All** acts like the **Ignore** button, except that it skips out-of-range errors that occur during this replacement.
- Click **Stop** to stop the replacement at the current record.

Replacement text not numeric

This alert message will appear when carrying out a replacement on the current graphics page when the field being replaced is a numeric field, and the replacement text contains a non-numeric value.

In the example below, the Fill Level Maximum contained a value of `23', and the replacement text was `fred'.

Do one of the following:

- Click **Ignore** to skip this operation, leave the entry in the list, and move on to the next replacement if it exists.
- Clicking **Ignore All** acts like the **Ignore** button, except that it skips any non-numeric errors that occur during this replacement.
- Click **Stop** to stop the replacement at the current record.

Field is read-only

This alert message appears when replacing an item on the current graphics page when the field being replaced is part of a linked object like a Genie or template.

In the example below, the Expression field was part of an object that was part of a genie.

Do one of the following:

- Click **Ignore** to skip this operation, leave the entry in the list, and move on to the next replacement if it exists.
- Clicking **Ignore All** acts like the **Ignore** button, except that it skips any read-only errors that occur during this replacement.
- Click **Stop** to stop the replacement at the current record.

Undetermined error

This alert message appears when carrying out a replacement on the current graphics page when a general error is detected, and not happen in normal operation.

Do one of the following:

- Click **Ignore** to skip this operation, leave the entry in the list, and move on to the next replacement if it exists.
- Clicking **Ignore All** acts like the **Ignore** button, except that it skips any undetermined errors that occur during this replacement.
- Click **Stop** to stop the replacement at the current record.

Troubleshooting Searches

If you don't find a result that you expected to find, check the following points, and then perform your search again:

- Did you spell the text string correctly?
- Did you include the correct number of spaces?
- Are you using the appropriate Look in option?
- Are you using the appropriate Search options?
- Are you searching in the correct project?
- Are you using the correct graphics search?
- If you are using the graphics page search, do you have the correct graphics page open?

Chapter: 10 Securing Projects

CitectSCADA projects represent a considerable investment. Once a commissioned project has been delivered, it usually needs to remain in the delivered state until modifications are performed by an authorized person. In order to help protect projects from modification by unauthorized personnel, CitectSCADA allows projects to be secured by an administrator as "read-only."

For large applications, or applications where access to certain processes or machinery needs to be restricted, you can build security into your system. You can then restrict access to commands that you do not want to be available to evry one of your operators; for example, commands that operate specialized machinery, acknowledge critical alarms, or print sensitive reports. There are 2 options available to you to configure security for your system.

These options are to use CitectSCADA native security or CitectSCADA integrated with Windows Security .

This section describes the following:

- Characteristics of read-write and read-only projects (see Overview).
- Scenarios that describe Securing a Top-level Project and Securing an Include Project.
- How to secure projects as read-only (see Making a Project Read-Only).
- The consequences of securing projects (see Read-Only Privileges on Projects).
- Using CitectSCADA native security.
- Using CitectSCADA integrated with Windows security

Overview

CitectSCADA has two types of project:

- **Read-write**: allows write and delete privileges to the project folder (or for any project file) for the current user.
- **Read-only**: projects that deny write and delete privileges to the project folder for the current user.

The table below shows the different characteristics of read-only and read-write projects:

Task	Read-only	Read-write
View existing files in project	x	x
Create new files in project		x
Delete existing files in project		x
Modify existing files in project		x
Delete project		x
Rename project directly		x

Read-only projects cannot be compiled as top-level projects (i.e., projects that are the main (root) project as opposed to an included project) and online changes are not supported.

Note: If the project folder is read-only for the current user, but one or more files in the project have read-write access for the current user, the project is considered to be a hybrid read-only/read-write project. CitectSCADA does not support this type of project. Running a hybrid project may result in your system becoming unresponsive. (This note does not include those folders or files that require read-write access in order to operate at runtime; see Using CitectSCADA with Windows Security for details.)

The security model used in enabling read-only projects does not replace the existing CitectSCADA user accounts; instead, it works in conjunction with user accounts like this:

- CitectSCADA user accounts govern runtime security to project elements.
- Windows user accounts govern the security of configuration project elements.

Securing a Top-level Project

This section describes a "real-world" situation that might require read-only privileges to be applied to a top-level project.

Note: Before securing a top-level project, read the section Read-Only Privileges on Projects for details on operational constraints. Pay particular attention to the section Read-only on top-level projects.

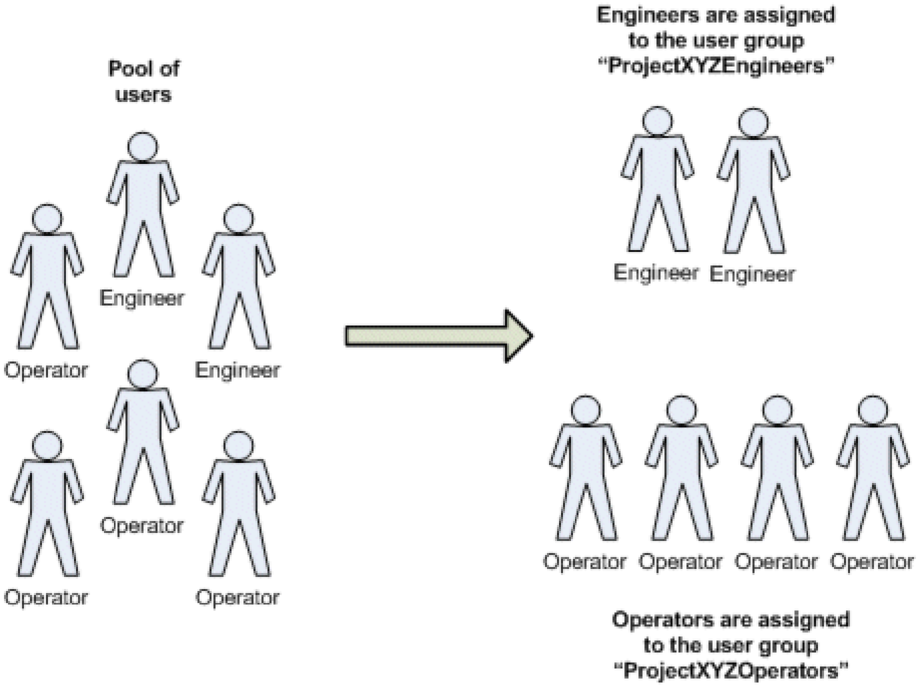
In this scenario, several onsite engineers are responsible for maintaining a top-level project, ProjectXYZ. Consequently they require read-write privileges for every project folder.

The operators responsible for monitoring plant operations will use the project at runtime only; consequently operators only have read-only access to the project.

The system administrator on site first identifies those employees who will use the project, and then divides this pool of users into two user groups:

- **Project Engineers** - responsible for project configuration.
- **Operators** - responsible for the project's runtime operations.

This is shown in the illustration below.


Pool of users
Engineer
Operator
Engineer
Operator
Operator
Operator
Engineers are assigned to the user group "ProjectXYZEngineers"
Engineer
Engineer
Operator
Operator
Operator
Operator
Operators are assigned to the user group "ProjectXYZOperators"

The administrator creates two user groups to make administering users easier: **ProjectXYZEngineers** and **ProjectXYZOperators,** and assigns engineers to the first group, operators to the second.

> **Note:** Creating user groups is optional and makes it easier to handle privileges for multiple users. Creating user groups may be unnecessary if you only have a few users.

The administrator then assigns engineers read-write privileges to the top-level project folder, and operators read-only privileges, like this.

1. Select the project folder of the top-level project and display its properties.
2. Select the **ProjectXYZEngineers** user group and allow read-write privileges. (Remember that in order to use read-write projects, read, write, *and* delete privileges needs to be assigned.)

3. Select the **ProjectXYZOperators** user group and deny write privileges. See the section Making a Project Read-Only for the specific privileges assign.
4. Apply and save the changes.
5. Review the changes to verify that engineers and operators have the correct privileges for their roles.

See Also

Securing an Include Project

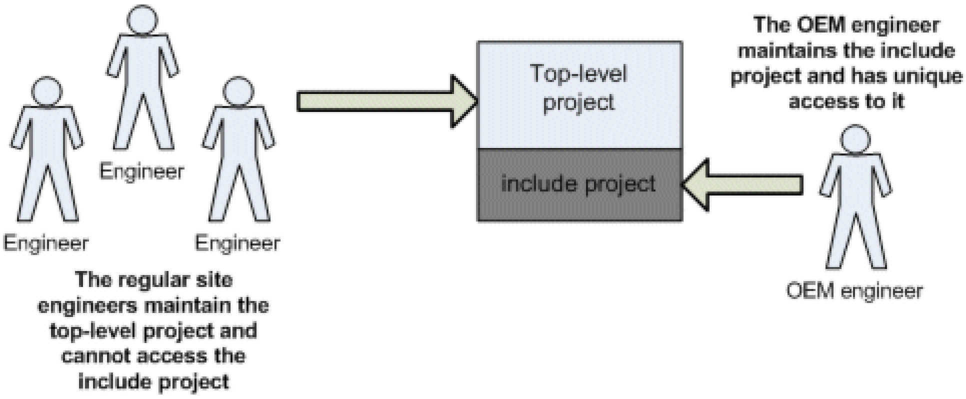
Securing an Include Project

This section describes a "real-world" situation that might require read-only privileges to be applied to an include project.

> **Note:** Before securing an include project, read the section Read-Only Privileges on Projects for details on operational constraints. Pay particular attention to the section Read-only on include projects.

In this situation, an OEM has configured and delivered an include project that is part of a larger (top-level) project. Because the OEM engineer is solely responsible for maintaining the include (and *only* the include) project, the site administrator assigns the OEM engineer read-write access to the include project, but read-only access to the top-level project. Conversely, the site's regular engineers can access the top-level project but not the include project.

This scenario is shown here:


Engineer
Engineer
Engineer
The regular site engineers maintain the top-level project and cannot access the include project
Top-level project
include project
The OEM engineer maintains the include project and has unique access to it
OEM engineer

To set up this scenario the administrator does the following:

1. For ease of administration, assigns the site engineers to the user group **AcmeTopEngineers**.

Note: Because there is only one OEM engineer, the administrator did not create a user group for this single user.

2. Selects the project folder of the top-level project and displays its properties.
3. Selects the **AcmeTopEngineers** user group and allows read-write privileges for this folder. (Remember that in order to use read-write projects, read, write, *and* delete privileges needs to be assigned.)
4. Applies and saves the changes.
5. Selects the include project.
6. Selects the user name of the OEM engineer and allows read-write privileges for this folder.
7. Applies and saves the changes.
8. Reviews the changes made to verify the correct privileges have been assigned. In particular, the administrator has to confirm that the privileges assigned to the **AcmeTopEngineers** user group deny read-write access to the include project.

See Also

Securing a Top-level Project

Making a Project Read-Only

This section describes how to make a project folder read-only by modifying the file security settings for selected users and/or user groups. The read-only project configuration is identical whether you are using Windows 2003 Server, Windows XP, or Windows 2000.

Notes

- This procedure assumes that you have already configured your users and user groups (optional), and added them to the User groups for the project folder list for the project you want to secure. For details on creating users and user groups, refer to your Windows documentation.
- Before making your project read-only, verify that read, write, delete, and execute privileges have been applied appropriately to allow the CitectSCADA configuration and runtime environments to operate correctly. For details, see Using CitectSCADA with Windows Security.
- If you are using Windows XP or Windows 2000 and your file system is FAT32, you need to convert the file system to NTFS in order to be able to specify the correct user privileges on that workstation. For details, refer to your Windows documentation.
- To avoid unexpected results, read Read-Only Privileges on Projects before making your project read-only.

To make a project read-only:

1. In Windows Explorer, select the project folder you want to make read-only. By default project folders are located in the folder

```
C:\ProgramData\Citect\CitectSCADA 7.10
```

2. Right-click the folder and choose **Properties** from the context menu. The Properties dialog appears.
3. Select the **Security** tab.
4. Select the user and/or user group you want to modify security settings for.
5. Click **Advanced**. The Advanced Security Settings dialog appears for the selected user/user group for the project folder.
   Verify that the user or user group you want to modify permissions settings for is selected.
6. Click **Edit** to display the Permission Entry dialog box.
7. Click **Clear All** to clear the current selections and then select the **Allow** check box for the following options:
   - **Traverse Folder/Execute File**
   - **List Folder/Read Data**
   - **Read Attributes**
   - **Read Extended Attributes**
   - **Read Permissions**
8. Click **OK**.
9. Click **Apply** to apply the permissions to the selected user/user group, and then click **OK** to dismiss the Advanced Security Settings dialog.
10. Click **OK** to close the Properties dialog box.

The project folder has now been specified as read-only for the selected user(s) and/or user group(s).

See Also

Securing a Top-level Project
Securing an Include Project

Read-Only Privileges on Projects

This section describes the operational constraints of making a project read-only. This section also describes issues specific to securing top-level and include projects.

Note: Before Making a Project Read-Only, make sure you're familiar with the issues

> described here. Also make sure that the correct privileges have been set in order for the configuration and runtime environments to operate; for details, see Using CitectSCADA with Windows Security.

- Startup
- General
- Graphics and pages
- Backup and restore
- Project upgrades
- Debugging
- Web deployment
- Runtime issues

Most of the issues discussed above are common to both top-level projects and include projects. The sections listed below discuss issues specific to these types of projects:

- Read-only on top-level projects
- Read-only on include projects

Startup

A project is determined to be read-only when Citect Explorer starts up. If the security permissions on the project folder are modified after Citect Explorer has started, the CitectSCADA configuration applications may not be able to determine accurately that the project is read-only.

See Also

Using CitectSCADA with Windows Security

General

When a read-only project is opened using the Graphics Builder, Project Editor, or Citect Explorer. the title bar shows the name of the project and a Read-Only message to indicate the project is read-only.

Opening the Express Wizard for a read-only project displays a message on the first page indicating the project is read-only:

In addition, any menu commands, toolbar buttons, and other operations that perform a write function are grayed out and/or unavailable. For example, the Copy command is available in the Project Editor for a read-only project, but the Cut and Paste commands are not.

When using the Process Analyst, you cannot create views to a project folder that is read-only and an alert message is displayed.

Graphics and pages

You cannot update graphics documents or pages in read-only projects. (You can, however, update pages in a read-write project if that top-level project includes one or more read-only projects.)

Opening an updated graphics page in a read-only project in Graphics Builder will attempt to show the upgraded symbol, but only at the presentation (not disk) level.

> **Note:** Updating graphics documents and/or pages in mid-level include projects is not recommended because it will not update pages in the top-level project. However, issuing an update pages at the top level iterates through included projects recursively. (If these need to be updated, make the relevant projects read-write first).

Backup and restore

Read-only projects can be backed up by any user, regardless of their privileges. However, the backup functionality does not archive the current security permissions. Consequently, if the project is restored on another machine, the security settings need to be reapplied.

Projects cannot be restored into an existing read-only project; attempting to do so will display an alert message advising that the project is read-only and cannot be restored.

Project upgrades

A project upgrade occurs when any of the following occurs:

- `[CtEdit]Upgrade=1` is added to the `citect.ini` file.
- A project link is added via Citect Explorer.
- When a project is restored.

When CitectSCADA detects that the include, system, or CSV_Include project is read-only and the version of CitectSCADA that the project was created under does not match the current version of CitectSCADA, a message box is displayed to advise you of this.

In addition, when CitectSCADA detects that a user project is read-only and the version of CitectSCADA that the project was created under does not match the current version of CitectSCADA, a message box is displayed to advise you of this.

Any links to the project will be removed and the project tree in Citect Explorer will be updated to indicate this.

If you plan to upgrade a top-level project, you need to log on as a user with the appropriate read-write security privileges for this project, add a link to the project in Citect Explorer, and then perform the project upgrade again.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Do not leave non-upgraded projects in your project environment.
- If security privileges prevent a successful upgrade of projects related to your system, restore the system to its prior state before resuming operations.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Debugging

In read-only projects you can set breakpoints when debugging code, but these breakpoints aren't saved when you exit the Cicode Editor.

Web deployment

You cannot perform Web deployment with read-only projects.

Runtime issues

By default most output operations during runtime occur in the `[DATA]` or `[RUN]` location (see below for details). By default CitectSCADA configures the `[RUN]` location to the project directory. If you intend on making the project directory read-only, you need to modify the path(s) to a suitable read-write location.

- **`almsav.dat`** - Alarm data by default is saved in the [RUN] location, which is usually the project folder. You need to change this location if you intend on making the project folder read-only. Alternatively use the `[Alarm]SavePrimary` and `[Alarm]SaveSecondary` options in the `citect.ini` file to control the location of the output.
- **Disk PLCs** - If a user does not have the correct privileges for the `[RUN]` path, communications will be offline for disk PLCs. You need to change this location if intending to make the project folder read-only.
- **User Cicode functions** - Making a project read-only prevents the use of the following user functions: `UserCreate`, `UserDelete`, `UserEditForm`, `UsetrSetPassword`, `UserSetPasswordForm`. Attempting to use these functions results in an error code 262 (0x0106) ("Cannot open file").

- **Alarm Cicode functions** - Making a project read-only prevents the use of the following alarm functions: `AlarmSetDelay`, `AlarmSetDelayRec`, `AlarmSetThreshold`, `AlarmSetThresholdRec`. Attempting to use these functions results in a hardware alarm 400 (0x0190) ("Project or file is read-only"). You also cannot modify alarm properties such as threshold or delay.

Any files in your top-level project that require runtime read-write access have to be located outside of the project folder.

Read-only on top-level projects

Before applying read-only to top-level projects, note the following:

- Before applying read-only to a top-level project, perform a full compile and then apply the relevant security attributes. Once you apply read-only privilege, you cannot compile a top-level project (see following bullet). For projects that needs to manipulate files in the top-level project, you can modify the security of individual files to read-write; the project will be regarded as read-only as long as the project folder is denied write privileges.

> **Note:** Any files in your top-level project that require runtime read-write access have to be located outside of the project folder.

- Projects that are read-write that have read-only include projects as a component can be compiled as usual.
- You can only run a read-only project if it is a top-level project.
- Applying read-only to top-level projects prevents online changes being made to alarms, users, trends, and pages.

See Also

Making a Project Read-Only
Securing a Top-level Project

Read-only on include projects

Note the following before applying read-only to include projects:

- When compiling a top-level project with a read-only include project, the object IDs of the included projects are skipped and not changed.
- Updating graphics documents and/or pages in mid-level include projects is not recommended because it will not update pages in the top-level project. However, issuing an update pages at the top level iterates through included projects recursively. (If these need to be updated, make the relevant projects read-write first).

See Also

Making a Project Read-Only
Securing an Include Project

Securing Runtime Computers

The CitectSCADA runtime system is a Windows-based application that runs in the standard Windows environment. Typically, Windows allows you to run several applications at the same time, however, this may impact performance or obscure runtime messages and information if you require a computer to be dedicated to Runtime.

For example, an operator display panel may be used to present alarm notifications. You don't want the Runtime screen minimized or hidden behind another window.

There are several different ways can limit access to software other than CitectSCADA.

See Also

Client Start up Restrictions
Running a client as a shell
Disabling Windows keyboard commands
Disabling control menu commands
Removing the Cancel button

Client Start up Restrictions

On start up CitectSCADA will establish initial communication with the server using a view-only login. By default this communication between the client and the server is in view-only mode. In this mode the user cannot write to any tag, acknowledge any alarm or use Cicode functions.

> **Note**: View-only mode is applied to the whole control client process, including any Cicode task that is running.

Write only access is available after a user has successfully logged in. Once the user logs out it returns to view-only mode.

Users can configure login by modifying the [Client]AutoLoginMode parameter.

See Also

Securing Runtime Computers

Running a client as a shell

To limit certain operators from switching a client over to a different application during runtime, you can configure Windows to launch with CitectSCADA running as the shell. This will deploy CitectSCADA Runtime as the only available interface for a computer, limiting access to within the context of the current project.

For information on how to set up a client as a shell, contact your local support office.

See Also

Disabling Windows keyboard commands

Disabling Windows keyboard commands

The Windows environment provides commands to switch between applications running on the computer at the same time. When using CitectSCADA, these commands might not be desirable - they allow an operator access to other Windows facilities without your direct control. You might be able to disable some of these commands with the *Computer Setup Wizard*. consult the Citect Knowledge Base for the latest information on disabling Windows keyboard commands.

See Also

Disabling control menu commands

Disabling control menu commands

The Control Menu (in the top left corner of an application window) provides commands to position and size the application window, and in some applications to control the application. The runtime system's Control Menu can be tailored to give access to several commands specific to CitectSCADA, such as Shutdown (to shut down the runtime system), or Kernel (to display the Kernel).

You can enable and disable these commands with the *Computer Setup Wizard*.

See Also

Removing the Cancel button

Removing the Cancel button

When the CitectSCADA runtime system starts, a message box displays the status of the system startup. This message box normally contains a **Cancel** button that allows you to cancel the startup. This button is useful when you are debugging or testing the system. When you have completed testing, you can remove the **Cancel** button from the message box with the *Computer Setup Wizard*, so that there will not be an unintentional cancellation of system startup.

Chapter: 11 Using CitectSCADA Security

To set up security in CitectSCADA you need to consider the following:

- Areas - An area is a section of the plant. It can be defined geographically or logically.
- Privileges - Level of access applied to system elements within your project. A user assigned a role that possesses the matching privilege can control it.
- Roles - A defined set of permissions (privileges and areas) that are assigned to users.
- Users - A person or group of persons that need to access to the runtime system.

Before configuring security within your project you will need to have a thorough understanding of these four aspects, and how they work together.

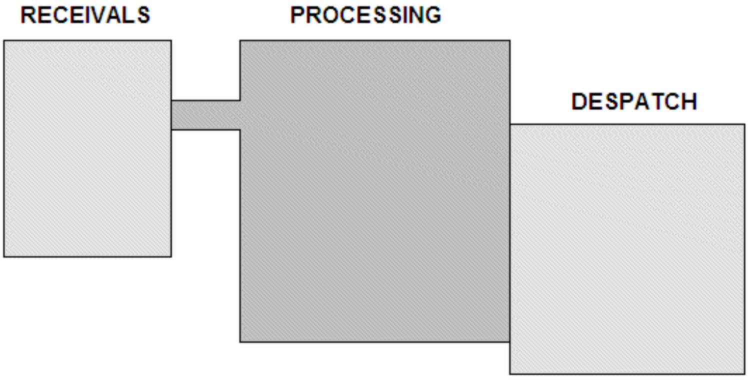
See Also

Areas
Privileges
Roles
Users
Using CitectSCADA integrated with Windows Security

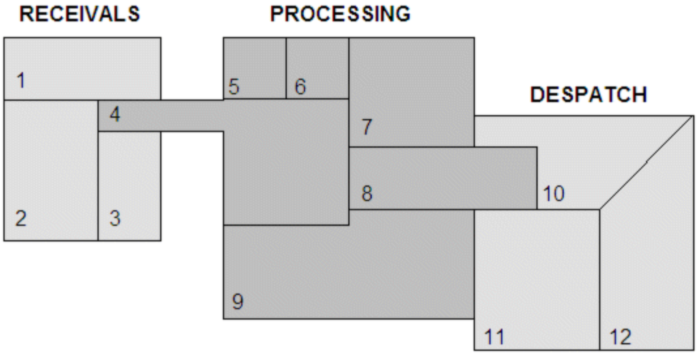
Areas

When implementing CitectSCADA for a large application, you can visualize the plant as a series of discrete sections or areas. You can define areas geographically (especially where parts of the plant are separated by vast distances or physical barriers) or logically (as discrete processes or individual tasks).

Small plants, for example a simple manufacturing plant can be divided into just three areas - raw product arrives in the receivables area, is transported to an area for processing, and is then transported to a packaging or despatch area.

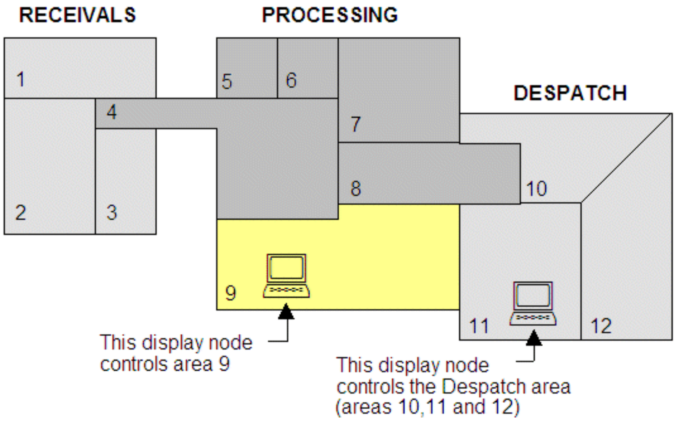


However, with larger or more complex plants you might need to define several areas, like this:



When defining an area, you would usually encompass a section of the plant that is controlled by one operator (or controlled from one CitectSCADAControl Client).

You can also define smaller areas that are collectively controlled by an operator or Control Client. This method can increase flexibility, but can introduce a higher level of complexity to your system.


RECEIVALS
PROCESSING
DESPATCH
1
2
3
4
5
6
7
8
9
10
11
12
This display node controls area 9
This display node controls the Despatch area (areas 10,11 and 12)

You can define up to 255 separate areas. You can then refer to these areas by number (1 to 255) or use a label to assign a meaningful name to the area (for example receivables, pre-process, conveying, etc).

After you have defined your areas, you then configure the system elements (commands, objects, alarms, reports, etc). your operators will use in those areas. For example:

For example:

Command	CONVEYOR = 1;
Area	8
Comment	This command belongs to Area 8

In this example, an operator without access to Area 8 will not be able to send the command. Refer to Roles for more information on how areas and roles work together.

> **Note:** Any system element that is not assigned to an area between 1 and 255 is automatically placed in a default area known as Area 0. Every user can view the system elements in Area 0, but without the matching privilege will be unable to control them.

See Also

Configuring Areas
Privileges
Roles
Users

Privileges

CitectSCADA provides eight privileges, numbered 1 to 8, that are used to restrict access to parts of the project. To implement privileges into your project:

- Assign a privilege to a particular system element (command, object, report, alarm etc)
- Assign the privilege or privileges to the role or roles that will need to control that system element.

> **Note:** Global privileges apply to every area.

You can allocate different privileges to different types of operation, as in the following example:

Privilege	Command
1	Operate the conveyors
2	Operate the mixers
3	Operate the ovens
4	Acknowledge alarms
5	Print reports
6	Operate box machine

To allow a user to operate the conveyors, you assign privilege 1 to the role associated to that user, for example:

Global Privilege	1

To allow a user to acknowledge alarms, you assign privilege 4 to the role associated to that user, for example:

Global Privilege	4

To allow a user to acknowledge alarms and operate the conveyors, you assign both privilege 1 and privilege 4 to the role associated to that user record:

Global Privilege	1,4

Privilege classifications needs to be separated by commas (,).

To allow a user access to every command in your system, allocate every privilege in the role associated to that user, for example

Global Privilege	1, 2, 3, 4, 5

Note: In assigning a role a global privilege, that role is granted view access to every area automatically. Any user assigned that role will then be able to view every area of the plant.

After you have allocated privileges, you can define the privilege requirements of your system elements (commands, reports, objects, alarms, etc.):

Command	CONVEYOR = 1;
Privilege	1
Comment	An Operator with Privilege classification 1 can operate the conveyor

Command	Report("Shift");
Privilege	5
Comment	An Operator with Privilege classification 5 can print the report

Not every system element needs a privilege classification. At least one command needs to be issued by users, a command to log in to the system:

Command	LoginForm();
Privilege	
Comment	A blank Privilege (or Privilege 0) means that the command has no classification - it is available to every user who performs this role.

See Also

Roles

Users

Roles

When creating a role, consider the tasks the users who are assigned this role will be expected to perform within the project, and what system elements that user will need to have access to, or be restricted from. Using the areas and privileges defined previously, a number of example roles are outlined below.

> **Note:** Area 0 is assigned by default to every role. This means users can view any system element in Area 0.

Role	Description	Viewable Areas assigned	Privileges
Receiver controller	Needed to monitor the receiver area, print out reports and track progress of supplies. Also will need to control system elements that have a privilege level of 1,4,5.	1,2,3	1,4,5
Operations Manager	Needed to monitor the Processing area of the plant. Needs to be also able to control elements within processing. At times will need to be able to check on the Receivals area.	1,2,3,4,5,6,7,8,9	2,3,4,5

Role	Description	Viewable Areas assigned	Privileges
Despatch Handler	Needs to be able to organize the packing and distribution of final product. Will need to track product production to schedule dispatch tasks for packer.	8,10,11,12	1,2,5,6
Engineer	Engineer is necessary to have access to every area of the plant, and will be able to operate any system element.	1,2,3,4,5,6,7,8,9,10,11,12	1,2,3,4,5,6
Conveyor Operator	Needs to be able to operate the conveyor.	8	1
Mixer Operator	Needs to have access to each type of mixer operation. Will have access to areas 4 to 7 and will be able to operate any system element with a privilege of 2.	4,5,6,7	2
Packer	Needed to be able to package goods	12	6

See Also

Users

Adding Roles

Users

A user can be a Windows or CitectSCADA user. Each user is assigned zero or more roles depending on the activities and processes they will have to operate and monitor. If zero roles are allocated to the user, this is the same as configuring the user with no privileges.

Name	Role
John Smith	Despatch Handler
Jack Smith	Mixer operator
Tom Smith	Engineer
Jerry Smith	Conveyor Operator
Joan Smith	Conveyor operator, Mixer operator

See Also

Configuring CitectSCADA Security

Adding Roles

Configuring CitectSCADA Security

To configure CitectSCADA security, you need to do the following:

- Configuring Areas
- Adding Roles
- Configuring Privileges
- Adding Users

Configuring Areas

When configuring areas within a plant you have the option of labeling areas, grouping areas and naming the group, and granting users view-only access to particular areas.

- Using labels to name areas
- Using groups of areas
- Viewing areas of the plant

Using labels to name areas

It might be easier to remember an area by a meaningful label (name) rather than a number. For example:

Label Name	DespatchAccum
Expression	10
Comment	Label Area 10 as "DespatchAccum"

In this case, "DespatchAccum" could be used whenever area 10 is referred to, for example:

Command	CONVEYOR = 1;
Area	DespatchAccum
Comment	This command belongs to Area 10 (DespatchAccum)

Note: If you leave the Area field blank on a form, the command does not belong to any particular area - it is assigned to every area of the plant.

To label an area:

1. Choose **System** | **Labels**.
2. Enter a **Name** for the label.
3. Enter an expression to be substituted for the label.
4. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Using groups of areas

Using groups of areas

You can group several areas and define a name for the group.

Group Name	Despatch
Association 1	DespatchAccum
Association 2	11
Association 3	12
Comment	Areas 10, 11, 12 = "Despatch"

In the above example, areas 10, 11, and 12 are associated with the name "Despatch". Any command assigned to "Despatch" belongs to areas 10, 11, and 12.

Command	CONVEYOR = 1;
Area	Despatch
Comment	This command belongs to Areas 10, 11 and 12

You can also define a group that includes other groups.

Group Name	Plantwide
Association 1	Receivals
Association 2	Process

Association 3	Despatch
Comment	Associate every area with "Plantwide"

In this example, the name "Plantwide" refers to every area defined in the "Receivals", "Process", and "Despatch" groups.

To define a group of areas:

1. Choose **System** | **Groups**. The Groups dialog box appears.
2. Complete the Groups dialog box.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Groups properties

Groups properties

Use the Groups dialog box to configure properties of groups:

Group Name

The name of the group. You can use this facility, for example, to define multiple areas or multiple devices. Enter a value of 16 characters or less.

After you have defined a group, it can be used anywhere that an individual entity can be used. You can also specify complex groups by defining a group of groups.

Association 1 . . . Association 10

A list of the entities associated with the Group Name. Enter a value of 16 characters or less. An Association can be a number, a name, or another group. You can also specify a range of numbers in the format <n1..n2> for example:

Association 1	4..10

Specifies numbers 4,5,6,7,8,9,10.

You can also define a group of devices to be accessed with a single name, for example:

Group Name	AlarmInfo
Association 1	AlarmPrint
Association 2	AlarmLog
Association 3	AlarmDBF

In this case, when the group name (AlarmInfo) is used as a device, the information is sent to three devices - AlarmPrint, AlarmLog, and AlarmDBF.

Comment

Any useful comment. Enter a value of 48 characters or less.

Viewing areas of the plant

You might need to provide a user access to information from other areas of the plant even though you do not want that user to control any of the processes within those areas. For example, the user may need to monitor the processes in one area as they might directly affect another area.

In the following example, John Smith has been assigned the role of Despatch Handler and as such has been granted control of:

- System elements located in **Despatch**, with a privilege level of **1**; and
- System elements located in **DespatchAccum**, with a privilege level of **4**.
- View-only access Plantwide to track when product is due to arrive in Despatch for packing and distribution.

User Name	J Smith
Global Privilege	
Viewable Areas	Plantwide
Areas for Priv 1	Despatch
Areas for Priv 2	
Areas for Priv 3	
Areas for Priv 4	DespatchAccum
Areas for Priv 5	
Areas for Priv 6	
Areas for Priv 7	
Areas for Priv 8	
Comment	Login for John

Alternatively, you could restrict users access to a group of areas (for example, "Receivals") or to a single area (for example, 12).

Adding Roles

Create a role for those people or groups of people you want to use your system. When creating a role you determine what permissions (privileges and areas) to set for each based on the tasks the user assigned that role needs to be able to perform within the project and plant.

To add a Role record:

1. Choose **System | Roles** to display the Roles dialog box.
2. Complete the Roles dialog box.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

Use the Roles dialog box to define properties for your roles.

Role Name

Enter a value of 16 characters or less, for example "Operator". Role Names are restricted to using the same syntax as Tag names. See Tag name syntax.

Windows Group Name (Users using Windows Authentication only)

Enter the name of the group that you intend link to the Windows security group. Verify that this name is same as the group name in Windows which you want the role link to. It can contain up to 254 uppercase or lowercase characters. Verify that only use the characters that are allowed for Windows group account name in Windows.

The Windows group name can include a domain name or a local computer name in the format of "domainname\operator", or "localcomputername\operator". If either are specified in the group name CitectSCADA runtime will only validate for the groups on the server specified in the name or the local computer.

Comment

Any useful comment. Enter a value of 48 characters or less.

Global Privilege

The privilege assigned globally to the role. Enter a value of 16 characters or less.

In the privilege field you can separate numbers with commas or you can enter a range separated by two periods Ie.g. 1..8

As you configure your system, you can assign privileges to the various elements, such as graphics objects, alarms, accumulators, commands, and so on. For example, a role with a Global Privilege of 3 will be able to issue any command that is assigned a privilege of 3, or action any alarm with a privilege of 3, or click any button that is assigned a privilege of 3, etc. Unless you are using areas, if you do not specify a global privilege, the role cannot access any command with a privilege assigned.

> **Note:** (For users using windows authentication) When you have completed the fields in this dialog and if you have not already done so, add the users to the group in Windows security that you want to have the privileges of this role.

See Also

Adding groups and users in Windows security.

Additional fields on this dialog using extended forms (press **F2**).

Additional Fields

Viewable Areas

The areas the user assigned the associated role is permitted to view. Enter a value of 16 characters or less.

> **Note:** Do not set Viewable Areas in conjunction with Global privileges, as global privileges give roles view access to areas automatically.

Remember, you need to still assign privileges to the elements in these viewable areas, such as graphics objects, alarms, accumulators, commands, etc. If you do not, the user will have full access to them. For example, if you do not assign a privilege to a command in one of these areas, the user will be able to issue it regardless if you want them to or not.

To make an element (such as a button on a expression) view only for a particular user, assign it an expression and a privilege. Add the area to the user's list of Viewable Areas, but don't give the user the necessary privileges in that area (or the necessary global privilege).

Multiple areas can be defined using groups.

If you do not specify “Viewable Areas”, the user will have viewable access to area 0. See Privilege and Area combinations for more information.

Areas for Priv 1 . . . Priv 8

The privileges (by area) assigned to the user. Enter a value of 16 characters or less. Using this combination of areas and privileges, you can assign a user different privileges for different areas. For example, users assigned a role with privilege class 6 in areas 29 and 30 will only have access to commands in those areas that require privilege class 6.

In the privilege field you can separate numbers with commas or you can enter a range separated by two periods Ie.g. 1..8

Note: In assigning a privilege to an area, you are making that area viewable to users assigned that role.

If you do not specify areas with associated privileges, access is defined by Viewable Areas or Global Privileges only

Entry Command

A Cicode command that is executed when the user assigned this role logs in. You can use any Cicode command or function. Enter a value of 254 characters or less.

Exit Command

A Cicode command that is executed when the user assigned this role logs out. You can use any Cicode command or function. Enter a value of 254 characters or less.

Configuring Privileges

When configuring privileges you have the option of changing the default from non-hierarchical to hierarchical. You may also need to assign privileges to a specific area or multiple areas and as such need to have an understanding of how certain combinations of privileges and areas will affect the security of your project.

- Using hierarchical privilege
- Implementing system security
- Privilege and Area combinations
- Using multiple areas and privileges

Using hierarchical privilege

By default, privileges are non-hierarchical (i.e. users with privilege 3 only have access to commands with classification 3).

When privileges are changed to hierarchical (using ini parameter [Privilege]Exclusive), privilege 1 becomes the lowest and 8 is the highest privilege. This means users with privilege 3 have access to commands with privilege classification 3, 2, and 1, whilst to allocate every privilege, you only need to specify privilege 8.

Global Privilege	8

Implementing System Security

Each of your system elements (objects, alarms, reports, accumulators, etc.) can be assigned a privilege level and allocated to a specific area. For a user to be able to acknowledge an alarm, for example, the role they are assigned, need to have access to the correct area, and have the necessary privileges that match the alarm.

Command	CONVEYOR = 1;
Privilege	1
Area	8
Comment	This command belongs to Area 8, and requires privilege 1

In this simple example, an operator without privilege 1 in Area 8 will not be able to issue the command.

See Also

Viewing areas of the plant

Privilege and Area combinations

Outlined below are four general rules regarding the use of privileges and areas within CitectSCADA.

1. Global privileges apply to every area.
2. Assigning a privilege to an area within a role, means any user assigned that role will gain viewable access to that area automatically. However the user can only operate system elements in that area that have a matching privilege. As a result of the first rule, if users are assigned a global privilege they will also be able to view every area.
3. Area 0 includes every privilege a role may have been assigned in other areas. In other words if granted a privilege 3 for use in another area that role can also control those system elements in Area 0 that have a privilege set as 3.
4. All users can view Area 0.

These rules will assist you in understanding how the various privilege and area combinations between system elements and roles will affect your security. The table below outlines numerous scenarios, and the resulting security for a simple on/off button.The first two columns Area and Privilege refer to the button.

Area	Priv	Role	Area	Priv	Security
No	No	Conveyor Operator	No	No	Operator can view and control the system element.
No	Yes	Conveyor Operator	No	No	Can view the system element but cannot operate it as role does not have the necessary privilege
No	Yes	Conveyor Operator	No	Yes (matching)	Can view the system element and control it as role been granted the matching global privilege. Role will be able to control those system elements that also have the matching privilege in other areas of the plant.
Yes	No	Conveyor Operator	No	No	Role cannot view the system element, as it is no longer assigned to Area 0.
Yes	No	Conveyor Operator	Yes (matching)	No	Role can view and control the system element, as no privilege restriction has been set.
Yes	Yes	Conveyor Operator	Yes (matching)	Yes (not matching)	Can view the system element in the relevant area but cannot operate it as role does not have the necessary associated privilege.
Yes	Yes	Conveyor Operator	Yes (matching)	Yes (matching)	Can view the system element and control it within the relevant area, as role has been

Area	Priv	Role	Area	Priv	Security
					assigned a matching associated privilege.

See Also

Using multiple areas and privileges

Using multiple areas with privileges

By combining area and privilege restrictions, you can select what control an operator has within a specific area. You can still assign privileges to each of your operators without using areas - to allow them access to the entire plant (global privileges), but by combining Areas and Privileges, you add an extra level of flexibility.

User Name	J Smith
Global Privilege	1,2
Viewable Areas	
Areas for Priv 1	
Areas for Priv 2	
Areas for Priv 3	Despatch
Areas for Priv 4	DespatchAccum
Areas for Priv 5	DespatchAccum, 11
Areas for Priv 6	
Areas for Priv 7	
Areas for Priv 8	
Comment	Login for John

In this first example, John Smith has been assigned the role of Despatch Handler. This role has global privileges 1 and 2. Privilege 3 in the "Despatch" areas (10, 11, and 12), privilege 4 in the "DespatchAccum" area (10) and privilege 5 in areas 10 and 11. This means he can:

- Due to rule 1 and 2, view evry area of the plant.
- Due to rule 1 control every system element in the plant with a value of 1 and 2.
- In area 10, control system elements with a privilege level 1, 2, 3, 4 or 5.
- In area 11, control system elements with privilege level 1, 2, 3 or 5.

Also, in this example, Groups and Labels have been used to make the security configuration intuitive.

Example 2:

User Name	J Smith
Global Privilege	
Viewable Areas	7,8,9
Areas for Priv 1	
Areas for Priv 2	
Areas for Priv 3	Despatch
Areas for Priv 4	DespatchAccum
Areas for Priv 5	DespatchAccum, 11
Areas for Priv 6	
Areas for Priv 7	
Areas for Priv 8	
Comment	Login for John

In this second example John Smith has been assigned the role of Despatch Handler. This role has no global privileges. Viewable areas, (7,8,9). Privilege 3 in the "Despatch" areas (10, 11, and 12), privilege 4 in the "DespatchAccum" area (10) and privilege 5 in areas 10 and 11. This means he can:

- View areas 7,8,9,10,11,12
- In area 10, control system elements with a privilege level 3, 4 or 5.
- In area 11, control system elements with privilege level 3 or 5.
- In area 12, control system elements with privilege level 3.
- Due to rules 3 and 4, control system elements with privilege level 3,4,5 in Area 0

See Also

Adding Users

Adding users

For each person you want to have access to your project you need to add their user information into the system.

To add a user:

1. Choose **System** | **Users**. The Users dialog box appears.
2. Complete the Users dialog box.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

Use the Users dialog box to define properties for your users.

User Name

The user's name. Enter a value of 16 characters or less. You can assign a user record for a single user, for example:

User Name	JackSmith
User Name	JohnSmith

Each operator needs to enter the **User Name** and **Password** to use the system.

User Names are restricted to using the same syntax as Tag names. See Tag name syntax.

Full Name

The full name of the user or class of user. Enter a value of 32 characters or less. This name is used as a comment and for display in alarm logs and command logs.

Password

The user's password. Enter a value of 36 characters or less. When you enter the password, an asterisk (*) will display for each character entered. When you save the user record, the password will be encrypted before it is saved to the Users.dbf.

Each operator needs to enter the **User Name** and **Password** to use the system.

Use the `[General]PasswordExpiry` parameter to specify when the password will expire.

Confirm Password

Re-enter the user's password to confirm the text entered in the **Password** field. Enter a value of 36 characters or less. If the contents of the **Password** and **Confirm Password** fields are different when the record is saved, a message will be displayed that indicates a mismatch and invites you to try again.

Type

The generic type of user. Enter a value of 16 characters or less. For example:

Type	Operator
Type	Supervisor
Type	Manager

The Type field is used in configuration only to specify a class of user that can then be used as the basis for creating new users in runtime via the UserCreate() cicode function. When this function is run it displays a form where you can select the user Type. When you do this, your new user will inherit the properties of the chosen user class record that you have already created. In doing this it uses the first user record with a matching Type value.

In configuration, decide what user classes you need. Each class (or Type) would contain any specific Global Privilege, Viewable Areas and Areas for Privilige and Entry and Exit Commands values . Maintain just one user record for each class and then base other individual users on this. If you create new users in configuration in the usual way by using Add, Replace technique, then you will get multiple user records with the same Type field value. Whilst this will not directly cause a problem for indiviual records it could confuse development or provide scope for inconsistencies and unexpected behaviour if other field values are changed. When adding records in configuration, it is therefore recommended that you remove duplicate Type values from additional user records.

Roles

Each user is assigned roles. The Roles field will accept zero or more comma-separated role names. If zero roles are specified for the user, this is the same as configuring the user with no privileges.

Note: When a Windows or CitectSCADA user who is linked to multiple roles logs onto runtime the privileges and areas that the user will be assigned are the combined privileges of the linked roles.

Comment

Any useful comment. Enter a value of 48 characters or less.

Notes: To login a user, you need to use the `Login()` or `LoginForm()` Cicode functions.

User records and project restoration

For those users not using windows authenticated security if you restore a project from a backup, or install a new project from a compiled offline master, the user records are reset to match those originally configured in the project. If the runtime user creation, password change ability, or password expiry functions are used, the runtime details might be thrown out of synchronization with master offline projects.

Here, you need to have procedures in place to use the current `Users.dbf` file (which is running live in the plant) when offline project compilations are performed. This minimizes the likelihood of either deleting users created at runtime, or of having expired user records locked when a new system is deployed and run up.

Note: Online changes arising from user creations and modifications are reflected only in the local `_Users.rdb` and `Users.dbf` files. Perform user administration activities on a central node so that user records remain synchronized across a distributed network. Other nodes will use the Copy= functionality in CitectSCADA or custom engineered database replication.

See Also

Adding users

Using CitectSCADA integrated with Windows Security

In CitectSCADA you have the ability to incorporate CitectSCADA users and security options with the standard Windows security system. You can still use the existing CitectSCADA security if you prefer to define users in the project and logon to CitectSCADA runtime.

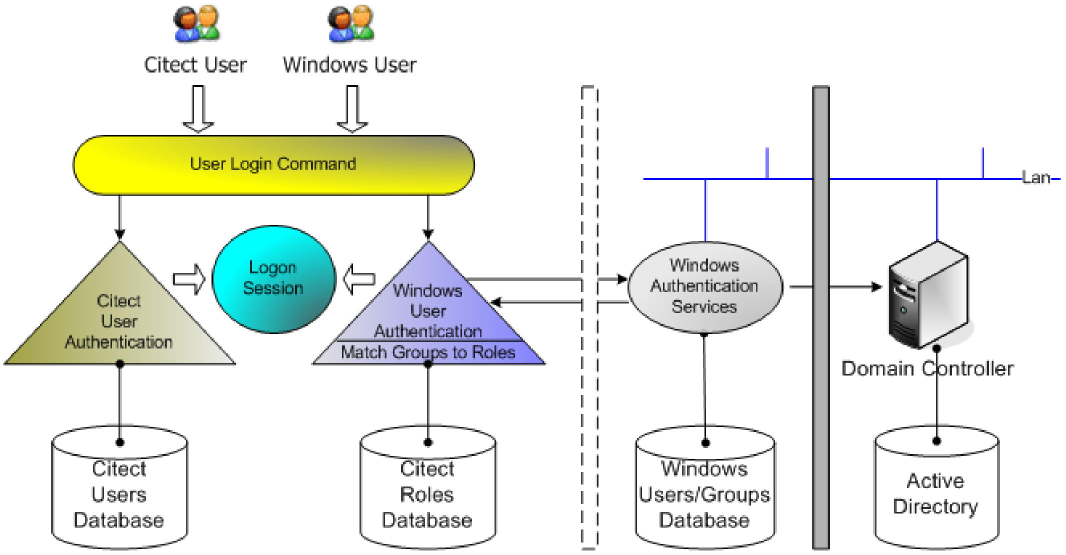
Using the integrated Windows security feature, the Windows user can logon to CitectSCADA runtime with runtime privileges and areas configured within the project. For a Windows user to be able to logon to runtime it needs to be linked to a CitectSCADA "Role" which is defined in the project with associated privileges

In order to link a Windows user to a CitectSCADA role add the "Role" that specifies the Windows group of which the Windows user is a member. The SCADA machines have to be added/belong to the domain that has the Windows users.

The pre-existing AutoLogin capability has also been extended to include the client, when the user is a Windows user, having an associated Citect role. In order to invoke this functionality for a Windows user you need to set the [Client]AutoLoginMode parameter in the Citect.ini file.

A CitectSCADA user will always take priority over a Windows user when logging in at runtime if the user is also included as a Windows user. However if a valid CitectSCADA user login does not succeed for some reason, the Windows user credentials will not be checked and an alert will be generated to advise that the login was not effective.

Conceptual diagram of CitectSCADA security and Windows security



Multi Signature support

When a Windows user is logged on to a runtime system with the associated privileges and areas of the role to which the user belongs, there are times when a higher level authorization is necessary for the user to perform certain actions. When this situation occurs the MultiSignatureForm Cicode function can be displayed through Cicode to allow authorization of an operation by another user who has the necessary level of privilege.

See Also

Adding Roles

Using Security

Adding Groups and Users

The Windows groups and users are defined by a Windows administrator or an authorized power user through the Windows administration function, as distinct from CitectSCADA administration.

Domain groups and users are defined or created on the domain server by the domain administrator. Local groups and users are defined or created on the local computer by the local administrator. To link a Windows user to a CitectSCADA role the user needs to be a member of the Windows group and the name of the group has to be same as the Windows group name specified in the CitectSCADA role.

For information on how to add groups and users to Window security, refer to the Windows documentation appropriate to your operating system.

A Windows user need not be a CitectSCADA user. However if a Windows user is added to the Windows group that is linked to a CitectSCADA role, then that Windows user will have the privileges as are assigned to the role.

The Windows administrator can control which Windows user can or cannot login to runtime by choosing whether to add the user to the linked Windows groups.

See also

Adding Roles

Scenarios and Usage

The following scenarios show how a user will be allowed to negotiate access to a CitectSCADA runtime system in various situations when using Windows groups and CitectSCADA roles.

Local User Login

When authenticating Windows users, if the Role|Group Name does not contain a domain path, then any domain will be used to authenticate. Therefore, specify a domain.

Domain Login

When authenticating Windows users and the Role|Group Name contains a domain name, windows will attempt to authenticate the Windows domain users and user groups. If the domain controllers are unavailable, then cached credentials and Citect group names will be used if available.

> **Note:** Cached credentials are not supported on the Web Client.
> Windows 2000 will only utilize cached credentials if the user is logged on with SE_TCB_NAME privilege.

Local Client Authentication

When a CitectSCADA Windows login is performed on a Control Client or View-only Client that is part of a domain, the client itself is responsible for authenticating with the domain. The CitectSCADA server only verifies the account exists – it does not perform the authentication.

Remote Client Authentication

When a CitectSCADA Windows login is performed on a remote client that is part of a domain, or a trusted domain, the client itself is responsible for authenticating with the domain. The CitectSCADA server only verifies the account exists – it does not perform the authentication. Essentially this mechanism is the same as a local client authentication.

Web Client Authentication

When a CitectSCADA Windows login is performed on a web client that is not a member of the configured domain, the server is responsible for authenticating the user on the domain. No local Windows authentication occurs on the web client machine. No auto-login can occur in this situation.

Multiple Domain Authentication

When a CitectSCADA Windows login is performed on a Control Client or View-only Client that is part of a domain, the client itself is responsible for authenticating with the domain. When that client has access to CitectSCADA servers on more than one domain, it is possible that the client will only be authenticated on one of the domains.

CtAPI Authentication

In this release CtAPI does not support operations with Windows security.

Authenticating a Trusted Network

Computers that are part of the trusted SCADA network will now use a shared secret password to authenticate each other. This machine password is configured using the Computer Setup Wizard, and the INI parameter [Client]PartofTrustedNetwork.

If the password has not been configured, and is necessary at runtime due to the INI setting, the servers will become inoperable. An appropriate message will then be recorded in the syslog.

See Also

Running the Computer Setup Wizard

Setting the Super User Password

You configure the Super User password using the Computer Setup Wizard. You will only set this password once, and only for a system running in multi-process mode. Computer Setup Wizard will automatically configure the [Server]AutoLoginMode INI parameter in line with the user's settings.

See Also

Authenticating a Trusted Network

Chapter: 12 Configuring Your System

Before you run your project you will need to configure each computer in your CitectSCADA system. Configuration information is stored on each machine in a `Citect.ini` file based on your installation, database configuration and compiled project. Configuration is done with the Computer Setup Wizard.

The Computer Setup Wizard contains a series of pages allowing configuration of the computer specific settings including:

- The role the computer has in the system network
- The project being run
- The CPU Configuration
- The CitectSCADA Events enabled for each component
- The Cicode run for each component on startup
- The cluster configuration
- The security settings applied

The wizard uses the configuration stored in the project databases to provide information to you. Selected options are written to the `Citect.ini` file.

The wizard needs to be run on each computer in your system to appropriately configure CitectSCADA for each particular machine. Run after compiling your project and as the last step before running the system.

See Also
Running the Computer Setup Wizard

Running the Computer Setup Wizard

To start the Citect Computer Setup Wizard:

1. Open Citect Explorer.
2. In the project list area, select **My Projects** - designated by a computer icon.
3. Double-click the **Computer Setup Wizard** icon, or choose **Tools** | **Computer Setup Wizard**. The Citect Computer Setup Wizard is displayed.
4. Select **Express Setup** or **Custom Setup**.

The pages that are displayed depend on the configuration of the machine and include:

- Project Configuration
- Computer Role Configuration
- Network Model
- Server Password Configuration
- Server User Configuration
- Internet Server Configuration
- Alarm Configuration
- Reports Configuration
- Trends Configuration
- CPU Configuration
- Events Configuration
- Startup Functions Configuration
- Cluster Connections Configuration *
- Control Menu Security Configuration *
- Keyboard Security Configuration *
- Miscellaneous Security Configuration *
- General Options Setup *

* Only available in Custom Setup mode.

Each screen of the wizard is described in the sections that follow.

See Also

Project Configuration

Project Configuration

The dialog box for project configuration will vary depending on whether you are configuring a project from a development and configuration environment, or from a runtime only environment.

Development and configuration environment

Select the project to run on this CitectSCADA computer. The Computer Setup Wizard will show you the compiled projects defined in the project list, apart from the include projects.

If there is only one compiled project present, it will be automatically selected. If there are no compiled projects present, an alert message is displayed and the wizard will terminate. If this occurs, return to Citect Explorer and confirm that the necessary project is saved locally and has compiled without errors.

See Also

Computer Role Configuration

Runtime only environment

Before configuring the runtime only environment, and running the computer setup wizard, it is assumed you have transferred a project from the configuration environment, including compiled projects to your local machine. If not refer to Backing up a project for more information.

Using the .CTZ files from your backup included projects and target project, you can restore the project. It is recommended you restore the included project before the target project. The restore project tool is available from the start menu shortcut Runtime Configuration ->Project restore

In the Project Restore dialog :

1. In the Backup file field, browse or enter the name of the project to restore.
2. Under To, the option 'Current project', may be unavailable, this is due to no project having been run on the local machine before. The option 'New project' will be selected to restore a backed up project as a new one.
3. In the Name field, enter a name of the restored project.
4. In the Location field, enter or browse to the location of the project to restore (restore the project under the 'User' folder).
5. Leave the remaining settings as default and click OK.
6. The project will be restored.

You are now ready to run the computer setup wizard available from the start menu shortcut Runtime Configuration ->CitectSCADA.Computer Setup

In the RUN path edit box enter, or use the select button to browse for, the path to the local project that you wish to the client to run.If you enter the run path set the run folder under the ‘User’ Folder.

If you wish to maintain a synchronized local version of an external project select the Enable RUN/COPY deployment check box. Then in the COPY path edit box enter, or use the select button to browse for, the path to the external project that you wish maintain synchronization. This feature uses the [CtEdit]RUN and [CtEdit]COPY commands to maintain synchronization between the two projects.

> **Note**: The Run/Copy mechanism does not transfer custom files such as meta-data, icons, images or runtime DBF files. Therefore, it is recommended deploying a project before Run/Copy to help ensure that those files at least exist. You will still need to redeploy a project, when you want to update the custom files.

See Also

Computer Role Configuration

Computer Role Configuration

Use the Computer Role Setup page to specify the role of the computer running CitectSCADA. Select one of the options described below.

> **Note:** In order to use CitectSCADA's multi-process capabilities, networking needs to be enabled.

Option	Description
Server and Control Client	This computer will be a standalone or networked I/O Server and Control Client. This option is disabled if this computer has no Server components assigned to it to run. Selecting this option enables the **Multi-Process** check box. Select the **Multi-Process** check box to separate your client and server components into individual processes. This option can be used for distributing the components across multiple CPUs. If you leave the **Multi-Process** check box unselected, CitectSCADA will run the client and server components in one process. If the **Multi-Process** check box is selected the [General]MultiProcess parameter in the Citect.ini file is saved with the value 1. If not selected, the parameter is saved with the value 0.
Control Client	This computer will only be a Control Client. This option is disabled if this computer has been assigned a Server component to run. Selecting this option enables the **Full License** check box. Select the **Full License** check box if you want this Control Client to use a full license. This sets the [Client]FullLicense parameter in the Citect.ini file to 1 (the default value).
View-only Client	This computer will only be a View-only Client. This read only option is disabled if this computer has been assigned a Server component to run.

Some of these options may be disabled depending on what servers have been configured to run on this computer. The Computer Setup Wizard cross-references your computer's network identification with the network addresses configured for each server in your project configuration.

See Also

Network Model
CPU Configuration

Network Model

Select the network model to be applied to this CitectSCADA computer. Options include:

- Stand alone (no other SCADA computers))
- Networked (connect to other SCADA computers)

From Version 7.0 CitectSCADA uses TCP/IP to facilitate communications across a network.

Note: TCP/IP address information for Citect servers is configured within the Citect project itself. See Network Address Definitions for more information.

When you complete the Computer Setup Wizard, the chosen network model is written to the `[LAN]` section in the `citect.ini` file; for example:

```
...
[LAN]
TCPIP=1
...
```

See Also

Internet Server Configuration

Configure Server Password

If networking is enabled use this page to configure the machine password. The machine password is used by computers to authenticate each other and create a trusted network between servers. Configuring the password automatically sets the [Client]PartofTrustedNetwork INI parameter.

If a server process exists and networking has been enabled:

The Configure Server Password checkbox is selected and unavailable.

Enter the Password, and confirm the password in the fields provided, before clicking Next.

If a client process exists and networking is enabled:

The Configure check box is unchecked. Select to enable the password fields.

Enter the password and confirm password in the fields provided, before clicking Next

> **Note**: If a server password has already been configured for the machine, the ‘Password’ and “Confirm Password” fields will be pre-filled.

See Also

Configure Server User

Configure Server User

Use this page to define the user to log in for the server processes running on the machine. The user can be the default server user, none (view-only), or a specified user. The Computer Setup wizard will automatically configure the [Server]AutoLoginMode INI parameter in line with the user’s settings.

Select 'Specific User' to enable the Configure Server User fields. The fields will remain disabled if either the 'Default Server User' option or 'None' option are selected.

See Also

Configure Server Password

Internet Server Configuration

Select the **This computer is an IDC Server** option to make the computer an Internet Server. To allow communication with a remote Internet Display Client, an Internet Server requires a permanent Internet connection and a static IP address (or hostname).

1. In the **Client Connection Information area**, type the **Internet Server IP address or hostname**. This adds a `Primary=<Internet Server IP address>` entry to the `[DNS]` section of the `citect.ini` file if there was no pre-existing entry. For details, see [DNS]Primary in the Parameter online help.

   > **Note:** To determine the TCP/IP address of the Internet Server computer, choose **Start | Run**. Type **CMD** and press **Enter**. Then at the DOS prompt type **IPCONFIG** and press **Enter**.

2. Type the **Alternate Internet Server IP address or hostname**. This adds a `Standby=<Alternate Internet Server IP address>` entry to the `[DNS]` section of the `citect.ini` file if there was no pre-existing entry. For details, see "[DNS]Standby" in the Parameters online help.

The Internet Display Client automatically connects to this alternate server if connection to the primary server is lost.

Note: that this can only happen automatically if an initial connection has previously been made to the primary Internet Server.

See Also

Alarm Configuration
DNS Parameters

Alarm Configuration

The Alarm Configuration page will only be displayed if this machine is configured as an Alarm Server in the Project Editor.

CitectSCADA has several options available for alarm processing. These are set in the Alarm section of the Citect.ini file. Refer to the Parameters help for information on these parameters.

Option	Description
Alarm scan time	Determines the rate at which alarms are scanned and processed. A value of 500 (the default value) indicates that CitectSCADA tries to process the alarms every 500 ms. However, if CitectSCADA cannot read the alarm data from the I/O Device within 500 ms, the alarms are processed at a slower rate. For example, if it takes 800 ms to read the alarm data from the I/O Device, CitectSCADA processes the alarms every 800 ms. If you select a larger value for the alarm scan time, the alarms server uses less CPU (because it does not need to process the alarm records as often). The amount of data read from the I/O Device is also reduced, so that other processes (Trends, Reports, and the current page) get their I/O Device data more quickly. You can enter any value from 0 to 60000 (milliseconds).
Alarm save period	The period for saving alarm and event data (to disk). You can save alarm and event data periodically so that the data is restored after a planned or unplanned system shutdown. The smaller the period, the greater is the load on the system.

Summary length	The maximum number of alarm summary entries that can be held in memory. You can view these alarm summary entries on the alarm summary page. Each event requires approximately 300 bytes of memory, including the length of the comment. 10,000 events require 3 to 4 MB of memory. If you use many events, have enough memory to keep them in RAM.
Summary timeout	The length of time that alarm summary entries remain in the alarm summary queue.
Primary alarms server save path	The path to the primary save files. CitectSCADA uses two save files for each alarms server, ALMSAV.DAT and ALMINDEXSAVE.DAT. The save primary path is the directory where the primary alarms server creates its save files. When restoring the files, the most recent (of the primary and secondary) save files will be used.
Standby alarms server save path	The path to the secondary save files.

To minimize the chance of conflicts between alarm files used by multiple Alarm Servers from different clusters running on the same machine, the alarm files have a dynamic naming convention based on the following format:

```
<ProjectName>_<ClusterName>_<filename>.DAT
```

See Also

Reports Configuration

Reports Configuration

The Reports Configuration page will only be displayed if this machine is configured as a Reports Server in the Project Editor.

> **Note:** For a networked computer to be a Reports Server it needs to also be the I/O Server or needs to be able to communicate with the I/O Server on the network.

CitectSCADA has several options available for report processing:

Option	Description

Startup report	Defines the name of the report to run when CitectSCADA starts up.
Inhibit triggered reports on startup	For example, you might have a report that is triggered off the rising edge of a bit on startup. The Reports Server notices the bit come on, and runs the report. If this option is checked, the Reports Server does not run this report until it has read the I/O Devices a second time.
Run reports concurrently with primary Reports Server	Enables or disables tandem processing of reports. If this server is the standby Reports Server, it can process every report in tandem with the primary server, or it can remain idle until called.

See Also

Trends Configuration

Trends Configuration

The Trends Configuration page will only be displayed if this machine is configured as a Trends Server in the Project Editor.

> **Note:** For a networked computer to be a Trends Server it needs to also be the I/O Server or needs to be able to communicate with the I/O Server on the network.

CitectSCADA has one option available for trend processing:

Option	Description
Inhibit triggered trends on startup	You might have a trend that is triggered off the rising edge of a bit on startup. If this option is enabled, the trends server does not display the trend until it has read the I/O Devices a second time.

See Also

CPU Configuration

CPU Configuration

The CPU Setup page is used to assign client and server components to specific processors in a multi-processor machine.

This page lists each component's full name, including the cluster to which it belongs, the priority and the CPU assignment. If the Multi-process option was not selected on the Computer Role Configuration page there will only be one entry listed, either Client or Client and Servers. If the Multi-process option was selected, you have the option of selecting specific CPUs for the Client, I/O Server, Alarm Server, Trends Server and Reports Server.

To assign a CPU to a component:

1. Select one or more components from the list (hold the **Ctrl** key down to select multiple components).
2. Click **Modify**.
3. Type the number of the CPU and click **OK**.

When you complete the Computer Setup Wizard, the CPU assignations are written to each component section in the `Citect.ini` file; for example:

```
...
[Alarm.Cluster1.AlarmServer1]
CPU=1
Clusters=Cluster1
...
[Trend.Cluster1.TrendServer1]
CPU=2
Clusters=Cluster1
...
```

See Also

Events Configuration

Events Configuration

Events are used to trigger actions, such as a command or set of commands. For example, an operator can be notified when a process is complete, or a series of instructions can be executed when a process reaches a certain stage. Select the **Enable Events on this computer** check box if events are to be enabled on this CitectSCADA computer.

The Events Setup page lists each component's full name, including the cluster to which it belongs, alongside a list of events that can be enabled for each component. If the Multi-process option was not selected on the Computer Role Configuration page there will only be one entry listed, either Client or Client and Servers. If the Multi-process option was selected, you have the option of enabling events for each component on this computer.

Note: The Computer Setup Wizard only displays named events from the selected project. If you are using events in included projects you will need to edit your `Citect.ini` file to add these under the `[Events]` section header.

Note: Events named 'Global' or events with no title will not appear as these are global events. These events will run on computers that have events enabled. These events will run in the client process.

To enable an event for a component:

1. Select the component from the list.
2. Select the events you want to enable for that component, or click **Enable All** or **Disable All**.
3. Click **Next** when finished.

When you complete the Computer Setup Wizard, the events are written to each component section in the `Citect.ini` file; for example:

```
...
[Alarm.Cluster1.AlarmServer1]
CPU=1
Clusters=Cluster1
Events=CSV_AlarmClient
...
[Trend.Cluster1.TrendServer1]
CPU=2
Clusters=Cluster1
Events=CSV_TrendXClient,CSV_TrendXServer
...
```

See Also

Startup Functions Configuration

Startup Functions Configuration

The Startup Functions Setup page is used to define the Startup Cicode that is executed by each CitectSCADA process.

The Startup Functions Setup page lists each component's full name, including the cluster to which it belongs, the priorities of the components and the startup function assigned to each component. If the Multi-process option was not selected on the Computer Role Configuration page there will only be one entry listed, either Client or Client and Servers. If the Multi-process option was selected, you have the option of assigning startup functions for each component on this computer.

If the StartupCode parameter value for a process is invalid, the CitectSCADA Runtime Manager will simply ignore it on start up.

To assign a startup function to a component:

1. Select the component from the list. To select multiple components, hold down the Ctrl key as you select each item.
2. Click **Modify**.
3. Type the name of the Cicode function you want to call on startup for that component.
4. Click **OK**.

When you complete the Computer Setup Wizard, the events are written to each component section in the `citect.ini` file; for example:

```
...
[Alarm.Cluster1.AlarmServer1]
CPU=1
StartupCode=alarmServerStartup
...
[Trend.Cluster1.TrendServer1]
CPU=2
StartupCode=trendServerStartup
...
```

See Also

CPU Configuration
Cluster Connections Configuration

Cluster Connections Configuration

The Cluster Connections Setup page is used to specify the clusters that each component connects to on startup. This controls what data streams a component can see in the system.

The Cluster Connections Setup page lists each component's full name, including the cluster to which it belongs, the priorities of the components and the clusters assigned to each component. If the Multi-process option was not selected on the Computer Role Configuration page there will only be one entry listed, either Client or Client and Servers. If the Multi-process option was selected, you have the option of assigning clusters for each component on this computer.

By default, every component will connect to every cluster unless otherwise modified.

If the Clusters parameter value for a process is invalid, then the CitectSCADA Runtime will simply ignore it on startup.

To assign a cluster to a component:

1. Select the component from the list. To select multiple components, hold down the Ctrl key as you select each item.
2. Click **Modify**.
3. Select the clusters you want the component to connect to on startup.
4. Click **OK**.

When you complete the Computer Setup Wizard, the clusters are written to each component section in the `Citect.ini` file; for example:

```
...
[Alarm.Cluster1.AlarmServer1]
CPU=1
Clusters=Sydney
...
[Trend.Cluster1.TrendServer1]
CPU=2
Clusters=Sydney,Tokyo
...
```

See Also

Implementing Clustering

Control Menu Security Configuration

The CitectSCADA window property options allow you to control an operator's access to system features.

This allows for flexibility with system security at run time.

Option	Description
CitectSCADA configuration environment on menu	Allows the operator to use the control menu (top left-hand icon) to access the Citect Editor, Project Editor, Graphics Builder, and Cicode Editor from CitectSCADA at run time. Disabling this provides better security.
FullScreen	Allows the operator to set whether pages will be displayed in fullscreen or restored state. When checked “FullScreen” will set the ini parameter [Animator]FullScreen to 1. If “FullScreen” is set.
Show title bar	Allows the operator to set whether pages will be displayed in fullscreen mode with the title bar. The [Animator]FullScreen ini parameter is set as follows: 0 if “Fullscreen” is unchecked; 1 if “Fullscreen” is checked and “Show title bar” is unchecked; 2 if both “Fullscreen” and “Show title bar” are checked. “Show title bar” cannot be modified if the option“Fullscreen” is unselected.

Shutdown on menu	Allows the operator to use the control menu (top-left icon) to shut down CitectSCADA at runtime. The shutdown is not password- or privilege-protected. Disabling this provides better security.
Kernel on menu	Allows the operator to use the control menu (top left icon) to display the CitectSCADA Kernel at run time. Disabling this provides better security.

See Also

Keyboard Security Configuration

Keyboard Security Configuration

Windows has a set of standard task-swapping shortcut commands that are (optionally) supported by CitectSCADA at run time. This option allows the Alt-Space Windows command to be enabled or disabled at run time. Alt-Space provides access to the Windows control menu (even if the title bar has been disabled).

> **Note:** The ability to disable Alt-Escape, Ctrl-Escape and Alt-Tab is not currently available.

See Also

Miscellaneous Security Configuration

Miscellaneous Security Configuration

Some standard Windows features may interfere with the secure operation of your system. Use the Miscellaneous Security page to disable these features.

Option	Description
Inhibit screen saver while CitectSCADA is running	Stops the screen saver from blanking out important screens that have to be always visible. Alternatively the screen saver password can add additional security features
Display Cancel button at startup	Provides the ability to stop CitectSCADA from starting up automatically. Automatic startup is a potential security concern

See Also

General Options Setup

General Options Setup

Use the General Options Setup page to specify general options.

Option	Description
Data Directory	The directory where the CitectSCADA data files are located. The data files are the files that are generated at run time: trend files, disk PLC etc.
Backup project path	The backup directory that is used if a runtime database cannot be located (due to inoperative hardware or a file that has been moved, corrupted, or deleted).
Startup page	The Page Name of the graphics page to display when CitectSCADA starts up.
Page scan time	The delay (in milliseconds) between updating a graphics page and starting the next communications cycle. The Page Scan Time sets the default for how often your graphics pages are updated. When a page is updated, relevant data (variable tags etc. represented on the graphics page) is scanned to determine if field conditions have changed. This setting is overridden by the Scan Time value specified in Page Properties (if applied). A value of 250 (the default value) indicates that CitectSCADA will try to update the page every 250 ms. However, if CitectSCADA cannot read the entire data from the I/O Device within 250 ms, the page is processed at a slower rate. For example, if it takes 800 ms to read the data from the relevant I/O Device, CitectSCADA processes the page every 800 ms. Under some conditions, you might want to slow the update of your pages to reduce the load on the I/O Servers. By reducing the page scan time, you allow more communication bandwidth to other CitectSCADA tasks or Clients. For example, you might want fast response on your main operator computers, while slowing the response time on manager computers. You can enter any value from 0 to 60000 (milliseconds).

See Also

Finish

Finish

Click **Finish** to save the setup to the `Citect.ini` file, **Cancel** to quit the wizard without saving, or **Back** to navigate to a page that requires adjusting.

Chapter: 13 Implementing Clustering

Once you have designed the clustering for your system, including the configurations of servers you need, you can proceed to implement that design. You will need to configure:

- Cluster Definitions
  Each cluster needs to be defined by giving it a unique name in the project.
- Network Address Definitions
  Each physical server in your system needs to be identified with a unique name and IP address.
- Alarm Server Definitions
  Each Alarm Server needs to be named, and assigned to a cluster and physical server. Identify each server as Primary or Standby.
- Reports Server Definitions
  Each Reports Server needs to be named, and assigned to a cluster and physical server. Identify each server as Primary or Standby.
- Trends Server Definitions
  Each Trends Server needs to be named, and assigned to a cluster and physical server. Identify each server as Primary or Standby.
- I/O Server Definitions
  Each I/O Server needs to be named, and assigned to a cluster and physical server. Identify each server as Primary or Standby.

See Also

Rules of Clustering
Assigning tags to a cluster at Runtime

Rules of Clustering

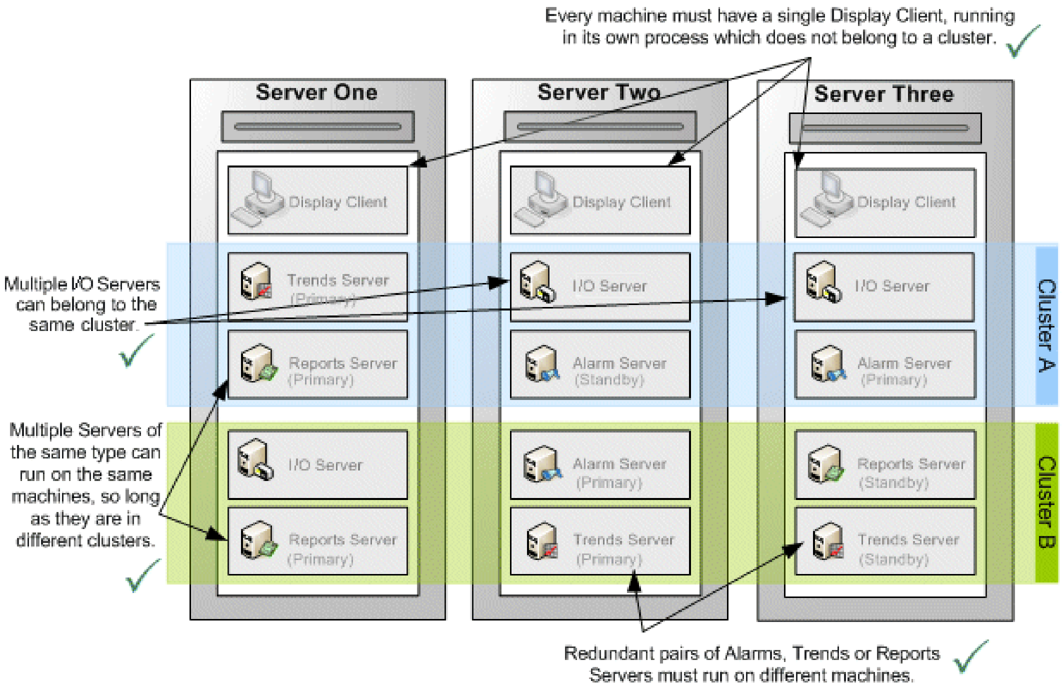
When configuring CitectSCADA the following clustering rules apply:

- Each cluster to have a unique name.
- Each server component to have a unique name.
- Each server component needs to belong to one cluster.
- Each cluster can contain only one pair of redundant Alarm Servers. They need to reside on different machines.
- Each cluster can contain only one pair of redundant Reports Servers. They need to reside on different machines.

- Each cluster can contain only one pair of redundant Trends Servers. They need to reside on different machines.
- Each cluster can contain an unlimited number of I/O Servers.

There are countless variations in how a clustered system can be configured. The most appropriate configuration will depend on the **requirements** for the solution to be deployed and the **environment** in which it is being deployed. For more information, refer to Typical system scenarios.

The diagram below is an example of a system running with two clusters across three machines. Every server and client component have been deployed in accordance with the clustering rules.



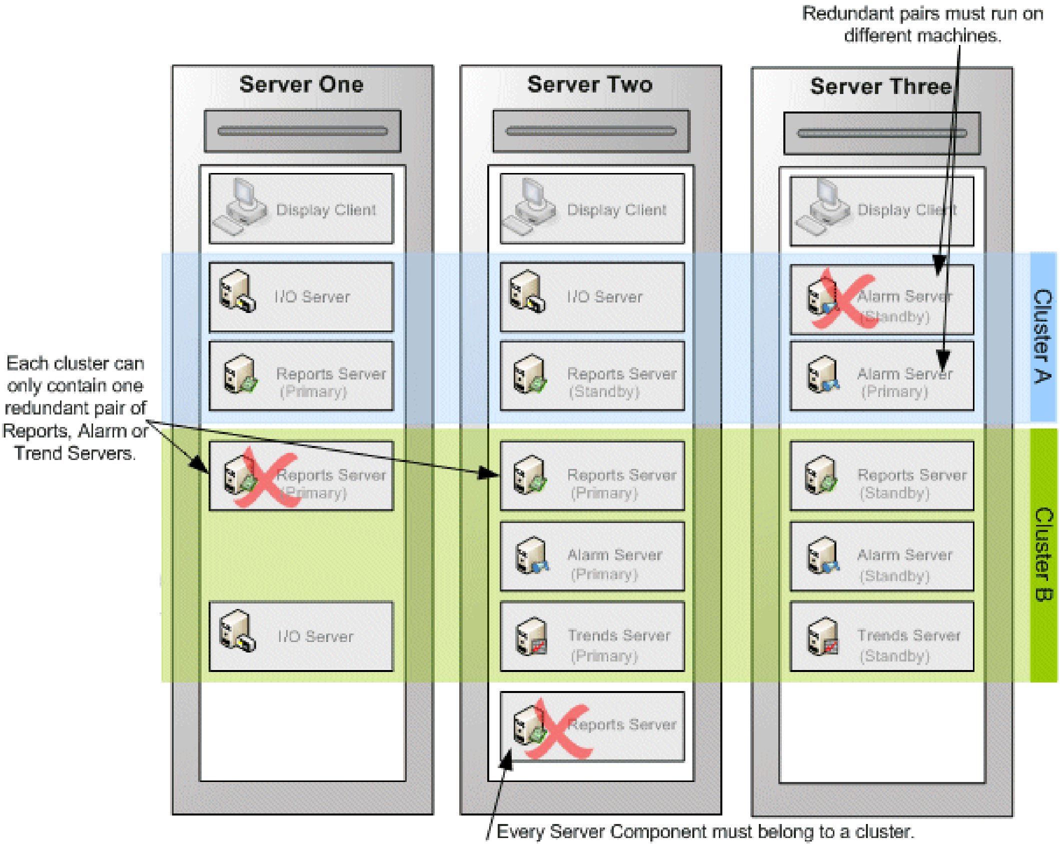
Every machine must have a single Display Client, running in its own process which does not belong to a cluster. ✓

Multiple I/O Servers can belong to the same cluster. ✓

Multiple Servers of the same type can run on the same machines, so long as they are in different clusters. ✓

Redundant pairs of Alarms, Trends or Reports Servers must run on different machines. ✓

The next diagram demonstrates circumstances which do not correctly follow the clustering rules.



The CitectSCADA compiler or the CitectSCADA Runtime Manager detects when the rules of clustering are not being observed and advises the user accordingly.

See Also

About cluster context

Cluster Definitions

See Rules of Clustering for additional information.

To define a cluster:

1. In the Project Editor, choose **Servers** | **Clusters**.
2. In the Cluster dialog box, complete the cluster properties:

Option	Description
Cluster Name	The name of the cluster. The name needs to be unique to the project and not contain spaces.

Comment	Any useful comment. This property is optional and is not used at runtime.

3. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

See Also

Network Address Definitions

Network Address Definitions

To configure a network address:

1. In the Project Editor, choose **Servers | Network Addresses**.
2. In the Network Addresses dialog box, complete the properties:

Option	Description
Name	The name of the machine at the network address being configured. The name needs to be unique to the project and not contain spaces.
Address	The IP address or computer name of the machine or network card being configured. For machines with dual network cards, add a network address for each card in each machine to which you want to communicate. See Network Redundancy for information.
Comment	Any useful comment. This property is optional and is not used at runtime.

3. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

See Also

Alarm Server Definitions

Alarm Server Definitions

Refer to the default server port numbers under Configure Servers.

> **Note**: The Alarm server reuses the Alarm Properties port. As a result alarm properties are now published for configured alarm servers.

To configure an Alarm Server:

1. In the Project Editor, choose **Servers** | **Alarm Servers**.
2. In the Alarm Servers dialog box, complete the properties:

Option	Description
Cluster Name	The name of the cluster to which this Alarm Server will belong. If there is only one cluster defined in the project, you can leave this field blank. The Alarm Server will default to the defined cluster
Server Name	The name of the server. The name needs to be unique to the project and any included projects, and not contain spaces.
Mode	The mode for this server, either Primary or Standby. If this property is left blank, the default value will be Primary. The Primary and Standby servers need to run on different computers, and only one Primary and one Standby can be defined per cluster.
Network Addresses	The network address(es) of the server being configured. To specify dual network connections to the server, use a comma delimited list. See Network Redundancy.
Port	The port this server will listen on. You can leave this field blank if you are running only one Alarm Server on the machine, in which case the default port number will be used.
Comment	Any useful comment. This property is optional and is not used at runtime.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

> **Note**: Servers listen on legacy ports only if the INI parameter [LAN] EarliestLegacyVersion has been enabled.

Option	Description
Legacy Port	Allows legacy connections to the server

3. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

See Also

Reports Server Definitions

Reports Server Definitions

Note the default server port numbers under Configure Servers.

To configure a Reports Server:

1. In the Project Editor, choose **Servers** | **Reports Servers**.
2. In the Reports Servers dialog box, complete the properties:

Option	Description
Cluster Name	The name of the cluster to which this Reports Server will belong. If there is only one cluster defined in the project, you can leave this field blank. The Reports Server will default to the defined cluster
Server Name	The name of the server. The name needs to be unique to the project and not contain spaces.
Mode	The mode for this server, either Primary or Standby. If this property is left blank, the default value will be Primary. The Primary and Standby servers need to run on different computers, and only one Primary and one Standby can be defined per cluster.
Network Addresses	The network address(es) of the server being configured. To specify dual network connections to the server, use a comma delimited list. See Network Redundancy.
Port	The port this server will listen on. You can leave this field blank if you are running only one Reports Server on the machine, in which case the default port number will be used.
Comment	Any useful comment. This property is optional and is not used at runtime.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

> **Note**: Servers listen on legacy ports only if the INI parameter [LAN] EarliestLegacyVersion has been enabled.

Option	Description
Legacy Port	Allows legacy connections to the server

3. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

See Also

Trends Server Definitions

Trends Server Definitions

Note the default server port numbers under Configure Servers.

To configure a Trends Server:

1. In the Project Editor, choose **Servers** | **Trend Servers**.
2. In the Trends Servers dialog box, complete the properties:

Option	Description
Cluster Name	The name of the cluster to which this Trends Server will belong. If there is only one cluster defined in the project, you can leave this field blank. The Trends Server will default to the defined cluster.
Server Name	The name of the server. The name needs to be unique to the project and not contain spaces.
Mode	The mode for this server, either Primary or Standby. If this property is left blank, the default value will be Primary. The Primary and Standby servers need to run on different computers, and only one Primary and one Standby can be defined per cluster.
Network Addresses	The network address(es) of the server being configured. To specify dual network connections to the server, use a comma delimited list. See Network Redundancy.

Option	Description
Port	The port this server will listen on. You can leave this field blank if you are running only one Trends Server on the machine, in which case the default port number will be used.
Comment	Any useful comment. This property is optional and is not used at runtime.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Note: Servers listen on legacy ports only if the INI parameter [LAN] EarliestLegacyVersion has been enabled.

Option	Description
Legacy Port	Allows legacy connections to the server

3. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

See Also

I/O Server Definitions

I/O Server Definitions

Note the default server port numbers under Configure Servers.

To configure an I/O Server

1. In the Project Editor, choose **Servers** | **I/O Servers**.
2. In the I/O Servers dialog box, complete the properties.

Option	Description
Cluster Name	The name of the cluster to which this I/O Server will belong. If there is only one cluster defined in the project, you can leave this field blank. The I/O Server will default to the defined cluster. 16 characters maximum.

Server Name	The name of the server. The name needs to be unique to the project and any included projects, and not contain spaces. 16 characters maximum.
Network Addresses	The network address(es) of the server being configured. To specify dual network connections to the server, use a comma delimited list. See Network Redundancy. 70 characters maximum.
Port	The port this server will listen on. You may leave this blank in which case the default port number will be used. 16 characters maximum.
Comment	Any useful comment. This property is optional and is not used at runtime. 48 characters maximum.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Note: Servers listen on legacy ports only if the INI parameter [LAN]EarliestLegacyVersion has been enabled.

Option	Description
Legacy Port	Allows legacy connections to the server

1. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

Assigning tags to a cluster at Runtime

CitectSCADA includes functionality that allows you to assign the tags on a page to a specific cluster during Runtime.

If the tags on a page exist within a number of different clusters, you can use Cicode to pass a cluster name to the page as it opens. Any tags that do not have a cluster explicitly defined will then be assigned to the specified cluster as the page is launched.

This provides a practical solution for a replicated system, where your project controls a number or identical sites or production lines. As the sites contain the same equipment, the tags representing the hardware architecture will potentially be the same for each. The ability to pass a cluster name to a page at runtime means just a single mimic page is necessary to represent multiple sites, reducing configuration time and simplifying project deployment.

For example, a menu page could be created to launch the mimic page for three identical production lines, each based on the same page called "ProductionLine".

To achieve this, you would use the PageDisplay function to configure the following buttons on your menu page:

Button one	Text	**Production Line A**
	Command	PageDisplay("ProductionLine","Cluster_A")
	Comment	Display the mimic page in the context of production line A
Button two	Text	**Production Line B**
	Command	PageDisplay("ProductionLine","Cluster_B")
	Comment	Display the mimic page in the context of production line B
Button three	Text	**Production Line C**
	Command	PageDisplay("ProductionLine","Cluster_C")
	Comment	Display the mimic page in the context of production line C

In each case, PageDisplay would pass the name of the host cluster to the page, depending on which button is selected. As each production line shares a common architecture, any tags that do not have a cluster explicitly defined will be assigned to the specified cluster as the page is opened.

This functionality is supported by the following Cicode functions:

- PageDisplay
- PageGoto
- WinNew
- WinNewAt

You can also use the PageInfo function to determine the cluster context that has been set for a page.

This functionality extends to any Cicode that may be called from a graphics page. You can write Cicode that only specifies variable tag names, allowing the cluster to be defined by the current context of the page from which it is launched.

If the Cicode is executed by the function TaskNew, you have the option to specify a cluster by setting the ClusterName parameter.

Note: The cluster context for Cicode cannot be changed once the code is running.

Chapter: 14 Building Redundancy Into Your System

Redundancy in CitectSCADA can be defined at many different levels. When building redundancy for your system, it is important to consider the degree of protection necessary to meet your requirements, by:

- Defining how important your processes are;
- Determining the overall likelihood of system downtime, and the important processes and equipment that will take the longest to restore to service;
- Deciding which components you would like to implement redundancy for; and
- Considering design and maintenance consequences of redundancy.

The section covers the following redundancy concepts:

- I/O Server Redundancy
- I/O Device promotion
- Redundancy and Persistence
- Data Path Redundancy
- Network Redundancy
- Alarms, Reports, and Trends Server Redundancy

> **Note:** Using the Computer Setup Wizard will allow you to define the level of redundancy you require by defining the function of each computer (See Running the Computer Setup Wizard).

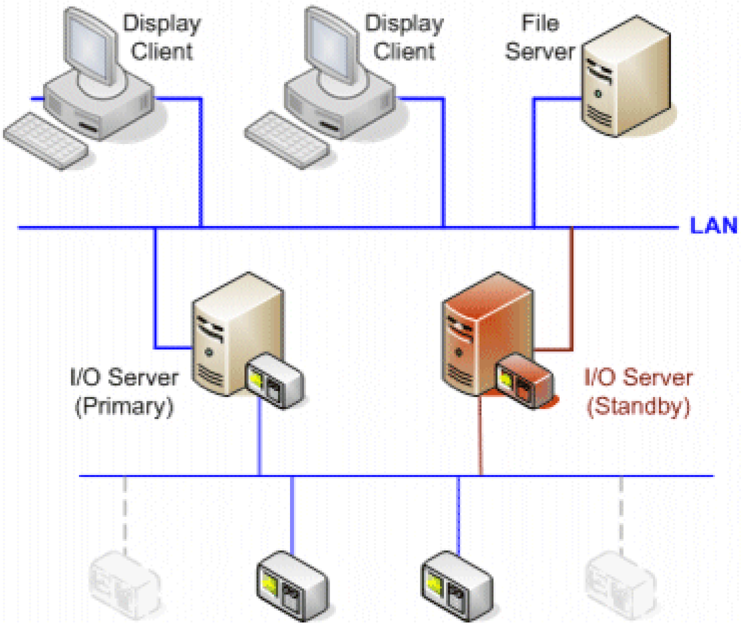
See Also

Alarms, Reports, and Trends Server Redundancy
How CitectSCADA handles file server redundancy
How CitectSCADA handles FTP server redundancy
Redundancy of Standalone Systems

I/O Server Redundancy

Systems with a single I/O Server can be interrupted by the inoperability of a single device or process. If the primary server isn't operating normally, control and monitoring of the system is lost. By introducing a second I/O Server and dedicating it to communicating with the same I/O Devices, the ability of a single device to influence the system as a whole is minimized. You now have a *primary* and *standby* I/O Server in your system where the standby I/O Server will assume operations in case the primary I/O Server becomes inoperative.

The diagram below illustrates the introduction of a standby I/O Server into the existing system. When the system is in operation, both I/O Servers are identically maintained.



> **Note:** Although both I/O Servers are identical, it is important to recognize that the standby server is not duplicating the primary server's functions. If it were, the load on the PLC portion of the network would be double and would significantly reduce performance. Therefore, only the primary server communicates with the PLCs at any given time.

Redundancy is provided as follows:

- When the system is in operation and the primary I/O Server becomes inoperative, or if you wish to perform some maintenance and take it offline, every client will be reverted to the standby server with minimal or no interruption to the system.

- When the primary server is brought back online, the system returns control of the I/O Devices back to the primary server. This is done by copying the disk image from the standby server to the primary, allowing the clients to reconnect to it, and thereby resuming control of the system.

When the system is running, you can also use redundant I/O Servers to split the processing load. This would result in higher performance as every I/O Server would be running in parallel when servicing the I/O Devices.

See Also

I/O Device promotion

I/O Device promotion

I/O Device promotion refers to when a system event forces a standby I/O Device to assume a primary role during Runtime.

If there is more than one standby device available within a cluster, the order they are promoted in will depend on how your project has been configured.

If necessary, you can manually set the order your standby devices are promoted in via the **Priority** field on the extended I/O Devices Properties form. If priorities are not set, the compiler automatically allocates a priority value to each standby device based on the following rules:

- user-configured priority settings take precedence
- any standby devices that do not have a priority setting will follow those that do, their priority being allocated in the order they were configured
- if no standby devices have a priority setting, they will be assigned priority according to the order they were configured.

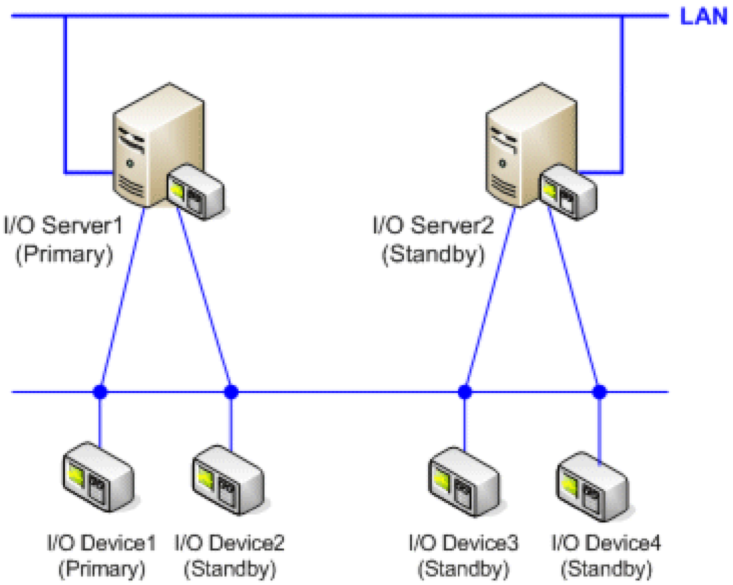
These rules apply regardless of the connections between the I/O Devices and I/O Servers configured in a cluster.

The standby device priorities are confirmed and/or allocated during project compilation. Notification is provided for each allocation issued by the compiler. Compile errors will occur under the following circumstances:

- if a primary device has a priority setting other than the default value of 1
- if standby devices have duplicated priority values.

Example

The following diagram shows four I/O Devices connected to two I/O Servers, with just one primary device configured.



If no priorities were set for the standby devices, the compiler would allocate the following:

- I/O Device1 is allocated **Priority 1** by default (no compiler alert)
- I/O Device2 is allocated **Priority 2** (compiler warning alert)
- I/O Device3 is allocated **Priority 3** (compiler warning alert)
- I/O Device4 is allocated **Priority 4** (compiler warning alert)

This presumes the devices were configured in numerical order.

If I/O Device3 has been manually set to priority 2, the compiler would allocate the following:

- I/O Device1 is allocated **Priority 1** by default (no compiler alert)
- I/O Device2 is allocated **Priority 3** (compiler alert generated)
- I/O Device3 is allocated **Priority 2** (no alert generated)
- I/O Device4 is allocated **Priority 4** (compiler alert generated)

In this case, the setting for I/O Device3 has taken precedence. The remaining standby devices are set for promotion based on the order they were configured.

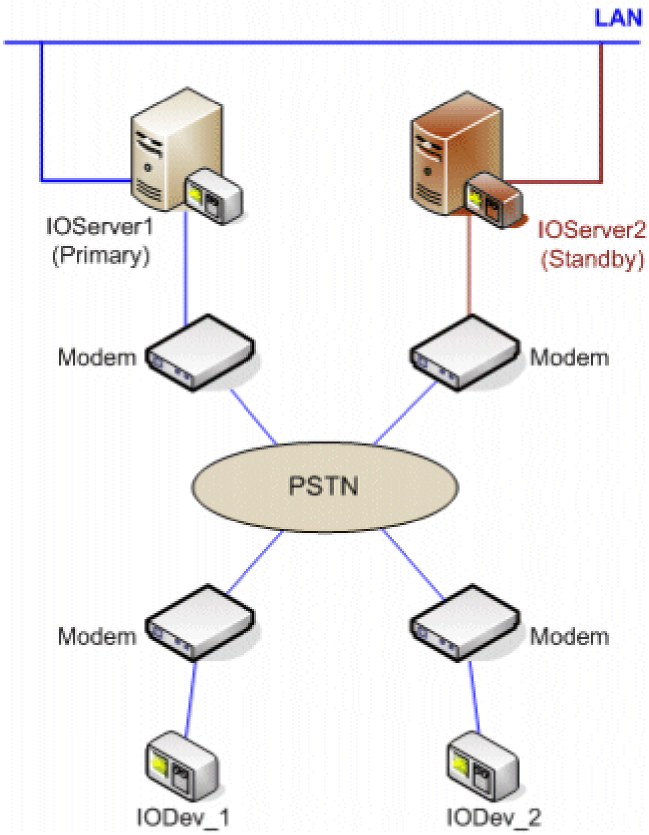
> **Note:** The automated priorities work on an n+1 equation; if I/O Device3 has been set to Priority 5, the remaining devices would have been allocated priority 6 and 7.

See Also

Redundancy and Persistence
Data Path Redundancy

Redundancy and Persistence

If you are using Server redundancy, Persistence Caches (I/O Server cache) keep standby servers updated with the most recently read device data. A Persistence Cache, or I/O Server Cache, is created for each cached I/O Device. The following diagram introduces the concept of a Persistence Cache.



The diagram shows that there are two I/O Servers, namely IOServer1 (primary) and IOServer2 (standby). Each connects to the public switched telephone network (PSTN) via a modem, which is in turn connects to the I/O Devices, also over a modem. Persistence Caches work as follows:

1. Every IODevices->Cache Time period, data from an I/O Device is stored temporarily in the memory of the I/O Server (I/O Server cache).
2. For every [IOServer]SavePeriod, IOServer1 saves its in-memory cache to disk.
3. The cache is saved in Persistence Caches -one for each cached device.
4. IOServer1 broadcasts to other I/O Servers the UNC path of the Persistence Caches (set with [IOServer]SaveNetwork).

5. From these Persistence Caches, IOServer2 updates its in-memory cache for its I/O Devices.
6. Depending on the value of the I/O Server parameter of `[IOServer]SavePeriod' (determines how often the Persistence Cache is saved to the hard disk in seconds), IOServer1 saves its in-memory cache to the hard disk every x amount of seconds.

> **Note:** You can define an I/O Device on an I/O Server using the Express Communications Wizard, or by adding a device in the I/O Devices form in CitectSCADA's Project Editor.

You are not limited to just one Standby Server, since the UNC path name set in [IOServer]SaveNetwork is broadcast to I/O Servers. Each I/O Server updates its cache from the Persistence Caches only for the I/O Devices defined on that server. It is then possible, therefore, set up several I/O Servers which update their in-memory caches with the most recently read data.

For example, we set the [IOServer]SaveFile and [IOServer]SaveNetwork parameters as follows:

On IOServer1	On IOServer2
[IOServer]	[IOServer]
SaveFile=C:\Data\IOServer1.dat	SaveFile=C:\Data\IOServer2.dat
Save- Network=\\IOServer1\Data\IOServer1.dat	Save- Network=\\IOServer2\Data\IOServer2.dat

IOServer1 would broadcast the following UNC path of the Persistence Cache to other I/O Servers: '\\IOServer1\Data\IOServer1.dat'. IOServer2 would then use the Persistence Caches to update its in-memory cache with the device data most recently read by IOServer1.

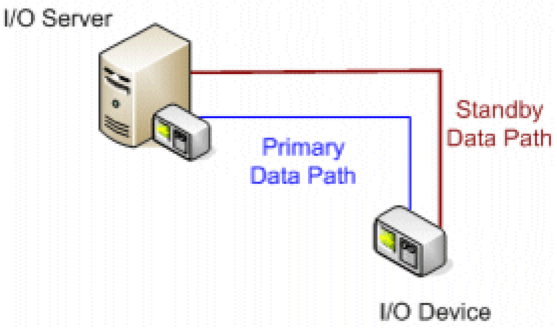
See Also

Data Path Redundancy

Data Path Redundancy

Data path redundancy is another form of redundancy involving defining data paths between the I/O Server and the connected I/O Devices. By providing a second (parallel) data path, you improve the chances that if one data path to the I/O Device is disconnected, the other can be used.

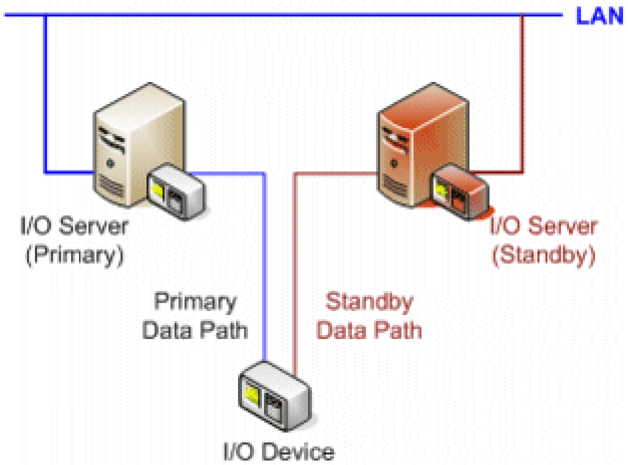
Most brands of PLCs have the facility to allow you to install a parallel data path from the I/O Server to the I/O Device.



The diagram above shows that an additional data path (running in parallel) has been defined. The redundancy is provided as follows:

- When you start your runtime system, CitectSCADA connects to the I/O Device using the **primary data path**.
- If communications with the I/O Device is lost at any time (for example if the communications cable is disconnected), CitectSCADA will **switch to the standby data path** with minimal or no interruption to the system.
- CitectSCADA reconnects through the primary data path when it is returned in to service.

On a larger system (such as one running on a network), you can also use data path redundancy to maintain device communications with multiple I/O Server redundancy, as shown in the following diagram.



The redundancy is provided as follows:

- By using a redundant data path from the I/O Device (one path to each I/O Server), you can maintain I/O Device communication.

- If communications with either the primary I/O Server or standby I/O Server be disconnected, the I/O Device is still accessible.

See Also

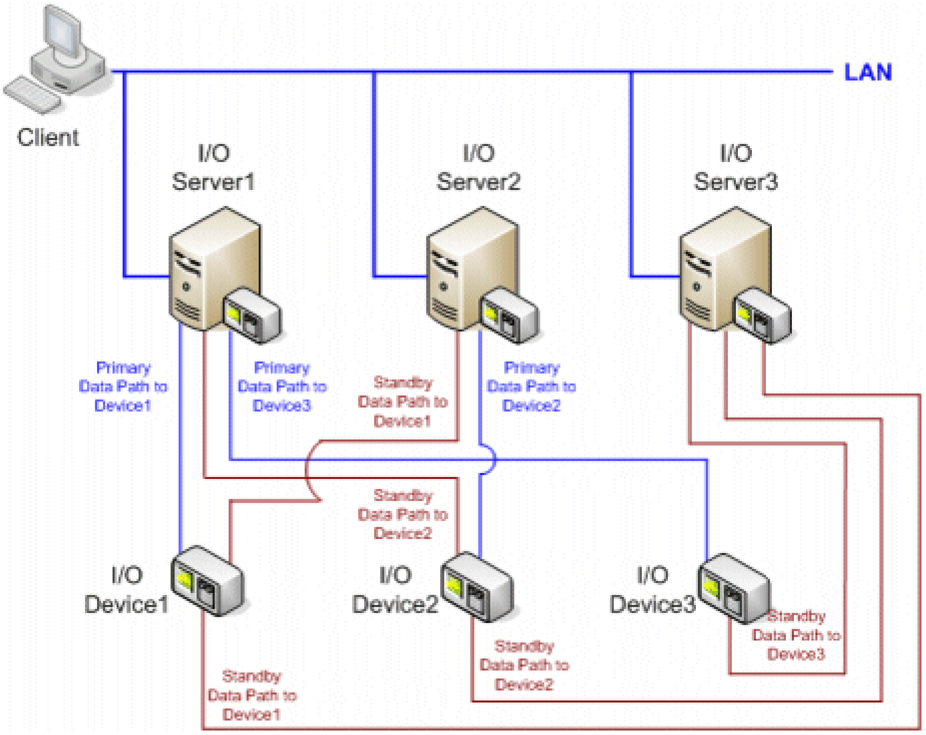
Multiple Device Redundancy (Standby Data Paths)

Multiple Device Redundancy (Standby Data Paths)

If your I/O Devices support peer-to-peer communication, you can add another level of redundancy to your system by duplicating the I/O Devices.

Note: Although I/O Servers are not assigned the Primary or Standby role based on the I/O devices to which they are connected, it is common practice in redundant I/O systems to connect the Primary I/O Devices to the Primary I/O Server and the Standby I/O Devices to the Standby I/O Server. One I/O Server can connect to a mixture of Primary and Standby I/O Devices. The I/O Server can support any number of Standby Data Paths.

The following diagram demonstrates multiple device redundancy and Standby Data Paths:


Client
LAN
I/O Server1
I/O Server2
I/O Server3
Primary Data Path to Device1
Primary Data Path to Device3
Standby Data Path to Device1
Primary Data Path to Device2
Standby Data Path to Device2
I/O Device1
I/O Device2
I/O Device3
Standby Data Path to Device3
Standby Data Path to Device2
Standby Data Path to Device1

In this scenario, we have three I/O Servers connected to three I/O Devices in the following manner:

I/O Server	I/O Devices Connected
IOServer1	I/O Device1 (Primary) I/O Device2 (Standby) I/O Device3 (Primary)
IOServer2	I/O Device1 (Standby) I/O Device2 (Primary)
IOServer3	I/O Device1 (Standby) I/O Device2 (Standby) I/O Device3 (Standby)

The following is known:

- CitectSCADA clients communicate with every configured I/O Server at the same time (on startup, the clients try to connect to each configured I/O Server. If they cannot establish communications with an I/O Server, a hardware error is generated).
- When every device is running, CitectSCADA processes the I/O on the Primary I/O Devices (this reduces the I/O load on the I/O Device (and PLC network), which is important in improving performance).
- The client creates network sessions to the three I/O Servers.
- The client then sends requests for I/O Device1 and I/O Device3 to I/O Server1, and requests for I/O Device 2 to I/O Server2.

Redundancy is provided as follows:

- If I/O Device1 become inoperative on I/O Server1, the client will send requests for I/O Device1 to I/O Server2 through the standby data path. It continues to send requests for I/O Device3 to I/O Server1.
- If I/O Device also becomes inoperative on I/O Server2, the client sends requests to I/O Server3.
- If the connection between I/O Device1 and I/O Server1 be re-established, the client resumes sending requests for I/O Device1 to I/O Server1.

The Standby I/O Devices will be activated strictly in the order in which they are first created in the project. This can be viewed by looking in the Units.dbf file in the project directory.

Since we can place primary and standby I/O Devices on various I/O Servers, share the primary I/O Devices between your I/O Servers to balance the loading across the I/O Servers. However, this might not apply for every protocol because the loading could be dependent on the PLC network and not the I/O Server CPU. In this case, more than one active I/O Server on the same PLC Network can degrade the PLC network and therefore, slow the total response.

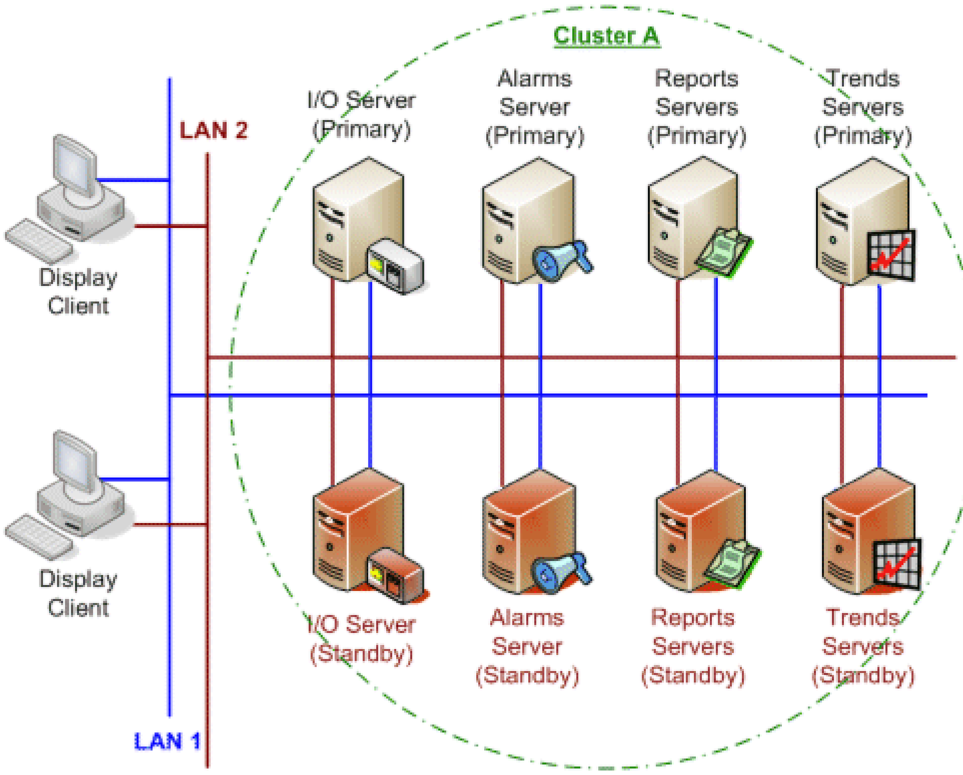
See Also

Alarms, Reports, and Trends Server Redundancy

Network Redundancy

You can use the dual NIC (or multiple network interface) capabilities of each client or server, enabling you to specify a complete and unique network connection from a client to a server.

Consider the following diagram:



The above example shows two levels of redundancy:

- network redundancy
- server redundancy

Network Redundancy

Each of the system components in the cluster are connected to a second network (LAN 2). This is achieved by using the dual end points of each server. This provides connection to two separate LANs to provide for LAN redundancy as follows:

- If LAN 1 is suddenly inoperative, each component in the cluster can easily maintain connection by using LAN 2.
- In turn, if LAN 2 becomes inoperative, LAN 1 remains in operation.

Server Redundancy

For information about server redundancy, see Alarms, Reports and Trends Server Redundancy.

See Also
Configuring network redundancy

Configuring network redundancy

To connect to machines using dual network connections for redundancy you need to first define network addresses for each network interface and then specify which network addresses to use for each server.

To configure network redundancy:

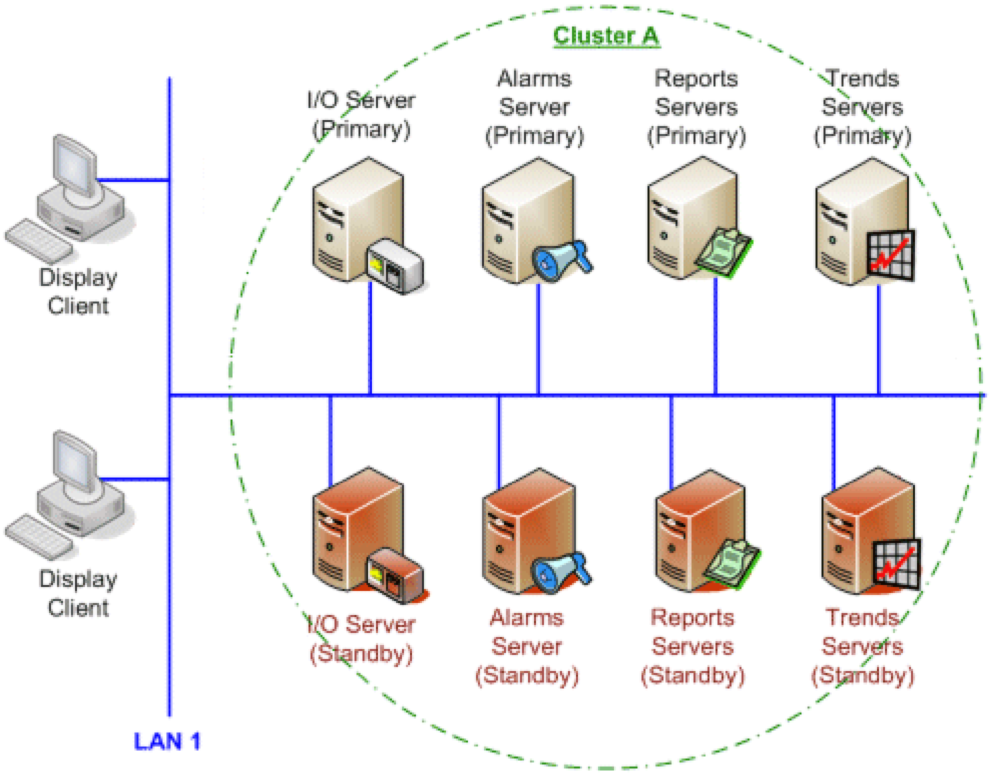
1. In the Project Editor, choose **Servers | Network Addresses**.
2. In the Network Addresses dialog box, define the network address for each network interface card. See Network Address Definitions.
3. In the Project Editor, choose **Servers** then the server type you want to connect to the network.
4. In Network Addresses field of the server configuration dialog box, type the dual addresses for the server separated by a comma. For example, "AlarmPrimaryLAN1,AlarmPrimaryLAN2".

See Also
Network redundancy

Alarms, Reports and Trends Server Redundancy

Redundancy of Alarms, Reports, Trends, and I/O Servers is achieved by adding standby servers, within a defined cluster, to provide redundant system components. In addition you can also utilize the dual end point (or multiple network interfaces) capabilities of each component, effectively enabling you to specify a complete and unique network connection from a client to a server. See Network Redundancy.

Consider the following diagram:



Server redundancy of each component in the diagram is achieved by providing a corresponding standby server on the same network (LAN 1). Redundancy is provided as follows:

- If any of the servers become inoperative or communications become inoperative, their standby counterpart assumes operation.
- For I/O Server redundancy, when the inoperative I/O Server comes back online, it resumes control based on the individual I/O device primary or standby priority configuration..
- For Alarm, Trend and Report servers, when the inoperative primary server returns online, the client will remain with the standby unless the primary has a higher priority.
- For Alarms, Reports, and Trends Server redundancy, clients connect to either the Alarms, Reports, or Trends primary server or standby server. On startup, clients try to establish a connection with the primary server. If a connection with the primary server cannot be established, they will try to establish a connection with the standby server. If the primary server becomes available, any clients connected to the standby server remain connected to the standby server unless the primary has a higher priority. If the standby server becomes inoperative the client will revert to the primary server. The priority is set using the connectivity parameters described below.

Connectivity Parameters

Two Citect.ini parameters determine how a client will behave if it is unable to establish or maintain a connection with a primary Alarms, Reports or Trends server. Each server type has access to these parameters [Type.ClusterName.ServerName]Priority and [Type.ClusterName.ServerName]DisableConnection, where Type is the relevant server type (Report, Trend or Alarm). The default setting for these parameters provides behavior as described above, in that a client redirected to a standby server will retain a connection to that server, providing that it is operable, even when the primary server is restored. This is also the behavior that will occur if you do not add these parameters for a given server type.

Using the parameters

When a primary server has the parameters [Type.ClusterName.ServerName]Priority=1 and [Type.ClusterName.ServerName]DisableConnection = 0 and a control client cannot establish a connection to the primary server, the control clients will switch to the standby server with its parameters set at [Type.ClusterName.ServerName]Priority=0 and [Type.ClusterName.ServerName]DisableConnection = 0. When the primary server is restored, the control clients will switch back to the primary server since it has a higher priority.

If both servers have their [Type.ClusterName.ServerName]Priority set to the same value and [Type.ClusterName.ServerName]DisableConnection = 0 then the client will switch to the standby server and will remain connected to the standby server even when the primary server has been restored. This is the default behavior. The connectivity parameters can be set at the system level using the Parameters form of the project, or specifically for each client in the client local Citect.ini file.

If either server has its [Type.ClusterName.ServerName]DisableConnection = 1, then any client connected to the other server with [Type.ClusterName.ServerName]DisableConnection = 0 will establish a non-redundant connection to that other server. If that connection becomes inoperative, the client will not be redirected and will have no connection until the server is restored.

For specific detail on setting these parameters see:

[Report.ClusterName.ServerName]Priority

[Alarm.ClusterName.ServerName]Priority

[Trend.ClusterName.ServerName]Priority

[Report.ClusterName.ServerName]DisableConnection

[Alarm.ClusterName.ServerName]DisableConnection

[Trend.ClusterName.ServerName]DisableConnection

in the Parameters online help

See Also

Alarm Server redundancy

Reports Server redundancy

Trends Server redundancy

File server redundancy

FTP server redundancy

Alarms server redundancy

It is possible to configure two Alarm Servers in a project. That is, a primary Alarm Server and a standby Alarm Server. With two Alarm Servers, you have improved (mirrored) redundancy on your system.

When both Alarm Servers are in operation, alarms are processed on both servers in parallel, and are logged by the primary Alarm Server. If the Primary Alarm Server becomes inoperative, the Standby Alarm Server starts to log alarms to devices.

When one alarm server task is shutdown the redundant alarm server task continues to process the alarm states. When the alarm server task is restarted, it tries to establish a connection to an alternate Alarm Server. If it can connect, it transfers the dynamic alarm data from the running Alarm Server to the other Alarm Server (this data includes summary data and the current alarm states). If a connection to an alternate Alarm Server cannot be established, the Alarm Server opens the save file (defined with the [Alarm]SavePrimary parameter) and restores the data from the file. If two save files exist, one from the primary Alarm Server and one from the standby Alarm Server, CitectSCADA uses the save file with the later date, or in other words, the newest save file. If no save file is configured, the Alarm Server cannot obtain the initial status (state) of the alarms and no summary information will be available. If this is the case, the Alarm Server starts processing the alarms, and then acknowledges the new alarms.

While both Alarm Servers are active, they will both read data from the I/O Server and process the alarms. The on/off status of each alarm is not passed between the two servers. When operators perform functions on alarms (for example, acknowledge, disable, enable, add comments, etc.), this information is passed between the two Alarm Servers (if an operator acknowledges an alarm on one server, that server tells the other server to acknowledge the same alarm).

Under normal operation, it is recommended to design your system to have an uninterrupted link between the alarm server tasks. This allows each server task to be able to communicate operational state changes (such as alarm acknowledgment) to the redundant peer. For alarm server task communication, network design that includes redundancy at the network layer helps the alarm server tasks continue operation even during a single network disconnection.

Alarm redundancy is designed to allow the user to resolve the arbitration of an alarm list that is different between redundant alarm server tasks. If a mismatch occurs, the user is able to select which alarm server task is incomplete and restart that server process. During the restart, the state of the alarm server task will match the online peer.

See Also

Reports Server Redundancy

Reports server redundancy

It is possible to configure two Reports Servers in a project. That is, a primary Reports Server and a standby Reports Server.

When both Reports Servers are in operation, the scheduled reports only run on the primary Reports Server. If the primary Reports Server becomes inoperative, the scheduled reports run on the standby Reports Server (you can also configure the standby Reports Server so that is also runs the scheduled reports in parallel with the primary Reports Server). Please be aware that no report data is transferred between the primary and standby Reports Servers (CitectSCADA does not synchronize the report data because reports can write their data to any type of device.

See Also

Trends Server Redundancy

Trends server redundancy

It is possible to configure two Trends Servers in a project. That is, a primary Trends Server and a standby Trends Server.

When both Trends Servers are in operation, trends are processed on both servers in parallel, and written to disk (each server needs to write to its own disk or its own private area on the file server).

When a Trends Server starts up, it tries to establish a connection to the other Trends Server. If it can establish a connection, it will transfer the trend data from the last time it was shutdown until the current time (this minimizes the chance that trend data will be lost).

See Also

File Server Redundancy

File server redundancy

CitectSCADA allows for redundancy of the File Server. The [CtEdit]Backup parameter specifies a backup project path. If CitectSCADA cannot find a file in the Run directory (i.e. as specified by the [CtEdit]Run parameter), it will look in the backup path. If the file is found in the backup path, CitectSCADA will assume that the run path has become unavailable for some reason (for example, the file server has become inoperative). It will then look for relevant files in the backup before changing over. When CitectSCADA changes over to the backup path, it will call event number 11 and generate the hardware error: **File server failed, to Standby**.

File Server redundancy will only operate correctly if the redirector (or shell) on the computer can respond appropriately if the File Server becomes inoperative. The Novell Netware shell cannot do this and will cause Windows itself to become inoperative or report what it views as serious Network Errors - if the file server becomes inoperative. Microsoft LAN Manager-based networks and peer-to-peer networks correctly handle instances when the File Server becomes inoperative. Therefore, CitectSCADA File Server redundancy will operate correctly with these networks.

Note: Only CitectSCADA switches to a backup path. Any other applications that are using files on the File Server will become inoperative when the File Server becomes inoperative. This may cause the computer to wait for long periods for the File Server (or to itself become inoperative). This includes Windows itself, so install Windows on a local drive.

To enable File Server redundancy, set the **[CTEDIT]Backup** parameter to a backup database path. For example, if your primary path is **F:\CITECT\USER\DB**, set the backup path to another File Server or a local drive, such as **C:\CITECT\USER\DB**.

You will want to verify that the project in the Backup path is the same as the one in the Run directory - each time you compile the project in the run directory copy it into the backup directory.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Before placing your CitectSCADA system into service, confirm that the Standby File Server has an identical copy of the current project, and that the [CTEDIT]Backup parameter is correctly set.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

See Also
FTP Server Redundancy
[CtEdit]Copy
[CtEdit]Run

FTP server redundancy

CitectSCADA supports FTP Server redundancy. If the primary FTP Server goes down, CitectSCADA will attempt to connect to the FTP Server on the standby machine. This occurs independently of I/O Server redundancy, so the two FTP Servers need to have the same passwords and the same directory structure.

FTP Server redundancy is configured by setting parameters in the [CLIENT] and [DNS] sections of the Primary FTP Server's Citect.ini file. These parameters are downloaded by the Internet Display Client (IDC) to its own Citect.ini file if the Primary FTP Server becomes inoperative, provided the [INTERNET]Redundancy parameter has not been set to 0 (zero). The IDC then uses the downloaded redundancy information to connect to the standby FTP Server.

> **Note:** Standby FTP Servers need not be Internet Servers. The Standby FTP Server can be any server using TCP/IP that the IDC can connect to, provided there are IDC licenses present in the network.

See Also
Redundancy of Standalone Systems

Redundancy of Standalone Systems

If you are using CitectSCADA as a standalone system, in the single process model, you can still achieve redundancy by implementing your standby systems on another CPU, provided you have a multi-CPU system. You may also wish to implement this for load-balancing purposes.

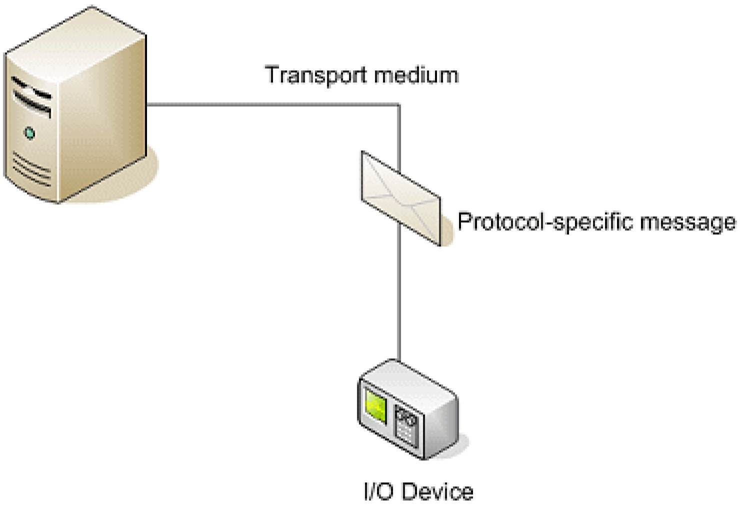
See Also
Configuring Your System

Chapter: 15 Communicating with I/O Devices

CitectSCADA can communicate with any control or monitoring I/O Device that has a communication port or data exchange interface, including PLCs (programmable logic controllers), loop controllers, bar code readers, scientific analyzers, remote terminal units (RTUs), and distributed control systems (DCS).

Typical CitectSCADA communications consists of four key parts:

1. A CitectSCADA I/O Server;
2. A target I/O Device;
3. A physical means for transporting messages between them (transport);
4. The messages exchanged between them (protocol).



These components work in unison to expose the inputs and outputs of an I/O Device to a CitectSCADA system.

- Inputs to the I/O Device provide information about your plant, such as the speed of a machine, status of a conveyor, or the temperature of an oven.
- Outputs from the I/O Device usually initiate tasks that control the operation of your plant, such as starting electric motors, varying their speed, or switching valves and indication lamps.

The transport medium does not need to be a direct cable as shown in the diagram; it can be any means of carrying the message – such as a high-speed wireless link or an FDDI network. Similarly, the protocol-specific message could be a simple ASCII message or a complex object-based message such as DNP 3. Engineers are free to assemble different combinations of I/O Devices, transports, and protocols.

See Also

The Role of the I/O Server

The Role of the I/O Server

The CitectSCADA computer that directly connects to an I/O Device is an I/O Server. I/O Servers are responsible for servicing the read, write and subscription requests from clients. A project can have many I/O Servers, with each cluster able to include multiple I/O Servers.

An I/O Server keeps up-to-date information on its connected I/O Devices by regularly retrieving data from each and storing it in a cache (I/O Device data cache). Whenever a CitectSCADA client requires data from an I/O Device, it will use the information stored in the I/O Server cache. Clients do not retrieve data directly from an I/O Device.

An I/O Server will read the necessary data from the I/O Device to execute a requested Cicode task or process. For example, when you schedule a report, CitectSCADA reads the I/O Device data that the report might need before the first line of the report starts running.

CitectSCADA performs writes to the I/O Device asynchronously, allowing other operations to continue while a write is taking place.

See Also

The Role of the I/O Device

The Role of the I/O Device

CitectSCADA is predominantly a supervisory system. It is the I/O Devices that directly monitor and control automation equipment. In most I/O Devices (such as PLCs) a program stored in the I/O Device controls the outputs. The logic (control strategy) of this stored program and the status of the inputs determine the value of each output.

The value of each input and output is stored in a separate memory register in the I/O Device. Each memory register is referenced by its address.

By reading and writing to memory registers in I/O Devices, CitectSCADA collects data from your plant or factory for monitoring and analysis, and provides high-level (supervisory) control of your equipment and processes.

You usually don't need to read (or write) to every register in the I/O Device: CitectSCADA lets you specify which inputs and outputs you want to monitor or control. After defining these register addresses, you can use them for system control, operator displays, trend analysis, data logging and alarm indication.

The choice of I/O Device is typically not open to the person engineering the communications system. In most cases, the I/O Device hardware is purchased in advance of the integration effort, or it is legacy equipment. However, if you have the luxury of influencing the choice of I/O Devices, communications capability is recommended to be one of the factors you consider.

Note: I/O devices such as programmable logic controllers (PLCs) usually have an internal program that controls the low-level processes within your plant. A PLC program continually scans the input registers of the PLC, and sets the output registers to values determined by the PLC program logic. While CitectSCADA can replace any PLC program, this is not recommended. PLCs are designed for high-speed response (typically 1 to 100 ms) and replacing this functionality with CitectSCADA could negatively affect your control system’s performance. Only use CitectSCADA to complement your PLC program (that is, for high level control and system monitoring).

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

Do not use CitectSCADA or other SCADA software as a replacement for PLC-based control programs. SCADA software is not designed for direct, high-speed system control.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

See Also

The Role of the Transport Medium

The Role of the Transport Medium

The term ‘transport’ refers to both the physical communications medium and the low-level logic necessary to drive it. As far as CitectSCADA is concerned, it simply defines how to package a message, how to send it, and where to send it.

The available transport options are usually governed by the kind of ports available on the target I/O Device. However, they are also influenced by the geography of the automation system, specific domain requirements (such as safety, performance, redundancy), and the amount of money the owners are willing to invest.

The three main categories of available transport are:

1. Simple serial – such as RS-232, RS-422, RS-485;
2. Ethernet – the dominant frame-based inter-network protocol; or
3. Proprietary – using intermediary PC-communications hardware or software (such as OPC servers).

Of these, Ethernet is by far the transport technology of choice.

Fortunately, by using the concepts of packet encapsulation and intelligent gateway/routing devices, you are able to mix different transport layers. For example, you could use a serial cable from CitectSCADA to a modem, a PSTN line from the modem to another modem, then a serial cable to an Ethernet gateway, and Ethernet to the I/O Device.

Note: Modems are treated as a special case in CitectSCADA since they are recognized as logical devices by Windows. Refer to the specific online help on setting up modems.

CitectSCADA uses the 'COMx' driver to implement simple serial transports. Similarly, the 'TCPIP' driver implements the Ethernet transport. These are commonly used transport drivers, and both have a range of tuneable options and settings. Each proprietary board has its own specific transport driver – which is typically integrated with the protocol driver.

See Also

The Role of the Protocol
Working With Device Drivers

The Role of the Protocol

I/O Devices support at least one protocol – which governs the kind of commands and data you can exchange with the device. They vary significantly in functionality and in complexity. However, because CitectSCADA supplies the protocol drivers, the engineer does not need to know the details of the protocol.

Most modern devices support two or more protocols. This gives engineers flexibility in designing appropriate communications architectures. Sometimes there is a simple protocol that provides basic access to device functionality, and a more comprehensive and complex protocol. Occasionally the protocol is closely linked to the transport layer used – particularly in the case of proprietary protocols and communications hardware.

Which protocol you choose depends on:

- The protocols supported by the I/O Device;
- The kind of data you need to access;

- The commands you need to execute; and
- Any constraints stemming from the transport chosen.

In many cases, a specific card or module is necessary in the I/O Device to support additional protocols.

Using Industry Standard Protocols

The automation and associated industries have developed a number of standardized protocols for communicating with I/O Devices. Because these protocols are open and developed collaboratively, they allow a greater degree of connectivity and longevity than other options.

The following list shows some of the prominent communication standards used in the automation industry that are supported by CitectSCADA:

- **ASCII** – for simple serial communications;
- **Modbus** – a widely used simple serial protocol for automation;
- **DNP 3.0** – a protocol for distributed networks such as RTUs;
- **BACNet** – specifically for the building automation control industry;
- **OPC** – a technology for sharing automation data at the PC level;
- **IEC870-5** – communication profile for sending basic telecontrol messages;
- **EIB** – European installation bus;
- **Profibus** – field bus communications protocols for automation;
- **SNMP** – widely used protocol for network devices.

As CitectSCADA allows you to choose whatever protocol you need for the situation, you can choose to use these protocols where supported by your I/O Device.

This is particularly useful when you have a mix of I/O Device brands but you want to simplify and use one common protocol.

See Also

Communication Configuration
Working With Device Drivers

Communication Configuration

CitectSCADA uses a logical and structured communications configuration, that maps closely to the elements shown in the diagram below.

It requires the engineer to:

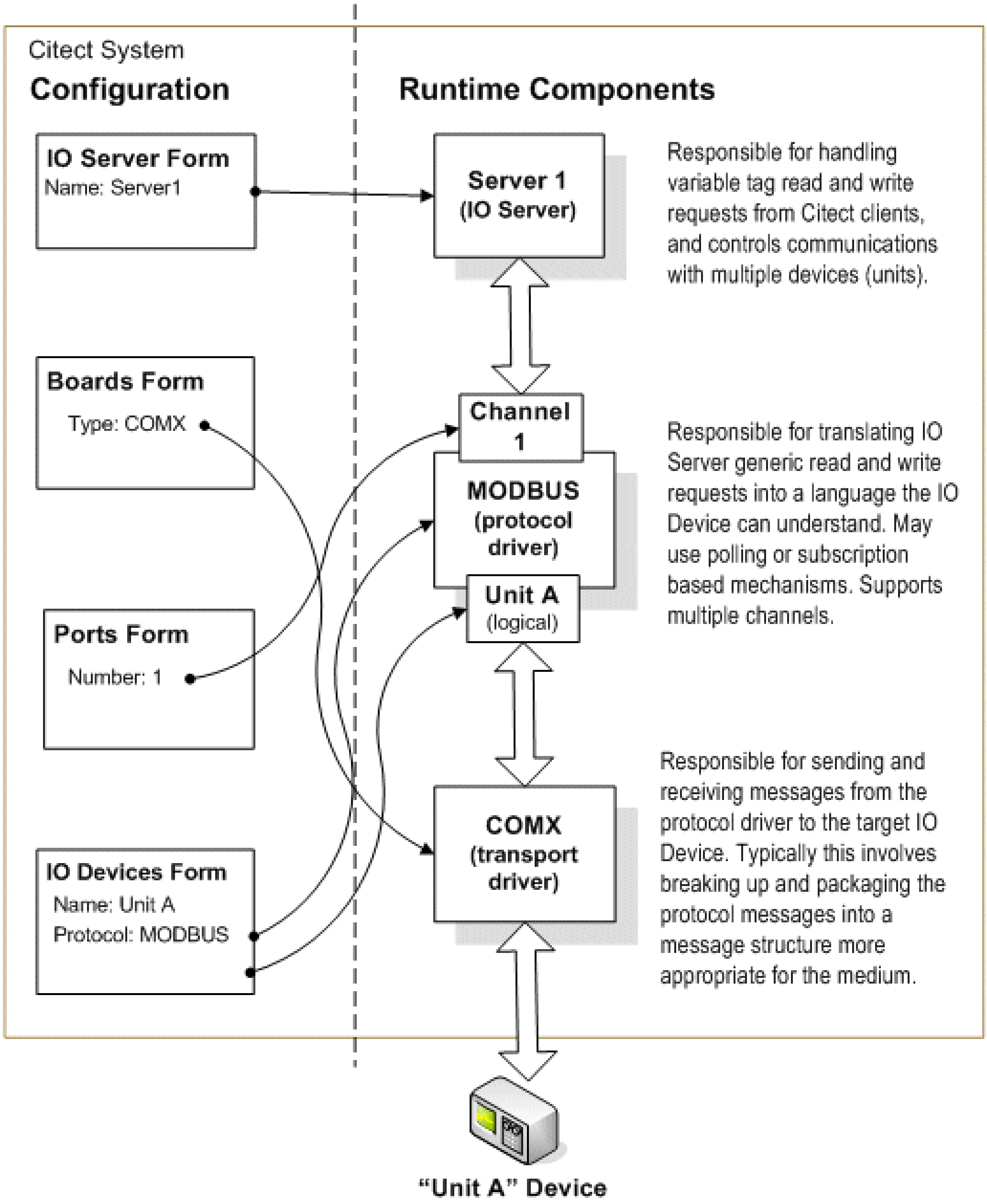
1. Define an I/O Server in the I/O Server form;
2. Define the transport type in the Boards form (COMx, TCPIP, PROFI…);

3. Define the port to communicate out of in the Ports form.
4. Define the protocol to use (MODBUS, DNP, ASCII…).

Note: Each of these steps is simplified in order to illustrate the mapping of necessary elements to the basic communication architecture. In reality, some other tasks need to be performed to setup communications.

The following diagram shows how the elements in the configuration environment map to the software components in the runtime system.

Although this is not essential to know, it does help understand how drivers fit into the system.



Responsible for handling variable tag read and write requests from Citect clients, and controls communications with multiple devices (units).

Responsible for translating IO Server generic read and write requests into a language the IO Device can understand. May use polling or subscription based mechanisms. Supports multiple channels.

Responsible for sending and receiving messages from the protocol driver to the target IO Device. Typically this involves breaking up and packaging the protocol messages into a message structure more appropriate for the medium.

Note: This example omits many logical runtime components for clarity (such as clients). In addition, the runtime components shown are not visible to the user at runtime. However, driver information can be accessed at runtime via the CitectSCADA Kernel.

A good place to start when setting up simple communications is with the Express I/O Device Setup wizard. Once you understand how the setup works, you can manually configure arrangements that are more complex. Much of the functionality of the protocols and transports can be modified using the driver options and parameters. Each driver has its own specific section in the online help that will guide you through configuration options.

See Also

Setting up communications
Working With Device Drivers

Retrieving time-stamped data from field devices

With the release of version 7.20, CitectSCADA supports the retrieval of time-stamped data directly from field devices. This capability is enabled by the Driver Runtime Interface (DRI), a component that is used by the following device drivers:

- OPC
- OFSOPC
- SISCOOPC
- DNPR

Note: Check with Technical Support if new drivers are available that support the DRI.

The DRI allows a driver to push time-stamped data from field devices into a CitectSCADA system. This means time-stamped digital alarms, time-stamped analog alarms and event-based trends can be updated directly from devices.

No configuration is necessary, as when these drivers start up they scan the system for time-stamped digital alarms, time-stamped analog alarms and event-based trends. If the driver receives updated information for any detected tags, it will pass it directly on. Regular I/O server polling is no longer used.

This mechanism can work in tandem with the tag extension feature (also introduced in version 7.20) to enable access to field-generated timestamp and quality tag values (see Tag Extensions).

Note: Prior to version 7.20, time-stamped data was manually pushed into CitectSCADA using the Cicode functions AlarmNotifyVarChange and TrnSetTable. If you are upgrading a project to a version 7.20 system with a driver that uses the new DRI push mechanism, you will no longer need to use these functions.

IO Server parameters

Events pushed from a DRI-supported driver can be buffered on an I/O server while an alarm or trend server is offline. The way this mechanism operates can be configured via the following parameters:

- [IOServer]EventsQueued
- [IOServer]EventsDropped
- [IOServer]MaxTimeInQueueMs

For more information, see IO Server Parameters.

See Also

Communications Configuration
Working With Device Drivers

Setting Up Communications

Setting up communications between a device and a CitectSCADA I/O Server typically involves a process that includes the following steps:

1. **Choose the Method of Communication**
By examining the capabilities of your I/O Devices and control systems networks, select an appropriate communication mechanism. This mechanism will usually imply the transport and protocol drivers that needs to be con figured in CitectSCADA.
See The Role of the Transport Medium and The Role of the Protocol for more information.
2. **Prepare the Device**
Confirm that your device meets the hardware and software requirements needed to communicate with CitectSCADA. The Driver Reference Help provides hardware setup information specific to each supported device.
3. **Prepare the I/O Server for Communication**
This involves preparing the I/O Server to communicate with a device. This includes setting up a COM port, a serial board, or a proprietary board in an I/O Server.
4. **Create a Test Project**
Before configuring your CitectSCADA project, confirm communications between CitectSCADA and your system devices. Creating a test project allows you to test the communication path in isolation, and verifies that CitectSCADA can bring your devices online.
5. **Configure communications using the Communications Express Wizard**
The Express Communications Wizard provides default values and a setup tailored to the communications requirements of a selected I/O Device.

OR

Configure Communications Manually
You can manually configure the communications to your I/O Device using the Boards, Ports and I/O Devices forms in the CitectSCADA Project Editor. This often gives more flexibility, but requires a more detailed knowledge of the driver settings and parameters.

Once you have performed these steps, you are ready to run and verify your communications.

See Also

Customizing a Project Using Citect.ini Parameters
Using a Disk I/O Device

Preparing a Device

You need to verify your system meets the hardware and software requirements needed to establish communication between a device and CitectSCADA.

This may include a variety of possibilities, including:

- the installation of a proprietary communications card
- the establishment of a device server
- the installation of configuration software, etc.

The **Driver Reference Help** provides the hardware and software requirements for each device, and includes any additional information you need to know about setting up specific devices. To access this help, select **Driver Help** from the **Help** menu in Citect Explorer and Project Editor.

See also

Preparing the I/O Server for communication

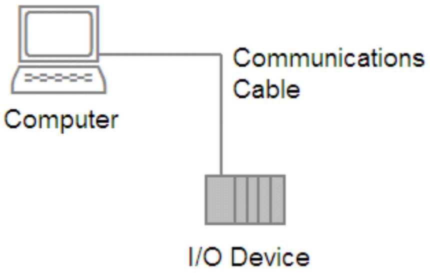
Preparing the I/O Server for Communication

You need to verify a CitectSCADA I/O Server is enabled to support the necessary communications method. This involves installing and configuring any communications hardware and/or software necessary for communications. Common scenarios are:

- Using a COM port
- Using a Serial board
- Using an Ethernet card
- Using a Proprietary board/intermediary software

Using a COM port

The simplest CitectSCADA systems use a single computer connected to the I/O Device(s). You can connect an I/O Device directly to a communications port with a standard RS-232 communications cable.



How to set up CitectSCADA to use your computer's COM port:

1. Verify that the **Boards** configuration has **COMx** as the**Type**, and the **Address** set to 0. The **I/O Port, Interrupt**and **Special Opt** can be left blank.
2. Enter the **Port Number** in the **Ports** configuration. The COM port number will usually be either 1 or 2, and is set in the Ports section of the Control Panel. Use the **Special Opt** field to modify the behaviour of the COMx driver. (See COMx driver special options reference for more information).

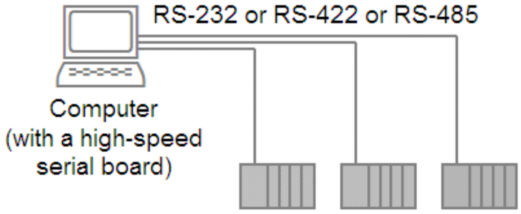
> **Note:** You only need to define the COMx board once. You can then add several ports that use the same CitectSCADA board. For example, a COM port and two serial boards could be defined as one COMx board in CitectSCADA, with multiple ports.

See Also

Debugging a COMx Driver
Using a Serial Board

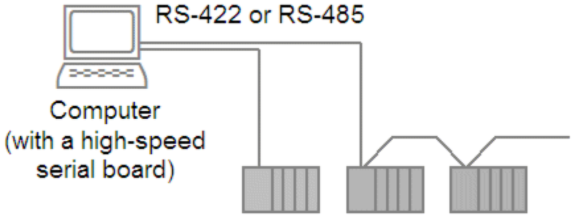
Using a Serial Board

The communications port on the computer is not designed for high-speed communications and reduces system performance. Instead, install a high-speed serial board (such as a Digiboard). High-speed serial boards have several ports (usually 4, 8, or 16) to let you connect several I/O Devices to your CitectSCADA system.

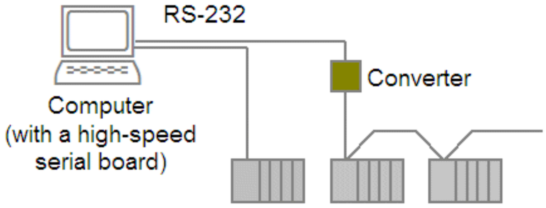


You can use identical I/O Devices or I/O Devices supplied by different manufacturers; CitectSCADA supports most popular I/O Devices. You can connect any number of I/O Devices; the only limitation is the size of your computer. High-speed serial boards are available for RS-232, RS-422, or RS-485 communication.

If you have several I/O Devices from the same manufacturer and these I/O Devices support multi-drop communication, you can connect them to an RS-422 or RS-485 high-speed serial board installed in your computer. (The RS-232 standard does not support multi-drop communication.)



Not every high-speed serial board supports RS-422. You can use an RS-232/RS-422 or RS-232/RS-485 converter to achieve the same arrangement.



Note: Using a converter can introduce handshaking/timing considerations.

To set up CitectSCADA to use a serial board:

1. Install the board in your computer and set it up under Windows as per the accompanying instructions. Use the latest driver from the board manufacturer.
2. Make sure that the boards configuration has COMx as the Type, and the Address set to 0. The **I/O Port**, **Interrupt**, and **Special Opt** can be left blank.
3. Enter the **Port Number** in the Ports configuration. The COM port number is usually greater than 2 and set in the **Ports** section of the Control Panel. Use the **Special**

Options field to modify the behavior of the COMx driver. (See COMx driver special options reference for more information).

Notes

- If using your computer's COM port, you don't need to install additional software.
- You only need to define the COMx board once. You can then add several ports that use the same CitectSCADA board. For example, a COM port and two serial boards could be defined as one COMx board in CitectSCADA, with multiple ports.

See Also

Using Proprietary Boards

COMx driver special options reference

Special Options (in the Ports form) are space separated and start with the dash character (-) immediately followed by the option characters. Only a small percentage of users will need to use the following options:

- **cATS0=1**: Send the string 'ATS0=1<CR>' on startup to allow the modem to detect the baud rate the port is running at. Abandon transmit if DCD is low. Wait for incoming call to raise DCD.
- **d**: Data will be transmitted only when DSR is high
- **di**: Received data is ignored when DSR is low.
- **dMS**: When transmitting a message the driver will wait up to 2000 ms for DSR to go high, then wait another MS milliseconds before transmitting.
- **-e**: Provides access to the output signal lines. The format is "~EIA**WXYZ**" where **WXYZ** represents one of the following options:

  S Simulate XOFF received
  S Simulate XON received
  RT Set RTS high
  RT Set RTS low
  D Set DTR high
  D Set DTR low
  B Set the device break line
  B Clear the device break line
  Example: "~EIADTR1" sets DTR high
- **-h**: Data will be transmitted only when CTS is high.
- **-hMS**: When transmitting a message the driver will wait up to 2000 ms for CTS to go high, then wait another MS milliseconds before transmitting.
- **-i**: The string sent whenever the port is initialized. The tilde (~) and '\M' characters represent special instructions:
  - ~: Delay for 500 milliseconds
  - \M: Send carriage return

Examples:
`~Fred`: Wait 500 milliseconds and then send 'Fred'
`Fred\MMary`: Send 'Fred', a carriage return, and then 'Mary'
Note: This option is not available for dialable devices (i.e. when the port number is -1).

- **-nt**: With some serial interfaces, line interruptions can cause the COMx read thread to shutdown. If this happens, the driver does not recover after the interruption. However, with the -nt (no terminate) option set, the thread is not shutdown, allowing the system to recover when the interruption is rectified.
- **-nts**: If errors occur when the COMx driver is starting up, it will not terminate, but will continue attempts to open the COMx port.
- **-r**: Driver will raise DTR only when transmitting.
- **-ri**: DTR is raised when there is enough room in the input buffer to receive incoming characters and drop DTR when there is not enough room in the input buffer.
- **-rPRE,POST**: When transmitting a message the driver will raise DTR for PRE milliseconds, transmit message, wait for POST milliseconds then drop DTR.
- -sc: Activates software flow control using XON and XOFF

XonLim: A number in bytes. This represents the level reached in the input buffer before the XON character is sent (30 bytes)

. XoffLim: The maximum number of bytes accepted in the input buffer before the XOFF character is sent. This is calculated by subtracting (in bytes) 100 from the size of the input buffer

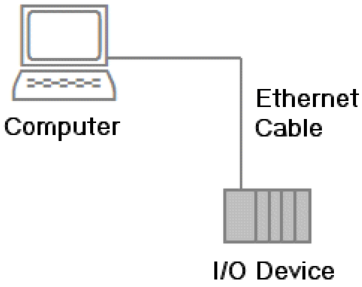
- **-t**: Driver will raise RTS only when transmitting.
- **-ti**: RTS is raised when there is enough room in the input buffer to receive incoming characters and drop RTS when there is not enough room in the input buffer.
- **-to**: RTS is raised only when there are characters to transmit.
- **-tPRE,POST**: When transmitting a message the driver will raise RTS for PRE milliseconds, transmit message, wait for POST milliseconds then drop RTS..

See Also

Using a Com Port
Debugging a COMx Driver
Using a Serial Board

Using an Ethernet Card

Ethernet is one of the most popular methods of communicating with a number of I/O Devices at high speed. It is certainly one of the easiest to configure. You can typically connect to Ethernet capable I/O Devices using a standard Ethernet card, or integrated Ethernet port.



To set up CitectSCADA to use your computer's Ethernet card:

1. Make sure that your Ethernet card is installed correctly and working under Windows.
2. Make sure that the **Boards** configuration has **TCP/IP** as the **Type**, and the **Address** set to 0. The **I/O Port, Interrupt**and **Special Opt** can be left blank.
3. In the **Ports** form, use the **Special Opt** field to enter the destination IP address using the following format:
   -Ia -Pn -T
   where:
   a = the destination IP address in standard Internet dot format. (For example 192.9.2.60)
   n = the destination Port number. Often one physical port has several virtual ports, used for different purposes. Use this option only if you want to override the default of 2222.
   T = forces the driver to use TCP (the default), rather than UDP (-U).
   (See TCP/IP driver special options reference for more information)
   Leave other fields blank. You can now define your I/O Device units.

> **Note:** You only need to define the TCP/IP board once. You can then add several ports that use the same CitectSCADA board.

See Also
Using a Serial Board

TCP/IP driver special options reference

Special Options (in the Ports form) are space separated and start with the dash character (-) immediately followed by the option characters. Use the following special options for TCP/IP:

- **A:** Allows the TCPIP driver to be used from Cicode.

- **-Ia.b.c.d**: defines remote IP address to connect to.
- **-FC**: Allows for a fake connection. This creates a pretend connection (no actual IP connection is made). Its intended use is when a driver wants to support a dummy connection but not talk to a device, or have a virtual unit. A virtual unit would allow access to driver addresses which do not need to talk to a device.
- **-K**: sets socket SO_KEEPALIVE flag.
- **-LIa.b.c.d**: defines local IP address.
- **-LPn**: defines local PORT.
- **-Ma.b.c.d**: defines multicast IP address.
  This option can be set using the [TCPIP]PortName.MulticastAddress parameter. See Debugging a TCP/IP driver.
- **-Pn**: defines remote PORT to connect to.
- **-PXn**: defines an extra port to listen to only in UDP mode.
- **-RC**: Activates the reconnection retries on reception of FD_CLOSE event. FD_CLOSE is only received if the Keepalive option is activated. -RC can resolve issues where the TCP/IP driver is notified of the connection close. For a more comprehensive handling, the -k option is needed. Retries can also be activated with the following high-level driver call:

```
COMSetParam((SHORT)ChannelNumber, (UCHAR*)("DO_RECONNECT"), NULL);
```

- **-T**: sets this port for TCP (stream) operation.
- **-U**: sets this port for UDP (datagram) operation

where:

a.b.c.d = standard IP address in dot notation using decimal numbers 0- 255. (Do not use a leading 0 when adding an IP address).

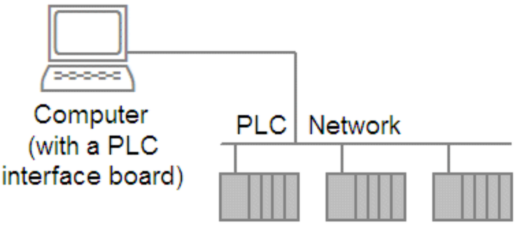
n = decimal value for the port ID for the necessary service.

See Also

Using an Ethernet Card
Using a Serial Board

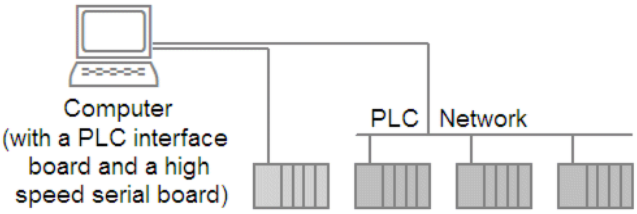
Using a Proprietary Boards and/or Intermediary Software

With some brands of I/O Devices you can install a proprietary interface board in your computer, or intermediary software. This PLC interface board/software is supplied by the PLC manufacturer; you can connect it to a single PLC or a PLC network.



Note: With some PLCs, a high-speed serial board provides better performance than a PLC interface board when the system is connected to more than one PLC.

You can mix both PLC interface boards and high-speed serial boards in a single computer. You can, for example, connect a PLC network to a PLC interface board, and individual I/O Devices to a high-speed serial board.



There are many possible hardware arrangements for a CitectSCADA application. CitectSCADA is a flexible system and imposes few restraints on the type (or manufacturer) of I/O Devices that you can use, or on the way you connect them to the computer.

To set up CitectSCADA to use a proprietary board/software:

If you are using a proprietary board/software (that is, supplied by the PLC manufacturer):

1. Install the board/software in your computer and set it up under Windows as per the accompanying instructions. Use the latest driver from the manufacturer.
2. If possible, run diagnostics on the board before configuring CitectSCADA to check that the board works correctly.
3. Check that the **I/O Port** and **Interrupt** settings are correct.
4. Configure the **Boards** and **Ports** as instructed by the PLC-specific help.

Creating a communications test project

To verify that CitectSCADA can bring your I/O Devices online, it is recommended you create a test project.

The Test Project needs to be as simple as possible. For example, if your system will eventually be communicating through a COM port, a KTX card, an SA85 card, and via TCP/IP do not try and make this work on your first try. Make the Test Project with 1 element at a time.

First add and test the COM port. When that works, make a new test project for the KTX card, then make a new test project for the SA85 card, and finally one for the TCP/IP. Once you are happy that each individual element works properly, start to add them together.

While checking that you have only the absolute minimum information in the project to enable communications, verify that there are no duplicated records.

A good way to do this is to open a form and then check the Record Number shown on the bottom left of the form. Check that it is on Record **1**, and then click the button in the scroll bar on the right hand side of the form and drag it down. When you get to the bottom of the scroll bar, let the button go. If it pops back up to the top of the form and the record numbers stays at 1 then you have only 1 record.

It is necessary to drag the scroll bar as the forms are indexed on the I/O Server name. Quite often there will be 'orphaned' records from a previous I/O Server name still in the database files. If you find any extra records, delete them and then pack the project. Duplicate or orphaned records in the communications database can negatively impact communications performance.

See Also

Running Your Test Project

Running Your Test Project

Before running your test project, verify that the Kernel is enabled on CitectSCADA startup so that you can follow the startup procedures step by step.

To do this, edit your citect.ini file to contain the following:

```
[DEBUG]

Kernel=1
```

This will show the Kernel on startup of CitectSCADA.

Next, run the Computer Setup Wizard. Verify that the computer is defined as a standalone CitectSCADA system and configured to run your test project.

Start the project running. When the kernel appears, double-click the title bar in the Main window, then double-click the title bar of the kernel. Doing this in order is important, as it will maximize the Main window, then Maximize the whole kernel. If you don't do this then you will not see what is happening as the information in the Kernel cannot be scrolled, and once it is off the screen it is gone. This might require a bit of practice as the startup procedure might only take 1-2 seconds or less on a fast machine.

Once you have the project running (and assuming that everything worked) the last line in the Main window in the Kernel will tell you that your I/O Devices are online. Do not confuse this with the message telling you that your Port channels are online. Citect-SCADA will first report that it can communicate through the port you have setup, then report that it can communicate with the I/O Device.

See Also

When Your Test Project Does Not Compile

When Your Test Project Does Not Communicate

Check everything and run it again. Depending on the type of board driver you are using there are different methods for debugging them. However, most of them are basically the same. Add one item at a time and test.

See Also

Troubleshooting Device Communications

Using the Communications Express Wizard

You use the Express Communications Wizard to set up communications with your I/O Devices.

Based on your selections, the Express Communications Wizard provides default values and a setup tailored to your I/O Device communications requirements.

To access the Communications Express Wizard:

1. Open the Project Editor.
2. Choose **Communication** | **Express Wizard** or open Citect Explorer.
3. Double-click the **Express I/O Device Setup** icon in the Communications folder of the current project.
4. Complete the Wizard.

Note: Each I/O Device/protocol combination requires a unique setup for the boards, ports, and I/O Devices forms. See the specific Help for each device.

See Also

Express Communications Wizard - introduction

Express Communications Wizard - introduction

You will be asked to select an I/O Server, choose a name, and indicate the type of I/O Device (External, Memory, Disk). From the list of available manufacturers you choose the manufacturer, model and communications method for the I/O Device. If you are connecting external devices or using a proprietary board in your computer you may be requested to nominate addresses and communications port.

After completing your setup, the Summary Page summarizes the configuration of your I/O Device and/or internal boards. Click **Finish** to save the listed configuration, or click **Back** to change a previous selection.

See Also

Express Communications Wizard - Server selection

Express Communications Wizard - Server selection

Select an existing I/O Servers as defined in the current Project, or create a new I/O Server.

When you create a new I/O Server, CitectSCADA automatically suggests the name **IOServer1**. You can enter a different name if you want. The name you specify needs to be 16 characters or less and use alphanumeric characters (A-Z, a-z, 0-9). You can also use the underscore character (_).

> **Note:** If you add a new I/O server, you will need to run the Computer Setup Wizard on the designated computer before you attempt to run the project. This is necessary to enable the computer to function as an I/O Server.

See Also

Express Communications Wizard - Device selection

Express Communications Wizard - Device selection

Enter the name of the new I/O Device that you want to create, or accept the name **IODev** which CitectSCADA automatically suggests.

See Also

Express Communications Wizard - I/O Device type

Express Communications Wizard - I/O Device type

Select the type of I/O Device. You may choose from the following options:

- External I/O Device
- Persisted Memory I/O Device
- Disk I/O Device

> **Note:** Using a Persisted Memory I/O Device is recommended as an alternative to using a Disk I/O Device, as full synchronization is supported if a server becomes unavailable for a period of time. See Persisted I/O Memory Mode for more information.

See Also

Express Communications Wizard - I/O Device communications selection

Express Communications Wizard - I/O Device communications selection

From the list of available manufacturers, select the manufacturer, model, and communications method specific to the I/O Device.

If a memory or disk I/O Device has been selected, the CitectSCADA Generic Protocol is included at the top of the tree.

See Also

Express Communications Wizard - TCP/IP address

Express Communications Wizard - TCP/IP address

Enter the IP address for the I/O Device, in standard Internet dot format (for example, 192.9.2.60). This address is set on (or specified by) your I/O Device.

The Port number and Protocol (TCP or UDP) fields have been set to the default values for the I/O Device. change these fields only if necessary.

For details about addressing your specific I/O Device, click **Driver Address Help** to browse driver-specific information for your I/O Device.

See Also

Express Communications Wizard - I/O Device address

Express Communications Wizard - I/O Device address

Enter the address for the I/O Device. What you enter in this field is determined by the type of I/O Device (and protocol) used, as each has a different addressing strategy.

For details about addressing your specific I/O Device, click **Driver Address Help** to browse driver-specific information for your I/O Device.

See Also

Express Communications Wizard - I/O Device connection schedule

Express Communications Wizard - I/O Device connection schedule

This form allows you to define the details of the communications schedule for your I/O Device and indicate that your I/O Device is remote by checking the PSTN box.

Options	Description
Connect I/O Device to PSTN	Check this box to indicate that the I/O Device is a dial-up I/O Device (connected to a PSTN - Public Switched Telephone Network). **Note:** Even if your I/O Device is not connected via a modem, you need to still check this box to schedule communications (but leave the Phone number to dial and Caller ID fields blank). Once you have completed your I/O Device setup using this Wizard, you need to go to the Ports form and change the Port number to the actual number of the COM port. You can choose to define the communication period in terms of **months**, **weeks**, **days**, or **hours, minutes, and seconds**. Alternatively, you can choose to communicate only at **startup** (persistent connection). Click a radio button to make your selection, then enter the start time and period as described below.
Synchronize at	The I/O Server will attempt to communicate with the I/O Device at this time, and then at intervals as defined below. This time is merely a marker for CitectSCADA. If you run up your project after this time, the I/O Server won't wait until the next day to begin communicating. It will operate as if your project had been running since before the start time.
Repeat Every	The time between successive communication attempts. (See Examples below.)
Phone number to dial	The telephone number that needs to be dialed to initiate contact with the I/O Device.

Note: These values can also be set using the I/O Devices form in the Project Editor.

Examples

All based on a **Synchronize at** time of 10:00:00:

- If you enter 12:00:00 in the **Repeat every** field, and start your project at 9 a.m., the I/O Server will communicate with the I/O Device at 10 a.m., then once every 12 hours after that, i.e. 10 p.m., then again at 10 a.m. of the following day, etc.
- If you enter 12:00:00 in the **Repeat every** field, and start your project at 4 p.m., the I/O Server will communicate with the I/O Server at 10 p.m., then again at 10 a.m. of the following day, etc. It will assume that communications were established at 10 a.m., so it continue as if they had been - communicating once every 12 hours after 10 a.m.
- If you enter 3 days in the **Repeat Every**, and start your project at 9 a.m. on a Wednesday, the I/O Server will communicate with the I/O Device at 10 a.m., then once every 3 days after that, i.e. 10 a.m. on the following Saturday, then at 10 a.m. on the following Tuesday, etc.
- If you enter the 6th of December in the **Repeat every**, and start your project during November, the I/O Server will communicate with the I/O Device at 10 a.m. on December 6, then again on December 6 of the following year, etc.
- Select **On Startup** for a persistent connection. To disconnect a persistent connection, you need to call the IODeviceControl() function with type 8.

See Also

Caller ID and commands
Express Communications Wizard - Link to external database

Caller ID and commands

Caller ID

A unique identifier which identifies a remote I/O Device when it dials back to the I/O Server. The caller ID can be any combination of alpha-numeric characters and/or the character '_' (underscore).

This ID will only be used if the I/O Device initiates the call to the I/O Server. If the modem initiates the call, you need to set the caller ID on the modem.

> **Note:** If you are multi-dropping off a single modem, use your I/O Devices to issue the caller ID, not the modem. This is because using the modem to issue the ID will send the same ID no matter which I/O Device the call is relevant to, which makes it difficult to identify the I/O Device that triggered the call.

By using the I/O Device to issue the ID, the I/O Server will receive a unique caller ID for each I/O Device. However, not every I/O Devices are capable of issuing caller IDs. If you are multi-dropping, use I/O Devices that can issue caller IDs.

Option	Description

[Event Commands] On connect	Cicode to be executed once the modem is connected and the I/O Device has come online (that is, before any read or write requests are processed).
[Event Commands] On disconnect	Cicode to be executed before the connection to the I/O Device is terminated (and after read and write requests are processed).

Note: These values can also be set using the I/O Devices form in the Project Editor.

See Also

Express Communications Wizard - Link to external database

Express Communications Wizard - Link to external database

This screen allows you to link to an external data source.

Option	Description
Link I/O Device to an external tag database	Determines whether or not you want to link the I/O Device to an external data source. If you link to an external data source, CitectSCADA is updated with any changes made to the external data source when a refresh is performed. If you disconnect an existing link, you can choose to make a local copy of the tags in the data source or you can delete them from CitectSCADA's variable tags data source altogether.
Database type	The format of the data referenced by the external data source.
External tag database	Specify the location of the external database. This could either be a path and filename of the external data source for the I/O Device, or the IP address/directory, computer name, or URL of a data server, etc. (for example "Work.CSV" or "127.0.0.1", "139.2.4.41\HMI_SCADA" or "http://www.abicom.com.au/main/scada" or "\\coms\data\scada"). Depending on the database type selected, the browse button could: display a standard Windows file browse dialog to browse for the external database.

Option	Description
	display a modal dialog to browse in a tree view of servers on the network connected to the computer. display the Unity Link Configuration Properties dialog.
Connection string	Enter a connection string to provide connection details for the data source. This is similar to an ODBC connection string. For example: UserID = XXX; Password = YYY or ServerNode=111.2.3.44; Branch=XXX Not every data source requires a connection string.
Add prefix to externally linked tags	Check this box if you want to insert a prefix in front of the names of linked tags in your Variable.DBF.
Tag prefix	The prefix that will be inserted in front of the names of linked tags in your Variable.DBF (for this I/O Device only). To change the prefix, delete it first, perform a manual refresh, then add the new prefix.
Automatic refresh of tags	Determines whether the linked tags in CitectSCADA's variable tags database will be updated when the external data source is changed (i.e. you manually change a field, etc.). This refresh will occur the first time you link to the data source, and then whenever you attempt to read the changed variables (for example you compile your project, display the variable using the Variable Tags form, or paste the tag, etc.). Without an automatic refresh, you will need to perform a manual refresh to update the linked tags in CitectSCADA.
Live Update	This field is only available if you have installed one of the CitectSCADAFastLinx products. It controls whether or not the linked tags in CitectSCADA and an external tag database will be synchronized if either database is changed. To enable live linking, choose **Yes** from the **Live Update** menu, and verify that the **Automatic refresh** check box is not selected. (Live Update and Automatic Refresh are mutually exclusive.) When Live Update is enabled and the CitectSCADA variable tag database is accessed (for example, during project compilation or when a dropdown list is populated), CitectSCADA queries the external tag database to

Option	Description
	determine if it has been modified. If so, CitectSCADA merges the changes into the local variable tag database. Conversely, any changes made to the local tag variable database will be incorporated seamlessly into the external tag database.

See Also

Express Communications Wizard - Serial device

Express Communications Wizard - Serial device

Since your protocol is based on serial communications you need to select which port on your computer will be used for the I/O Device.

The serial ports listed have been detected from your operating system registry. If you have correctly installed a proprietary serial board (for example a Digiboard) and the associated driver, the available port will be listed.

See Also

Express Communications Wizard - Summary

Express Communications Wizard - Summary

Summarizes the I/O Device setup using the information you provided. The summary shows the CitectSCADA communications setup and the recommended configuration of your I/O Device and/or internal boards.

Manually Configuring Communications

Usually the Express Communications Wizard is sufficient to set up your communications. However, if you need to manually configure communications, do the following:

1. Define an I/O Server in the I/O Server Properties form. This defines the name of the CitectSCADA server that the I/O Device will communicate with.
2. Complete the Boards Properties. This defines which board (on your CitectSCADA computer) to use to communicate (mother board, network card, serial board or a PLC communication card). Dial-up remote I/O Devices need to use a COMx board.
3. Complete the Ports Properties. Often boards have multiple communication ports and you need to specify the port to use. Some equipment can have several logical (virtual) ports assigned to the one physical port. If using modems, you need to specify a

unique port name for each and -1 for the port number. You need to also specify the communication parameters and any special behavior of that port.

4. Complete the I/O Devices Properties. This defines the I/O Device that CitectSCADA is talking to, by specifying the address. The protocol is also defined at this level.
5. Run the Computer Setup Wizard to complete configuration. This allows you to define your CitectSCADA computer as the I/O Server defined above. This is usually done after you compile the project.
6. If using dial-up remote I/O Devices, you need to complete the Modems dialog box. This defines how CitectSCADA uses a modem to communicate with remote I/O Devices.

Note: If there is no data to read or write, CitectSCADA will not communicate with an I/O Device regardless of whether it is defined or not. You need to create a variable tag and use it before CitectSCADA will do a read request. For example, use an integer variable to display a number on a page.

I/O Server Properties

To define an I/O Server, you need to specify a name on the I/O Server form. This name will be used to reference the I/O Server and has to be logical. For example, for a single I/O Server, you might use the name IOServer. When specifying the server name, you cannot use localized characters. See I/O Server Definitions.

If you are using multiple I/O Servers for redundancy (or to split the communications), you need to add a database record for each. Select a unique name for each I/O Server, for example IOServer1 and IOServer2.

Note: You need to add the record to the project database (use the Add Button at the bottom of the form) or replace the record (use the Replace Button at the bottom of the form) if you have changed the record.

Adding an I/O Server

To define a computer as an I/O Server, you need to run the Computer Setup Wizard on that computer. The wizard allows you to choose from a selection of the I/O Servers you have configured in your project.

See also

I/O Server Properties

Boards Properties

The properties of a board depend on the type of board installed in the I/O Server computer.

> **Note:** The term "Board" is a legacy term from the time when most ports on a computer were provided by additional boards. For example, an Ethernet card. The logical board concept is still useful even though a physical board may not exist (it may be software).

Boards have the following properties:

Board Name

A name for the board. 16 characters maximum. For example Server1_Board1.

If you have more than one board in your I/O Server computer, the name of each board needs to be unique. If you have multiple I/O Servers, the board name need only be unique within each server. For example Server1_Board2

Board Type

The type of board. 16 characters maximum.

If you are using a serial board or your computer's COM port, enter COMx.

Address

The starting address of the Board. For example 0xCC00. 8 characters maximum.

You need to specify the address to match the switch settings on the board when it was installed in your computer. If you are using a serial board or your computer's COM port, enter 0 as the address.

> **Note:** If more than one board is installed in the same computer, use a different memory address for each board.

I/O Port

The I/O port address of the Board. 8 characters maximum.

You need to specify the address to match the switch settings on the board when it was installed in your computer.

> **Note:** If you are using your computer's COM port not enter the port address here. specify the port number in the Ports form.

Interrupt

The interrupt number used by the Board. This is not necessary if using your computer's COM port.

Special Opt

Any special options supported by the board. 32 characters maximum. Please check the Hardware Arrangements Help Topic for your specific I/O Device to see if specific options are necessary.

Comment

Any useful comment. 48 characters maximum.

Ports Properties

The properties of a port depend on the type of board installed in the I/O Server, and on the I/O Device connected to the port. Ports have the following properties:

Port Name

A name for the port connected to your I/O Device(s). 31 characters maximum. Each port needs to have a unique name (i.e. you cannot assign the same Port Name to two ports in your system). You can use any name, for example: Board1_Port1

If you have more than one board in your computer, you can use the port name to identify the board, for example: Board2_Port1

Port Number

The port number that the I/O Device is connected to. 4 characters maximum. Do not assign the same Port Number to two ports on a board, unless connecting to a dial-up remote I/O Device via a modem (see the note below). Ports on different boards can be assigned the same number.

If you are using your computer's COM port enter the port number here - the port number is defined in the Ports section of the Windows Control Panel.

> **Note:** If you are connecting to a dial-up remote I/O Device (via a modem), you need to define a unique port name on the I/O Server for each dial-up remote I/O Device, and the port number needs to be -1 for each.

Board Name

The name you used for the board. 16 characters maximum. This is necessary to link the port to the board. For example Server1_Board1

Baud Rate

The baud rate of the communication channel (between the CitectSCADA I/O Server and the I/O Device). 16 characters maximum.

> **Note:** The I/O Device hardware and serial board may support other baud rates. If you do choose an alternative baud rate, verify that both the I/O Device and serial board support the new baud rate.

Data Bits

The number of data bits used in data transmission. You need to set your I/O Device to the same value or a communication link cannot be established.

Stop Bits

The number of stop bits used to signify the completion of the communication. You need to set your I/O Device to the same value.

Parity

The data parity used in data transmission.

Special Opt

Any special options supported by the port. 32 characters maximum. Please check the Hardware Setup Help Topic for your specific I/O Device to see if specific options are necessary.

Comment

Any useful comment. 48 characters maximum.

I/O Devices Properties

Configuring an I/O Device involves specifying its properties using the Project Editor. The properties depend on both the protocol and I/O Device.

To configure an I/O Device:

1. In the Project Editor, select **Communication | I/O Devices**. The I/O Devices dialog displays.
2. In the **Name** field, type in a name for your I/O Device (PLC). The name needs to be unique in the CitectSCADA system, unless the I/O Device is defined in other I/O Servers (to provide redundancy). If redundancy is used, the I/O Device needs to then have the same I/O Device number and address for each I/O Server. use different I/O Device names for your primary and standby I/O Devices, otherwise I/O Device Cicode functions cannot differentiate between them. 31 characters maximum.
3. In the **Number** field, enter a unique number for the I/O Device (0-16383). 8 characters maximum. The number needs to be unique in the CitectSCADA system, unless the I/O Device is defined in other I/O Servers (to provide redundancy). If redundancy is used, the I/O Device needs to then have the same I/O Device number and address for

each I/O Server. You may use the same device name, but if you want to use I/O Device Cicode functions, it is easier to have different I/O Device names.

> **Note:** For Numbers, Protocol and Port Type - The same network number is used for redundancy (as stated above), however this implies that the Protocol is usually the same (though there maybe some special circumstances where devices support multiple protocols) and the Port modes are similar. i.e. a real world port reference, DISKDRV or MEMORY . If DISKDRV is being used, then redundant units (i.e. the same NUMBER) needs to be DISKDRV. This also applies for MEMORY mode.

4. In the **Address** field, enter the address of the I/O Device. 64 characters maximum. What you enter in this field is determined by the type of I/O Device (and protocol) used, as each has a different addressing strategy.
5. In the **Protocol** field, select the protocol you are using to communicate to the I/O Device. 16 characters maximum. Many I/O Devices support multiple protocols, dependent on the communication method chosen.
6. In the **Port Name** field, specify the port on the board to which the I/O Device is connected. 31 characters maximum. This is necessary to link the I/O Device to the port. For example Board1_Port1.

> **Note:** There is a limit of 255 COMx ports on a server. To avoid this limitation restricting your number of remote I/O Devices, you can connect multiple remote I/O Devices to the same port as long as communication details (Telephone number, baud rate, data bits, stop bits, and parity) are identical.

7. In the **Startup mode** field select the type of I/O Device redundancy:

Primary	Enable immediate use of this communications channel. This is the default mode if no mode is specified.
Standby	This channel will remain unused until the I/O Device configured with the primary channel becomes inoperative.
StandbyWrite	This channel will remain unused until the I/O Device configured with the primary channel becomes inoperative. Write requests sent to the primary channel are also sent to this channel. **Note:** Do not use this mode for scheduled I/O devices because there is the possibility that the Standby I/O

	device write queue will never be cleaned up. When writing to a scheduled I/O device which it is not communicating, write requests are queued up until communication resumes. In case of using I/O device redundancy and StandbyWrite mode, Primary and Standby I/O devices have their own queues of pending write requests. If I/O device communication is controlled automatically by the communication schedule, then both queues are flushed when the schedule dial-time occurs. Where manual communication is controlled by the IODeviceControl Cicode function, only the Primary I/O device write queue is flushed and the Standby I/O device write queue will never be cleaned up.

> **Note:** To use Standby or StandbyWrite modes, you need to also configure a Primary I/O Device with the same I/O Device name, number and address.

8. In the **Priority** field type the order standby devices are promoted in if a primary I/O Device becomes inoperative during runtime. 8 characters maximum.
   If there is more than one standby I/O Device configured for a cluster, you can use this field to give precedence to a particular standby device. If this field is left blank, priority will be automatically allocated to the device during project compilation, based on the priority settings of other standby devices and/or the order the devices were configured in.
   See I/O Device promotion.
9. In the **Memory** field, specify whether the I/O Device runs in Memory mode. 8 characters maximum. The default value is FALSE. If you select TRUE, the I/O Device will be created in memory and its values stored in memory at runtime. This may be useful if you are testing your system, before connecting physical I/O Devices, as memory mode stops CitectSCADA from communicating with physical I/O Devices. See Using Memory Mode.

   > **Note:** If a device is configured with **Memory** set to TRUE, the **Port Name** and **Address** fields can be left blank as they will not be used by the compiler or the I/O server at runtime.

10. In the **Comment** field, type in any useful comment. This field is optional and is not used at runtime.
11. Click Add to add a new record, or Replace to replace an existing one.

See Also

Fields implemented with extended forms (press **F2**).

Communicating with Dial-up Remote I/O Devices

I/O Devices Properties Extended

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Linked

Determines whether or not the I/O Device is linked to an external data source. If you link to an external data source, CitectSCADA is updated with any changes made to the external data source when a refresh is performed.

If you disconnect an existing link, you can choose to make a local copy of the tags in the database or you can delete them from CitectSCADA's variable tags database altogether.

If an I/O Device is linked to an external data source the Database Type, External Database, Connection String, Tag Prefix and Live Update fields will be greyed out.

Database Type

The format of the data referenced by the external data source.

External Database

The path and filename of the external data source for the I/O Device. Alternatively, you can enter the IP address/directory, computer name, or URL of a data server, etc. (for example "C:\Work.CSV" or "127.0.0.1", "139.2.4.41\HMI_SCADA" or "http://www.abicom.com.au/main/scada" or "\\coms\data\scada").

Click **Browse** to select a path and filename.

Connection String

Enter a connection string to provide connection details for the data source. This is similar to an ODBC connection string. For example:

```
UserID = XXX; Password = YYY
```

or

```
ServerNode=111.2.3.44; Branch=XXX
```

Not every data source needs a connection string.

Tag Prefix

The prefix that will be inserted in front of the names of linked tags in CitectSCADA's variable tags database (for this I/O Device only). To change the prefix, delete it first, perform a manual refresh, then add the new prefix.

Automatic Refresh

Determines whether the linked tags in CitectSCADA's variable tags database will be updated when the external data source is changed (i.e. you manually change a field, etc.). This refresh will occur the first time you link to the data source, and then whenever you attempt to read the changed variables (for example you compile your project, display the variable using the Variable Tags form, modify or paste the tag, etc.)

Without an automatic refresh, you will need to perform a manual refresh to update the linked tags in CitectSCADA.

Live Update

> **Note:** This field is only available if you have installed one of the CitectSCADA FastLinx products.

Controls whether or not the linked tags in CitectSCADA and an external tag database will be synchronized if either database is changed. To enable live linking, choose Yes from the Live Update menu.

When Live Update is enabled and the variable tag database is accessed (for example, during project compilation or when a drop-down list is populated), CitectSCADA queries the external tag database to determine if it has been modified. If so, CitectSCADA merges the changes into the local variable tag database. Conversely, any changes made to the local tag variable database will be incorporated automatically into the external tag database.

Log Write

Enables/disables the logging of writes to the I/O Device. When enabled, writes are logged to the SYSLOG.DAT file in the **logs** folder of the CitectSCADA**User and Data** folder selected during installation, also specified in the INI file as [CtEdit]Logs. See Tag Functions for information about TagWriteEventQue and logging tag data.

> **Note:** Logging writes to every I/O Device may slow communications as the CitectSCADA system will be writing large amounts of data to disk. However, logging of writes is useful when debugging a system.

Log Read

Enables/disables the logging of reads from the I/O Device. When enabled, reads are logged in the CitectSCADA SYSLOG.DAT file.

Note: Logging reads to every I/O Device may slow communications as the CitectSCADA system will be writing large amounts of data to disk. However, logging of reads is useful when debugging a system.

Cache

Enables/disables data caching. When enabled, a cache of the I/O Device's memory is retained at the I/O Server, thus improving communications turn-around time.

Note: Data caching has to be enabled for scheduled I/O Devices, but disabled for memory or disk I/O Devices.

Cache Time

The cache time specified in milliseconds. When caching is enabled, data that is read from a I/O Device is stored temporarily in the memory of the I/O Server. If another request is made (from the same or another client) for the same data within the cache time, the CitectSCADA I/O Server returns the value in its memory - rather than read the I/O Device a second time. Data caching results in faster overall response when the same data is needed by many clients.

A cache time of 300 milliseconds is recommended.

The Cache Time for a scheduled I/O Device is automatically calculated. You can view a scheduled I/O Device's cache time using the Kernel Page Unit command.

Background Poll

Specifies that tags for a particular device are continuously polled at a minimum background poll rate. The options are True or False. Set this field to True if you are operating on a slow network, with a slow device, or where tags may normally only be polled infrequently.

Background Rate

When Background Poll is set to True, specifies the minimum rate by which the device will be polled. You may select any predefined value from the drop-down list, or enter your own in the format of HH:MM:SS. If you do not enter a value, the default value of 30 seconds will be used.

Min Update Rate

The DataSource will send tag update value notifications to subscription clients after a pre-defined period of time expires. You may select any predefined value from the drop-down list, or enter your own in the format of HH:MM:SS. If you do not enter a value, the default value of 0 seconds will be used and no update value notifications will be sent.

Staleness Period

Staleness period represents total number of seconds that will elapse after the last update before extended quality of the tag element is set to "Stale". You may select any predefined value from the drop-down list, or enter your own in the format of HH:MM:SS. If you do not enter a value, the default value of 0 seconds will be used and tag elements will not be set to "Stale".

Note: For further information on Minimum Update Rate and Staleness Period refer to Reading and Writing Tags.

Scheduled

Determines whether the I/O Device is configured for scheduled communications. This is normally set using the Express Communications Wizard.

Note: If you do not specify a schedule for a dial-up remote I/O Device, the connection will be established at startup and will remain connected until shut-down.

See Scheduled Communications.

Time

The I/O Server will attempt to communicate with the I/O Device at this time, and then at intervals as defined below. This time is merely a marker for CitectSCADA. If you run up your project after this time, the I/O server will not wait until the next day to begin communicating. It will operate as if your project had been running since before the start time.

Connect Action (254 Chars.)

Cicode to be executed once communication with the I/O Device has been established (and before any read or write requests are processed).

Period

The time between successive communication attempts. The period needs to be long relative to the driver's watchtime and the [Dial]WatchTime. If necessary, decrease these watchtimes or increase Period to help ensure that Period is more than double the watchtimes.

Examples (based on a Synchronize at time of 10:00:00):

1. If you enter 12:00:00 in the Repeat every field, and start your project at 9 a.m., the I/O Server will communicate with the I/O Device at 10 a.m., then once every 12 hours after that, i.e. 10 p.m., then again at 10 a.m. of the following day, etc.
2. If you enter 12:00:00, and start your project at 4 p.m., the I/O Server will communicate with the I/O Server at 10 p.m., then again at 10 a.m. of the following day, and so on. The I/O Server will assume that communications were established at 10 a.m., so it continue as if they had been - communicating once every 12 hours after 10 a.m.

3. If you enter 3 days, and start your project at 9 a.m. on a Wednesday, the I/O Server will communicate with the I/O Device at 10 a.m., then once every 3 days after that, i.e. 10 a.m. on the following Saturday, then at 10 a.m. on the following Tuesday, etc.
4. If you enter the 6th of December in the Repeat every, and start your project during November, the I/O Server will communicate with the I/O Device at 10 a.m. on December 6, then again on December 6 of the following year, etc.

Select On Startup for a persistent connection. To disconnect a persistent connection, you need to call the IODeviceControl() function with type 8.

Disconnect Action (254 Chars.)

Cicode to be executed once communication with the I/O Device has been terminated (and after read and write requests are processed).

Phone Number (64 Chars.)

The telephone number that needs to be dialed to initiate contact with the I/O Device. (i.e. for dial-up remote I/O Devices)

Caller ID (32 Chars.)

A unique identifier which identifies a remote I/O Device when it dials back to the I/O Server. The caller ID can be any combination of alpha-numeric characters and/or the character '_' (underscore).

This ID will only be used if the I/O Device initiates the call to the I/O Server. If the modem initiates the call, you need to set the caller ID on the modem.

Note: If you are multi-dropping off a single modem, use your I/O Devices to send the caller ID, not the modem. This is because using the modem to send the ID will send the same ID no matter which I/O Device the call is relevant to. This makes it difficult for you to identify the I/O Device that triggered the call.

By using the I/O Device to send the ID, the I/O Server will receive a unique caller ID for each I/O Device. However, some I/O Devices are not capable of sending caller IDs. If you are multi-dropping, use I/O Devices that can send caller IDs.

Persist

Set to TRUE or FALSE to indicate if the persistence of the tag cache file is on or off. Default is TRUE and the data will be written to the XML file as described in "File Name" below. If one of the primary/standby I/O devices (identified by the same network number) has persistence set to FALSE, then every device has it turned off.

Persist Period

Indicates how often to persist the XML cache file to disk. Default is 10 minutes (00:10:00). Normally, the period used is the highest value specified for one of the primary/standby I/O devices (identified by the same network number).

File Name

Indicates the path to the file on disk where the XML cache will be stored. You can specify just the path or the path with the actual file name. The default is Data directory using the format <ClusterName>.<IOServerName>.<IODeviceName>.cache. If the location cannot be created, then either the default path, default file name or both is used. Normally, the path used is the first non-empty path specified for one of the primary/standby I/O devices (identified by the same network number).

See Also

Communicating with Dial-up Remote I/O Devices

Working With Device Drivers

Drivers enable CitectSCADA to communicate with the devices in your system. Every communication setup in CitectSCADA needs drivers to implement the protocol and transport. Often this means two separate drivers, but in some cases both are combined into one driver – termed a ‘board’ driver.

CitectSCADA has over 140 device drivers available, enabling communication with a vast array of production devices across several communication types. These include generic drivers that support industry-standard protocols such as OPC and Modbus.

The driver you will need to communicate with a particular device is determined by:

- the device itself
- the communication protocol the device supports
- the communication channel used to connect the device to the CitectSCADA I/O Server.

To add a particular device to your CitectSCADA project, you will need to determine the driver necessary to support it. If the particular driver is not installed with your current version of CitectSCADA, you will have to obtain a driver pack and install it separately.

> **Note:** The word ‘driver’ is often used when referring to communications. A driver is a component piece of software that implements a protocol or transport (or both). Drivers are modular, such that new drivers can be easily ‘plugged-in’ to existing systems.

See Also

Determining Which Driver To Use

Installing a Driver Pack

Determining Which Driver to Use

To connect a device to CitectSCADA, you will initially need to determine which driver is necessary to support communication with the I/O Server. In most cases, you can use one of the following methods:

- a native driver, engineered specifically for the device
- a generic driver, available for devices that support industry-standard protocols such as Modbus or DNP
- an OPC Server, with communication taking place via a third-party application

initially check if a native driver is available. To do this, go to the **Supported Devices** page in the **Driver Reference Help** and locate the device you are trying to connect to. The list is sorted alphabetically by manufacturer name. (To access the Driver Reference Help, select **Driver Help** from the **Help** menu in Citect Explorer and Project Editor.)

If the device is not included on the list, consider the communication capabilities of the device to determine if it supports any industry-standard protocols, such as Modbus, DNP, COMLI or SNMP. If this is the case, you may be able to use one of the generic CitectSCADA drivers that support these standards.

Similarly, you can use OPC to connect a device to CitectSCADA, however this will require the purchase of a third party application to establish an OPC Server. If you take this approach, use an OPC Server application recommended by the manufacturer of the device you are trying to connect.

If these options do not provide an appropriate solution, you can contact Technical Support for this product.

Note: Check the Citect Driver Web at http://www.Citect.com/driverweb for information about driver updates and new driver pack releases.

See Also

Installing a Driver Pack

Installing a Driver Pack

If a necessary driver is new, or has been recently updated, it may not be included among those installed with your current version of CitectSCADA. If this is the case, you will need to obtain a Driver Pack and install it separately. See the Citect Driver Web at http://www.citect.com/driverweb.

You need to install a driver pack on the computer that will connect to the target device. This computer needs to be configured as a CitectSCADA I/O Server. It needs to be also be installed on any machines where the CitectSCADA project is compiled.

Before installing the driver pack, close your CitectSCADA runtime environment and Citect Explorer. You need administrator permissions for the I/O Server PC.

To install a driver pack:

1. Save the driver pack to an appropriate location on the I/O Server PC.
2. Double-click the EXE file to launch the installation.
3. Follow the instructions provided by the installation Wizard.
4. You are prompted to select the CitectSCADA installation folder. If you installed CitectSCADA in a different folder to the default, browse to that location. An alert message will alert you if you have selected the wrong folder.
5. Click **Finish** to complete the installation.

After you have installed a Driver Pack, the associated help will be automatically merged into the Driver Reference Help.

Note: If you are installing a driver pack on an older version of CitectSCADA that precedes the initial release of the driver, the supporting help may not appear automatically in the Driver Reference Help. If this is the case, you can locate the driver help in the CitectSCADA bin directory. The help file will be named <drivername>.chm.

See Also

The Driver Update Utility

The Driver Update Utility

The Driver Update Utility is a tool that determines whether an update is available for the CitectSCADA drivers you have installed on a computer. It achieves this by scanning a PC and comparing the locally installed drivers with those available on the Citect Driver Web.

When a scan is finished, a merged list of drivers is displayed. Each will fall into one of the following categories:

- Recommended Updates: drivers that you have installed for which there is a newer version available.
- Newly Available: drivers that you do not have installed that were recently released.
- Installed And Current: drivers that you have installed that match the latest version available.

- Unofficial Version Installed: drivers that you have installed that are newer than the version available on the DriverWeb. This would usually mean that you are running a tech preview or field test version, or you have a Hotfix installed.

Once you have viewed the list, you can download a driver update (or any of the new drivers) directly from the Update Utility.

Installation

The Driver Update Utility operates independently to CitectSCADA and is installed separately. The installer is available in the Extras directory on the CitectSCADA installation disk, or you can obtain a copy from the Citect DriverWeb at http://www.citect.com/driverweb.

Note: The Driver Update Utility needs an Internet connection to operate successfully. It uses a secure 128-connection when connecting to the DriverWeb.

See Also

Installing a driver pack

Using the Driver Reference Help

Each driver, whether it is installed with CitectSCADA or as part of a driver pack, is supported by the Driver Reference Help.

This where you will find the specific reference information related to each driver. This information includes:

- the device(s) each driver supports
- the address used to identify and locate a device
- any necessary device setup information
- the information necessary to configure a device in your CitectSCADA project
- the data types each driver supports
- parameters you can use to customize a driver
- driver-specific errors

Much of this information you will need to establish communication with a device; some of it is more suited to advanced engineering tasks and troubleshooting.

To access the Driver Reference Help, select **Driver Help** from the **Help** menu in Citect Explorer and Project Editor.

If you familiarize yourself with the processes explained in Setting Up Communications, you will be instructed on how to use this information to configure devices within a CitectSCADA project.

Customizing Communication Using Citect.ini Parameters

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Do not under any circumstances change or remove any of the undocumented Citect.ini parameters.
- Before deleting sections of the Citect.ini file, confirm that no undocumented parameters will be deleted.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Note: Always seek the advice of Technical Support personnel regarding undocumented features.

Citect.ini parameters can be used to tune the performance of a CitectSCADA driver and perform runtime maintenance diagnostics.

You can customize the way CitectSCADA communicates with the a driver (and even individual PLCs) by creating or editing the section of the Citect.ini file named after a specific driver, for example, [MODBUS].

There are some ini parameters that are common to every CitectSCADA driver, as well as custom parameters unique to each driver.

See the Driver Reference Help (**Help** menu | **Driver Help**) for the ini parameters related to each driver, and their default values.

When CitectSCADA starts at runtime, it reads configuration values from the Citect.ini file that is stored locally. Therefore, any configuration settings needs to be included in the Citect.ini file located on the computer acting as the I/O Server to the devices in the system.

By default, CitectSCADA looks for the Citect.ini file in the CitectSCADA project\bin directory. If it can't find the file there, it searches the default Windows directory.

Using a Disk I/O Device

In addition to connecting to actual I/O devices, CitectSCADA supports the configuration of disk I/O devices, which exist only within your computer. The value of each variable in the disk I/O device is stored on your computer's hard disk.

> **Note:** With the release of V7.20, a feature called persistence can be applied to I/O devices in memory mode. Using a Persisted Memory I/O device is recommended as an alternative to using a disk I/O device, as full synchronization is supported if a server becomes unavailable for a period of time. See Persisted I/O Memory Modefor more information.

Once configured, disk I/O devices appear exactly as any other I/O device in your system, but are not connected to any field equipment. Disk I/O devices can contain any type of variable supported by CitectSCADA, and you can configure them to emulate any supported I/O device. You can also specify a generic protocol for a disk I/O device.

A disk I/O device is useful when the status of your plant needs to be restored after a planned or unplanned system shutdown. You can configure your system to continually update a disk I/O device with the subset of variables that defines the status of your plant. When you restart your system after a shutdown, CitectSCADA can restore this status immediately.

You can also use disk I/O devices for storing predefined data that needs to be recalled immediately when a process is necessary (for example, in a simple recipe system).

> **Note:** If you create a RAM disk in the computer for the disk I/O device, you do not need to create or copy the disk file to the RAM disk. CitectSCADA automatically creates a disk file on startup. Do not set up your Disk I/O device on a RAM disk if you want the data written to it to survive a power cycle of the Servers.

See Also

Disk I/O Device setup
Redundant Disk I/O Devices
Using Memory Mode

Disk I/O Device setup

To set up communications with a device, follow the basic steps given in the I/O Device setup procedure below.

Sometimes you might need to edit the communications forms directly. They require the following specific information.

- You do not need to complete a Boards dialog box.
- You do not need to complete a Ports dialog box.
- Complete the I/O Devices dialog box as follows.

Perform the setup by entering the following information:

Text Box	Required information
I/O Device name	A name or your Disk I/O Device, for example: DISK_PLC. Each I/O Device needs to have a unique name in the CitectSCADA system.
I/O Device number	A unique number for the disk I/O Device (1-4095). Each I/O Device needs to have a unique number in the CitectSCADA system.
I/O Device address	The path and filename of the disk file, for example: [RUN]:DSKDRV.DSK If you are using redundant disk I/O Devices, specify the path to both the primary file and the Standby file in the configuration of both disk I/O Devices. For example, if this is the primary disk I/O Device, enter: Primary_File, Standby_File If this is the Standby Disk I/O Device enter: Standby_File, Primary File **Primary_File** is the name (and path) of the primary disk I/O Device file. You may use path substitution in this part of the field. **Standby_File** is the name (and path) of the Standby Disk I/O Device file. You may use path substitution in this part of the field. **Notes:** ● If the specified disk I/O device file is not found, CitectSCADA creates a new (empty) file with the specified filename. ● To use redundant disk I/O devices, you need to use a network that supports peer-to-peer communication. ● Disk files need to have write permission (on both primary and standby servers). ● The frequency with which CitectSCADA writes data to the disk I/O device(s) is determined by the [DiskDrv]UpDateTime parameter.
I/O Device protocol	To use the CitectSCADAgeneric protocol, enter: **GENERIC** - or - To select a specific protocol supported by CitectSCADA, see the I/O Devices online Help.
I/O Device port name	You need to use: **DISKDRV**
I/O Device comment	Any useful comment.

Text Box	Required information
I/O Device startup mode	If not using redundant disk I/O Devices, leave this property blank. If you are using redundant disk I/O Devices, use either: **Primary**: Enable immediate use of this disk I/O Device **StandbyWrite**: This disk I/O Device will remain unused until the computer with the primary disk I/O device configured becomes inoperative. Every write request sent to the primary disk I/O Device is also sent to this disk I/O Device. To use StandbyWrite mode, you need to also configure an I/O disk device in the primary server. Both I/O Devices need to have the same I/O Device name and number.
I/O Device Log Write, Log Read, Cache and Cache Time	Leave these properties blank.

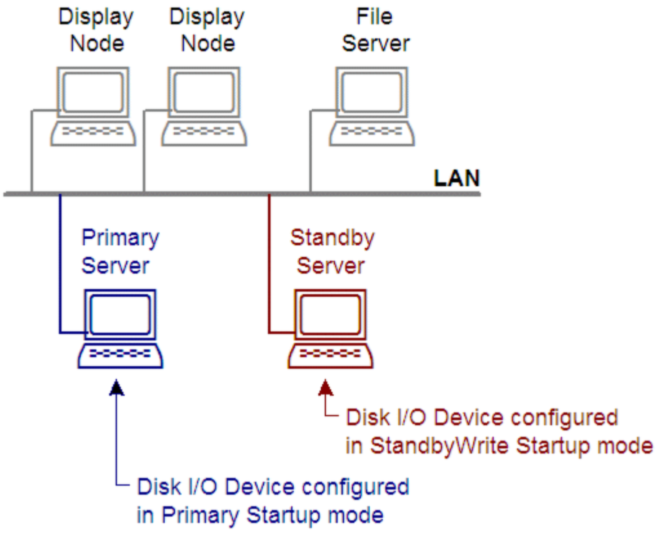
See Also

Redundant Disk I/O Devices

Disk I/O Device Errors

Redundant Disk I/O Devices

If using a network, you can configure a redundant disk I/O Device to minimize the potential for data loss (in the event the server becomes inoperative). This diagram illustrates the use of redundant disk I/O Devices:


Display Node
Display Node
File Server
LAN
Primary Server
Standby Server
Disk I/O Device configured in StandbyWrite Startup mode
Disk I/O Device configured in Primary Startup mode

When the system is operating, CitectSCADA reads and writes runtime data to the disk I/O Device configured in the primary server. It also writes runtime data to the disk I/O Device configured in the standby server. (CitectSCADA maintains both disk I/O Devices identically.)

If the primary server becomes inoperative, the Disk I/O Device in the standby server is activated without interrupting the system. When the primary server becomes active, CitectSCADA automatically returns control to the primary server, and copies the Disk I/O Device from the standby server to the primary server. The Disk I/O Device in the standby server reverts to its standby role.

To define a redundant disk I/O Device:

1. Configure a new disk I/O Device
2. Select **StandbyWrite** for the Startup Mode.

For redundant disk I/O Devices, you need to use Microsoft Networking (or another peer-to-peer network), and the hard disk of the standby server (the directory where the disk I/O Device file is stored) needs to be shared. Use Windows Explorer to set the directory to shareable.

See Also

I/O Devices Properties
Performance Considerations
Disk I/O Device Errors

Disk I/O Device Errors

The following errors, listed in order, are specific to the CitectSCADADISKDRV driver

Errors	Message
48	Error Opening DiskFile
49	Error in DiskFile Header
50	DiskFile Header contains invalid variable
51	DiskFile out of memory error
52	Read Error in DiskFile
53	Write Error in DiskFile
54	Seek Error in DiskFile
55	Error creatng queue
56	Error creating thread
57	Error creating event

See Also

Disk I/O Device Setup
Redundant Disk I/O Devices

Using Memory Mode

When configuring an I/O Device, you have the option to set them to memory mode. This means that the I/O Device will be created in memory and its values stored in memory at runtime.

Refer to I/O Devices Properties for more information on how to configure an I/O Device.

Devices using memory mode are not connected to any hardware and write their values to a cache. The I/O Device values can be read by many processes. The difference between a local variable and a device in Memory Mode is that an I/O Device in Memory Mode will reside in the I/O Server's memory and will observe standard networking and redundancy rules of a standard I/O Device.

Memory mode is useful when you are configuring a system for the first time, as you can design and test your system before connecting a physical I/O Device.

An I/ODevice with Memory set to TRUE or FALSE will behave differently in instances where Variable Tags share the same address.

For example:

Define a MODNET IODevice:
1 - INT1 (INT Tag)
2 - DIG1 (DIG Tag for bit0 of INT1)
3 - DIG2 (DIG Tag for bit1 of INT1)

At runtime the Display Client will show:

1. IODevice Memory mode = FALSE, at runtime setting DIG1 = 1, DIG2 = 1, INT1 = 3, modifiyng DIG1 and/or DIG2 affects INT1.
2. IODevice Memory mode= TRUE, at runtime setting DIG1 = 1, DIG2 = 1, INT = 0 or cached value, modifiyng DIG1 and/or DIG2 does not affect INT1.

In both instances the Override and Control Mode settings for each tag (INT1, DIG1, DIG2) did not affect the other tag. Setting INT1 in Override or Control Inhibit mode did not set DIG1 or DIG2 in Override or Control Inhibit mode.

As with local variables, values in an I/O Device using only memory mode are not retained when you shut down. However, if you set the Persist field to TRUE in the in the extended section of the I/O Devices Properties dialog their values will be retained. For more information on local variables, refer to Configuring Local Variables.

See Also

I/O Devices Properties

Persisted I/O Memory Mode

Using Persisted I/O Memory Mode

With the release of V7.20, a feature called persistence can be applied to I/O devices. When implemented, the tag cache for an I/O device is written to an XML file at a set period, allowing the last available data to be reloaded following a shutdown.

Persistence is enabled using the **Persist** field in the extended section of the I/O Devices Properties dialog.

You can also create a persisted memory I/O device using the Express Communications Wizard. When this type of device is selected, the wizard creates a new I/O device with the **Address** and **Port Name** properties left blank, **Memory** set to “TRUE”, and **Persist** set to “TRUE”. The **File Name** field is left empty to allow runtime to generate a default cache file name. This file will be located in the [DATA] directory.

Migrating Disk I/O devices

When applied to I/O Devices in memory mode, persistence provides an improved alternative to a disk I/O device, as synchronization is supported in scenarios where a server becomes unavailable for a period of time.

Many customers use disk I/O devices to provide system-wide global variable tags that are managed by I/O servers and are persisted to disk to maintain their latest values. Disk I/O devices take advantage of the standard I/O system redundancy features, such that, if one I/O server is unavailable, another can provide client(s) with tag values. They also perform a level of synchronization by using features such as standby write and by providing redundant paths to the persisted binary data files, so that, at startup of an I/O server, the latest value can be read into the system from the latest modified data file.

However, there is no synchronization when network connections are inoperative and regained, resulting in several scenarios in which redundant disk I/O devices can end up with different values for the same tag. For this reason, it is recommended that data assigned to disk I/O devices be migrated to the new persisted memory I/O mode available in V7.20.

To migrate disk PLC devices to persisted I/O memory mode:

1. Perform the procedures listed in Upgrading to CitectSCADA v7.20, as is appropriate.
2. On the I/O Device dialog, for disk I/O devices set Background Poll to TRUE and set Persist to TRUE (the default setting).
3. Start the I/O server up and wait for a few minutes so that the background poll updates every tag.
4. Shutdown the I/O server.

5. On the I/O Device dialog, for disk I/O devices set Background Poll back to FALSE, set Memory to TRUE and clear out the Port Name field as it is not necessary for Memory Mode.
6. Restart the I/O server to finish the migration to persisted memory I/O mode, with your devices now containing your original values from the disk I/O device.

See Also

Using Memory Mode

Troubleshooting Device Communications

Debugging device communications involves the following processes:

- gathering system information about current communication status
- analyzing the available information to identify problems

The information needed to facilitate this is available from the following resources:

- hardware alarms
- driver errors
- log files (system, trace, debug and driver logs)

This information can be analyzed directly to identify problems, or you can use the CitectSCADA Kernel to perform low-level diagnostic and debugging operations.

There are also procedures you can follow to debug low-level communications protocols, such as COMx and TCP/IP.

See also

Gathering communications information
Debugging communications

Gathering information about device communication

If you cannot establish communication with a device, check the following resources for evidence of a problem:

- Hardware alarms
- CitectSCADA log files
- Driver errors
- Citect Kernel

Hardware Alarms

When an error occurs in a CitectSCADA operation, a hardware alarm is generated. Hardware alarms are typically displayed on a dedicated page within a project, allowing an operator to monitor the status of a system's infrastructure.

Hardware alarms will indicate if break in communications has occurred, if Cicode can't execute, if a server becomes inoperative, and so on. If you are having problems communicating with a device, initially check the hardware alarms page for evidence of an error.

See Hardware Alarms.

CitectSCADA log files

CitectSCADA supports a number of log files that track operational status during runtime.

The primary log file, syslog.dat, contains a log of system information; from low-level driver traffic and Kernel messages, to user-defined messages.

The **Log Read** and **Log Write** fields in the I/O Devices Properties dialog determine if logging is enabled for each I/O device.

Driver logs are also available. These relate to the operation of a particular driver and are named accordingly. For example, the OPC driver is logged in 'OPC.dat'.

For a description of the available log files and where they are stored, see Log files.

Driver errors

There are two types of driver errors that are logged to syslog.dat:

- generic
- driver-specific

Generic errors are common to many drivers, and are mapped to general hardware errors.

Driver-specific errors are unique to a particular driver. A driver will convert a specific error into a generic error so that it can be recognized by the hardware alarm system. However, if further investigation is necessary, you can refer to a description of the original error in the Driver Reference Help. This includes a description of the errors associated with each driver.

See Driver Error Messages.

Citect Kernel

The Citect Kernel is a gateway into the internal workings of CitectSCADA at runtime, and is provided for diagnostics and debugging purposes.

The Kernel can display several different diagnostic windows each providing an active view of the CitectSCADA runtime system.

The Citect Kernel Driver window, launched via the Page Driver Kernel command, displays information about each driver in the CitectSCADA system. The statistics it presents include read requests, physical reads, digital reads per second, register reads, cache reads, error count, timeouts, and so on.

See Using the Citect Kernel.

See also

Debugging communications

Debugging I/O Devices and Protocols

Before commissioning any system, test communications between CitectSCADA and the I/O Devices. Many people leave this last, only to uncover communication issues that they cannot resolve.

The following information is provided to encourage you to test your communications thoroughly before it becomes a time critical-element in the job. It will also help you to debug communications and protocol conditions yourself.

See Also

Creating a communications test project
Debugging a COMx driver
Debugging a TCP/IP driver
Debugging a protocol driver using serial communications
Debugging proprietary board drivers
Contacting Technical Support

Debugging a COMx driver

A COMx driver is the board driver typically used for serial communications. Since version 2.01, the COMx driver allows you to dump debug information.

Three files are produced for each com port: a write file, a read file and status file. The debug files are configured by settings in the `Citect.ini` and are written and are written to the path specified in [CtEdit]Logs.

The following `Citect.ini` entries are used:

Parameter	Default	Comments
WritePortName	(no value)	Port name as defined in the Ports form
WriteFileSize	1000	Size in KB
WriteDebugLevel	0	1 to enable debugging, 0 to disable
ReadPortName	(no value)	Port name as defined in the Ports form
ReadFileSize	1000	Size in KB
ReadDebugLevel	0	1 to enable debugging, 0 to disable
StatusPortName	(no value)	Port name as defined in the Ports form
StatusFileSize	1000	Size in KB
StatusDebugLevel	0	1 to enable debugging, 0 to disable
CharTimeOut	11	CharTimeOut specifies the maximum acceptable time (in milliseconds) to elapse between the arrival of two characters on the communication line. During a ReadFile operation, the time period begins when the first character is received. If the interval between the arrival of any two characters exceeds this amount, the ReadFile operation is finished and any buffered data is returned. A value of zero indicates that interval time-outs are not used.
PassIfAnyInitPortsPass	0	Set to 1 if you wish to start when a COMx port is missing.

Example

```
[COMx]
WritePortName=PORT_1,PORT_2
WriteDebugLevel=1
WriteFileSize=2000
ReadPortName=PORT_1
ReadDebugLevel=1
ReadFileSize=1000
StatusPortName=PORT_2
StatusDebugLevel=1
StatusFileSize=10
```

The above example would:

- Log to "write" files up to 2000 kb of the data sent by the CitectSCADA driver to both PORT_1 and PORT_2;
- Log to a "read" file 1000 kb of the data received by the CitectSCADA driver from PORT_1; and
- Log up to 10 kb of status data from PORT_2 to a "status" file.

This would also result in the following text files being created:

- WPORT_1.dat
- WPORT_2.dat
- RPORT_1.dat
- SPORT_2.dat

In general, the format for the file names is "R", "W" or "S" followed by the Port name requested, followed by ".dat". The file names represent the corresponding "Read", "Write" and "Status" files.

File formats

The **Write** file will adopt the following format:

LINE 1	WRITE Debug file Started PORTNAME Debug Level 1 DOW MONTH DOM HH:MM:SS.sss
LINE 2	HH:MM:SS.sss
LINE 3	Out W In X nBytes Y Status Z
LINES 4…	AA BB CC ..

where:

W & X =	Values of buffer pointers
Y =	Number of bytes written
Z =	Return status of the WriteFile
AA BB CC =	Values in hex of each byte written

Example:

```
WRITE Debug file Started PORT1_BOARD1 Mon Dec 15 16:07:.998
16:07:09.810
Out 0 In 8 nBytes 8 iStatus 997
0e 02 00  00 10 00 00
16:07:10.802
Out 8 In 16 nBytes 8 iStatus 997
0e 02 00 00 00 10 00 00
..
```

The **Read** file will adopt the following format:

LINE 1	READ Debug file Started PORTNAME Debug Level 1 DOW MONTH DOM HH:MM:SS.sss
LINE 2	HH:MM:SS.sss
LINE 3	Out W InRx X nBytes Y Status Z
LINES 4…	AA BB CC ..

where:

W & X =	Values of buffer pointers
Y =	Number of bytes read
Z =	Number of characters remaining in the buffer
AA BB CC =	Values in hex of each byte written

Example

```
READ Debug file Started PORT1_BOARD1 Mon Dec 15 16:07:07.998
16:07:09.830
Out 0 In 0 nBytes 8 iStatus 0
0e 02 00 00 00 10 00 00
16:07:10.822
Out 0 In 8 nBytes 8 iStatus 0
0e 02 00 00 00 10 00 00
```

The **Status** file will adopt the following format:

LINE 1	STATUS Debug file Started PORTNAME Debug Level 1 DOW MONTH DOM HH:MM:SS.sss
LINE 2	HH:MM:SS.sss
LINE 3	modemStatus X

Example

```
STATUS Debug file Started PORT_1 Debug Level 1 Wed Nov 05
15:28:55.310
15:29:55.950
modemStatus 34
..
```

More parameters might be added later, so check the COMx information and the Citect Knowledge Base regularly.

See Also

Debugging a TCP/IP driver

Debugging a TCP/IP driver

A TCP/IP driver is the low-level driver that is used for any TCP/IP communications. It might be over Ethernet, Token Ring or Arcnet. This driver communicates between CitectSCADA and Winsock, so it uses the normal networking functionality of Windows. The parameters used for TCP/IP are:

- **[TCPIP]Log**
  Dumps the traffic between the CitectSCADA TCPIP.DLL and Winsock to a text file called TCPIP.DAT in your Logs directory. The size of the log is governed by the LogsSize parameter
  Allowable Values:
  1 - Enables logging
  0 - Disables logging
  Default value = 0
- **[TCPIP]LogSize**
  Sets the default maximum size (in kilobytes) of the TCP/IP driver's log file before wraparound occurs.
  Allowable Values: any integer
  Default value: 200
- **[TCPIP]LogTxRx**
  Log actual data Tx & Rxs besides setup info.
  Allowable Values:
  1- Enables logging of Tx & Rx
  0 - Disables logging
  Default value -= 0
- **[TCPIP]NoBufferSleepTime**
  Sleep time in ms to wait AFTER getting a WSAENOBUFS (no system buffers message) before reading another buffer. There is no need to adjust this.
  Default value: 0
- **[TCPIP]PortName**
  PortName to log.

Default value: * (by default every port is traced)

- **[TCPIP]Timeout**
  Defines the timeout period between connection retries.
  Default value: 1000 (ms)
- **[TCPIP]PortName.MulticastAddress**
  This parameter allows you to implement the TCPIP driver option "-Maa.bb.cc.dd"in the Citect.ini file. This may be necessary if you have run out of characters in the Special Options field of the Ports Form, which is limited to 16 characters.
  You need to specify the following in the Citect.ini file:

```
[TCPIP]PortName.aaa.bbb.ccc.ddd
```

where:
PortName = the name of the port as entered in the Ports Form
aaa.bbb.ccc.ddd = the multicast Class D IP address
See TCP/IP driver special options reference.

Procedure for debugging TCP/IP

Use the following steps for debugging:

1. Check if you can PING your target I/O Device. If you can't, CitectSCADA cannot talk to it. To do this open up a Command Window and type **PINGaaa.bbb.ccc.ddd** where a.b.c.d is the IP address of the I/O Device you are trying to connect to. If PING does not work, you need to go back to your Windows Networking and fix that.
2. Keep using your Simple As Possible Project (SAPP).
3. Delete any old `tcpip.dat` files.
4. Set the `Debug=1` and `Log=1` parameters in your `Citect.ini` file.
5. Start the project. From the information in the maximized Main window of the Kernel and the TCP/IP Debug window, you can see if CitectSCADA is sending requests to your I/O Device to initialize communications with it.

If there are no requests being sent, your software is improperly configured, and check that there were no errors on startup of CitectSCADA. If there were errors on startup, look them up in the Help. Also check that your computer is an I/O Server (and that it matches the one in your project). To do this run the Computer Setup Wizard, and configure the computer for a standalone configuration.

If there are requests, you can see communications between CitectSCADA and Winsock in a separate window, which displays the requests made by CitectSCADA to connect to the I/O Device and the corresponding response from the I/O Device. Any errors have a Winsock Error code that you can look up in the Microsoft Knowledge Base. If you see a **Connection OK** message, CitectSCADA will be able to come online.

See Also

Debugging a protocol driver using serial communications

Debugging a protocol driver using serial communications

To debug a protocol driver that uses serial communications, do the following:

1. Keep using your Simple As Possible Project (SAPP).
2. Set the "DebugStr=* all" for your protocol.
3. Backup and delete both the `syslog.dat` and `syslog.bak` files. The system will recreate a fresh version of this file the next time CitectSCADA is started.
4. Start the project. From the information you can see in the maximized Main window of the kernel be able to see if CitectSCADA is sending requests to your I/O Device to initialize communications with it.

If there are no requests being sent then your software is improperly configured, and check that there were no errors on Startup of CitectSCADA. If there were errors on startup look them up in the Online Help. Also check that your computer is an I/O Server (and that it matches the one in your project). To do this run the *Computer Setup Wizard*, and configure the computer for a standalone configuration.

If there are requests being sent but no reply, then CitectSCADA is trying to communicate. When CitectSCADA is sending requests but getting no reply, these are the most common causes:

- **The request CitectSCADA is sending is not getting to the I/O Device** - Check the Address field in the I/O Devices form, and verify that it is correct. If the I/O Device is one that needs a unique identifier (such as a node address), or you need some type of routing path, then verify that it is correct.
  Check that you have the same parameters in the Ports form that the I/O Device is using. If you have 8 data bits and the I/O Device uses 7 data bits, communications will not work.
  Check that your cable is OK. The easiest way to do this is to create a new project and use the Loopback protocol. You can use this to verify the **Tx** and **Rx** lines' integrity by placing a jumper on these lines. Initially test this with a jumper between pins 2 and 3 on your PC. Then plug in your cable and test again with the jumper between the **Tx** and **Rx** lines. Keep moving the jumper until it is at the end of your communications bus. You can find more information on using the Loopback protocol in the Citect Knowledge Base.
  Even if the Loopback protocol shows no errors, your cable might still be responsible for the loss of communications. CitectSCADA usually places a far higher constant load on serial communications than programming software does, this usually means that CitectSCADA will require much more stringent handshaking than the programming software. So it is possible that the cable you use to program your I/O Device works fine for programming, but not for CitectSCADA. Check the Wiring Diagram for your Protocol in the help.

Another major cause of improper cable connections is 9-pin to 25-pin converters. Many of these converters are made specifically for serial mice. These typically only use the **Tx**, **Rx** and **Ground** signals. If you use one of these converters they do not support any handshaking and will most likely not work for your Protocol.

If the above checks OK, use the parameters for COMx (as mentioned above) to create log files. Examine these log files and verify that what CitectSCADA thinks it is sending is actually what it is sending. The log files produced by using these parameters get their information from a lower level than CitectSCADA and show you exactly what is going through the COMx driver.

- **The Response from the I/O Device is not getting to CitectSCADA** - This is unlikely and usually caused by cabling that is damaged, has insufficient bandwidth, or is connected improperly.. Check your cabling as above. Also, check that you are specifying everything you need within CitectSCADA. Many protocols require CitectSCADA to send a unique identifier in its request packet. If this identifier is incorrect then the response cannot get back to CitectSCADA.
- **The I/O Device does not understand the Request** - Every CitectSCADA protocol can check if an I/O Device is running. Typically the protocol attempts to read data from the I/O Device, usually a status register or other register that is there. However, many pseudo-standard protocols, such as Modbus, do not conform to the exact specification for that protocol. Many protocols supplied with CitectSCADA have some extra parameters to allow you to choose the specific initialization request from CitectSCADA. These can be found in either the Online Help or the Knowledge Base. Check with the manufacturer of your I/O Device to confirm that it will respond to the request that CitectSCADA sends. If you are unsure of the request being sent for initialization, use DebugStr=* to get the actual variable address that CitectSCADA is asking for in its initialization.

  Check the protocol you are using. CitectSCADA may have many different protocols for communicating to an I/O Device. PLCs such as the AB PLC5 can use different serial protocols, depending on the method you are trying to use. Make sure you are using the correct one. If you are unsure, try the other possible protocols.
- **The I/O Device is not functioning properly** - There is usually some sort of software from the I/O Device manufacturer that can be used to diagnose any problems with the I/O Device.

See Also

Debugging proprietary board drivers

Debugging proprietary board drivers

Proprietary board drivers (such as the Allen Bradley KTX card, Modicon SA85 card, Siemens TIWAY card, and so on) have their own low-level drivers. Each driver has debugging parameters to make it easier to debug device behavior. Check the Citect-SCADA Knowledge Base and Help for parameters. The Knowledge Base has articles describing how to debug these board drivers.

The debugging process is exactly the same as with a serial connection:

1. Keep using your Simple As Possible Project (SAPP).
2. Set any debugging parameters for the protocol and board drivers.
3. Start CitectSCADA with clean log files.
4. Find any errors and then look them up in the manufacturer's documentation.

Serial Port Loop-Back Test

You can use the serial port loop-back test to test your serial hardware configuration. This test can be used with any COM port, whether it is local, or on a multi-port serial board (such as a Digiboard). The test can be performed internally or externally with loop-back cable attached.

See Also

Test Setup

Serial Port Loop-Back Cable

Test setup

1. Do not configure any other protocols. Temporarily delete other boards and units while performing this test.
2. Configure a unit for each port to be tested. Make the I/O Devices form look as follows:
   - **Name:** <unique name for the I/O Device>
   - **Number:** <unique network number for the I/O Device>
   - **Address:** NA
   - **Protocol:** LOOPBACK
   - **Port Name:** <the "Port Name" in the Ports form>
3. The following Citect.ini options are supported:

- [LOOPBACK]

LoopBack - Set this to 1 if internal loop-back is to be performed (make sure this is deleted after running the test). When set to 0, a loop-back connector which ties pins 2 and 3 together is necessary at each port.

> **Note:** The COMx driver does not support internal loop-back. The external loop-back is the default mode.

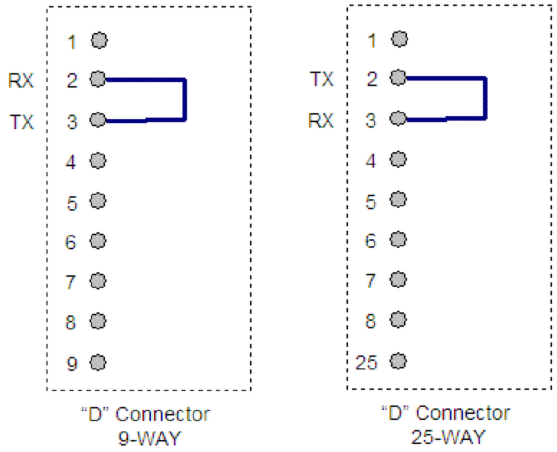
Size - Sets the maximum frame size. The length of each frame transmitted is random between 1 to 'Size'-1. The default size is 512.

4. Start up CitectSCADA. Each port will transmit a frame of random length. This process is repeated when the entire frame is received.
5. Open the kernel, type "page driver" and press **Enter**. Type **V** to set the display mode to 'verbose'. The following statistics appear:
   - Number of characters transmitted.
   - Number of frames transmitted.
   - Number of characters received.
   - Elapsed time in milliseconds.
   - Characters received per second.
   - Start time in milliseconds.
   - Total number of errors.
   - Error code of last detected error.

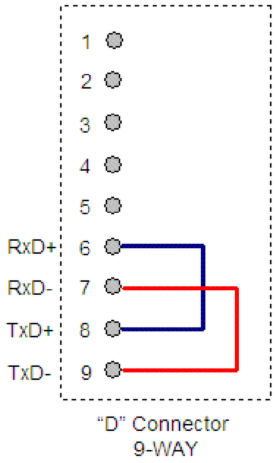
> **Note:** The total number of errors should be 0. If the number of errors reported is not zero, your serial hardware is not operating properly.

Serial port loop-back cable

The diagram below shows the loop-back connections to use with RS-232.



The diagram below shows the loop-back connections to use with RS-422 or RS-485.



Contacting Technical Support

If your debugging efforts do not succeed within a reasonable amount of time, contact Technical Support for this product and provide the following information:

- a Solution Request which you can obtain through Technical Support.
- `syslog.dat`, `syslog.bak`, `tcpip.dat`, or `COMx` log files.
- A copy of the SAPP, and the `Citect.ini` file.

See Getting Technical Support.

Performance Considerations

Many external factors influence the performance of control and monitoring systems. The capabilities of the computer, the I/O Device(s), and the communication pathway between them are obvious factors. The faster they can transfer data, the faster your system operates. (CitectSCADA always attempts to maintain a data transfer rate as fast as the I/O Device hardware can support.) The data transfer rate is hardware dependent: once you have installed the hardware, your ability to influence this characteristic is limited.

However, there is one area where you can directly affect the performance of your runtime system: the arrangement of data registers in the I/O Device(s).

See Also

Caching data
Grouping registers
Remapping variables in an I/O Device

Caching data

On large networked systems with many clients, you can improve communications turnaround time by using memory caching.

When caching is enabled, data that is read from an I/O Device is stored temporarily in the memory of the I/O Server. If another request is made (from the same or another client) for the same data within the cache time, the CitectSCADA I/O Server returns the value in its memory, rather than rereading the I/O Device. Data caching results in faster overall response when the same data is necessary by many clients.

A cache time of 300 milliseconds is recommended. Avoid using long cache times (in excess of 1000 milliseconds), as this may negatively impact the timely delivery of information to the system.

> **Note:** Do not use data caching for disk I/O devices as they implement a cache themselves.

How data caching works

Data caching prevents unnecessary rereads of I/O Device data within a short period of time. Unnecessary reads can be generated when more than one client requests the I/O Server to read data from a PLC or similar I/O Device within a short (typically 300 ms) period of time.

Normally, upon request from a client, an I/O Server reads status data from an I/O Device, and passes it back to the requesting client.

If the server receives subsequent requests from other clients before the original data is returned to the first client, it optimizes the read by automatically sending the original data back to requesting clients. (Page General Blocked Reads shows this count).

If a client requests the same data immediately after the server returned the data to a client, the server rereads the device unnecessarily.

Setting the data cache time to 300 ms (or similar) prevents identical repetitive reads within that cached time frame. If further clients request the identical data from the same server up to 300 ms after the server has sent that data to an earlier client, the cached data on the server is sent immediately in response to the subsequent requests.

> **Note:** Multiple clients don't have to be separate CitectSCADAs on a network. They may be the alarms and trend clients in the same computer, so this optimization will affect even a single node system.

CitectSCADA also uses read-ahead caching. When the data in the cache is getting old (close to the cache time), the I/O Server will re-request it from the I/O Device. This optimizes read speed for data that is about to be re-used (frequent). To give higher priority to other read requests, the I/O Server requests this data only if the communication channel to the I/O Device is idle.

Keeping a persistent record of the data

To keep a persistent copy of cached device data, you can save the I/O Server's cache to disk. For every `[IOServer]SavePeriod`, the data is saved to s, one for each cached device.

Saving the data to disk will, in most circumstances, allow you to shut down and restart the I/O Server without having to contact each I/O Device again to get its current values. Instead, you can read the values from the device's persistence cache.

> **Note:** When read-through caching is enabled for a remote or scheduled I/O Device, the persistence cache for that device is saved to disk when the active I/O Server disconnects from the device. This occurs regardless of the value set in `[IOServer]SavePeriod`. You can enable read-through caching by setting the parameter `[Dial]ReadThroughCache`.

See Also

Grouping registers

Grouping registers

When you configure a CitectSCADA system, you need to define each variable (register address) that CitectSCADA will read when your system is running. When your runtime system is operating, CitectSCADA calculates the most efficient method of reading registers. CitectSCADA optimizes communication based on the type of I/O Device and the register addresses.

When CitectSCADA requests data from an I/O Device, the value of the register is not returned immediately; an overhead is incurred. This overhead (associated with protocol headers, checksum, device latency, and so on) depends on the brand of I/O Device, and is usually several times greater than the time necessary to read a single register. It is therefore inefficient to read registers individually, and CitectSCADA usually reads a contiguous block of registers. Because the overhead is only incurred once (when the initial request is made), the overhead is shared across every register in the block, increasing the overall efficiency of the data transfer.

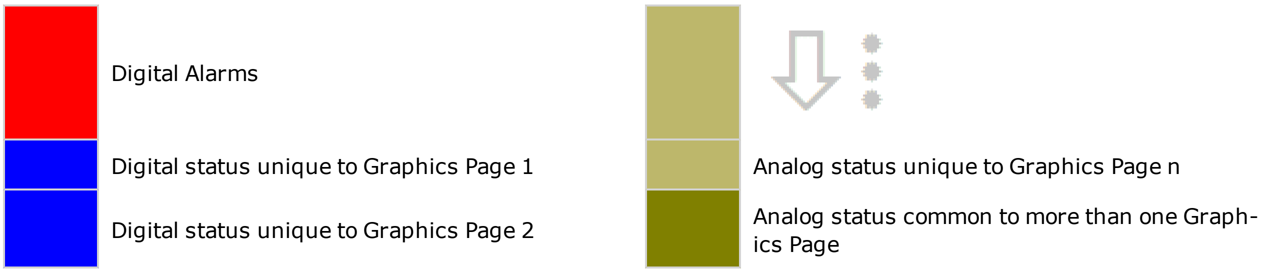
However, reading a block of registers where only a small percentage of the block is actually used is also inefficient. If the registers that your CitectSCADA system will read are scattered throughout the memory of your I/O Device, excessive communication will be necessary. CitectSCADA needs to either read many contiguous blocks (and discard the unused registers), or read registers individually, degrading system performance. You can avoid this by grouping the registers that CitectSCADA will read.

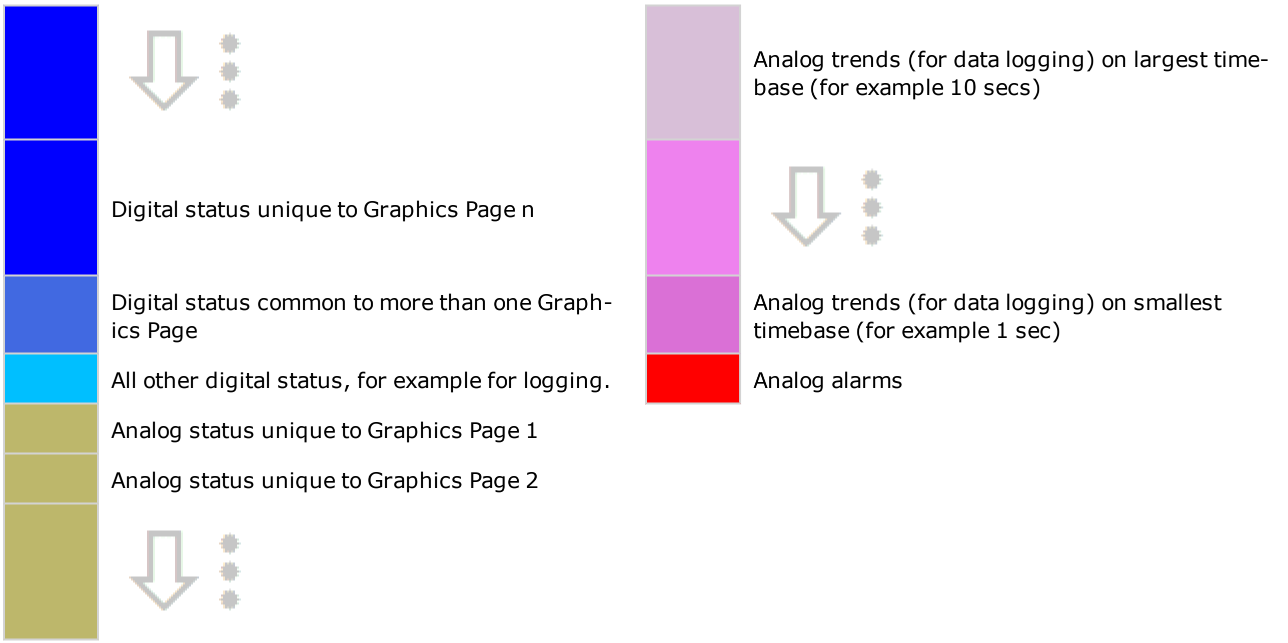
CitectSCADA continually reads registers associated with alarms. (If an alarm condition occurs, CitectSCADA can display the alarm immediately.) therefore group registers that indicate alarm conditions.

Registers associated with status displays (objects, trends, and so on) are only read as they are necessary (that is, when the associated graphics page is displayed) and are appropriately grouped according to the pages on which they are displayed.

Registers used for data logging are read at a frequency that you define. They are grouped according to the frequency at which they are read.

The following table shows an ideal register grouping for a CitectSCADA system:


Digital Alarms
Digital status unique to Graphics Page 1
Digital status unique to Graphics Page 2
Analog status unique to Graphics Page n
Analog status common to more than one Graphics Page


Digital status unique to Graphics Page n
Digital status common to more than one Graphics Page
All other digital status, for example for logging.
Analog status unique to Graphics Page 1
Analog status unique to Graphics Page 2
Analog trends (for data logging) on largest timebase (for example 10 secs)
Analog trends (for data logging) on smallest timebase (for example 1 sec)
Analog alarms

While memory constraints and the existing PLC program might impose limitations, grouping registers into discrete blocks, even if they are not consecutive blocks, will improve system performance.

When designing your system, allow several spare registers at the end of each block for future enhancements.

See Also

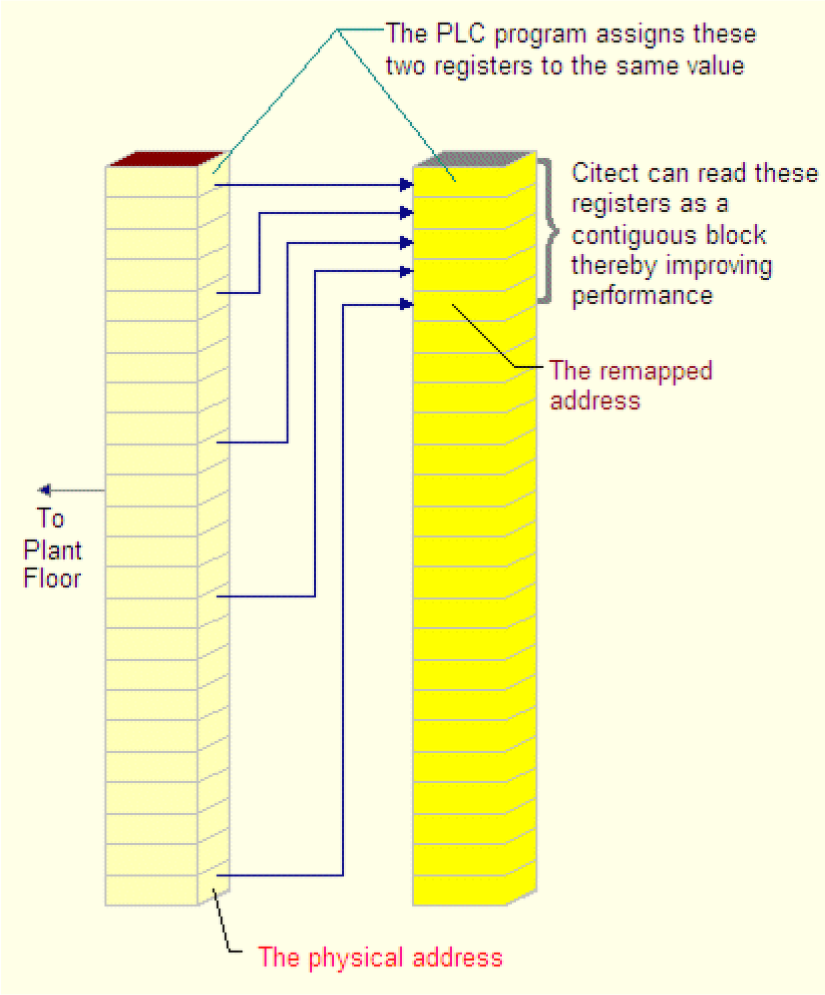
Remapping variables in an I/O Device

Remapping variables in an I/O Device

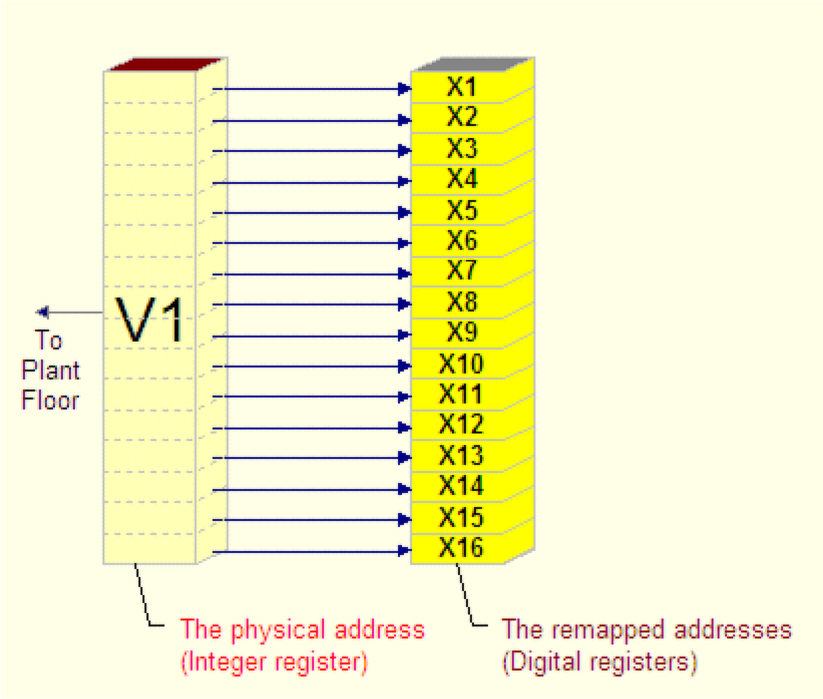
Some PLCs allow you to remap (or copy) an I/O Device variable to another register address. CitectSCADA allows you to remap to:

- Group registers more efficiently to increase performance.
- Allow CitectSCADA to interpret a variable type (for example, an analog variable) as a different variable type (for example, a digital variable). For example, you can create additional digital addresses if an I/O Device has run out of digital addresses.

To remap a variable in your PLC, you need to design (or modify) the logic in the PLC to associate both addresses. CitectSCADA can then read or write the variable to and from the remapped address instead of the physical address.


The PLC program assigns these two registers to the same value
Citect can read these registers as a contiguous block thereby improving performance
The remapped address
To Plant Floor
The physical address

You can also reassign one type of variable (for example, an integer) to another type of variable (for example, a digital variable).



To remap in CitectSCADA, first create the variables in your project as you would normally. Then you can set up the remapping, specifying that any variable with an address in the desired range will be remapped. The I/O Server will redirect the addresses at runtime as per the remapping instruction.

> **Note:** Not every PLC and/or CitectSCADA driver supports remapping. It is not recommend unless necessary. Contact Schneider Electric (Australia) Pty. Ltd. Support if you need to evaluate whether the PLC and/or driver support remapping.

To remap a variable in CitectSCADA:

1. Open Project Editor.
2. Choose **Communication** | **Remapping**. The Remapping dialog box appears.

Option	Description
CitectSCADA variable	The first remapped (CitectSCADA) variable defined in the variable tags database (using the Tags dialog box); for example: `Motor_1_Run`. (79 characters maximum). Alternatively, use the direct <Unit Name>|<Address>| format (using values specific to your I/O Device); for example: IODev|X1|. The address entered here is remapped. At runtime the I/O

Option	Description
	Server will read/write data through the physical address instead.
Length	The number of remapped variables. CitectSCADA reads enough physical variables to remap this number of variables (10 characters maximum). The length needs to be less than the maximum request length of the protocol. The protocol overview displays the maximum request length of the protocol.
Physical Variable	The first physical variable in the PLC, for example: ReMapIntV7. (79 characters maximum). This variable does not need to be defined in the variable tags database. You can use the <Unit Name>|<Address>| format (using values specific to your I/O Device). For example, IODev|V7|.
Remap Read	Determines whether to perform the remapping for reads. Set to TRUE or FALSE. FALSE - Read normal (not remapped) variables. The actual address of the CitectSCADA variables will be read directly from the I/O Device, instead of through the Physical Variable. (Use this mode if your I/O Device does not support remap reads.) TRUE - Read remapped variables (through the physical variable).
Remap Write	Determines whether to perform the remapping for writes. Set to TRUE or FALSE. FALSE - Write to normal (not remapped) variables. The actual address of the CitectSCADA variables will be written directly in the I/O Device, instead of through the physical variable. (You need to use this mode if your I/O Device does not support remap writes.) TRUE - Write to remapped variables (through the physical variable).
Comment	Any useful comment (48 characters maximum).

3. Complete the dialog box, and then click **Add**.

Note: To determine if the device supports remapped reads or writes, see the I/O Device data type Help.

See Also

Remapping example

Remapping example

Using the CCM protocol with a GE 9030 PLC, the following remapping could be used to optimize communication when reading some digital register values. This example is based on the assumption that the PLC is set up correctly for remapping.

Variable tags database

The digital register values are defined as follows.

Variable Tag Name	Motor_1_Running_Feedback
Data Type	Digital
I/O Device Name	Area1
Address	M1
Variable Tag Name	Motor_1_Overload
Data Type	Digital
I/O Device Name	Area1
Address	M3
Variable Tag Name	Motor_4_ Running_Feedback
Data Type	Digital
I/O Device Name	Area1
Address	M4
.	
.	
.	

Variable Tag Name	Motor_4_ Running_Feedback
Data Type	Digital
I/O Device Name	Area1
Address	M16

Notice that the address M2 is not used. Skipping addresses does not affect performance.

Remapping database

The remapping is defined as follows.

CitectSCADA Variable	Area1|M1|
Length	16
Physical Variable	Area1|R10|
Remap Read	True
Remap Write	False

The physical variable is an integer data type; it does not need to be defined in the variable tags database (but it can be).

Advanced Driver Information

This section provides advanced information about drivers.

See Also

Variable (Digital) Limitations
Validating distributed project data for tag-based drivers
Write Delay Issue

Variable (digital) limitations

Devices often have memory areas that are of a designated data type, like Byte, Integer, or Word. Some protocols do not support the reading and writing of data in these memory areas using a different data type. This situation is most common in the case of reading and writing of individual bits within the data types like Bytes, Integers, and Words.

In this case, reading individual bits within these larger data types is done by reading the designated data type and getting the CitectSCADA driver to sub-divide it into individual bits. Writing to bits within the larger data types is more complicated, as writing to one bit within the larger data type will at the same time overwrite the other bits within that same data type. To prevent overwriting existing bits when writing a new bit value, a 'read-modify-write' scenario can be used to write to a bit within the larger data type. Using this approach, the CitectSCADA driver will read the larger data type, modify the appropriate bit within the larger data type, and then write the larger data type back to the device.

While the 'read-modify-write' approach is necessary to avoid overwriting existing bits in the registers of larger data types, it can create an issue if the device being written to is also configured or programmed to modify these same registers. For example, if a PLC device modifies the registers of one of its larger data types after the CitectSCADA driver has read these same registers, but before CitectSCADA has written the modified value, the changes made by the PLC will be overwritten. This outcome can be avoided if CitectSCADA and any devices using these data types are configured so that only one or the other has write access at any given time.

This 'read-modify-write' method has a serious operational concern: if the device modifies the larger data type after the CitectSCADA driver has read it, but before CitectSCADA has written the new value, any changes made by the device are overwritten. This issue could be serious in a control system, and it is recommended that the device and CitectSCADA be configured so that only one of these systems writes to the data types of this kind.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

When the read-modify-write method will be used to alter a data type's bit values, configure your system so that Citect and the host device do not have simultaneous write access to the affected memory ranges.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Consider the following example:

1. The initial state of a PLC register is 0x02h.
2. The CitectSCADA driver reads the value of this register (effectively making a copy) in preparation for a change to bit 3.
3. However, before the driver writes its change back to the PLC, the PLC code changes the value of bits 0 and 4 of this register to 0x13h.

4. The CitectSCADA driver then changes bit 3 of its copy of the register to 0x0Ah. When it writes to the PLC, it overwrites the PLC's copy of the whole register (not just the changed bit). Because the PLC code modified bits 0 and 4 in the interval between CitectSCADA's read and write, these changes are overwritten.

Validating distributed project data for tag-based drivers

CitectSCADA uses numeric index values to uniquely identify the variable tags in a project. They are used as a reference point when requesting data from the I/O Server for a tag-based driver.

These index values are automatically generated when a project is compiled. Circumstances may arise where a distributed project has index values that represent different tag addresses on different computers. For this reason, CitectSCADA has a number of automatic checks in place that validate a project's tag index values and flag any discrepancies.

An initial security check takes place on client machines at a unit level, allowing a tag index mismatch to be isolated to a particular client before any requests are sent to the I/O Server. This confirms that the unit address, the unit type, the raw data type and the tag address match for index values across the client and server machines. Any discrepancies found are flagged by a hardware alarm on the client machine.

Each page is also checked to confirm that it was compiled against the current version of the variables database. There is also a check performed whenever a tag-based driver loads the variable database to test whether it matches the current tag addresses. The parameter TagAddressNoCase allows you to adjust the case-sensitivity of these checks.

In addition, CitectSCADA will also check if a project is currently running on the local machine when a compile is attempted, as this is one of the circumstances that may lead to mismatched index values.

If the project uses a tag-based driver and is currently in runtime, CitectSCADA will stop the compiler and generate an error in the error database noting that Citect32.exe was still running. The .ini parameter [General]CitectRunningCheck allows you to override this feature, however it is recommended that you leave it enabled so that your tag index values are assigned as intended.

Write delay effects

CitectSCADA performs writes to the I/O Device asynchronously (that is, when you write to the I/O Device, the write takes time to get to the I/O Device, during which CitectSCADA continues to perform other operations). If the other Cicode assumes that the write has completed immediately, you might encounter some side effects such as inconsistencies in the value of a variable tag across two Cicode threads.

If you have the following Cicode:

```
PLC_VAR = 1234;
Prompt("Variable is " + PLC_VAR : ####);
```

the first line is a write to the PLC.

When CitectSCADA executes the first line, it generates a request to the I/O Server to write the value 1234 into the PLC variable `PLC_VAR`. CitectSCADA then executes the next line of Cicode before the PLC write is completed. CitectSCADA does this so that the Cicode is not stopped while waiting for a slow I/O Device. As the write to the PLC has not completed, you might think that the next line of Cicode will display the last value of `PLC_VAR` and not the value 1234. However, this Cicode will display the correct value (1234) because whenever CitectSCADA writes to the PLC, it first updates its local copy of the variable: any following Cicode will get the correct value.

Sometimes this solution will not work as CitectSCADA might keep multiple copies of an I/O Device variable, and only update the one associated with the current Cicode. The other variables will contain the old value of the I/O Device variable until they are refreshed (with a read from the I/O Device). There is a separate data area for each display page, Cicode file, for alarms, trends and reports. If you write to an I/O Device variable from a page keyboard command, the copy of the I/O Device variable associated with that page will be updated; however, the copy associated with other pages and the Cicode functions is not updated until the next read (as determined by the `[Page]ScanTime` parameter). If you call a Cicode function that assumes the write has completed, it will get the last value.

The workaround is to write to the I/O Device variable in the Cicode function. Every Cicode function shares the same I/O Device variables, so the writes will operate as expected.

```
FUNCTION
TestFunc(INT nValue)
        PLC_VAR = nValue;
END
```

Another side effect of the delays inherent in asynchronous writes to I/O devices is that you might assume that CitectSCADA has successfully written to the I/O Device, when in fact the write operation might not yet have been completed or even attempted. In such cases, it is still possible that a hardware- or configuration-based error will prevent the success of the write operation. While such an unsuccessful write operation will generate an error, your Cicode cannot be notified or use the `IsError()` function because the Cicode has continued to execute past the initial I/O device write command. Therefore, when it is important to check the success of a write operation before allowing code execution to continue, you might insert the function `TagRead()` after the write and then verify the value of the variable. `TagRead()` forces Cicode to re-read the I/O Device variable so that you can check the new value. `TagRead()` is a blocking function. It blocks the calling Cicode task until the operation is complete.

```
PLC_VAR = 1234;
sTestStr = TagRead("PLC_VAR");
IF sTestStr <> 1234 THEN
        Prompt("Write not completed");
END
```

Here the data will be read from the physical PLC, not from the I/O Server cache, as the I/O Server will invalidate any cached data associated with a PLC write. This will allow you to test for a completed write. Please be aware that other Cicode tasks running at the same time will not be waiting on the TagRead, so they might see the old or new value, depending on if they are using the same copy of the PLC variable. You can also stop CitectSCADA from writing to the local copy of the variable by using the function `CodeSetMode(0, 0)`.

Communicating with Remote Devices via Modems

A dial-up remote I/O Device is one which is connected to CitectSCADA through a PSTN (Public Switched Telephone Network), and is accessible through pre-selected and pre-configured modems.

Once connected, CitectSCADA can write to, and read from, dial-up remote I/O Devices just as it does with any other I/O Device: local or remote.

Communications can be:

- **On request**: initiated by CitectSCADA using IODeviceControl() or by the remote I/O Device (for instance, to report an alarm condition).
- **Periodic**: for instance, to transfer the logged events for a period.
- **Persistent**: for instance, to monitor and control the water level at a remote dam.

The only limiting factor would be an inability to connect a modem to an I/O Device due to incompatible communications capabilities. Many I/O Devices have fixed settings and can only communicate at a pre-set rate determined by the manufacturer. If a modem cannot match these settings, communication cannot be established.

To make communications setup easier, you can connect dial-up remote I/O Devices with identical communications to the same modem and port. Where I/O Devices are connected to the same modem, CitectSCADA can communicate with each I/O Device one after the other using the same phone connection, rather than hanging up and re-dialing. This reduces the number of necessary telephone calls and increases the speed and efficiency of communications.

You need to have at least one modem at the I/O Server end, and at least one at the I/O Device end.

See Also

Modems at the I/O Server
Modems at the I/O Device
I/O Device constraints for multi-dropping
Configuring multidrop remote I/O Devices
I/O Server redundancy for dial-up remote I/O Devices
Troubleshooting dial-up remote I/O Device communications

Modems at the I/O Server

To decide how many modems to use at the I/O Server end, decide what function each modem will perform. A single modem can do any one of the following functions:

Dial-out	Makes calls to remote I/O Devices in response to a CitectSCADA request; for example, scheduled, event-based, operator request, and so on. Also returns calls from remote I/O Devices.
Dial-in	Only receives calls from remote I/O Devices, identifies the caller, then hangs up immediately so it can receive other calls. CitectSCADA then returns the call using a dial-back or dial-out modem.
Dial-back	Only returns calls from remote I/O Devices.
Dial-in and dial-out	Receives calls from remote I/O Devices and makes scheduled calls to remote I/O Devices.
Dial-in and dial-back	Receives and returns calls from remote I/O Devices.

Your modem setup depends on your system requirements. When making your decision, consider the following guidelines:

- If you need to communicate with multiple remote I/O Devices at once, you will need a separate modem for each I/O Device. Otherwise you'll have to wait as I/O Devices are contacted one after the other, and incoming calls may be held up.
- If you have scheduled requests to I/O Devices and you also need to urgently return calls from remote I/O Devices that dial in, you will need at least one modem for each

of these functions. For example, if you have a large number of remote I/O Devices and you require fast responses by CitectSCADA, provide an additional modem at the I/O Server. This would reduce the chances of it being engaged when an I/O Device dials in (say, with an urgent alarm).

- In a big system with many remote I/O Devices or a system where calls from remote I/O Devices can be time sensitive, it's a good idea to dedicate at least one modem to Dial-Back. This will give you quick responses to Dial-In calls (from remote I/O Devices). It also means your dial-out schedules won't be disrupted (if you use the same modem for returning calls and scheduled calls, the scheduled calls are forced to wait until the dial-back call is complete).

See Also

Modems at the I/O Device

Example configurations for modems at the I/O Server

Modems at the I/O Device

You can have multiple I/O Devices connected to a single modem if they have the same communication requirements (phone number, baud rate, data bits, stop bits, and parity). A separate port and modem needs to be used for each remote I/O Device with different communication requirements.

When deciding how many modems to use, consider the following:

- Once an I/O Device has been contacted, the connection can be retained for other I/O Devices in the group. If the I/O Server needs to request data from another I/O Device with the same communication details, it will wait until the current request has been completed, then it will use the same connection to make the second request.
- You can configure your modem to initiate telephone calls (call CitectSCADA), and/or receive telephone calls. This is done independently of CitectSCADA.

> **Note:** Wherever your modem is, you need to verify that its **Data bits**, **Parity**, **Stop bits**, and **Serial Port Speed** settings are compatible with the remote I/O Device as defined by the device's manufacturer or your communications link will not work.

See Also

Modems at the I/O Server

Example configurations for modems at the I/O Server

The examples below demonstrate how to set up the modems at your I/O Server to accommodate different combinations of I/O Devices.

Example 1

All your remote I/O Devices have the same communication requirements (data bits, stop bits, parity, and baud rate) - 19200 8 E 1.

You don't expect any important calls from your I/O Devices, or you only have a few remote I/O Devices. This means you can use a single modem at the I/O Server end. This modem would be set up to answer and return incoming calls and make scheduled and other CitectSCADA initiated calls.

To configure your modem, define it in Windows. Assuming that the logical modem is called 'Standard Modem', configure it as follows:

Port	Modem Name	Max. Speed	Data Bits	Parity	Stop Bits
COM1	Standard Modem	19200	8	E	1

You would then configure it in CitectSCADA as a dial-out modem and dial-in modem:

Modem Name	Dial-out	Dial-in	Dial-back
Standard Modem	TRUE	TRUE	FALSE

Example 2

In this example, your I/O Devices use a total of two different communication specifications - 9600 7 O 1 and 19200 8 E 1.

You don't expect important calls from I/O Devices or you have only a few I/O Devices. This means you can get by with a single modem at the I/O Server end. This modem has to receive and return calls from I/O Devices as well as initiate calls (dial out) to I/O Devices.

To configure your modem, you need to first define it in Windows (through the Windows Control Panel). Remember, you're not just defining the physical modem here. You have to define a separate Windows (virtual) modem for each communication specification.

So far, this gives you two virtual modems - one for 9600 7 O 1 and one for 19200 8 E 1. However, Windows won't let you define both of these modems as dial-in. It only lets you define one dial-in modem per port. If you choose the first, it won't be able to receive calls with the second, and vice versa.

This means you have to set up a separate virtual modem that can answer calls no matter which communication specification is used. This modem would be set with a generic communication specification of 9600 8 N 1.

So in Windows, you'll end up with three logical modems (two for Dial-Out and one for Dial-In). Assuming that the logical modems are called 'Standard Modem' to 'Standard Modem #3', you would configure them as follows:

Port	Modem Name	Max. Speed	Data Bits	Parity	Stop Bits
COM1	Standard Modem	9600	7	O	1
COM1	Standard Modem #2	19200	8	E	1
COM1	Standard Modem #3	9600	8	N	1

You would then configure the modems in CitectSCADA as follows.

Modem Name	Dial-out	Dial-in	Dial-back
Standard Modem	TRUE	FALSE	FALSE
Standard Modem #2	TRUE	FALSE	FALSE
Standard Modem #3	FALSE	TRUE	FALSE

Example 3

In this example, there are five different communications frameworks - 9600 7 O 1, 19200 8 E 1, 4800 8 N 1, 9600 8 N 1, and 19200 8 N 1.

If you expect important calls from I/O Devices or you have many I/O Devices, you would set up three modems at the I/O Server end:

- One on COM3 dedicated to receiving calls from **9600 7 O 1** I/O Devices.
- One on COM2 for dialing out to **4800 8 N 1**, **9600 8 N 1**, and **19200 8 N 1** I/O Devices.
- One on COM1 for dialing out to **9600 7 O 1** and **19200 8 E 1** I/O Devices.

The two dial-out modems would return calls as well as initiate calls in response to scheduled requests, and so on.

To configure your modems, you need to first define them in Windows (through the Windows Control Panel). Remember, you're not just defining the physical modem here. You have to define a separate Windows (virtual) modem for each communication framework.

Assuming that the logical modems are called 'Standard Modem' to 'Standard Modem #6', you would configure them as follows:

Port	Modem Name	Max. Speed	Data Bits	Parity	Stop Bits
COM1	Standard Modem	9600	7	O	1
COM1	Standard Modem #2	19200	8	E	1
COM2	Standard Modem #3	4800	8	N	1
COM2	Standard Modem #4	9600	8	N	1
COM2	Standard Modem #5	19200	8	N	1
COM3	Standard Modem #6	9600	7	O	1

You would then configure the modems in CitectSCADA as follows:

Modem Name	Dial-out	Dial-in	Dial-back
Standard Modem	TRUE	FALSE	FALSE
Standard Modem #2	TRUE	FALSE	FALSE
Standard Modem #3	TRUE	FALSE	FALSE
Standard Modem #4	TRUE	FALSE	FALSE

Standard Modem #5	TRUE	FALSE	FALSE
Standard Modem #6	FALSE	TRUE	FALSE

Example 4

In this example, your I/O Devices use three different communication frameworks: 9600 7 O 1, 19200 8 E 1, and 9600 8 N 1. However, in this example, you are expecting important calls from I/O Devices, so you need a modem dedicated to returning calls.

Here you need to configure your modems like this:

- One modem on COM1 to dial remote I/O Devices (for scheduled calls, and so on).
- One modem on COM2 to receive calls from remote I/O Devices.
- One dedicated modem on COM3 to return these calls.

To configure your modems, first define them in Windows (through the Windows Control Panel). Remember, you're not just defining the physical modem here: you need to define a separate Windows (virtual) modem for each communication framework. This means you have to configure:

- Three logical modems on the port to which the physical dial-out modem is attached.
- One logical modem on the port to which the physical dial-in modem is attached.
- Three logical modems on the port to which the physical dial-back modem is attached.

Assuming that the necessary total of seven logical modems are called 'Standard Modem' through to 'Standard Modem #7', configure these modems as follows:

Port	Modem Name	Max. Speed	Data Bits	Parity	Stop Bits
COM1	Standard Modem	9600	7	O	1
COM1	Standard Modem #2	19200	8	E	1
COM1	Standard Modem #3	9600	8	N	1
COM2	Standard Modem #4	9600	8	N	1

COM3	Standard Modem #5	9600	7	O	1
COM3	Standard Modem #6	19200	8	E	1
COM3	Standard Modem #7	9600	8	N	1

You would then configure the modems in CitectSCADA as follows:

Modem Name	Dial-out	Dial-in	Dial-back
Standard Modem	TRUE	FALSE	FALSE
Standard Modem #2	TRUE	FALSE	FALSE
Standard Modem #3	TRUE	FALSE	FALSE
Standard Modem #4	FALSE	TRUE	FALSE
Standard Modem #5	FALSE	FALSE	TRUE
Standard Modem #6	FALSE	FALSE	TRUE
Standard Modem #7	FALSE	FALSE	TRUE

I/O Device constraints for multi-dropping

If you are multi-dropping off a single modem, use your I/O Devices to issue the caller ID, not the modem. This is because using the modem to issue the ID will send the same ID no matter which I/O Device the call is relevant to. This makes it difficult to identify the I/O Device that triggered the call.

By using the I/O Device to issue the ID, the I/O Server will receive a unique caller ID for each I/O Device. However, not every I/O Device is capable of issuing caller IDs. If multi-dropping, use I/O Devices that can issue caller IDs.

To configure dial-up remote I/O Devices for communication with CitectSCADA:

1. Run the Express Communications Wizard.
2. Complete the wizard, selecting the relevant I/O Server, then the I/O Device, creating each new instance when necessary.
3. On the Scheduling page of the wizard, select the **Connect I/O Device to PSTN** check box.
4. Select an appropriate schedule for CitectSCADA to communicate with the remote I/O Device. (For a persistent connection - whenever CitectSCADA is running - select **On Startup**.) For example (all based on a **Synchronize at** time of 10:00:00):
   - If you enter **12:00:00** in the **Repeat every** field, and start your project at 9 a.m., the I/O Server will communicate with the I/O Device at 10 a.m., then once every 12 hours after that; that is, 10 p.m., then again at 10 a.m. of the following day, and so on.
   - If you enter 12:00:00 and start your project at 4 p.m., the I/O Server will communicate with the I/O Server at 10 p.m., then again at 10 a.m., of the following day, and so on. CitectSCADA will assume that communications were established at 10.a.m., so will continue as if they had been, communicating once every 12 hours after 10 a.m.
   - If you enter 3 days and start your project at 9 a.m. on a Wednesday, the I/O Server will communicate with the I/O Device at 10 a.m., then once every 3 days after that; that is, 10 a.m. on the following Saturday, then at 10 a.m. on the following Tuesday, and so on.
   - If you enter the 6th of December in the **Repeat every** field and start your project during November, the I/O Server will communicate with the I/O Device at 10 a.m. on December 6, then again on December 6 of the following year, and so on.
5. Select **On Startup** for a persistent connection. To disconnect a persistent connection, call the `IODeviceControl()` function with type 8.
6. Type in the phone number necessary for CitectSCADA to dial the remote modem attached to the remote I/O Device. Include any pre-dial numbers necessary to obtain a connection to an outside PSTN line (dial tone) before dialing (for example, **0** (zero)) - if appropriate.
7. On the next wizard page, if the device is configured to dial-in to CitectSCADA, create a unique identifying caller name for the remote I/O Device so that it can be identified by CitectSCADA.
8. Follow the instructions on the next page of the wizard and click **Finish**.

See Also

Configuring multidrop remote I/O Devices

Configuring multidrop remote I/O Devices

Multidropping remote I/O Devices from the same remote modem enables CitectSCADA to communicate with each I/O Device one after the other, using the same phone connection, rather than hanging up and re-dialing.

Although you can configure multidrop remote I/O Devices using the Express Communications Wizard, we recommend that you do it manually. The wizard would create a new port for each I/O Device. This would mean you couldn't have any more than 255 I/O Devices.

1. Run the Express Communications Wizard to configure the first device.
2. Configure every other I/O device manually.
3. Open the Citect Project Editor.
4. Select **Communications | I/O Server** and scroll to the I/O Server that will be communicating with the I/O Device.
5. Select **Communications | I/O Devices**.Complete the dialog box.
6. To increase the efficiency and capacity of your system you can allocate the same port name to I/O Devices with the same communication settings.

> **Note:** If you are multi-dropping and you want to be able to dial in to the I/O Server, use your I/O Devices to issue the caller ID, not the modem. This is because using the modem to issue the ID will send the same ID no matter which I/O Device the call is relevant to. This makes it difficult to identify the I/O Device that triggered the call.

By using the I/O Device to issue the ID, the I/O Server will receive a unique caller ID for each I/O Device. However, not every I/O Device are capable of issuing caller IDs. If multi-dropping, use I/O Devices that can issue caller IDs.

To set up a modem connected to your dial-up remote I/O Devices:

You can connect multiple I/O Devices to the same modem. This means CitectSCADA can communicate with these I/O Devices one after the other using the same phone connection, rather than hanging up and re-dialing. This will reduce the number of necessary telephone calls and increase the speed and efficiency of communications.

1. Connect the modem to a PC with a telephony program installed (for example HyperTerminal or PhoneDialer). This is where you will configure the modem to answer calls from CitectSCADA and/or initiate calls.
2. If the modem is necessary to make calls to CitectSCADA, configure it to initiate the phone call to a pre-determined CitectSCADA I/O Server Dial-In type modem (following manufacturer instructions).

3. Depending on your hardware either the modem or an intelligent PLC can be responsible for initiating calls to CitectSCADA and identifying the caller. Whichever is responsible needs to have a caller ID set. The caller ID can be any combination of alpha-numeric characters and/or the character '_' (underscore).
   Some modems have dip-switch settings, and some have initiation strings which can include auto-dial-up numbers that are stored within the modem's non-volatile memory. Consult the manual provided with the modem for exact details.
   You can use either the Express Communications Wizard or the I/O Devices form to set the caller ID for an I/O Device.
   If multi-dropping off a single modem, use your I/O Devices to issue the caller ID, not the modem. This is because using the modem to issue the ID will send the same ID no matter which I/O Device the call is relevant to, making it difficult to identify which I/O Device triggered the call.
   By using the I/O Device to issue the ID, the I/O Server will receive a unique caller ID for each I/O Device. However, not every I/O Device is capable of issuing caller IDs. If you are multi-dropping, use I/O Devices that can issue caller IDs.
4. Set the modem's **Data bits**, **Parity**, **Stop bits**, and **Serial-Rate** to match manufacturer specifications for communication with the I/O Devices.
   Some modems do not allow you to manipulate their communications settings via methods such as extended AT commands or dip switches. If this is the case, the only way of setting the necessary values is to communicate with the modem using the values (for example, via HyperTerminal). Once this is done, the modem remembers the last values used to communicate with its serial port.
5. Connect the modem to the I/O Devices.

To configure a modem at the I/O Server you need to set it up in Windows and then set it up in CitectSCADA.

If every one of your I/O Devices are the same, you only have to do this once for each modem. However, if your I/O Devices talk using different communication specifications (data bits, parity, stop bits, and serial-rate), your modem has to be able to talk using each of these details as well. To set this up, you have to create a modem in Windows and CitectSCADA for each specification. (See Example configurations for modems at the I/O Server)

To set up a modem in Windows

1. Each modem connected to a CitectSCADA I/O Server PC needs to FIRST be configured within Windows using **Start | Settings | Control Panel | Phone and Modem Options**.
2. Select the **Modems** tab, and click **Add** to launch the Install New Modem wizard.
3. Check the box labeled **Don't detect my modem; I will select it from a list**, then click **Next**.

4. Select **Standard Modem Types** in the list of manufacturers.

> **Note:** Do not select a brand name modem from the Manufacturers list, even if the name of the modem you're installing is included in the list. Do not click **Have Disk**.

5. Select the **Standard xxxx bps modem** rate from the list of models to exactly match the bit per second rate of the I/O Device that is going to be communicating via this modem. Check the device to determine the device communication rate. If you are still unsure, select the 9600 bps model. This can be changed later if necessary.
6. Do not click **Have Disk**. Click **Next**.
7. Select the **COMx Port** that the modem is connected to. Click **Next**.
8. Click **Finish**. Windows displays the modem in the list of modems on the Modems Properties form.
9. No option was given for the selection and setting of the Data bits, Parity, or Stop bits information. The Modems wizard automatically defaults to 8-none-1 for Standard Modem types. To change these settings to match the Data bits, Parity, Stop bits requirements of the remote I/O Device, select a modem in the list, then click the **Properties** button.
10. Click the **Advanced** tab and click **Change Default Preferences**.
11. Click the **Advanced** tab at the next dialog to gain access to the Data bits, Parity, Stop bits settings for the modem.
12. Change the Data bits, Parity, and Stop bits settings using the drop-down options, so that they exactly match those being used by the remote I/O Device and its remote modem. Don't change any advanced settings. (The default is Hardware flow control.)
13. Click **OK**. If a modem of the same rate is installed to the same port as an existing modem, Windows asks for confirmation that you want to install the same thing more than once. Click **Yes** to install a duplicate copy of the modem.
14. Preconfigure the modem(s) to be used at the remote dial-up I/O Device(s). These will be used to test modem configuration settings in the next step.
15. With CitectSCADA not running, confirm that the local and remote modems will properly communicate with each other by using a terminal communications program such HyperTerminal or PhoneDialer (both supplied with Windows).

Once the modem(s) are set up and tested with proven communications in Windows, they can then be set up in CitectSCADA.

To set up a modem in CitectSCADA:

Before completing this procedure, verify that you have set up your modem in Windows (as described above).

1. Open the **Citect Project Editor**.
2. Select **Communications** | **I/O Server** and scroll to the I/O Server the modem is attached to.
3. Select **Communications** | **Modems**. The Modem Properties dialog will appear.

Option	Description
Server Name (16 Chars.)	The name of the I/O Server to which the modem is attached.
Modem Name (64 chars.)	The name of the modem you are configuring (as it appears in the Windows Control Panel | Phone and Modem Options).
Comment (48 Chars.)	Any useful comment.
Use this modem to make outgoing calls	Determines whether this modem will be used to initiate calls from the I/O Server to a dial-up remote I/O Device. (Dial-Out) This may include calls that are scheduled, event driven, or in response to I/O Devices that dial in.
Use this modem to answer incoming calls	Determines whether this modem will be used to receive calls from a dial-up remote I/O Device. (Dial-In)

The following fields are implemented with extended forms (press **F2**).

Use this modem to call back I/O Devices	Determines whether this modem will be used to initiate calls from the I/O Server to a dial-up remote I/O Device in response to a call received from the I/O Device. (Dial-Back)

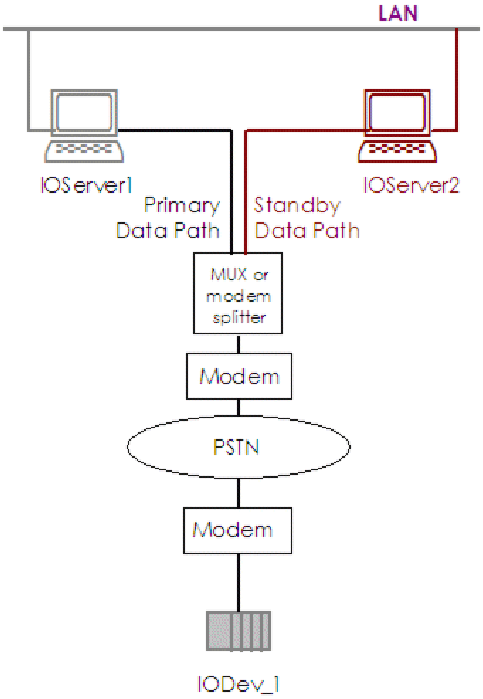
4. Complete the dialog box.

> **Note:** CitectSCADA allows you to set up a maximum of 256 modems on the I/O Server for communication with remote dial-up I/O Devices. Before it can successfully establish communication, any targeted remote I/O Devices need to also be properly configured within CitectSCADA on the I/O Server.

I/O Server redundancy for dial-up remote I/O Devices

You can change the number of rings the I/O Server will wait before answering the call (using the **[Dial]RingCount** parameter). If you are using redundant I/O Servers, the primary I/O Server will be called by default. However, by setting [Dial]RingCount to a different value on each of the I/O Servers, you can make the standby I/O Server answer the call if the primary does not.

Consider the following setup:



If you set the ring count to 3 on IOServer1 and 4 on IOServer2, IODev_1 will attempt to call IOServer1. If IOServer1 has not answered the call after 3 rings, IOServer2 will answer it.

See Also

Troubleshooting dial-up remote I/O Device communications

Troubleshooting dial-up remote I/O Device communications

The challenges most often encountered when using a dial-up remote I/O Device involve communications speed, parity, and control signals from the connected equipment. If changing the speed and parity does not resolve a communications interruption, evaluate the modem's answering codes and command echoing.

Following is a list of settings that might be helpful in resolving dial-up communications interruptions. (Since not every modem supports the same in commands in the same way, this is only a guide. Consult the modem manual for exact details.)

On the modem at the PC end

```
ATV1 //Enables long-form (verbose) result codes
ATQ0 //Result codes are sent on the RS-232 connection
ATE0 //Commands sent from the computer are not echoed back to the RS-232 connection
AT&C1 //DCD will follow carrier on the line
AT&K0 //Handshaking OFF
ATW0 //Upon connection, only DTE speed is reported
AT%C0 //Compression OFF
AT&D0 //DTR always on
```

If the modem at the PC end is configured so that calls are automatically answered even when your CitectSCADA project is off, data being reported from the I/O devices may be lost. Therefore, it is recommended that you turn off the PC modem's auto-answer feature before taking your project offline. To do this, set the following parameter to zero:

```
ATS0 = 0 // Auto answer OFF
```

Be aware, however, this will also impact applications that might use the modem other than CitectSCADA, as the modem cannot answer a call while CitectSCADA is not driving its functionality.

On the modem at the I/O Device end

```
ATV0 //Enables short-form result codes
ATQ1 //No result codes are sent on the RS-232 connection
ATE0 //Commands that are sent from the computer are not echoed
back to the RS-232 connection
AT&C1 //DCD will follow carrier on the line
AT&K0 //Handshaking OFF
ATW0 //Upon connection, only DTE speed is reported
AT%C0 //Compression OFF
AT&D0 //DTR always on
ATS0 //Set to greater than 0 (sets the number of rings necessary before the modem
answers
an incoming call).
```

Alternative (backward compatibility) method of persistent connection

If you are setting up your modem to dial a dial-up remote I/O Device, follow the procedures described in Communicating with Dial-up Remote I/O Devices. This method is available for backward compatibility.

Scheduled Communications

CitectSCADA allows you to schedule communications with your I/O Devices (regardless of the type of connection: modem, radio link, etc.). For example, if you have multiple I/O Devices on a single network or line, you can schedule reads so that the important I/O Devices are read more often than less important I/O Devices. Alternatively, a water supplier with radio link connections to dam level monitors might schedule hourly level reads from CitectSCADA in order to conserve band-width.

See Also

Specifying a schedule
Writing to the scheduled I/O Device
Reading from the scheduled I/O Device

Specifying a schedule

To configure scheduled communications with an I/O Device, you need to flag it as a "Scheduled" device. This is done using the **Express Communications Wizard**. If your I/O Device is not capable of scheduled communications, the scheduling options will not be presented by the wizard.

> **Note:** Scheduled communications will not work for drivers that are designed to handle Report-By-Exception protocols. For communicating with your I/O Device outside of schedule use the IODeviceControl function.

To schedule communications with an I/O Device:

1. Select the Project Editor (or press the Project Editor icon).
2. Choose **Communication | Express Wizard** or open Citect Explorer.
3. Double-click the **Express I/O Device Setup** icon in the Communications folder of the current project.
4. Follow the instructions to work through the screens, selecting the relevant I/O Device, and so on. When the scheduling screen displays, check the **Connect I/O Device to PSTN** box, even if your I/O Device is not connected via a modem (but leave the Phone number to dial and Caller ID fields blank).
5. Fill out the schedule fields to achieve the desired schedule. For example (all based on a **Synchronize at** time of 10:00:00).

If you enter 12:00:00 in the **Repeat every** field, and start your project at 9 a.m., the I/O Server will communicate with the I/O Device at 10 a.m., then once every 12 hours after that, i.e. 10 p.m., then again at 10 a.m. of the following day, etc.

- If you enter 12:00:00, and start your project at 4 p.m., the I/O Server will communicate with the I/O Server at 10 p.m., then again at 10 a.m. of the following day, etc. i.e. it will assume that communications were established at 10 a.m., so it continues as if they had been, communicating once every 12 hours after 10 a.m.
- If you enter 3 days, and start your project at 9 a.m. on a Wednesday, the I/O Server will communicate with the I/O Device at 10 a.m., then once every 3 days after that, i.e. 10 a.m. on the following Saturday, then at 10 a.m. on the following Tuesday, etc.
- If you enter the 6th of December in the **Repeat every**, and start your project during November, the I/O Server will communicate with the I/O Device at 10 a.m. on December 6, then again on December 6 of the following year, and so on.

6. Select **On Startup** for a persistent connection. To disconnect a persistent connection, you need to call the IODeviceControl() function with type 8.
7. If your I/O Device is not connected via a modem, you need to go to the Ports form (select Ports from the Communication menu) and change the Port number to the actual number of the COM port.

See Also

Writing to the scheduled I/O Device

Writing to the scheduled I/O Device

Whenever an I/O Device is actively communicating (as per its schedule), you can write to it directly. However, if you try to write to it when it is not communicating, your write request will be queued until it is. For example, you might decide to schedule one write per hour. If someone at a Control Client changes a tag's value during that hour, that change will not be written to the I/O Device until the hour has expired.

As write requests are not written to the I/O Device until it is communicating, confirm that pending writes have been completed in full before shutting down.

> **Note:** Don't control hardware using a scheduled I/O Device, as the exact state of the hardware may not be known. Although you can read the state from the cache, it may have changed since the cache was created.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

Do not use a scheduled I/O device to control hardware in a system managed by CitectSCADA.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

See Also

Reading from the scheduled I/O Device

Reading from the scheduled I/O Device

When the I/O Server initiates communication with the I/O Device, it immediately writes any queued write requests, then reads the I/O Device's tags. These values are then stored in a cache so that you can still access them between communications.

Note: Because the I/O Server reads tags on initiation of communication, the license point count is increased, even though some of the tags read may not actually be used.

Keeping data up-to-date during prolonged connections

Normally, communication is terminated as soon as every read and write request is complete. Sometimes, however, you may want to prolong the communication (for example by calling the IODeviceControl() function with Type 7).

In this situation (if Read-Through Caching is disabled), when client computers request device data from the I/O Server, it retrieves the data from its cache, not from the I/O Device. This occurs even though the I/O Server maintains a connection to the device.

To retrieve fresh data from the I/O Device, you can force a periodic read using Cicode, or change the cache timeout by setting the IODeviceControl() function to Type 11.

For example:

```
INT hTask;
// Initiate communications and read tags.
// Sleep time will depend on how fast your
// modems connect.
FUNCTION
DialDevice(STRING sDevice)
        INT bConnected = 0;
        INT nRetry = 5;
        hTask = TaskHnd("");
        IODeviceControl(sDevice, 7, 0);
        Sleep(20);
        WHILE bConnected <> 1 AND nRetry > 0 DO
                bConnected = IODeviceInfo(sDevice, 18);
                nRetry = nRetry - 1;
                Sleep(10);
        END
        IF bConnected = 1 THEN
                WHILE TRUE DO
                        Sleep(2);
                        IODeviceControl(sDevice, 16, 0);
                END
        END
END
// Kill the read task and terminate the connection.
FUNCTION
HangupDevice(STRING sDevice)
        TaskKill(hTask);
        IODeviceControl(sDevice, 8, 0);
END
```

You can also force the I/O Server to read data directly from an I/O Device by enabling Read-Through Caching. With `[Dial]ReadThroughCache` set, while the I/O Server is connected to a device it will supply data to requesting clients directly from the device. The cache is not updated during this time, but is refreshed with the most recent device data just before the server disconnects.

Note: If using modems, you might need to adjust or deactivate the inactivity timer in your modems to stop them from disconnecting while no data is being read. The inactivity timer is controlled by the S30 register. If your modem doesn't support this register, please consult your modem's manual.

Avoiding unnecessary multiple reads of I/O Device data

To avoid unnecessary reads of an I/O Device, you can use data caching to temporarily store data read from the device in the memory of the I/O Server. This means that if the I/O Server receives more than one request for device data in a short time period, instead of contacting the I/O Device a second time and reading identical data, it can retrieve the data from the cache.

Chapter: 16 Tagging Process Variables

You need to assign a variable tag to each I/O Device variable that CitectSCADA uses in your runtime system. To define your variable tags, you declare them in the variable tag database. The variable tag becomes a label, used to reference the data value of the I/O Device register at the specified address. Using labels has several benefits:

- You do not have to remember the address every time you want to use the variable. You use the tag name, which has to be logical and descriptive, and therefore less confusing.
- The address in the I/O Device is defined only once. If you change the address, you only need to update the variable tag definition, not every instance in your configuration.
- You can scale the raw data to an appropriate range in the same declaration.
- You can access extended data value, quality and timestamp information.

You need to define your variables as a specific data type. The most common variables supported by I/O Devices are digital and integer. CitectSCADA also supports real, string, byte, bcd, long, and longbcd data types.

After you have defined your variable tags, you can use them to:

- Display objects on a graphics page.
- Store data for trending and analysis. (See Trending Data.)
- Monitor Alarms. (See Configuring and Processing Alarms.)
- Control equipment and processes. (See Defining Commands and Controls.)
- Store data in memory (See Configuring Local Variables.)

The most common variables supported by I/O Devices are digital and integer variables, although some I/O Devices support other numeric variables and strings.

See Also

Tag name syntax
Tag Extensions
Tag Data Types
Using structured tag names
Configuring Local Variables
Configuring Variable Tags
Formatting numeric variables
Using arrays

Tag Naming

CitectSCADA puts a couple of restrictions on the names of variable tags:

- Tags are restricted to using a specific syntax. See Tag name syntax.
- Do not give Tags the same name as Cicode functions within the project or within any included projects. An error will result during compilation if a tag has the same name as a Cicode function and is placed on a graphic page. See Defining Variable Tag Names for a list of restrictions on naming.

In addition, using a tag naming convention will make your project easier and faster to design, configure, commission, and maintain. See Using structured tag names for recommendations about naming conventions.

Tag name syntax

CitectSCADA tags (variable tags, alarm tags and trend tags) need to have the following syntax:

```
[<alpha> | '_'] *[<alpha> | <digit> | '\' | '_']
```

That is, the tag name needs to begin with either an alpha character (A-Z or a-z) or the underscore character (_). Any following characters needs to be either alpha characters (A-Z or a-z), digit characters (0 - 9), backslash characters (\), or underscore characters (_). The use of any other characters will result in a compiler error.

For example, '_MyTag123' and 'my\New\Tag' are both valid tag names, whereas '\NewTag\' is invalid.

Tag names that begin with a numeric character, such as '12TagName', are only valid if the INI parameter **[General]TagStartDigit** is set to 1 (the default value is zero).

Note: The name of an Alarm Tag needs to follow this syntax but the Alarm Name does not.

Using structured tag names

The following naming convention is recommended for a system, to obtain maximum benefit when using features such as Genies and Super Genies. (If you are already using a naming system that differs from the following convention, you can still use Genies and Super Genies supplied with CitectSCADA by modifying the Genies that you want to use.)

Each tag name can contain up to 79 characters. To establish a convention, you need to divide the characters in the tag name into sections that describe characteristics of the tag, for example, the area where the tag is located, the type of variable, and any specific attributes. Four basic sections are suggested for a CitectSCADA naming convention:

```
Area_Type_Occurrence_Attribute
```

Area

The **Area** section identifies a plant area, number, or name. If you use a prefix that identifies tags within a particular area, you can easily duplicate CitectSCADA functions within the area. For example, if you have three boilers with the same controls on each boiler, you can configure the tags for boiler number one, and copy the tags to boilers two and three. You then only need to change the area section in the tag names to the area of the second and third boiler. The remainder of the tags remain unchanged, for example:

Boiler 1	Boiler 2	Boiler 3
B1_TIC_101_PV	B2_TIC_101_PV	B3_TIC_101_PV

If you do not need this facility, you can omit the Area section of the Tag Name to reduce the number of characters in the tag.

Type

The **Type** section identifies the Type of parameter, process equipment, or control hardware. The ISA standard naming system is recommended.

Variable Tag	Meaning
B1_**TIC_**101_PV	Temperature indicating controller
B1_**FIC_**101_PV	Flow Indicating controller

B1_**PUMP_**101_PV	Pump
B1_**VALVE_**101_PV	Valve

Occurrence

The **Occurrence** section identifies the loop number.

Variable Tag	**Meaning**
B1_TIC_**101_**PV	Temperature Indicating Controller 101
B1_TIC_**102_**PV	Temperature Indicating Controller 102
B1_PUMP_**101_**PV	Pump 101
B1_PUMP_**102_**PV	Pump 102

Attribute

The **Attribute** section identifies the attribute or particular parameter that is associated with the loop.

Variable Tag	**Meaning**
B1_TIC_101_**PV**	Process Variable
B1_TIC_101_**SP**	Setpoint
B1_TIC_101_**OP**	Output
B1_TIC_101_**P**	Gain or proportional band
B1_TIC_101_**I**	Integral
B1_PUMP_101_**CMD**	Command signal to start pump
B1_PUMP_101_**M**	Auto/Manual mode
B1_TIC_101_**V**	Value (running/stopped)

Recommended Attributes

Genies and Super Genies supplied with CitectSCADA use the following attribute convention. If you follow this convention, you can use the Genies without having to modify them.

Mnemonic	Discrete Control / Monitoring	Data Type	Range
_CMD	Command Signal to Start Device	Digital	0 = Off, 1 = On
_M	Control Mode	Digital	0=Man, 1=Auto
_V	Value	Digital	0=Off, 1=On
_FAIL	Device Failure	Digital	1=OK, 0=Failed
FAULT	Device Fault	Digital	1=OK, 0=Fault

Mnemonic	Process Alarms	Data Type	Range
_ALM	General Alarm	Digital	0=Active, 1=InActive
_HHALM	High High Alarm	Digital	0=Active, 1=InActive
_HALM	High Alarm	Digital	0=Active, 1=InActive
_LALM	Low Alarm	Digital	0=Active, 1=InActive
_LLAM	Low Low Alarm	Digital	0=Active, 1=InActive
_DALM	Deviation Alarm	Digital	0=Active, 1=InActive
_DLALM	Deviation Low Alarm	Digital	0=Active, 1=InActive
_DHALM	Deviation High Alarm	Digital	0=Active, 1=InActive
_HHTRIP	High High Alarm Trip Point	Analog	

_HTRIP	High Alarm Trip Point	Analog
_LTRIP	Low Alarm Trip Point	Analog
_LLTRIP	Low Alarm Trip Point	Analog
_DTRIP	Deviation trip Point	Analog
_LDTRIP	Low Deviation Trip Point	Analog
_HDTRIP	High Deviation Trip Point	Analog
_HHhyst	High High Alarm Hysteresis	Analog
_Hhyst	High Alarm Hysteresis	Analog
_Lhyst	Low Alarm Hysteresis	Analog
_LLhyst	Low Low Alarm Hysteresis	Analog
_LDhyst	Low Deviation Alarm Hysteresis	Analog
_HDhyst	High Deviation Hysteresis	Analog

Mnemonic	**Analog Control / Monitoring**	**Data Type**	**Range**
_PV	Process Variable	Analog	
_SP	Setpoint	Analog	
_RSP	Remote Setpoint	Analog	
_OP	Output	Analog	
_OPM	Output Mode	Digital	0=Manual, 1=Auto
_SPM	Setpoint Mode	Digital	0=Local, 1=Remote
_P	Gain (Proportional Band)	Analog	

_I	Integral (Reset)	Analog	
_D	Derivative (Rate/Preact)	Analog	
_KP	Gain Modifier	Analog	
_KI	Integral Modifier	Analog	
_KD	Derivative Modifier	Analog	
_SPTK	Setpoint Track Mode	Digital	0=OFF, 1=Track
_OPTK	Output Track Mode	Digital	0=OFF, 1=Track
_SPB	Setpoint Bias	Analog	
_SPR	Setpoint Ratio	Analog	
_DEV	Deviation		
_TOT	Totalizer Value	Analog	
_COUNT	Counter Value	Analog	
_CRESET	Counter Reset Command	Digital	0=Counting, 1=Reset
_CLIMIT	Counter Preset Limit	Analog	
_TIME	Timer Value	Analog	
_TRESET	Timer Reset Command	Digital	0=Timing, 1=Reset
_EXP	Timer Expired	Digital	
_TLIMIT	Timer Limit	Analog	
_CALC1	Calculation Result 1	Analog	
_LINZ1	Linearized Signal 1	Analog	
_Q	Data Quality Flag	Digital	1=OK, 0=BAD

Note: To keep the tag names shorter you can omit the underscore, but you would sacrifice readability; for example: B1TIC101PV instead of B1_TIC_101_PV.

Tag Extensions

A variable tag is a representation of data elements. Each element provides access to a view of the data value for the tag.

Each variable tag can be used on its own or by referencing a particular element to access the following information:

Element	Description
.field	The field element, which represents the latest field data received from the device (see Reading Tag Values).
.valid	The valid element, which represents the last field data which had 'Good' quality (see Reading Tag Values).
.override	The override element, which represents the overridden tag value (see Controlling and Overriding Tag Values).
.overridemode	The override mode, which is used to set the override behavior of the tag (see Override Mode).
.controlmode	The control mode, which is used to set the control inhibit mode of the tag (see Control Inhibit Mode).
.status	The tag status element, which is used to represent the current status of the tag (see Tag Status).

The tag and each element have items that can be referenced to access the following information:

Item	Description
v	The value, which will access the data value of the tag or element.
vt	The value timestamp, which will access the timestamp of when the value last changed.
q	The quality, which will access the quality of the value , either GOOD, UNCERTAIN or BAD. You can access further detail from the quality using the Cicode Quality functions.
qt	The quality timestamp, which will access the timestamp of when the quality last changed.
t	The timestamp, which will access the timestamp of when the tag or element was last updated.

The tag reference syntax is:

[Cluster.]Tag[.Element][.Item][[n]], where

Cluster	The optional cluster name.
Tag	The tag name or SuperGenie association.
Element	The optional element name. If the element name is not specified, the requested element will be determined at runtime.
Item	The optional item name. If the item name is not specified, the whole element is referenced.
n	The optional array index if the tag is defined as an array.

The array index is at the end of the reference (MyArray.v[n], MyArray.Field[n], MyArray.Field.v[n]). There is only a single quality and timestamp for each array, each member will return the same quality and timestamp.

> **Note:** Consider the impact on network traffic when configuring tag extensions, as the distribution of quality and value timestamps increases the amount of data being sent between servers.

You can access Tag data in the following ways:

1. Reference the tag data by using only the tag name, for example ‘MyTag’ (unqualified tag reference). This will provide default access to the Field element information, unless the tag is in one of the override modes.

2. Reference the tag data by using the tag name and the item name, for example ‘MyTag.q (unqualified tag reference). This will provide access to the item information for the tag, either default from Field or Override element.

3. Reference the tag data by using the tag name and the element name, for example ‘MyTag.Field’ (qualified tag reference). This reference will provide access to the specific tag element information.

4. Reference the tag data by using the tag name, element name and the item name, for example ‘MyTag.Field.vt’. This reference will provide access to the specific tag element item (qualified tag reference).

You can also access alarm data similar to tag data. See Using Alarm Properties as Tags.

Controlling Tag Extension behavior

By default, the tag data referenced without an element will provide access to the data value when the value is of quality is good and an error (#BAD, #COM, etc) when the quality is bad. The configuration parameter PageIgnoreValueQuality can be used to change this behavior, including automatically changing the background color of text and number graphics objects on a page with changes in quality of the tag.

The Tag Extensions behavior is controlled by several citect.ini file settings which are described in the .Parameter Help file. For each of these settings there is a corresponding setting in the parameters database (param.dbf). A citect.ini file setting specifies behavior for a particular machine and a parameter database setting is applied system-wide. The citect.ini file setting can be entered using the Computer Setup Editor, the parameter database settings are configured in the CitectSCADA Project Editor from the System, Parameter menu.

Note: By default, the TagSubscribe Cicode function is set to retrieve "lightweight" tag values that exclude quality and value timestamps. If you need a subscription to retrieve timestamp data, you need to set the "bLightweight" argument to 0 (false).

See Also

The Quality Tag Element
Reading and Writing Tags
Tag Data Types
Controlling and Overriding Tag Values

The Quality Tag Element

The majority of data which is contained within the variable tag is represented as an element which includes the items "Value", “ValueTimestamp”, “Quality”, “QualityTimestamp” and "Timestamp". With the exception of "Quality" these items are absolutes of a single value. However the 'Quality" item has several layers of information within it. These layers of information are covered by the following categories:

The primary layer identifies the **General Quality Values.**

Cicode label name	Value	Description
QUAL_BAD	0x00	Value is not useful for reasons indicated by the Substatus Bit Field.
QUAL_UNCR	0x01	The Quality of the value is uncertain for reasons indicated by the Substatus Bit Field.
QUAL_GOOD	0x03	The Quality of the value is Good.

These can be accessed from a text object on a graphics page at runtime or with Cicode using the QualityGetPart Cicode function and the necessary Part parameter.

Within each of these quality values is a second, substatus layer, and a third extended substatus layer. These layers of information listed below can also be accessed using the QualityGetPart Cicode function.

- QUAL_BAD Substatus Values
- QUAL_UNCR Substatus Values
- QUAL_GOOD Substatus values

There is also the additional informational value of "Quality Limit" and "Tag Status" accessed using the QualityGetPart Cicode function or the appropriate function shown below.

Quality Limit

Cicode label name	Value	Description
QUAL_LIMITED_ NOT_LIMITED	0x0	The value is free to move up and down.
QUAL_LIMITED_ LOW	0x1	The value has 'pegged' at some lower limit.
QUAL_LIMITED_ HIGH	0x2	The value has 'pegged' at some high limit.
QUAL_LIMITED_ CONSTANT	0x3	The value is a constant and cannot move.

Quality Tag Status

Cicode label name	Value	Description
QTS_OVERRIDE	0x01	The tag is in Override mode.
QTS_CONTROL_ INHIBIT	0x02	The tag is in Control Inhibit Mode.

QUAL_BAD Substatus Values

Use the the QualityGetPart Cicode function and the necessary Part parameter to return the details of a QUAL_ BAD tag. The following table identifies the possible information that will be returned.

Cicode label name	Value	Description
QUAL_BAD_NON_ SPECIFIC	0x00	The value is bad but no specific reason is known.
QUAL_BAD_CONFIGURATION_ ERROR	0x01	There is some server-specific problem with the configuration. For example, the item in question has been deleted from the configuration.
QUAL_BAD_NOT_ CONNECTED	0x02	The input is necessary to be logically connected to something but is not. This quality may reflect the fact that no value is available at this time, for reasons like the value may not have been provided by the datasource.

Cicode label name	Value	Description
QUAL_BAD_DEVICE_FAILURE	0x03	An inoperative device has been detected.
QUAL_BAD_SENSOR_FAILURE	0x04	An inoperative sensor has been detected (the 'Limits' field can provide additional diagnostic information in some situations).
QUAL_BAD_LAST_KNOWN_VALUE	0x05	Communication has been lost, however, the last known value is available. The age of the value may be determined from the TIMESTAMP in the OPCITEMSTATE.
QUAL_BAD_COMM_FAILURE	0x06	Communication has been lost. There is no last known value available.
QUAL_BAD_OUT_OF_SERVICE	0x07	The block is off scan or otherwise locked. This quality is also used when the active state of the item or the group containing the item is InActive.
QUAL_BAD_WAITING_FOR_INITIAL_DATA	0x08	After items are added to a group, it may take some time for the server to actually obtain values for these items. In such cases, the client might perform a read (from cache), or establish a ConnectionPoint based subscription and/or execute a refresh on such a subscription before the values are available. This substatus is only available from OPC DA 3.0 or newer servers.

See Also

QUAL_EXT Substatus Values

QUAL_GOOD Substatus Values

QUAL_UNCR Substatus Values

The Quality Tag Element

QUAL_UNCR Substatus Values

Use the the QualityGetPart Cicode function and the necessary Part parameter to return the details of an UNCERTAIN quality tag. The following table identifies the possible information that will be returned.

Cicode label name	Value	Description
QUAL_UNCR_NON_SPECIFIC	0x00	The value is uncertain but no specific reason is known.
QUAL_UNCR_LAST_USABLE_VALUE	0x01	Whatever was writing this value has stopped doing so. Regard the returned value as 'stale'. This differs from a BAD value with Substatus 5 (0x14) (Last Known Value). This status is associated specifically with a detectable communication error on a 'fetched' value and indicates the failure of some external source to 'put' something into the value within an acceptable period of time. The age of the value can be determined from the TIMES-

Cicode label name	Value	Description
		TAMP in OPCITEMSTATE.
QUAL_UNCR_ SENSOR_NOT_ ACCURATE	0x04	Either the value has 'pegged' at one of the sensor limits (in which case, set the limit field to 1 or 2) or the sensor is otherwise known to be out of calibration via some form of internal diagnostics (in which case,set the limit field to 0).
QUAL_UNCR_ ENG_UNIT_ EXCEEDED	0x05	The returned value is outside the limits defined for this parameter. In this case the Limits field indicates which limit has been exceeded but does NOT necessarily imply that the value cannot move farther out of range.
QUAL_UNCR_ SUBNORMAL	0x06	The value is derived from multiple sources and has less than the necessary number of Good sources.

See Also

QUAL_EXT Substatus Values

QUAL_GOOD Substatus Values

QUAL_BAD Substatus Values

The Quality Tag Element

QUAL_GOOD Substatus Values

Use the the QualityGetPart Cicode function and the necessary Part parameter to return the details of a GOOD quality tag. The following table identifies the possible information that will be returned.

Cicode label name	Value	Description
QUAL_GOOD_ NON_SPECIFIC	0x00	The value is good. There are no special conditions.
QUAL_GOOD_ LOCAL_OVERRIDE	0x06	The value has been Overridden. Typically this is means the input has been disconnected and a manually entered value has been "forced".

See Also

QUAL_EXT Substatus Values

QUAL_BAD Substatus Values

QUAL_UNCR Substatus Values

The Quality Tag Element

QUAL_EXT Substatus Values

Use the the QualityGetPart Cicode function and the necessary Part parameter to return the details of a BAD or UNCERTAIN quality tag. The following table identifies the possible information that will be returned.

Cicode label name	Value	Description
QUAL_EXT_ NON_SPECIFIC	0x00	There is no specific extended substatus value.
QUAL_EXT_ SCHEDULED_ OFFLINE	0x01	The device is a scheduled device that is offline and no cache value is available.
QUAL_EXT_ INVALID_TAG	0x02	The tag configuration is invalid.
QUAL_EXT_ INVALID_DATA	0x03	The value of the tag is invalid.
QUAL_EXT_ SOFTWARE_ ERROR	0x04	An internal software error occurred in the device driver.
QUAL_EXT_ TOO_MANY_ DEVICES	0x05	Too many devices are attached.
QUAL_EXT_ COMM_NO_INIT	0x06	Communication is not initialised.
QUAL_EXT_ COMM_BAD	0x07	Bad communication.
QUAL_EXT_ TAG_OUT_OF_ RANGE	0x08	Tag address is out of range.
QUAL_EXT_ WRITE_ONLY	0x09	Tag is not readable.
QUAL_EXT_ WRITE_PROTECTED	0x0A	Write operation is not authorised.
QUAL_EXT_NO_ CLUSTER_SPECIFIED	0x0B	No cluster is specified within a system or for a given tag.
QUAL_EXT_ CLUSTER_NOT_ FOUND	0x0C	The requested cluster is not known or no clusters are available.

Cicode label name	Value	Description
QUAL_EXT_CLUSTER_DISABLED	0x0D	The requested cluster is disabled.
QUAL_EXT_SESSION_NOT_CONNECTED	0x0E	Cannot connect to the requested session.
QUAL_EXT_TAG_RESOLVE_TIMEOUT	0x0F	Tag could not be resolved.
QUAL_EXT_VALUE_NOT_REPLICATED	0x10	Tag element value not replicated to every redundant DataSource.
QUAL_EXT_STALE	0x11	Tag element value is "stale.
QUAL_EXT_COMM_DEV_BAD	0x12	Unsuccessful communication between DataSource and PLC.
QUAL_EXT_INVALID_ARGUMENT	0x13	Invalid element used for referencing to a tag.
QUAL_EXT_VALUE_OUT_OF_RANGE	0x14	Tag element value is out of range.

See Also

QUAL_BAD Substatus Values

QUAL_GOOD Substatus Values

QUAL_UNCR Substatus Values

The Quality Tag Element

Reading and Writing Tags

CitectSCADA represents an I/O Device variable monitored or controlled by the system as a tag variable. You create a tag variable in the Project Editor specifying the tag variable name, data type, address, deadband and other attributes. After the tag variable has been created, you can reference it using the tag name. The tag name reference provides access to the value of the tag variable.

This functionality is available to be read from a tag with a text object on a graphics page and can be read, and in some cases written to, using Cicode functions or CtAPI functions.

Tag Extensions provide the following additional functionality:

- Access to the tag value quality and timestamp.
- Extended data associated with a variable tag.
- Make this data available to CitectSCADA client components: Control Client, Trend Server, Alarm Server, Report Server, Cicode and CtAPI.
- Be able to override and prohibit writing to the tag variable value.
- Be able to display and trend the tag values even if their quality is not “Good”.
- Be able to have the real PLC quality and timestamp in CitectSCADA and to make it available.
- Have Persistence and Replication of the tag data.

Each DataSource owns a local tag data cache. This cache is periodically saved to disk, thus, maintaining last known value (LKV) for each tag. When a DataSource is restarted its local tags are initialised to their corresponding LKVs, unless the tags have never been cached before – in which case they are initialised to the default values.

Tag data from the DataSource cache is periodically saved to disk. When the time comes to save the file, a temporary file is used, which after it is written simply gets renamed to the original file name. This is done to avoid partial master file updates or corruption. If the DataSource is shutdown abruptly, some tag value changes that are waiting to be persisted to disk may get lost. The Field data are currently saved to disk.

The Tag cache file is saved in an XML format, as shown here, preserving the necessary information about each tag’s value, quality, and timestamp. On load the original persisted quality is substituted by BadLastKnown quality value. The file is saved in the CitectSCADA Data directory using the name format of:

<ClusterName>.<IOServerName>.<IODeviceName>.cache

If during the file load the schema validation is unsuccessful, the entire file is considered invalid and the tags LKVs are lost. If during the file load a particular tag’s value, quality, or timestamp is invalid, the corresponding invalid entry is simply ignored (and logged as such). Persistence is configurable within the extended section of the CitectSCADA I/O Devices properties dialog.

Minimum Update Rate

The DataSource will send tag update value notifications to subscription clients after a pre-defined period of time expires. The configuration of the minimum update rate period is performed via the IODevice configuration dialog. Devices that are configured for redundant operation needs to have minimum update rate set according to the rules specified in the table below.

Primary Device Minimum Update Rate Value/Staleness Period Value	Standby Device Minimum Update Rate Value/Staleness Period Value	Compilation Result
Blank (Default)	Blank (Default)	OK
Blank (Default)	Value Defined	Error
Value Defined	Blank (Default)	OK
Value Defined	Same value as primary	OK
Value Defined	Value is different than on primary	Error

Stale Tag Values

When an I/O Data Consumer does not get a tag element value update from the DataSource for a specific period of time, as specified by the Staleness period, the tag element value is considered to be “stale”. The Extended Quality Substatus QUAL_EXT_STALE will indicate this condition. Processing of staleness period for tags is performed on the client side of the connection.

Configuration can be performed on the server using the IODevice properties dialog, or on the client using Citect.ini parameters.

Staleness Period

The Staleness Period represents the total number of seconds that will elapse after the last update before extended quality of the tag element is set to “Stale”. It can be configured on both the server and client. The server configuration is specified in the table above. For client configuration details refer to the ClientStalenessPeriod parameter.

Staleness Period Tolerance

Staleness Period Tolerance represents the percentage of the staleness period within a total of which time a check or stale client tag elements will be performed. The default value is 10%, which caters for a large number of configuration scenarios. You can reduce or increase the tolerance as necessary by your particular scenario by configuring this value in the client machine’s using the ClientStalenessPeriodTolerance parameter. This parameter is not available for server configuration.

Example:

In order to reduce CPU load on the client machine,the default Staleness Period Tolerance is 10%. This means that if staleness period is set to 600 seconds, a check for stale client tag elements will be performed every 60 seconds.

XML DataSource Schema

XML DataSource Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns="http://www.schneider-ele-
tric.com/Platform/PSI/DataSource/PersistenceCache/V1/"
       xmlns:dsps="http://www.schneider-
electric/Platform/PSI/DataSource/PersistenceCache/V1/"
       elementFormDefault="qualified"
       targetNamespace="http://www.schneider-elec-
tric.com/Platform/PSI/DataSource/PersistenceCache/V1/">



       <xs:simpleType name="DataType">
               <xs:restriction base="xs:string">
                       <xs:enumeration value="Boolean" />
                       <xs:enumeration value="SByte" />
                       <xs:enumeration value="Byte" />
                       <xs:enumeration value="Char" />
                       <xs:enumeration value="Double" />
                       <xs:enumeration value="Int16" />
                       <xs:enumeration value="Int32" />
                       <xs:enumeration value="Int64" />
                       <xs:enumeration value="Single" />
                       <xs:enumeration value="String" />
                       <xs:enumeration value="UInt16" />
                       <xs:enumeration value="UInt32" />
                       <xs:enumeration value="UInt64" />
                       <xs:enumeration value="Decimal" />
                       <xs:enumeration value="DateTime" />
               </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ElementName"
       <xs:restriction base="xs:string">
               <xs:enumeration value="" />
               <xs:enumeration value="Field" />
               <xs:enumeration value="Valid" />
               <xs:enumeration value="Override" />
               <xs:enumeration value="OverrideMode" />
               <xs:enumeration value="ControlMode" />
               <xs:enumeration value="Status" />
       </xs:restriction>
</xs:simpleType>

<xs:complexType name="DataSource">
       <xs:sequence minOccurs="1" maxOccurs="1">
               <xs:element name="properties" type="PropertyCollection">
                       <xs:unique name="UniquePropertyName">
                               <xs:selector xpath="dsps:property" />
                               <xs:field xpath="@name" />
                       </xs:unique>
               </xs:element>
               <xs:element name="tags" type="TagCollection">
                       <xs:unique name="UniqueTagName">
```

```
                                <xs:selector xpath="dsps:tag" />
                                <xs:field xpath="@name" />
                        </xs:unique>
                </xs:element>
        </xs:sequence>
</xs:complexType>
<xs:complexType name="PropertyCollection">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:element name="property" type="Property" />
        </xs:sequence>
</xs:complexType>

<xs:complexType name="TagCollection">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:element name="tag" type="Tag" />
        </xs:sequence>
</xs:complexType>

<xs:complexType name="Property">
        <xs:simpleContent>
                <xs:extension base="xs:string">
                        <xs:attribute name="name" type="xs:string" use="necessary" />
                        <xs:attribute name="type" type="DataType" use="necessary" />
                </xs:extension>
        </xs:simpleContent>
</xs:complexType>

<xs:complexType name="TagElement">
        <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="v" type="Value" />
                <xs:element name="q" type="Quality" />
                <xs:element name="t" type="xs:dateTime" />
                <xs:element name="qt" type="xs:dateTime" />
                <xs:element name="vt" type="xs:dateTime" />
        </xs:sequence>
        <xs:attribute name="name" type="ElementName" use="necessary" />
</xs:complexType>

<xs:complexType name="Value">
        <xs:sequence minOccurs="1" maxOccurs="unbounded">
                <xs:element name="item" type="xs:string" />
        </xs:sequence>
        <xs:attribute name="type" type="DataType" use="necessary" />
        <xs:attribute name="size" type="xs:positiveInteger" use="necessary" />
</xs:complexType>
```

```
<xs:complexType name="Quality">
        <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="generic" type="xs:integer" />
                <xs:element name="specific" type="xs:integer" />
        </xs:sequence>
</xs:complexType>
```

```
        <xs:element name="datasource" type="DataSource" />
</xs:schema>
```

Reading Tag Values

The data received from the field is represented by the Field and Valid tag elements.

- The Field element represents the latest tag field data received from the device.
- The Valid element represents the last field data which had "Good" quality.

> **Note**:The Valid tag element initially has bad quality. The quality will only become good when the first read request is successfully finished for the tag, which will only occur when the IO device is online and either background polling is enabled, the tag has been subscribed to, or a TagRead has beeninitiated.

You can access these elements by using a qualified tag reference (a tag referenced by the tag name and the element name, for example 'MyTag.Field').. The following tables describe each of these elements.

Field tag element

Reference Syntax	CitectSCADA Data type	Description
TagName.Field	INT REAL STRING	Implied value of Field element
TagName.Field.V	INT REAL STRING	Value
TagName.Field.VT	TIMESTAMP	Timestamp of when the value last changed
TagName.Field.Q	QUALITY	Quality
TagName.Field.QT	TIMESTAMP	Timestamp of when the quality last changed
TagName.Field.T	TIMESTAMP	Timestamp of when the element was last updated

Valid tag element

Reference Syntax	CitectSCADA Data type	Description
TagName.Valid	INT REAL STRING	Implied value of Valid element
TagName.Valid.V	INT REAL STRING	Value
TagName.Valid.VT	TIMESTAMP	Timestamp of when the value last changed
TagName.Valid.Q	QUALITY	Quality
TagName.Valid.QT	TIMESTAMP	Timestamp of when the quality last changed
TagName.Valid.T	TIMESTAMP	Timestamp of when the element was last updated

If you determine that the tag value is incorrect because of a sensor or communication becoming inoperative, the tag value may need to be overridden. The Override tag element is used to store the overridden tag value. For further information refer to Controlling and Overriding Tag Values.

Writing Tag Values

The tag elements which have read/write access can be modified. However, these elements can only be updated as a whole; writes to an individual element item is not allowed.

The following table shows the result of writing to each of the tag elements:

Element Name	Write allowed	Write Result
Not specified (unqualified tag reference)	Yes	Write to the Field element and propagate the value to the I/O Device.
Field	Yes	Write to the Field element and propagate the value to the I/O Device.
Valid	No	Will not succeed
Override	Yes	Write to the Override element
Override Mode	Yes	Write to the Override Mode element.

Element Name	Write allowed	Write Result
		See the appropriate function references for the available override mode values. **Note:** When the tag is in Override mode, an unqualified tag reference will refer to the Override element
Control Mode	Yes	Write to the Control Mode element The Value item can be one of the following: 0 – Control inhibit mode is Off. 1 – Control inhibit mode is On. **Note:** When Control inhibit mode is On, writing to the Field element is prohibited.
Tag Status	No	Will not succeed

If you determine that the tag value is incorrect because of a sensor or communication interuption, the tag value may need to be overridden. The Override element tag element is used to store the overridden tag value. For further information refer to Controlling and Overriding Tag Values.

Controlling and Overriding Tag Values

Tag extensions in CitectSCADA allow you to control the ability of a tag to be written to, or to override a value read from a tag if it is considered to be incorrect due to some loss of communication with the PLC.

These functions are managed by setting the tag into Control Inhibit Mode or Override Mode.

Control Inhibit Mode

⚠ **WARNING**

PERFORMING A CONTROL OPERATION WHICH DOES NOT IN FACT OCCUR.

When the tag is put into the control inhibit mode, writing to the Field element value is prohibited. If the system is configured in such a way that it does not give the operator any indication that the tag is in the control inhibit mode, the operator may assume that he is performing a control operation which does not, in fact, occur.

Configure the system in such a way that it provides a visual indication to the operator that a tag is in the control inhibit mode.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

To provide a visual indication to the operator that a tag is in the control inhibit mode the following can be done:

Set any of the following Citect.ini parameters in such a way that control inhibit mode will be indicated by changing the background color or overlaying the numeric or text graphics objects and symbol set objects with a dithered pattern:

[Page]ControlInhibitDitheringColor

[Page]ControlInhibitDitheringDensity

[Page]ControlInhibitTextBackgroundColor

and set [Page]IgnoreValueQuality parameter to a value of 0 or 2.

or

Use the Control Mode element value (0 if the tag is not in the control inhibit mode or 1 otherwise)

or

Use the Field element quality Tag Status ControlInhibit bit (1 if the tag is in the control inhibit mode or 0 otherwise)

Control Mode

Control inhibit mode allows you to prohibit writing to the Field tag element. Setting a tag in Control inhibit mode is applied system wide. Writing to the Field element will be prohibited for each of the I/O Data Consumers.

Control inhibit mode is controlled by the “Control Mode” element which is described in the following table.

Reference Syntax	CitectSCADAData type	Description
TagName.ControlMode	INT	Value of Control Mode element
TagName.ControlMode.V	INT	Value Allowed values are: 0 – Control inhibit mode is Off. 1 – Control inhibit mode is On. Default value: 0.
TagName.ControlMode.VT	TIMESTAMP	Timestamp of when the value last changed
TagName.ControlMode.Q	QUALITY	Quality. The Control Mode quality is QUAL_GOOD if the tag was put into the Control inhibit mode on the primary server and it was propagated to redundant servers. Otherwise the general quality status will be QUAL_UNCR and the extended susbstatus will be QE_NOT_REPLICATED to indicate

Reference Syntax	CitectSCADAData type	Description
		that not every redundant server is aware of the fact that the tag is in Control inhibit mode.
TagName.ControlMode.QT	TIMESTAMP	Timestamp of when the quality last changed
TagName.ControlMode.T	TIMESTAMP	Time of when the tag was put in or out of the Control inhibit mode. (equal to 0 at start up)

Note: The quality of tags referenced by items, ex. Tag1.v or Tag1.Field.t, is GOOD and its timestamps are 0 (INVALID TIMESTAMP). Therefore they give no visual indication of any not good quality, error or change in handling state such as control inhibit or override mode regardless of the setting used for the [Page] IgnoreValueQuality parameter.

Override Mode

⚠ WARNING

PERFORMING A CONTROL OPERATION BASED ON INACCURATE DATA.

When the tag is put into the override mode, the default tag element value will be equal to the Override element value instead of the Field element value. If the system is configured in such a way that it does not give the operator any indication that the tag is in override mode, the operator may assume that he is seeing the Field element value (live data) and consequently may operate the plant inappropriately.

Configure the system in such a way that it provides a visual indication to the operator that a tag is in override mode.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

To provide a visual indication to the operator that a tag is in the override mode the following can be done:

Set any of the following citect.ini parameters in such a way that override mode will be indicated by changing the background color or overlaying the numeric or text graphics objects and symbol set objects with a dithered pattern:

[Page]OverrideDitheringColor

[Page]OverrideDitheringDensity

[Page]OverrideTextBackgroundColor`

and set [Page]IgnoreValueQuality parameter to a value of 0 or 2.

or

Use the Override Mode element value (0 if the tag is not in the Override Mode or 1,2,3,4 otherwise)

or

Use the default or override element quality Tag Status Override bit (1 if the tag is in the override mode or 0 otherwise)

Using Override Mode

If you determine that the tag value is incorrect because of a sensor or communication interuption, the tag value may need to be overridden. The Override tag element is used to store the overridden tag value.

You can put the tag into the Override Mode which will cause each unqualified tag reference (a tag referenced only by the tag name, for example 'MyTag').in every I/O Data Consumer (Control Client, Trend Server, Alarm server, etc) to return the Override element.The quality of the tag will indicate that the tag is in override mode.

Tag override is controlled by the Override and Override Mode tag elements. The Override element contains the override value and the Override Mode tag element is used to set the specific override mode.

The following tables describe each of these elements:

Override tag element

Reference Syntax	CitectSCADA Data type	Description
TagName.Override	INT REAL STRING	Implied value of Override element
TagName.Override.V	INT REAL STRING	Value
TagName.Override.VT	TIMESTAMP	Timestamp of when the value last changed
TagName.Override.Q	QUALITY	Quality. The Override value has "Good" quality except when it is out of range
TagName.Override.QT	TIMESTAMP	Timestamp of when the quality last changed
TagName.Override.T	TIMESTAMP	Timestamp of when the element was last updated

Note: The quality of tags referenced by items, ex. Tag1.v or Tag1.Field.t, is constantly GOOD and its timestamps are constantly 0 (INVALID TIMESTAMP). Therefore they give no visual indication of any not good quality, error or change in handling state such as control inhibit or override mode regardless of the setting used for the [Page] IgnoreValueQuality parameter.

Override Mode tag element

Reference Syntax	CitectSCADA Data type	Description
TagName.OverrideMode	INT	Value of Override Mode element
TagName.OverrideMode.V	INT	Override Mode Value. See the appropriate function references for the available override mode values.
TagName.OverrideMode.VT	TIMESTAMP	Timestamp of when the value last changed
TagName.OverrideMode.Q	QUALITY	The Override Mode quality is QUAL_GOOD if the tag was put into the Override mode on the primary server and it was propagated to any redundant servers. Otherwise the general quality status will be QUAL_UNCR and the extended substatus will be QE_NOT_REPLICATED to indicate that some redundant servers are unaware of the fact that the tag has been overridden.
TagName.OverrideMode.QT	TIMESTAMP	Timestamp of when the quality last changed
TagName.OverrideMode.T	TIMESTAMP	Time when the tag was put in or out of Override mode. (equal to 0 at start up)

Override Modes

Override Mode Value	Description
0	The Override Mode is Off.
1	The tag is in Static Override Mode and the Override value is initially set to the value of the tag’s Field element.
2	The tag is in Static Override Mode and the Override value is initially set to the value of the tag’s Valid element.
3	The tag is in Static Override Mode and the Override will remain at the previously set value of the tag's Override element.
4	The tag is in the Dynamic Override Mode and the Override value will continually track the value of the tag's Valid element. The ability to write to the value of the tag’s Override element is disabled in this mode.

Tag Status

The Tag Status element is used to represent the current status of the tag. The value of this element is affected only for those tag operations that involve a physical device (for example, writing to the Field element will affect the status element, but writing to the Control Mode will not). Tag Status element is reset after I/O Server restart. The element value contains a set of bit flags.

The following table describes the Tag Status element:

Reference Syntax	CitectSCADA Data type	Description
TagName.Status	INT	Value of Tag Status element
TagName.Status.V	INT	Value A set of bit flags representing the following data: - 0x1: Read Pending. - 0x2: Write Pending. - 0x4: Write Successful. - 0x8: Write Unsuccessful. Default value: 0.
TagName. Status.VT	TIMESTAMP	Timestamp of when the value last changed
TagName.Status.Q	QUALITY	Quality
TagName.Status.QT	TIMESTAMP	Timestamp of when the quality last changed
TagName.Status.T	TIMESTAMP	Timestamp of when the element was last updated

Tag Data Types

Tags in CitectSCADA hold data values that can be defined as one of the following data types.

Data Type	Variable	Size	Allowed Values
BCD	Binary- Coded Decimal	2 bytes	0 to 9,999
BYTE	Byte	1 byte	0 to 255

Data Type	Variable	Size	Allowed Values
DIGITAL	Digital	1 bit or 1 byte	0 or 1
INT	Integer	2 bytes	-32,768 to 32,767
UINT	Unsigned Integer	2 bytes	0 to 65,535
LONG	Long Integer	4 bytes	-2,147,483,648 to 2,147,483,647
ULONG	Unsigned Long Integer (Only for display on a screen.Arithmetic operations are not supported.)	4 bytes	0 to 4,294,967,295
LONGBCD	Long Binary- Coded Decimal	4 bytes	0 to 99,999,999
REAL	Floating Point	4 bytes	-3.4E38 to 3.4E38
STRING	String	256 bytes (maximum)	ASCII (null terminated)

Tag values can be used with the Cicode Variable data types.

A Cicode variable of INT data type can be used to store Tag data types: BCD, BYTE, DIGITAL, INT, UINT, LONG, ULONG and LONGBCD.

A Cicode variable of the QUALITY or TIMESTAMP data type can be used to store the Tag quality and timestamp items.

See Also

Cicode Variable Data Types

Configuring Variable Tags

To configure a variable tag:

1. Start Citect Explorer.
2. Click **Variable Tags,** or choose **Tags** | **Variable Tags**. The Variable Tags form displays.

3. Enter the properties of the variable tag.
4. Click **Add** to append a new record, or **Replace** to modify a record.

At a minimum you need to enter the **Variable Tag Name**, **I/O Device Name**, **Data Type**, and **Address** fields.

You can paste any existing variable tag into forms in your project.

To select an existing variable tag:

1. Open the Project Editor.
2. Choose **Edit | Paste Tag**.

To configure a digital tag:

1. Open the Project Editor.
2. Click **Variable Tags**, or choose **Tags | Variable Tags**.
3. Complete the properties in the **Variable Tags** dialog box that appears, using **DIGITAL** as the **Data Type**.
4. Click **Add** to append a new record, or **Replace** if you have modified a record.

You need to at least enter the **Variable Tag Name**, **I/O Device Name**, **Data Type**, and **Address** fields. Leave the following properties blank:

- **Raw Zero Scale**, **Raw Full Scale**
- **Eng Zero Scale**, **Eng Full Scale**
- **Eng Units**, **Format**

To configure an analog tag:

1. Start the Project Editor.
2. Click **Variable Tags** or choose **Tags | Variable Tags**. The Variable Tags form appears.
3. Enter the properties, using **INT** (or Real, BCD, Long, LongBCD) as the **Data Type**.
4. Click **Add** to append a new record, or **Replace** to modify a record.

You need to at least enter the **Variable Tag Name**, **I/O Device Name**, **Data Type**, and **Address** fields.

See Also

Variable Tag Properties
Formatting numeric variables

Variable Tag Properties

You can use this dialog for Configuring Variable Tags. Variable tags have the following properties:

Variable Tag Name

You can use any name for a tag (79 characters) provided it follows the Tag name syntax and isn't the same as the name of a Cicode function within the project or any included projects (see Defining Variable Tag Names for a list of restrictions on naming). If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.When the tag name is referenced on a graphics page, Cicode, etc., it can be used with or without a specific tag element or item.

If you are using distributed servers, the name needs to be unique to the cluster (for example, you cannot have the same variable tag name in more than one cluster).

Cluster Name

The name of the cluster to which the tag applies (16 characters). Select a cluster name from the list of available clusters as defined under Cluster Definitions.

Data Type

The type of I/O Device variable (16 characters). I/O Devices support several data types that are used to exchange data with CitectSCADA. Because of the lack of an industry standard, most I/O Device manufacturers use individual naming conventions for their I/O Device variables. However, every variable corresponds to one of the Tag data types.

> **Note:** If you do not specify a range for your tag, then an out of range alert message will be generated if you write a value which is outside the range of the type.

You need to specify the correct data type that corresponds to the data type of the I/O Device variable you are configuring. Each data type has a unique address format. You need to use this format when you are specifying the address of the variable (as the Address property).

You will want to verify that you only use data types that are valid for your I/O Device. If you do not specify a data type, the variable will be treated as 16-bit integer.

CitectSCADA supports concatenation of I/O Device registers. For example, you can define a real data type (in CitectSCADA) as two contiguous int data types (in the I/O Device). CitectSCADA reads across the boundary of the two ints and returns a real.

If you use concatenation of registers, the address of the variables needs to be on odd boundaries or even boundaries. You cannot mix address boundaries. For example, V1, V3, V5 are valid addresses.

Be careful when using this feature: the I/O Device needs to maintain the integrity of the second register. (If the I/O Device writes to the second int, the value of the real could be corrupted.) The structure of some I/O Devices might not support this feature, and might therefore behave unpredictably.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not mix the use of odd and even variable addresses as boundaries when you are concatenating registers.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

String variables

While numeric variables are more common, some I/O Devices also support ASCII strings. You can use strings to store text data (for example, from a bar code reader).

All strings needs to be NULL-terminated in the I/O Device. CitectSCADA uses the NULL character to check for the end of a string, and if no NULL character is present, CitectSCADA reads (and displays) any extra characters in memory, after the end of the string.

When you are using a disk I/O Device, you can also specify a string data type for storage of recipes, or for operator display information.

Not every I/O Device supports strings. However, if your I/O Device does support integer data types, CitectSCADA can use these integer registers to store ASCII strings in your I/O Device. CitectSCADA strings can only be stored in contiguous blocks (consecutive registers), and are stored as an array.

Note: Non printable characters within string tag values will be substituted with spaces e.g. string [0x01, 0x41, 0x10, 0x42] will appear as " A B", so cache loading continues to operate.

To display the data types for an I/O Device, double-click the I/O Devices book from the **Help**, select your I/O Device from the list, and then select the **Data Types** topic.

I/O Device Name

The name of the I/O Device where the variable is stored (31 characters). If using I/O Device redundancy, you need to specify the primary I/O Device name here, not the standby.

Address

The register address in the I/O Device where the variable is stored (254 characters). The format and prefix of an address will depend on the protocol configured for the I/O Device, which can be determined by checking the **Protocol** field on the I/O Devices form in Project Editor.

Raw Zero Scale / Raw Full Scale

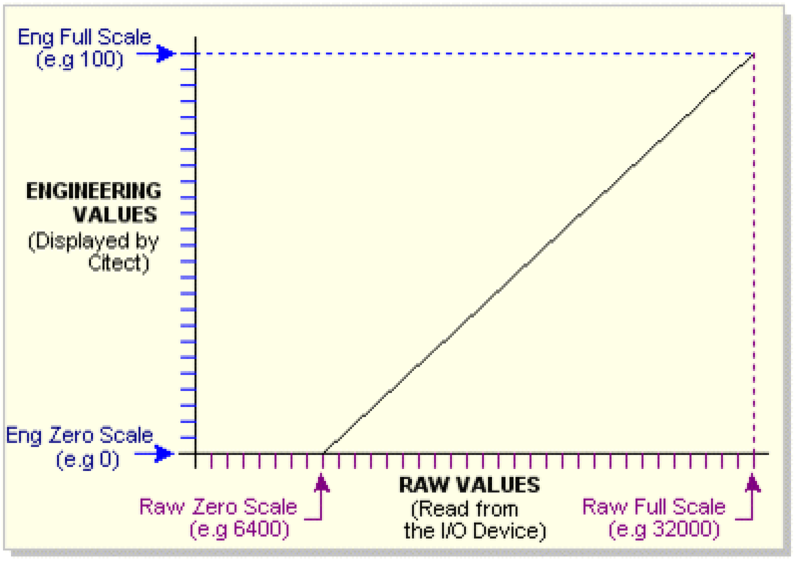
The unscaled (raw) values (of the variable) that represent the zero point and full scale point for the data (11 characters). The raw values are the values that CitectSCADA reads from the I/O Device.

Eng Zero Scale / Eng Full Scale

The scaled values that CitectSCADA calculates from the raw values (11 characters). The Raw Zero Scale is scaled to the Eng Zero Scale and the Raw Full Scale is scaled to the Eng Full Scale. These properties are represented in engineering units and are used as the upper and lower limits of trends and bar graphs.

Most I/O Devices return an integer to indicate the value of an analog input. To return a usable value, the I/O Device converts an input signal (usually 4-20 mA) to a raw scale variable, usually (but not always) in the range 6400 to 32000.

To present this variable as a meaningful value, you can specify a scaling calculation. CitectSCADA then scales every value accordingly, as in the following diagram.



The scaled value of the variable (engineering value), not its raw value, is used throughout the system.

The scaling properties are optional. If you do not specify scaling, Eng Zero Scale defaults to Raw Zero Scale, and Eng Full Scale defaults to Raw Full Scale; that is, no scaling occurs.

A value that is below the specified Raw Zero Scale or above the specified Raw Full Scale causes an "Out of Range" alert message in your runtime system.

Eng Units

(8 characters.) The engineering units that the value represents (for example %, deg, mm/sec, etc.). This property is optional. If you do not specify engineering units, no engineering units are used.

Format

(11 characters). The display format of the value (of the variable) when it is displayed on a graphics page, written to a file, or passed to a function (that expects a string). This property is optional. If you do not specify a format, the format defaults to ####.##.

Deadband

(11 characters). A deadband allows the value of a variable tag to fluctuate within a defined threshold without updates being sent through the system. This may be useful if a tag produces many small, insignificant value changes. The threshold is represented as a percentage of the tag's engineering range. The default value is 0 (zero), which captures every value change.

For example, if a variable tag has an engineering range of zero to 10000, a deadband of 1 would mean a change in value would have to be greater than 100 (or 1 percent of the range) to be recognized. If the current value was 5600, the tag in the PLC would have to change to a value greater than 5700 or less than 5500 before an update would be sent through the system.

Comment

Any useful comment (48 characters).

Linked

The **Linked** field in the status line of the Variable Tags form reads either **Yes** or **No** and indicates whether or not the variable tag is linked to an external data source. When you program an I/O Device using software other than CitectSCADA, an external data source is used to store tag data. Linked variable tags are updated whenever external tag data changes, meaning you do not need to enter the information again in CitectSCADA.

Defining Variable Tag Names

You cannot use certain reserved words when defining your tag name variables. You cannot use:

- Reserved words such as:

&	*	**	-
/	+	<	<=
>=	<>	=	>
ACTION	AND	ANY_NUM	ARRAY
BOOL	CASE	CONFIGURATION	CONSTANT
DATE	DATE	DINT	DO
DUT	DWORD	ELSE	EN
END_CASE	END_CONFIGURATION	END_FOR	END_IF
END_PROGRAM	END_VAR	END_WHILE	ENO
EXIT	FALSE	FOR	FUNCTION
IF	INT	INT	MOD
NOT	OF	OR	PROGRAM
REAL	REPEAT	STRING	STRING
TASK	TIME	TIME	TO
TRUE	TRUE	UNTIL	VAR_EXTERNAL
VAR_GLOBAL	VAR_IN_OUT	VAR_INPUT	VAR_OUTPUT
WHILE	WORD	XOR	

- Cicode function names. See Functions Reference for an entire list of Cicode functions.

Using reserved words or Cicode function names may cause a syntax error.

Formatting numeric variables

The value of a numeric variable (number) can be displayed on a graphics page or written to a file in many different formats (for example, 24, 0024, 24.000, or 24.0%).

Format specifiers

Format specifiers are keyboard characters that you use to define the format for the numeric variable. The specifiers that you can use are:

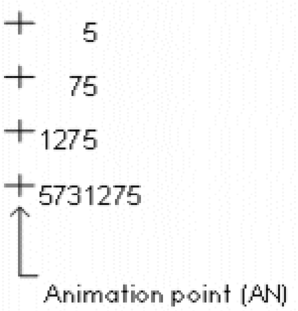
Specifier	Description	Function
#	The hash character	The number of characters to display
0	Zero	Padding
-	Minus	Justification
.	Period	Decimal notation
EU		Engineering units
S		Exponential notation

Specifying the number of digits

The hash character (#) specifies how many digits CitectSCADA displays. every numeric variable displays to the right of an animation point, for example:

Format: ####

In this example, CitectSCADA displays at least four digits (or spaces) to the right of the animation point. If the number contains more than four digits, the first digit is located at the animation point. The following figure illustrates several numbers in the above format.

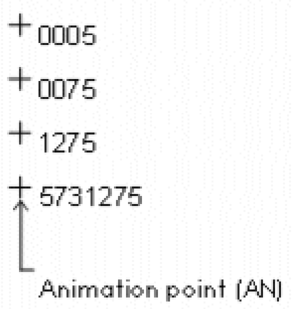


Padding with zeros

When a number contains fewer digits than your format specifies (5 or 75 in the above example), CitectSCADA only displays the meaningful digits. You can use the padding character, zero (**0**), as the second character in the format string, to fill the number with zeros, for example:

Format:#0##

This format string displays:

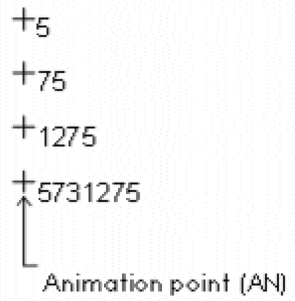


Changing justification

By default, numeric variables are right-justified (within their field). You can change the default justification by using a minus (**-**) sign as the second character in the format string, for example:

Format: #-###

This format string displays:



Specifying decimal notation

To specify decimal notation, use a period (**.**), for example:

Format: ###.##

In this example, CitectSCADA displays three digits before the decimal point and two digits after. If the variable is 12.3, CitectSCADA displays 12.30.

All numbers are automatically rounded, i.e. 12.306 displays as 12.31.

Specifying engineering units

You can specify engineering units (such as %, deg, rpm, M, mm/sec, etc.) when you define a variable tag. To include these units in the format of the number, type **EU** in the appropriate position.

Format: ####.##EU

> **Note:** If you do not specify an engineering unit for the variable, only the number is displayed (or logged).

Specifying exponential notation

To specify exponential notation, include the exponential character (**s**), for example:

Format: #s###

In this example, CitectSCADA displays the number in exponential notation format, for example: 1.234e+012.

Combining format specifiers

You can combine format specifiers, for example:

Format: #0-###.##EU

This format string displays six digits before the decimal point and two digits after. The number is left justified, padded, and displayed with engineering units (if specified).

Using shortform notation

As an alternative to the hash (#) notation, you can use shortform notation. With shortform notation, you use a number to specify the format of the numeric variable, for example:

Format: 3.2

This format string displays three digits before the decimal point and two digits after. This format string is equivalent to the ###.## format specification. You can also include engineering units with shortform notation, for example:

Format: 6.0EU

This format string displays six digits before the decimal point and no digits after. The number is displayed with engineering units (if specified). This format string is equivalent to the ######EU format specification.

You cannot use padding or left justification with shortform notation.

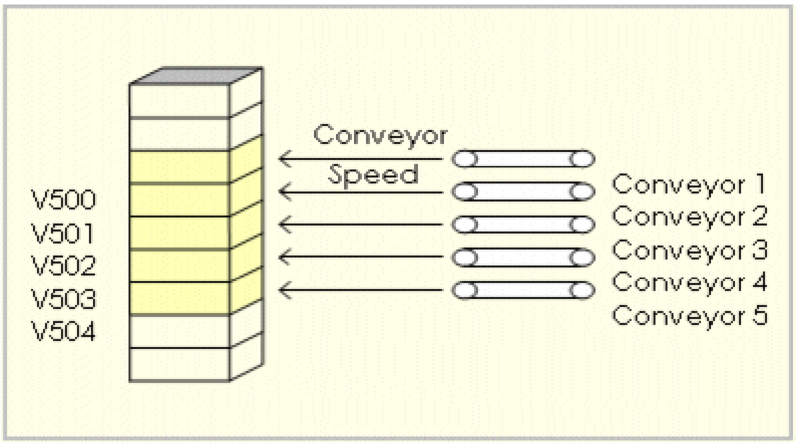
> **Note:** If the value of a numeric variable is extremely large, it is displayed in exponential notation (for example 1.2345E012).If no format is specified for a variable, the default ####.# (or 4.1) is used.

See Also

Using arrays

Using arrays

An array is a collection of variables (all of the same data type) that are stored in consecutive memory registers in your I/O Device. Numeric arrays are useful when you have separate devices (or processes) performing similar functions. You can program the I/O Device to store, in consecutive memory registers, variables that relate to each of these processes, for example:



Here, five consecutive variables (V500, V501, V502, V503, and V504) store the conveyor speed of five conveyors (1 to 5). Instead of configuring five separate variable tags (one for each conveyor), you can configure a single tag, as an array.

To specify a single variable tag for an array, define the first address and add the size of the array (the number of consecutive addresses) to the register address, for example:

Variable Tag Name	Conveyor_Speed
Address	V500[5]

Here, five register addresses are referred to by the variable tag Conveyor_Speed.

Referring to array variables

Each entry of an array is referred to by an index. You can extract individual variables (from the array) by specifying the tag name and index:

```
<Tag Name>[Index]
```

For example, to refer to the third variable of the array in the above example (Conveyor 3), use the following syntax:

Variable Tag	Conveyor_Speed[2]

The index of the first entry of an array is always 0 (zero). In this example, Conveyor_Speed[0] is the first entry of the array (Conveyor 1), and Conveyor_Speed[4] is the last entry (Conveyor 5).

Note the following when using arrays:

- Do not define large arrays, because each time an individual array entry is requested, CitectSCADA reads the entire array from the I/O Device. With large arrays, system performance could be reduced.
- The size of the array needs to be less than the maximum request length of the protocol. The I/O Device description help topic (for your I/O Device) displays the maximum request length of the protocol.
- declare digital arrays on a 16 bit boundary. CitectSCADA rounds each digital array down to the nearest 16 bit boundary. Consequently, digitals in the array are changed. For example, if you declare an array Test to start at bit 35, and CitectSCADA starts the array at bit 32. The index to the array also starts at bit 32; Test[0] refers to bit 32, not bit 35.
- Each individual array entry has its own data value for the field, valid and override elements, but the entries within an array share the override and control mode elements, quality and timestamps. Reading the quality or timestamp for an indexed array entry will return the quality or timestamp for the entire array. Writing to the value for an indexed array entry will change the timestamps and quality for the entire array. Changing the override mode and the control mode for an array entry will change it for the entire array.

See Also

Tag Extensions

String arrays

If you are using a CitectSCADA string data type, you need to specify an array of integer data types. One int register stores two string characters.

To calculate the size of an array (for string data types), use the following formula:

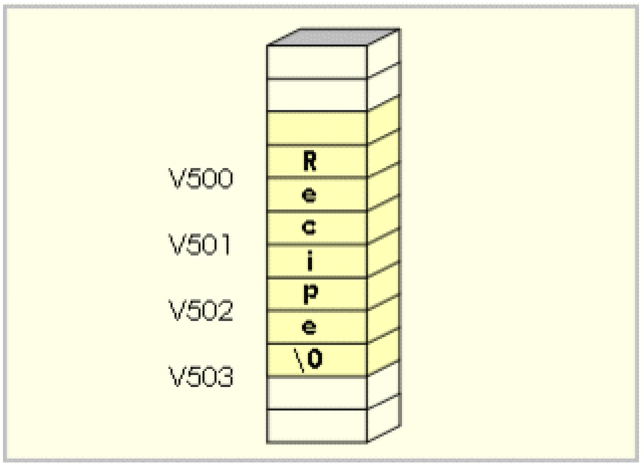
$$\text{Size of Array} = \frac{(\text{number of characters} + 1)}{2}$$

The last element of the array is always used to store the null character '\0'. This character marks the end of the string.

To store the word "Recipe", you need to specify an array with 4 elements, for example:

Variable Tag Name	Recipe_Tag
Address	V500[4]

Two characters are stored in each register, i.e:



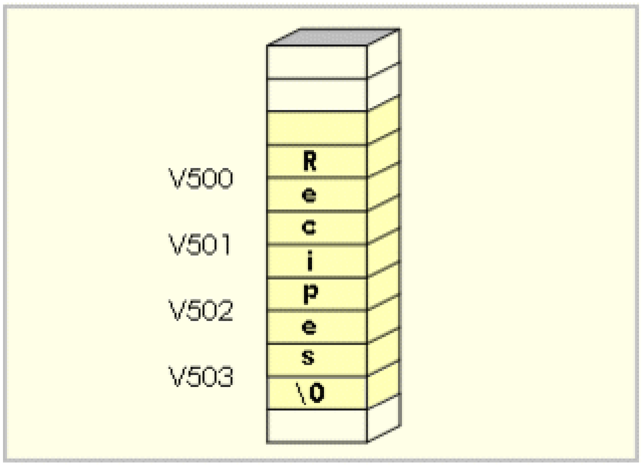
You can then refer to the entire string by specifying the tag:

```
<Tag Name>
```

For example:

Variable Tag	Recipe_Tag

To store the word "Recipes", you would also specify an array with 4 elements. The characters are stored as follows:



> **Note:** If your I/O Device supports string data types, you need to specify the address in the format determined by the I/O Device you are using.

See Also

Using structured tag names

Configuring Local Variables

Local variables allow you to store data in memory when you start your runtime system. They are created each time the system starts, and therefore do not retain their values when you shut down. Local variables may be of any data type supported by CitectSCADA, including two-dimensional arrays of standard CitectSCADA types.

Local variables are useful when you need each process to have a separate copy of the data. Each process has its own copy of each local variable configured in the project, and the values in a local variable are available only to the process that wrote them.

To configure a local variable:

1. In Project Editor, select **Tags | Local Variables.** The Local Variables dialog displays.
2. In the **Name** field, enter a name for the local variable. Variable names cannot include the '-', '/', '%' or <space> characters.
3. In the **Data Type** field (16 characters), select one of the following supported data types:

Data Type	Variable	Size	Allowed Values
BCD	Binary- Coded Decimal	2 bytes	0 to 9,999

BYTE	Byte	1 byte	0 to 255
DIGITAL	Digital	1 bit or 1 byte	0 or 1
INT	Integer	2 bytes	-32,768 to 32,767
UINT	Unsigned Integer	2 bytes	0 to 65,535
LONG	Long Integer	4 bytes	-2,147,483,648 to 2,147,483,647
LONGBCD	Long Binary-Coded Decimal	4 bytes	0 to 99,999,
REAL	Floating Point	4 bytes	-3.4E38 to 3.4E38
STRING	String	up to 256 bytes	ASCII (null terminated)

Numeric and digital variables have a default value of 0 and string variables default to "" (empty string). If you do not specify a data type, the local variable will be treated as 16-bit integer.

4. In the **Array Size** field (8 characters), enter the size of the array (number of elements) used to store the local variable. The array will be of the data type specified in the **Data Type** field. The array can be one or two-dimensional. Up to 32766 elements can be used, which, for a two dimensional array, means that the size of the first dimension multiplied by the size of the second is less than or equal to 32766. When specifying a multi-dimensional array, separate the dimensions with a comma, for example "20,30"."
5. In the **Zero Scale** field (11 characters), enter the value of the local variable that represents the zero point for the data. The zero scale value is used as the lower limit for trend and bar graphs, and values below the zero scale value will cause an "Out of Range" alert message in the runtime system.
6. In the **Full Scale** field (11 characters), enter the value of the local variable that represents the full scale point of the data. The full scale value is used as the upper limit for trend and bar graphs, and values above the full scale value will cause an "Out of Range" alert message in the runtime system.

7. In the **EngUnit** field, enter the engineering units that the value represents (for example %, deg, mm/sec, etc.). This property is optional. If you do not specify engineering units, no engineering units are used.
8. In the **Format** field, Enter the display format of the value (of the variable) when it is displayed on a graphics page, written to a file, or passed to a function (that expects a string). This property is optional. If you do not specify a format, the format defaults to `####.#`.
9. In the **Comment** field, enter any useful comment. This property is optional and is not used at runtime.
10. Click **Add**.

See Also

Configuring Variable Tags

Chapter: 17 Linking, Importing, and Exporting Tags

Because I/O Devices are often programmed independently of CitectSCADA, you can import, or link to, every the tag in an external data source. This means that you only have to define tag information once when you program your I/O Devices. You do not have to re-enter the same information again in CitectSCADA. Because the necessary information is already saved in an external data source, you can just import it or link to it.

There are two types of tags in the CitectSCADA variable database - linked tags and non-linked tags. When performing a tag import from Citect Explorer, only non-linked tags are updated. Linked tags remain unchanged. When performing a refresh or auto-refresh, only linked tags are updated. Non-linked tags remain unchanged. When performing a synchronization (Live Update), every tag for the I/O device are updated and linked.

CitectSCADA also lets you export tags to an external file, specifying the destination and format of your choice. You might then import this file into a third-party I/O Device programming package database, or simply use it as a backup.

> **Note:** The simultaneous update of a tag name and its fields is only supported when importing via OFS. In every other case, if you attempt to simultaneously update a tag's name and fields, only the name will be updated. In this case update a tag name in a separate operation from updating its fields.

See Also

Linking tags
Importing tags
Exporting tags
Unity Support Matrixes

Linking tags

Linking is an I/O Device specific operation. When you add an I/O Device record in CitectSCADA (using the Express Communications Wizard), you can choose to link it to an external data source. The external data source is where the tag data was saved when the actual I/O Device was programmed. If you link to the external data source, CitectSCADA automatically creates variable tag records for every tag in the I/O Device.

These variable tag records are dynamically linked to the tags in the external data source. CitectSCADA is updated whenever one of the external tags is changed. For example, you might program your I/O Devices, configure your project, then add some new tags or edit existing tags. In this situation, CitectSCADA is automatically updated with your changes.

This update occurs when you attempt to read the changed tags in CitectSCADA (for example you compile your project, display the tag using the Variable Tags dialog box, modify or paste the tag, or perform a manual refresh, etc.). For example, if you change the address of a tag using a third party I/O Device programming package, the external data source is changed. Then, when you display the variable tag record or compile your project in CitectSCADA, the change is copied from the external database to CitectSCADA's variable tags database.

You can tell if a tag is linked because the status line at the bottom of the Variable Tags form will read **Linked: Yes**. If it is linked and you change a field, your change will not be overwritten when the link is next refreshed. Generally, however, any field which takes its value from the external data source is disabled.

Note: Some properties defined for the external tags will not be relevant to CitectSCADA. Also, some will not be in a format that CitectSCADA can read. Each I/O Device has an associated format file in CitectSCADA. It is this file that determines what information is copied to CitectSCADA's variable tags database and how this information is to be converted. In most cases, you will not have to edit this file.

Note the following for Citect tags linked to Unity tags:

1. The addition of a new tag on either side, will result in the addition of a new tag on the other side.
2. The deletion of an existing tag on either side, will result in the deletion of the corresponding tag on the other side.
3. The modification of an existing tag on either side, will result in the modification of the corresponding tag on the other side.
4. The delete operation in either Unity or Citect will take precedence over other operations (such as modify).
5. If there is some contention between the Unity database and the Citect database for the same tag, that is modifications are done from both databases at the same time, the Unity database item will take precedence over the Citect database item.
6. The linked I/O Device live update synchronization will be triggered in the following cases:
    1. Unity configuration update.

2. Citect configuration update.
3. Manual refresh.

See Linked Tags for information about levels of support between Unity and Citect-SCADA.

Breaking the link to the external data source

You can break the link to the external data source from the I/O Devices form or through the Express Communications Wizard. If you break the link, you can choose to make a local copy of the tags or you can simply delete them altogether.

Deleting the I/O Device

If you delete an I/O Device record which is marked as linked, you can choose to make a local copy of the linked tags or you can simply delete them.

To link to tags in an external data source:

1. Start the Project Editor.
2. Choose **Communication** | **I/O Devices**.
3. Scroll to the relevant I/O Device and choose **Linked** | **True**.
4. Complete the remaining fields as necessary.

To link to tags in an external data source using the Express Communications Wizard:

1. Start the Project Editor and choose **Communication** | **Express Wizard**. (Alternatively, you can open Citect Explorer, and then click **Express I/O Device Setup** in the **Communications** folder of the current project.)
2. Complete the wizard screens one by one, selecting the relevant I/O Device, and so on. When the **Link to External Database** screen appears, select the **Link I/O Device to an external tag database** check box and complete the remaining details.

To manually refresh linked tags:

1. Open Citect Explorer.
2. Choose **Tools** | **Refresh Linked Tags** to display the **Refresh Linked Tags** dialog box.

See Also

Refresh Linked Tags properties
Importing tags

Refresh Linked Tags properties

Use this dialog to refresh linked tags. The dialog has the following field:

Select Linked I/O Devices

Every linked I/O Device in your project (and included projects) are listed here. To refresh the tags for an I/O Device, click the I/O Device, then click **Refresh**. This updates your project with the latest tag values for the selected I/O Device.

If you use CitectSCADA to modify any I/O Device tags, your modifications will not be overwritten on refresh.

Importing tags

Importing tags from an external data source lets you halve your data entry time. Instead of entering your tag information once when you program your I/O Device and once when you configure your project, you can program your I/O Device, then import the tags straight into CitectSCADA, where they are treated as regular tags. (CitectSCADA automatically creates variable tag records for every tag in the I/O Device.)

Note: CitectSCADA only supports the use of alphanumeric characters, the backslash and underscore characters (see Tag name syntax). Any other characters will be converted to underscores on import. This may result in Duplicate Tag errors on compilation.

Like linking, the importing of tags is an I/O Device specific operation: you import the tags for a particular I/O Device. Unlike linking, however, imported tag records are not linked in any way to the tags in the external data source. Therefore, importing is typically a one-time operation. To update imported tags, you need to import them again.

Note: Tags will be imported into the project in which the selected I/O Device is defined. This could be either the project selected in the Explorer, or in one of its included projects.

For most external data sources, the import process involves two steps. First you export the data from the I/O Device to a format that CitectSCADA can read, then you import the database into CitectSCADA. However, tag data saved using Mitsubishi or Unity FastLinx can be read directly by CitectSCADA. This means that no export is necessary.

Some characteristics of the import are defined in the format files in the `Config` directory (with the extension `.fmt`). There is one format file for each database type, and each file specifies how external data is converted to CitectSCADA variable data. In general, imported tags are either added to the CitectSCADA variable database if the tag does not already exist, or updated if the tag exists. However, if a field is listed in the `[EditableFields]` section of the format file corresponding to the import database type:

- If the tag already exists, the CitectSCADA field for the tag will not be updated with the external value.
- If the tag does not yet exist, the CitectSCADA field for the tag will be updated with the external value.

For example, if a tag has 10 fields to be imported and 5 of them are listed in the [EditableFields] section of the format file, then if the tag already exists in CitectSCADA the tag will be updated with the external values for the 5 fields that are listed in [EditableFields]. If the tag does not already exist then the tag will be created with the external values for every 10 fields.

Fields marked as editable in this way can be changed by CitectSCADA. To prevent internal changes being lost, changes to these fields in the external database are ignored.

Note: If the external database supports to LiveUpdate and the **LiveUpdate Link** has been selected for the I/O Device, existing CitectSCADA tags will be synchronized with the external database and the tags will be updated for the imported fields regardless of whether it is marked as an editable field. An I/O Device configured as **Linked** or **Auto-refreshed Linked** will behave as in a normal import and will not update fields marked as editable.

When you import tags into CitectSCADA, you have two options for dealing with existing tags:

- Delete tags associated with that I/O Device prior to the import.
- Update existing tags on import. Tags found in CitectSCADA and in the external data source are updated in CitectSCADA. Tags found in CitectSCADA but not in the external data source will remain untouched. New tags are appended.

If you import a data source that is already linked, every tag in the data source are duplicated. That is, you will have two copies of each tag: one local and one linked.

See Imported Tags for information about levels of support between Unity and CitectSCADA.

Some properties defined for the external tags will not be relevant to CitectSCADA. Also, some will not be in a format that CitectSCADA can read. To define what information is copied to CitectSCADA's variable tags database and how this information is to be converted, you need to edit the I/O Device's format file.

To import variable tags from an external database:

1. Open Citect Explorer.
2. Choose **Tools** | **Import Tags**.
3. Complete the **Import Variable Tags** dialog box as necessary.

See Also

Import variable tags properties

Import variable tags properties

Use this dialog for Importing tags from an external data source.

> **Note:** If an I/O Device is linked to an external data source the Database Type, External Database, Connection String, and Tag Prefix fields will be greyed out.

The Import Variable Tags dialog has the following fields:

I/O Device

The I/O Device for which you are importing tags. Use the menu to select an I/O Device that has been defined using CitectSCADA.

> **Note:** Tags will be imported into the project in which the selected I/O Device is defined. This could be either the project selected in the Explorer, or in one of its included projects.

Database type

The format of the data referenced by the external data source.

External database (128 Chars.)

References the source for the external database. Click the Browse button to either navigate to the file or to specify configuration options. The external database can be:

- An explicit path and file (for example, `C:\Data\Tags.csv`)
- An IP address and node (for example, `127.0.0.1\HMI_Scada`)
- A URL (for example `http://www.abicom.com.au/main/scada`)
- A computer name (for example `\\coms\data\scada`)

Tag data can be read directly from a datasources. Click the browse button to specify configuration options. See or Unity Link Tag Import.

Connection string (128 Chars.)

Connection details for the data source. This is similar to an ODBC connection string. For example:

```
UserID = XXX; Password = YYY
```

or

```
ServerNode=111.2.3.44; Branch=XXX
```

Only certain drivers require a connection string:

- Mitsubishi FastLinx
- Unity Fastlinx (Static/Dynamic)
- OPC

Selecting one of these drivers and then browsing for the external database will automatically populate the Connection String field where necessary.

Add prefix to imported tags

Select this box to insert a prefix in front of the names of imported tags in your *Variable*.DBF.

Tag prefix

The prefix (8 characters max.) that is inserted in front of the names of imported tags in the variable tags database.

Delete I/O Device tags prior to import

Select this box to delete every one of the I/O Device's tags (from the variable tags database) before importing.

If you do not select this checkbox, tags found in CitectSCADA and in the external data source are updated in CitectSCADA. Tags found in CitectSCADA but not in the external data source remain untouched. New tags are appended.

Purge deleted tags not found in data source

Select this box to delete tags which have been removed from the external database. In other words, if a tag is still part of the I/O Device in your project, but not in the external database, it is deleted from your project.

FastLinx for Mitsubishi Tag Import

Use the CitectSCADA FastLinx tag browser to:

- Browse the available Mitsubishi servers on your network.
- Generate the connection strings and server parameters that the Fastlinx database driver requires in CitectSCADA.

The FastLinx tag browser has the following fields:

Database tree display

Shows a hierarchical display of the MxChange servers (both local and networked), their Fastlinx databases, and associated nodes. To expand or collapse a tree entry, click the add [+] or minus [-] symbol, respectively, to the left of the entry.

When you open the browser, the information displayed depends on whether or not there is an existing FastLinx database with the same name as the current CitectSCADA project:

- If an FastLinx database exists with the same name as the CitectSCADA project and the database contains a GID node with the same name as the CitectSCADA I/O Device, the browse dialog highlights the GID node of that name. You can then click **Open** to generate connection strings or server parameters.
- If no FastLinx database exists with the same name as the CitectSCADA project, the **Create a new database** node is highlighted. Click **Open** to generate the connection strings, or click the highlighted node to create a new database.

Server

Displays the server name on which the database, if present, exists. If no database is found, this box displays defaults to the name of the local PC. (Read-only)

Database

Displays the database name (i.e., name of the current CitectSCADA project). (Read-only)

IO Device

Displays the name of the GID node (i.e., CitectSCADA I/O Device). (Read-only)

PLC Class

Indicates the type of project configured in the MxChange server.

PLC Type

Indicates the PLC type corresponding to the selected PLC class.

Password

Enter the password for the selected database to allow CitectSCADA to access the database for data retrieval. The password needs to be valid for read/write access as assigned by your database administrator.

See Also

Defining Variable Tag Names for Mitsubishi FastLinx

Defining Variable Tag Names for Mitsubishi FastLinx

Note: The functionality described in this topic relates only to FastLinx for Mitsubishi.

> You need to have purchased an appropriate CitectSCADA license for this functionality to be supported.

If you are using Mitsubishi FastLinx, you cannot use certain reserved words when defining your tag name variables. You cannot use:

- Reserved words such as:

&	*	**	-
/	+	<	<=
>=	<>	=	>
ACTION	AND	ANY_NUM	ARRAY
BOOL	CASE	CONFIGURATION	CONSTANT
DATE	DATE	DINT	DO
DUT	DWORD	ELSE	EN
END_CASE	END_CONFIGURATION	END_FOR	END_IF
END_PROGRAM	END_VAR	END_WHILE	ENO
EXIT	FALSE	FOR	FUNCTION
IF	INT	INT	MOD
NOT	OF	OR	PROGRAM
REAL	REPEAT	STRING	STRING
TASK	TIME	TIME	TO
TRUE	TRUE	UNTIL	VAR_EXTERNAL
VAR_GLOBAL	VAR_IN_OUT	VAR_INPUT	VAR_OUTPUT
WHILE	WORD	XOR	

- Cicode function names. See Functions Reference for a complete list of Cicode functions.

Using reserved words or Cicode function names may cause a syntax error in Mitsubishi FastLinx when you export tags to the FastLinx tag name database.

See Also

FastLinx for Mitsubishi Tag Import

Unity Link Tag Import

The Unity Link Configuration dialog specifies the parameters used to generate alarm and trend tags. It has the following fields:

I/O Device & Protocol Driver
The name of the destination I/O Device and its corresponding protocol. Unity FastLinx will only import tags to an I/O Device using either the MODBUS, MODNET, MBPLUS, or UNITE.

Unity Database Type
The import/export database driver type selected from the Import Variable Tags dialog.

PLC Family
The family name of the PLC you are importing from. It can be either Premium or Quantum. This option is only valid for the Unity FastLinx Static protocol.

Unity File
Path of the Unity project file from which you want to import the tags. If the database type is Unity FastLinx Dynamic, this is a *.stu file. If the database type is Unity FastLinx Static, this is a *.xsy file.

Unity Profile Enable
Enable this if the Unity database is restricted by user profile and a user name and password combination are needed to gain access. See the Access Security Management section of the Unity Pro online help for further information.

Unity Username and Password
The username and password, if it is necessary, for the Unity database from which you are importing.

DCOM Enable
Enable this if the database host is a DCOM server.

DCOM Server Name
If enabled, the DCOM server that the client uses to hosts the database from which you are extracting tags.

DCOM Username and Password
If enabled, the username and password for the DCOM server.

Log File Path
The path name for log files generated during the import or export process.

Logging Level
The level of detail necessary in the import or export logs.

Log Pool Size
Specifies the maximum number of log files for the given device.

Validate
Click the validate button to check the Unity Link settings are correct. This button enables the OK button on the page.

OPC Data Access Server Tag Browser

The OPC Data Access Server Tag Browser dialog generates the strings that the OPC Data Access Server database driver requires in CitectSCADA.

The OPC Data Access Server Tag Browser dialog has the following fields:

Machine Name

The location of the OPC Server which can be an IP address or the network path server. Leave blank to select the local machine.

Database tree display

Displays the Servers on the network that are running the OPC Server application, and the hierarchical database node structure for each.

Select a group of tags to import from an available database. OPC data can be either a tree (hierarchical) or a flat structure. The CitectSCADA OPC driver supports access to both types of storage. If the data is in a tree structure, it can be accessed to the first branch level down from the root node. Deeper levels of branching may be supported in the future.

> **Note:** The authentication values needs to be valid for read/write access, as assigned by the appropriate database administrator.

Exporting tags

The Export feature allows you to export I/O Device data to an external data source, specifying the destination and format of your choice (for example RSLOGIX driver). This file might then be imported into a third-party I/O Device programming package database. Alternatively, you might use it as a backup.

> **Note:** Exporting using the OPC driver is not supported.

The file to which you are exporting needs to already exist; otherwise the export will not work. You can choose to delete its contents before exporting, or you can leave it and create duplicate tags.

> **Note:** The export tags feature is not yet supported by every database type. If you have existing links to any external data source, the linked tags will also be exported. As the structure of each type of external data source differs, some tag data might not be exported. This is determined by the format file for the I/O Device.

See Exported Tags for information about levels of support between Unity and Citect-SCADA.

> **Note:** Tag names greater than 32 Chars cannot be exported to Unity.

To export variable tags to an external database:

1. Open Citect Explorer.
2. Choose **Tools** | **Export Tags**.
3. Complete the **Export Variable Tags** dialog box as necessary.

See Also

Export Variable Tags properties
External data source
Format file

Export Variable Tags properties

Use this dialog for Exporting tags to an external database. The Export Variable Tags dialog has the following fields:

External database

A reference (128 characters max.) to the external data source to which your variable tags are exported. This can be:

- An explicit path and file (for example, `C:\Data\Tags.csv`)
- An IP address and node (for example, `127.0.0.1\HMI_Scada`)
- A URL (for example, `http://www.abicom.com.au/main/scada`)
- A computer name (for example, `\\coms\data\scada`)

> **Note:** This data source needs to exist before the export can be performed.

Database type

The format of the data referenced by the external data source.

Connection string

Enter a connection string to provide connection details for the data source. This is similar to an ODBC connection string. For example:

```
UserID = XXX; Password = YYY
```

Not every data source requires a connection string.

I/O Device

The I/O Device for which you are exporting tags. Use the menu to select an I/O Device that has been defined using CitectSCADA.

Remove prefix from tags

Select this box to remove a known prefix from the front of the exported tag names.

Tag prefix

The prefix (8 characters max.) to be removed from exported tag names.

Delete existing tags

Select this box to delete any tags in the external database before exporting.

> **Note:** For the Unity driver existing tags are not deleted, even if you select this check box.

External data source

When setting up an import, export, or link, you need to provide a data source and the format of the data.

Your data source can be entered as:

- An explicit path and file (for example, `C:\Data\Tags.csv`)
- An IP address and node (for example, `127.0.0.1\HMI_Scada`)
- A URL (for example, `http://www.abicom.com.au/main/scada`)
- A computer name (for example, `\\coms\data\scada`)

The database type field specifies the format of the external data source. When CitectSCADA attempts to read from this data source, it will use the mechanism specified by the database type. The supported database types are:

- OPC (OPC1 is not supported)
- CSV (comma-separated values)
- Concept Ver 2.1 ASCII file

- Mitsubishi FastLinx (a specialized driver which allows you to connect directly to the PLC programming software)

To configure the external data source as a file

The example uses a CSV file, in this case an RSLOGIX database driver

In the Import/Export or Links dialog box, enter details as follows:

- **External database** C:\Data\Tags.csv
- **Database type** RSLOGIX Driver
- **Connection string** (leave blank)

To configure the external data source using a specialized driver

> **Note:** This example uses the supplied OPC driver.

In the Import/Export or Links dialog, enter details as follows:

- **External database**: The name of the OPC server process, for example, FactorySoft.InProc
- **Database type**: OPC
- **Connection string**: The parameters are **ServerNode** or **Branch**, though both are optional. Their use depends on the location of the OPC server and the scope of the necessary data. **ServerNode** can be an IP address or the network path to the server. For example:
  - ServerNode=127.0.0.1
  - ServerNode=\\Server
  - ServerNode=www.server.com

  For **Branch**, OPC data can be either a tree (hierarchical) or a flat structure. The OPC driver supports access to both types of storage. If the data is in a tree structure, it can be accessed to the first branch level down from the root node, by entering the name of the branch. For example:
  - Branch=device1

  Deeper levels of branching might be supported in the future.

  > **Note:** This example uses the supplied Mitsubishi Driver

In the Import/Export or Links dialog box, enter details as follows:

- **External database**: 127.0.0.1\HMI_Scada (you can also use the computer name instead of the IP address)
- **Database type**: Mitsubishi FastLinx
- **Connection string**: UserID=Citect;Password=Citect

The Connection String needs to be in the format which the specialized driver (in this case Mitsubishi FastLinx) expects; otherwise it might be ignored.

> **Note:** Tags in your external data source need to conform to the CitectSCADA standard. Tag names that are longer than 79 characters are truncated. If this truncation results in duplicate tags, you are informed when you compile your project. Characters other than (a to z, A to Z, and 0 to 9) and the underscore character "_", are removed on import/link (and before any truncation).

See Also

Format file

Format file

The format file defines the import/export/linking rules. The file maps columns from the external data source format to the internal CitectSCADA database format. In other words, it determines what information is imported/exported/linked and how this information is modified during the operation.

The format file also provides the information to allow CitectSCADA to use the correct driver for accessing the external data source.

Some format files are provided with CitectSCADA; however, sometimes you might need to write or modify a format file using an editor such as Microsoft Notepad.

Following is an example of how format file rules work. (For more information, see Format file layout.)

1. Column 3 in the external data source needs to be copied into the "Name" column in the CitectSCADA's tag database. (CitectSCADA names are restricted to alphanumeric characters (a to z, A to Z, and 0 to 9) and the underscore character "_", but this is not typically a consideration in the format file; CitectSCADA does the conversion automatically). However, if Column 3 in the external data source happens to be blank, there is no need to copy this record across (it needs to be rejected).
2. Column 1 in the external data source needs to be copied straight into the "Addr" column in CitectSCADA's tag database, because they both mean exactly the same thing.
3. Columns 4, 5, 6, and 7 in the external data source need to be copied into the "Comment" column in Variable.DBF. (It is not uncommon for external data sources to split the comments across several fields). The fields need to be copied in that order, so that if the data in Column 4 in the external data source is "**Loop**", Column 5 is "**1**", Column 6 is "**Process**", and Column 7 is "**variable**", these fields are copied across in order, so that the "Comment" column in Variable.DBF reads "**Loop 1 Process variable**". This process is called 'concatenation'. (For the "Comment" column, CitectSCADA automatically adds a space between each field from the external data source.)

4. The data in Column 1 in the external data source determines what CitectSCADA needs to write in the "Type" column. However the data cannot be copied across directly, because it would not make sense to CitectSCADA. Instead, it needs to go through a conversion (or filtering) process. This conversion needs its own set of rules, such as:
5. If Column 1 in the external data source is "BT%d:%d.%d" (where%d means "any decimal number"), CitectSCADA needs to write the string "DIGITAL" in the "Type" column.
6. If Column 1 in the external data source is "F%d:%d/%d" (where %d means "any decimal number"), CitectSCADA needs to write the string "DIGITAL" in the "Type" column.
7. If Column 1 in the external data source is "O:%e" (where %e means "any octal number"; that is, every digit from 0 to 7), CitectSCADA needs to leave the "Type" column blank. It still accepts the record (provided other columns pass any filtering tests) but it does not write anything in the "Type" column. The assumption is that CitectSCADA currently does not have (or does not need) a suitable corresponding type.
8. If Column 1 in the external data source is "PD%d:%d.%d", CitectSCADA needs to write the string "REAL" in the "Type" column.
9. If Column 1 in the external data source is "ST%d:%d", CitectSCADA needs to write the string "STRING" in the "Type" column.
10. If there are no rules covering the contents of Column 1 in the external data source, CitectSCADA needs to reject the whole record and not copy it into the CitectSCADA database.

See Also

Format file layout

Format file layout

The format file is divided into sections:

- [General] section
- [Columns] section
- [ImportFilterMap] and [ExportFilterMap] sections

Each section consists of a section header (the section name enclosed in square brackets (for example "[my_section]")) on a line by itself. This is followed by the body of the section, typically single line statements of the form:

```
"something = something_else -> something_else_again"
```

Any white space (or none at all) is acceptable around the "=" and the "->", but the whole statement needs to be on one line. Most statements within a format file follow this pattern, but in many cases there might be no "->" (the converter), or there might just be a converter without anything following it.

The following sections are necessary in every format file:

```
[General]
[Columns]
[ImportFilterMap]
[ExportFilterMap]
```

Other sections might be necessary depending on the complexity of the conversion between CitectSCADA and the external data source. This is determined by the contents of `[ImportFilterMap]` and `[ExportFilterMap]`.

Comments can also be added within or between sections. To do this, place a semicolon ";" as the first character on the line. The rest of the line is then considered a comment, and is ignored by CitectSCADA. For example:

```
; I am putting the [General] section here
[General]
```

[General] section

The General section consists of 4 lines:

```
[General]
Name=name
Description= description
DriverName= driver name
DriverInst="a special string"
```

The *name* and the *description* are not currently used by CitectSCADA. CitectSCADA uses the *driver name* to load the correct driver for accessing the external data source. This driver might be one that is part of the CitectSCADA installation, or it might be a customized driver (including a driver that you have written yourself), for accessing a particular data source (which could be a protocol, type of hardware, server or file type). The driver needs to be an OLE DB-compliant driver.

The *special string* allows extra information to be passed to the driver. It is added to the connection string (in Citect Explorer and the Project Editor). So the connection string can be used for information that is likely to change often, and this *special string* can be used for more persistent information (such as the comma "," delimiter for a .CSV file). The main use of this string is as a delimiter for an input file. To specify that a comma "," is used by an input file as the delimiter, the following syntax would be used:

```
DriverInst="delimiter=,"
```

[Columns] section

The Columns section defines the format of the columns in the external data source. It is structured as follows:

```
[Columns]
 External column name 1 = column width -> data type
 External column name 2 = column width -> data type
..
 External column name n = column width -> data type
```

The only restriction that CitectSCADA places on the data for *External Column name n* is that it needs to be unique within the section. For convenience, you can use the names that the external data source uses (such as "Description", "PLC_id", "Iotype") or you can just make up names like "Column1", "Column2", etc. The order in which these entries appear needs to be the same as the order of the fields in the external data source. The names used for the External Data Source columns in the [ImportFilterMap] and [ExportFilterMap] sections needs to come from this list.

Column width is the number of characters in the field, and *data type* is the type of data for that column. Currently the only acceptable data type is "STRING".

[ImportFilterMap] and [ExportFilterMap] sections

The `[ImportFilterMap]` and `[ExportFilterMap]` sections have identical syntax and functionality, except that the `[ImportFilterMap]` describes how to convert data from the External data source on import, while the `[ExportFilterMap]` describes the opposite conversion.

These are the most complex sections in the format file. (The rest of the text will just focus on the [ImportFilterMap], as the [ExportFilterMap] follows basically the same logic.)

The `[ImportFilterMap]` is structured as follows:

```
[ImportFilterMap]
 Import Rule 1 = External column name l -> Citect Column i
 Import Rule 2 = External column name m -> Citect Column j
..
Import Rule nn = External column name n -> Citect Column k
```

The values in *Import Rule nn* can be any name strings, but they needs to be unique within the section. Therefore, for convenience, you might want to use names like "ImportRule1", "ImportRule2", "Mapping1", "Filter1" etc., or you might want something that is descriptive of the conversion involved, such as "Description_to_comment".

The name used for *External column name n* needs to be identical to a name that appears in *External column name n* in the [Columns] section above.

The name used for *Citect Column k* needs to be the same as one of the columns in the CitectSCADA internal tags database, such as "Name", "Type", "Addr", "Comment" etc.

Thus the values for the external column and the CitectSCADA column provide information on how to transfer data from the external column to the Citect column during import.

For example:

```
ImportRule1 = Description -> Comment
```

This indicates that there is a relationship between the data in the "Description" field in the external data source and the data that needs to go into the "Comment" field in the CitectSCADA database.

The name that you use for *Import Rule nn* might be the same as the name of another optional section in the format file: here, the extra section provides CitectSCADA with more information. In the simplest case, if there is no section with that name in the format file, the rule simply states that the data in *External column name n* is to be copied directly into *Citect Column k* without modification or filtering.

So if the "Description" column in the external database contains "Truck Position 1" and the above entry appears in the `[ImportFilterMap]` section, and there is no section called `[ImportRule1]`, then after the import, the "Comment" column in the CitectSCADA database will contain the string "Truck Position 1".

Concatenation

To concatenate fields from the external database into one field in the CitectSCADA database, add separate entries to the `[ImportFilterMap]` section. Each section needs to contain the name of a relevant external column and the name of the destination column in CitectSCADA. The entries needs to appear in the order in which the fields are to be concatenated.

So, if the external data source has a field called "IOdev" containing the value "M", and another field called "IOaddr" containing the value "61", and you want to join them together so that the value "M61" is imported into the CitectSCADA "Addr" field, this is how it would be done:

```
[ImportFilterMap]
Addr1= IOdev -> Addr
Addr2= IOaddr -> Addr
```

Here, you need to verify that there are no sections in the format file called [Addr1] or [Addr2], unless you need some filtering or conversion.

See Also

Field conversion

Field conversion

To modify mapped data or to apply filtering (to reject certain records), create a new section using the name of the relevant line from the mapping section. For example, if you have the following mapping section:

```
[ImportFilterMap]
Test1_to_type = Test1 -> Type
```

and you want to convert the data from the Test1 column before importing it, somewhere in the file you need to have a section called `[Test1_to_type]`:

```
Test1_to_type]
```

followed by the necessary conversion rule.

> **Note:** The name needs to be from the import mapping section, not from the export mapping section. If you use a name from the export mapping section, the conversion applies to the export, not the import.

The basic format of this conversion/filtering section is as follows:

```
[Relevant mapping name]
Filtering Rule 1 = External Pattern 1 -> Citect String 1
..
Filtering Rule m = External Pattern m
..
Filtering Rule n = External Pattern n -> Citect String n
```

The name used for *Filtering Rule n* has no intrinsic significance to CitectSCADA (except that it uses it as a key to locate the entry). The only restriction is that it needs to be unique within the section, so you can use whatever is convenient.

The value in *External Pattern n* is a combination of characters which CitectSCADA will look for in the external data source column. This pattern can be any combination of the following:

Character in format file	Matches what string in external data source
<specific text>	<specific text>
*	Any string.
?	Any single character.
%d	Any decimal integer (nnn. . . where n is 0-9).
%e	Any octal number (0onnn. . . where n is 0-7).
%h	Any hexadecimal number (0xnnn. . . where n is 0-9, A-F or a-f).
%s	Any string.
{	Begin a "token string". Any characters enclosed by { } in the Input Pattern (including regular and special characters) represent a token string. The characters in the data stream that match a token string are referenced by the Output Data String and written directly to the output database as a group.
}	End of a token string.
\	Treat the following character as a literal. For example, if a literal * character was expected in the input data stream, you would use * to denote this. If a literal backslash \ is expected, use \\.

Any other characters need to literally match the same character.

If `External Pattern n` is found in the external column, `Citect String n` is written to the relevant column in CitectSCADA (as per the mapping).

In addition, two special characters can appear in the output data string:

Character in output string	Meaning
$	The pattern $*n* (where *n* is any integer) is replaced in the output data stream by the *n*th "token"; a token is a matching sequence of characters enclosed by { } in the input pattern. (An alert message will result if $ not followed by a token number.)

!REJECT!	This sequence needs to appear by itself in the output data pattern. The whole record is rejected. As the record had already been matched to the input pattern, no further rules are checked.
\	Treat the following character as a literal. This would be used if a literal $ sign was necessary (use \$) or if another digit immediately follows. For example, if the string "3August2001" needs to immediately follow the token, use "$1\3August2001".
**** (at end of line)	Insert a literal space ` ' character at the end of the output line. Without this provision, the system could not distinguish between the end of the input line (which is likely to be followed by characters, such as spaces, that Windows will ignore) and a space being necessary at the end of the output line.

Other characters are written literally to the output database.

CitectSCADA works through each filtering rule in the section, looking for a match. If a rule does not match, the next one is tried, then the next, and so on, until a match is found. If no match is found, the whole record is rejected; none of the data from any field is copied to CitectSCADA.

For example, to convert the string "FLOAT" in the external data source to "DIGITAL" in CitectSCADA, you could use the following entry:

```
[ImportFilterMap]
Test1_to_type = Test1 -> Type
..
[Test1_to_type]
Rule1 = FLOAT -> DIGITAL
..
```

For a more complex example, let us assume that the external data source has a column called "Tag" which is equivalent to the "Name" field in CitectSCADA. Let us also assume that the external database has no direct equivalent of CitectSCADA's "Type" field, yet CitectSCADA needs this field to be filled in. We need to use the "Tag" field to decide what goes into the "Type" field of the CitectSCADA database.

If the "Tag" column in the external data source has the value "I:060/07", we have determined that we write the string "DIGITAL" into CitectSCADA's "Type" field. In fact, if that field has "I:" followed by any octal value, followed by a slash "/", followed by any octal value, we want the string "DIGITAL" to appear in our "Type" field. How do we express this in the format file?

Firstly, there are two sets of relationships to consider, one connecting the "Tag" field in the external data source to the "Name" field in CitectSCADA, and the other connecting it to the "Type" field in CitectSCADA. So we need two "mappings" (entries) in the [ImportFilterMap] section:

```
[ImportFilterMap]
Tag_to_Name = Tag -> Name
Tag_to_Type = Tag -> Type
..
```

As we want the data in the "Tag" field to be copied directly into the "Name" field, we do this by not having a `[Tag_to_Name]` section anywhere in the format file.

But because we are not copying directly from the "Tag" field to the "Type" field, but are just using the data to decide what goes into the "Type" field, we need a `[Tag_to_Type]` section.

Recall the desired outcome: If the "tag" field has "I:" followed by any octal value, followed by a slash "/", followed by any octal value, we want the string "DIGITAL" to appear in our "Type" field.

We express this in the format file as follows:

```
[Tag_to_Type]
Rule1 = I:%e/%e -> DIGITAL
..
```

This will match "I:060/07" or "I:0453/02343445602" (and cause the string "DIGITAL" to be written to CitectSCADA's Type field), but will not match "I:060/98" or "I:054".

To give a few examples of how the wild-card characters (%s, * and ?) might be used, the pattern "HE%sLD" or "HE*LD" in the format file would match "HELLO WORLD" or "HE IS VERY BOLD" in the external data source. The pattern "HE???????LD" would match "HELLO WORLD" but not "HE IS BALD", as each question mark "?" needs to match exactly one character.

CitectSCADA will also handle multiple wildcard patterns, such as "%s/%s:%s".

For an example more useful than "Hello World", imagine that we need to copy the data straight across without modification, but we want to verify that no blank fields are copied across. The pattern "?%s" or "?*" will match any string that has at least one character, but will not match a blank.

Sometimes only part of the input stream is necessary in the output, or the input might need to be split up into different output columns. In these situations "tokens" are useful.

In this example of an export situation, the "Addr" field in the CitectSCADA database needs to be split among two fields in the external database: the "IOdev" (whose value is always "D" or "M"); and "IOaddr" (whose value is a decimal integer of no more than 3 digits). Values in the "Addr" field of the CitectSCADA database are strings such as "D62", "M546", etc.

This situation could be solved by concatenation, i.e. using one mapping to write to the IOdev field, and three other separate mappings to copy each digit separately into the IOaddr field of the external database. But this would be complex and in some situations would not work.

It is better to use a token to address the situation:

```
[ExportFilterMap]
..
Addr2IOdev = Addr -> IOdev
Addr2IOaddr = Addr -> IOaddr
..
[Addr2IOdev]
D = D* -> D
M = M* -> M
AnythingElse = * ->
..
[Addr2IOaddr]
Rule = ?{%d} -> $1
```

In the [Addr2IOaddr] section, the {%d} is the token string, and as it is the first (and only) token appearing in the rule, $1 is used to reference it on the output stream side. So if the "Addr" field of the CitectSCADA database contains "D483", "D" is written to the "IOdev" field of the external data sink, and "483" (the token) to the "IOaddr" field.

Here is another example illustrating the use of multiple tokens. Suppose we need to: convert period characters (.) to colons (:); remove the first two characters (which are blank); and remove any unnecessary characters from the data we are expecting; that is, convert "..BJ6452.78......" to "BJ6542:78". This can be achieved by using the following rule:

```
Rule = ??{*%d}.{%d}* -> $1:$2
```

At this point, we introduce another feature of the format file. If you use the following rule:

```
[Relevant mapping name]
Filtering Rule = External pattern
```

i.e., without "-> Citect String" included, CitectSCADA interprets this as "check that the string matches the External Pattern, if it does, copy it across unchanged".

If this rule is used:

```
Filtering Rule = External pattern ->
```

i.e., without "Citect String", it would mean: "If the string pattern matches then accept the record but copy a NULL string to the CitectSCADA database."

Using the above example again, we can add the restriction that any records with no data (i.e. a blank or NULL string) in the Tag field of the external data source will not be imported into CitectSCADA. We would add a [Tag_to_Name] section, and would have just one rule: that we accept everything except for a blank.

```
[Tag_to_Name]
RejectBlanks = ?*
..
```

Recall that CitectSCADA checks the pattern in each filter rule sequentially until a pattern that matches the string is found in the external data source. With this in mind, a huge range of conversions and filterings are possible by ordering the rules correctly and, in some cases, by making use of concatenation.

For instance, if certain string types need to be converted but others need to be copied unmodified to CitectSCADA, you could have a section with a set of rules at the top, followed by a final rule to let everything else through unmodified.

```
[Tag_to_Name]
Rule n = ...... -> ......
..
LetEverythingElseThru = %s
```

A single %s or *, without anything else, matches anything and everything, including blanks.

For an example of how to reject a particular string or pattern, let's suppose we want to reject any tags starting with "DFILE"(another real-life example). We would simply use the following:

```
[Tag_to_Name]
Rule1 = DFILE* -> !REJECT!
..
LetEverythingElseThru = %s
```

Clearly, it is pointless having the !REJECT! rule not followed by other rules concerning patterns that you do want to accept, as anything that does not match an input pattern is rejected. The logic behind the order that the rules appear can become particularly important when using a !REJECT! rule. You would typically have any reject rule(s) as the first rule(s) in the mapping. There would not be any point in putting a !REJECT! rule as the last rule in the mapping.

!REJECT! rules can also be useful where some text file generated by another system contains some sort of header lines that are not wanted, but the rest of the data is necessary.

See Also

Having CitectSCADA recognize format files

Recognizing format files

Your format file needs to be saved to the config folder of the CitectSCADA User and Data folder selected during installation.

For CitectSCADA to import, export, or link, it needs to know the name of the format file and the name of the driver that will access the external data source. It also needs the text of the string that will appear in the **Database type** field of the Import or Export dialog box. This information is stored in another file, called `tagdriv.ini`, in the same directory.

The format of `tagdriv.ini` is simple and based on the `odbc.ini` format. When it is installed it already has the necessary information for the format files and drivers that come shipped with CitectSCADA. You just need to copy the same layout for your new format file, and the driver that you are using.

`The tagdriv.ini` file has several sections. The first section is the `[External data sources]` section, with the following general layout:

```
[External data sources]
Section name 1 = the name you want to appear in the import/export/ link menu for
entry 1
Section name 2 = the name you want to appear in the import/export/ link menu for
entry 2
..
Section name nn = the name you want to appear in the import/export menu for entry nn
```

Each entry in this section refers to a combination of format file and driver necessary for a particular import, export, or link operation.

The text on the left of the "=" sign needs to refer to the name of another section which needs to appear in `tagdriv.ini`.

The text on the right of the "=" sign is exactly what will appear in the menu under "**Database type**" for importing, exporting or refreshing variable tags.

The other sections each refer to a type of import or export described in the `[External data sources]` section, and give details about the format file and driver pair. The general layout of these sections is as follows:

```
[Section name matching an entry in [External data sources] ]
driverid = Driver ID
datastring = The name of the format file
Currently the Driver ID is always CiTrans.
```

So if we assume that the version of CitectSCADA you are installing contains 4 format files, there would be 4 sections in `tagdriv.ini`, as shown in the following example:

```
; This file contains the driver name, driver prog id, and format file mappings
; The format file needs to reside in the BIN directory

[External data sources]
CSV = CSV Driver
RSLOGIX = RSLOGIX Driver
Concept ver 2.1 Ascii = Concept Ver 2.1 ASCII file
MxChange = Mitsubishi FastLinx

[CSV]
driverid = CiTrans
datastring = csv.fmt

[RSLOGIX]
driverid = CiTrans
datastring = rslogic.fmt

[Concept ver 2.1 Ascii]
driverid = CiTrans
datastring = concept ver 2_1.fmt

[MxChange]
driverid = CiTrans
datastring = MxChange.fmt
```

This adds 4 entries to the database type drop down menu: CSV, RSLOGIX, ASCII and Mitsubishi FastLinx.

The 4 entries in the menu match the strings on the right of the "=" sign in the `[External data sources]` section.

If you add another format file, you will also need to add a matching entry to `tagdriv.ini`. For example, if you add a new format file for a Simatic data source, you need to add a line similar to this to the `[External data sources]` section:

```
SIMATIC = Siemens SIMATIC Driver
```

You need to also add the following section to the bottom of the file:

```
[SIMATIC]
driverid = CiTrans
datastring = SIMATIC.fmt
```

Save the file, restart Citect Explorer, and "Siemens SIMATIC Driver " appears in the menu.

Selecting this entry causes the format file in the datastring entry under the `[SIMATIC]` section to be used for the import, export, or link; that is, `simatic.fmt`.

Unity Support Matrixes

The following sections show the levels of support between Unity and CitectSCADA:

- Imported Tags
- Exported Tags
- Linked Tags

Imported Tags

For Unity Fastlinx Dynamic and Static:

Unity PLCs	Unity Data Types	Citect Protocols	Support Status
QUANTUM	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Not Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Supported (Except BYTE type)
		OPC	Not Supported

PREMIUM	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Supported (Except BYTE type)
		OPC	Not Supported
M340	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Not Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Not Supported
		OPC	Not Supported

Note: Citect recommends that you use the Unity Fastlinx Dynamic driver when importing tags from Unity Pro into a Citect project.

Note: Arrays are supported for every data type except DATE, TOD, DT and STRING data types. For Unity data types marked with *, Citect does not have data types that directly match the Unity types. Instead use supported data types and Cicode to convert and simulate those data types.

In addition, the following Unity data types are not supported:

- Structured data types.
- Arrays which are not zero-based.
- Multi-dimensional arrays.
- Arrays of Unity BOOL data type are not supported for all protocols and will be excluded from the import.

Exported Tags

For Fastlinx Static:

Unity PLCs	Unity Data Types	Citect Protocols	Support Status
QUANTUM	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Not Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Supported (Except BYTE type)
		OPC	Not Supported
PREMIUM	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Supported

Modnet	Supported (Except BYTE type)		
Modbus	Supported (Except BYTE type)		
MBPlus	Supported (Except BYTE type)		
OPC	Not Supported		
M340	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Not Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Not Supported
		OPC	Not Supported

Note: Arrays are supported for data types except DATE, TOD, DT and STRING data types. For Unity data types marked with *, Citect does not have data types that directly match the Unity types. Instead use Technical supported data types and Cicode to convert and simulate those data types.

In addition, the following Unity data types are not supported:

- Structured data types.
- Arrays which are not zero-based.
- Multi-dimensional arrays.
- Arrays of Unity BOOL data type are not supported for all protocols.

Linked Tags

Linked I/O Device Live Update

For Unity Fastlinx Dynamic:

Unity PLCs	Unity Data Types	Citect Protocols	Support Status
QUANTUM	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Not Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Supported (Except BYTE type)
		OPC	Not Supported
PREMIUM	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Supported (Except BYTE type)
		OPC	Not Supported
M340	BOOL EBOOL BYTE WORD DWORD INT DINT UINT UDINT REAL TIME * DATE * TOD * DT * STRING	Unite	Not Supported
		Modnet	Supported (Except BYTE type)
		Modbus	Supported (Except BYTE type)
		MBPlus	Not Supported
		OPC	Not Supported

Note: Arrays are supported for data types except DATE, TOD, DT and STRING data types. For Unity data types marked with *, Citect does not have data types that directly match the Unity types. Instead use supported data types and Cicode to convert and simulate those data types.

In addition, the following Unity data types are not supported:

- Structured data types.
- Arrays which are not zero-based.
- Multi-dimensional arrays.
- Arrays of Unity BOOL data type are not supported for all protocols.

Chapter: 18 Defining and Drawing Graphics Pages

A runtime system usually comprises a series of graphics pages that display on your computer screen(s) and provide a "window into the process." You can design your pages to provide your operators with control of an area (or all) of your plant. Your graphics pages can also display the status of your plant by using various graphical items known as objects.

See Also

Creating a New Graphics Page
Using Page Templates
Using a Browse Sequence
Specifying a Startup Page
Sizing the Page
Defining Page Properties
Setting Default Page Settings
Understanding the Drawing Environment
Using Find and Replace in a project

Creating a New Graphics Page

You can display your graphics pages on a monitor individually, or display several pages at a time. You can display them in any order, controlled by operator commands or controlled automatically.

To create a new page:

1. From the Graphics Builder, click **New**, or choose **File** | **New**.
2. Click **Page**.
3. Choose a **Template** upon which to base the page.
4. Choose a **Style** for the page.
5. Check or clear the **Linked** and **Title bar** as necessary.
6. Choose the **Resolution** for the page.
7. Click **OK**.

Note: If you create a new page using the Graphics Builder, you need to edit the page record (with the Project Editor) to define a browse sequence.

See Also

New Dialog Box

Animation points

Each point on a graphics page where an object is displayed is called an animation point. When you add an object (text, symbols, pipes, etc.) to your page, it is automatically allocated a number (termed an AN) to the animation point. ANs can be linked to static items (a bitmap or an object without an animation expression), or animated items (where there is an animation expression defined); however only those ANs that belong to an animated item are available for reference at runtime.

The number of objects that you can use is limited by the performance of your computer, though this is unlikely to be a consideration on modern computers. A good rule of thumb is to try and keep the number of objects (and hence ANs) less than 3000.

The first 2 ANs are used for automatically displaying system information such as messages, alarm information and page details. In some applications, such as trend pages, some other ANs are reserved.

Genies are a combined set of one or more graphic objects.Each graphic results in one or more ANs and animation records being saved to the project configuration. The total number of genies that can be used on a single page is determined by what graphic objects are defined within each genie, and the total number of ANs and animation records.

You can add individual animation points to a graphics page by using the **Cicode Object** tool (add a Cicode Object without a command).

> **Note:** You can locate any object on a page by referencing its AN. To locate an object by its AN use the **Goto Object** tool in the **Tools** menu.

New Dialog Box

This dialog box is used to create a page, template, symbol, Genie, or Super Genie by selecting a button.

Working with pages

To open an existing page:

1. Click **Open**, or choose **File | Open**.
2. Choose **Type: Page.**

3. Select the **Project** where the page is stored.
4. Select the **Page.**
5. Click **OK**.

> **Note:** To delete a page from the project, select the page name and click **Delete**.

To save the current page:

1. Click Save, or choose File | Save.
2. Select the **Project** in which to store the page. (The first eight characters of the name needs to be unique to this page.)
3. Click **OK**.

To save the current page with a new name:

1. Choose **File** | **Save As**.
2. Select the project in which the page is stored.
3. Enter a name for the page in **Page** (64 characters maximum). The first eight characters of the name needs to be unique to this page.
4. Click **OK**.

To save current pages that are open

- Choose **File** | **Save All.** (The first eight characters of each page name needs to be different.)

See Also

Use Template (new page/template) dialog box
Open/Save As dialog box

To locate a graphics page:

1. Choose **File** | **Find**.
2. Select a project from the **Project Name** list.
3. Browse through the pages in the **Pages** list.
4. Click **OK** to view the page.

See Also

Using Find and Replace in a project

To print a graphics page on your printer:

- Choose **File** | **Print**. (This prints without a confirmation dialog.)

To close an existing page:

- Choose **File** | **Close**.

Use Template (new page/template) dialog box

This dialog box lets you create a new page or template based on an existing template.

Template

A table of templates on which you can base the new page or template. Use the scroll bar to locate the thumbnail image of the template, then choose the template and click **OK** (or double-click the thumbnail image).

> **Note:** To edit the template, select it and click **Edit**

Style

The style of the page. CitectSCADA templates are grouped into several styles and are available in various page resolutions. When you create a new project, you can choose the style that most suits your taste and application. For details of each style, refer to the "Presenting CitectSCADA " booklet, supplied with your CitectSCADA system.

Linked

To maintain the link with the original template, select this check box. A page or template that is linked with its original template is automatically updated if the template is changed.

> **Note:** You can cut the link to the template at any time with the **Cut Link** command from the Edit menu, but you cannot re-link a page or template with its original template after the link has been cut.

Title bar

Determines whether to display the Windows title bar (at the top of each graphics page). The title bar contains the title of the window, maximize, minimize and close buttons (at the right hand end of the title bar), and the control menu button (at the left hand end of the title bar). To display a page in fullscreen (without a title bar), the size of the page needs to be the same size as the display (or larger). If the page is smaller than the display, the title bar still displays, even if fullscreen mode is enabled. Standard templates styles are available for both page sizes.

Resolution

The screen resolution of the page or template:

Screen Type	Width (pixels)	Height (pixels)
Default	Width of the screen on the computer currently in use	Height of the screen on the computer currently in use
VGA	640	480
SVGA	800	600
XGA	1024	768
SXGA	1280	1024
User	User-defined size	User-defined size

Open/Save As dialog box

This dialog box lets you open or save a page, template, symbol, Genie, or Super Genie. (To select the type of entity, click the appropriate tab.)

Page/Symbol/Template/Genie

The name of the graphics page, template, symbol, Genie, or Super Genie.

If you are opening a page, template, symbol, Genie, or Super Genie, select its name from the large window.

If you are saving a page, template, symbol, Genie, or Super Genie, type a name into the smaller input box (or select a name from the large window if you want to overwrite an existing page, template, symbol, Genie, or Super Genie).

> **Note:** If you are using distributed servers, the name needs to be unique to the cluster (for example you cannot have the same page name in more than one cluster).

Project

The project in which to save the graphics page, template, symbol, Genie, or Super Genie.

Library

(For symbols, Genies, and Super Genies only.) The library in which to save the symbol, Genie, or Super Genie. To create a new library, click **New**.

Style

(For templates only.) The style of the template. To create a new style, click **New**.

Preview Enable

Displays a thumbnail image of the page, template, symbol, Genie, or Super Genie.

Title bar

(For templates only.) Specifies whether to include a space for the title bar. If you use a title bar, you will have slightly less display space on screen.

Resolution

(For templates only.) The screen resolution of the template:

Screen Type	Width (pixels)	Height (pixels)
Default	Width of the screen on the computer currently in use	Height of the screen on the computer currently in use
VGA	640	480
SVGA	800	600
XGA	1024	768
SXGA	1280	1024
User	User-defined size	User-defined size

To delete a page, template, symbol, Genie, or Super Genie, select it and click **Delete**.

Using Page Templates

You can use many different page types in your runtime system.

- Mimic pages display the status of the plant, with buttons and other controls to give your operators control of processes within the plant.
- Alarm pages display alarm information.
- Trend pages display a visual representation of past and current activity in the plant.

> **Note:** You need to create default pages for your alarms (including alarm summary and hardware alarms pages).

To enable you to create your graphics pages quickly, there are templates for common page types. Templates help you create project pages that have a consistent 'look-and-feel'. (Consistency in your project reduces the time your operators need to learn how to use your runtime system.)

See Also
Choosing a page style
Linking templates
Creating your own templates

Choosing a page style

Templates are grouped into several styles and are available in various page resolutions. When you create a new project, you can choose the style that most suits your taste and application.

See Also
Linking templates

Linking templates

When using a template to create a new page, a link can be kept to the template. A page (or template) that is linked with its original template is automatically updated if the template is changed.

When a page is linked to a template, the objects that form the template cannot be accessed from the page by the usual double-click. To display the properties of these objects, hold the **Control** (CTRL) key down and double-click the object you want to view properties for. Alternatively, choose **Tools** | **Goto Object**, select the group, and click **OK**. However, most of these properties are read-only.

> **Note:** You can cut the link to the template at any time using **Edit** | **Cut Link**, but you cannot re-link a page or template with its original template after the link has been cut.

See Also
Creating your own templates

Creating your own templates

If your project contains several pages that are similar (for example, menu pages, common processes, or common equipment), you can create your own template (containing common objects) to use as a base for the pages. You can then create the pages based on the template, and add individual objects to each page.

If you want to delete or change the location of a common object, or to add a new common object, you do not have to change each page; instead, you can change the template. Pages based on that template are automatically updated.

Note: When you create a template, save it in the project directory. It is then backed up when you back up the project. Don't modify the supplied standard templates. When you edit a template, you need to use the **Update Pages** command (from the **Tools** menu) to update each page based on the template. The properties of the template are not updated automatically.

To create a new template:

1. Click **New**, or choose **File** | **New**.
2. Click **Template.**
3. Choose a **Template** upon which to base the template.
4. Choose a **Style** for the template.
5. Check or clear the **Linked** and **Title bar** as necessary.
6. Choose the **Resolution** for the template.
7. Click **OK**.

To open an existing template:

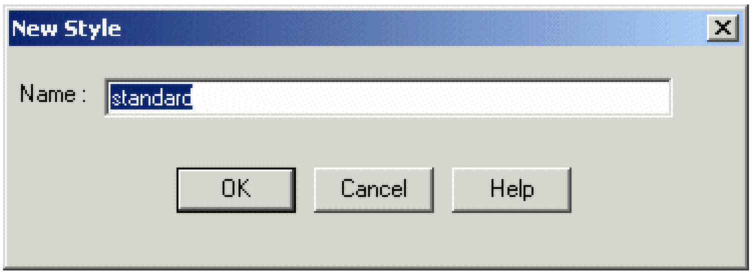
1. Click **Open**, or choose **File** | **Open**.
2. Select **Type: Template.**
3. Select the **Project** where the template is stored.
4. Select the **Template.**
5. Click **OK**.

Note: To delete a template from the project, select the template name and click **Delete**.

To save the current template:

1. Click **Save**, or choose **File** | **Save**.
2. Select the **Project** in which to store the template.
3. Click **OK**.

Note: To create a new style for the template, click **New**. You create a new template



standard

See Also

New Style dialog box

New Style dialog box

This dialog box lets you create a new style of templates. Enter a name for your new style in the **Name** text box.

See Also

Creating your own templates

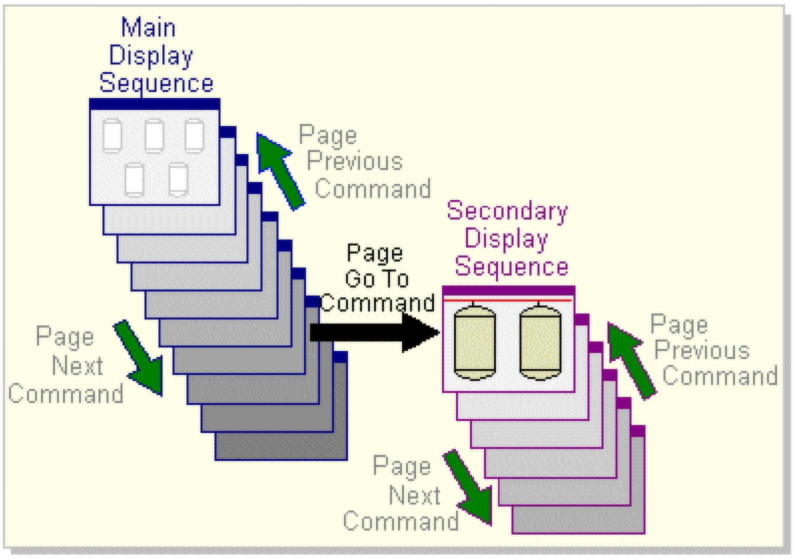
Using a Browse Sequence

You can link related pages together with a browse sequence. A browse sequence creates a linear navigation sequence for the pages in your system.

When you define a graphics page, you can specify where in the browse sequence the page displays. Within a browse sequence, an operator can display a preceding or following page by choosing **Page Previous** and **Page Next** commands (or a similar set of buttons defined on each page).

When you save a page for the first time it is automatically added to the browse sequence.

You can also use multiple browse sequences by defining a Page GoTo command that displays a page in another (secondary) sequence. The Page Next and Page Previous commands then display the next and previous pages in the secondary sequence, as in the following diagram:



You do not have to use a display sequence. You can define several Page GoTo commands that display specific pages in an hierarchical structure.

See Also

Specifying a Startup Page

Specifying a Startup or Splash Page

Every system needs to have a startup page and may have a splash screen.

A splash screen is the very first window that is displayed when CitectSCADA starts. The splash window has a thin windows border, is non-movable and is often set to "always on top". Usually, the splash window displays brief project information, such as company logo, software version, project version, while the system is initializing. By default, it is only displayed for a brief period of time and then automatically dismissed.

The startup page is displayed before or after the splash window is dismissed. The timing is controlled by a set of parameters:

- [Page] SplashTimeout# - how long the splash window will be displayed
- [Page] StartupDelay# - how long the Startup page will wait before it is displayed.

If the StartupDelay is set to less than the SplashTimeout, the Startup page will be displayed while the splash screen is still shown. On the other hand, setting the StartupDelay to be greater than the SplashTimeout will display the Startup page after the splash screen is dismissed.

The example project makes use of the new splash screen parameters to specify a page to be displayed as splash screen, and the new Cicode functions to display product and project information on the splash screen.

The following parameters are used to set up the splash screen in the example project:

```
[Page]
 Splash = !Splash
 SplashWinName = Splash
 SplashTimeout = 5000
 Startup = Menu
 StartupWinName = Main
 StartupDelay = 5100
 HomePage = Menu
```

See Also

Sizing the Page

Sizing the Page

By default, new pages in the Graphics Builder take up your entire display area. You can resize them if you want. You can:

- Specify the size of a page when you create it.
- Change the size of a page at any time after it is created.
- Specify that new pages default to a different size.

See Also

Page (screen) resolution

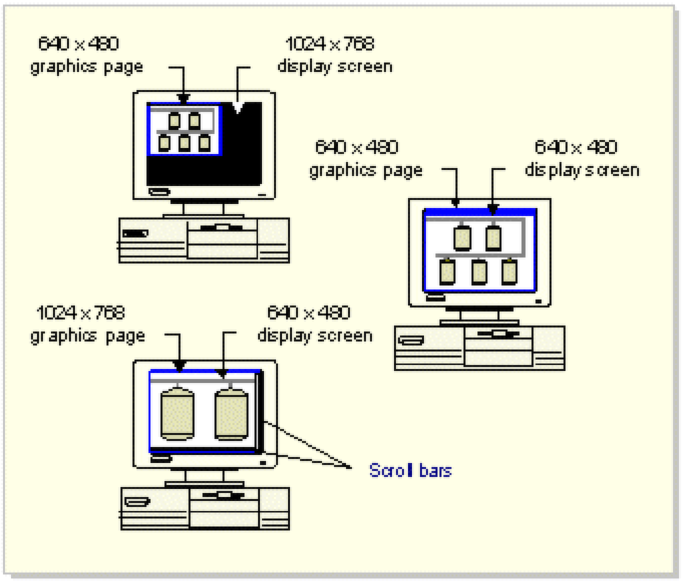
Page size at runtime

Page (screen) resolution

When you draw a graphics page, determine the resolution of the computer you are using to draw the page, and the resolution of the screen that will display the page in your run-time system.

If a page displays on a screen with a resolution which is greater than the page's resolution, the page will be smaller than the display area. For example, if you draw a page on a VGA screen (640 x 480) and then display it on a XGA screen (1024 x 768), the image displays in the top left corner of the screen, and occupies a little more than half of the screen.

Conversely, if a page displays on a screen with a resolution which is lower than the page's resolution, the page will be larger than the display area. For example, if you draw a page on a XGA screen and then display it on a VGA screen, it occupies more than the entire screen; use the scroll bars to scroll to the area of the page that is not displayed.



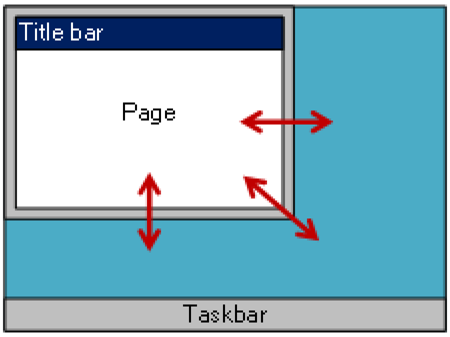
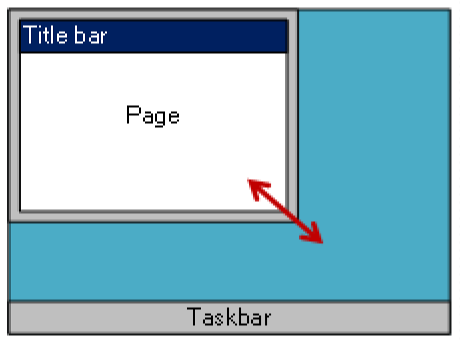
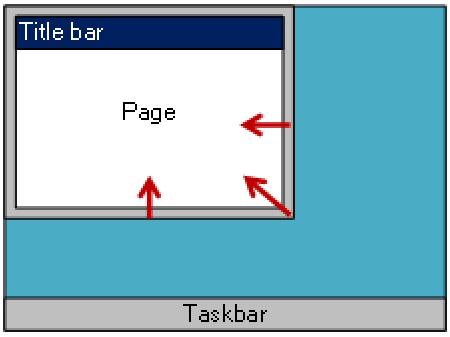
See Screen examples for some examples of parameter settings and how the screen is displayed on screen.

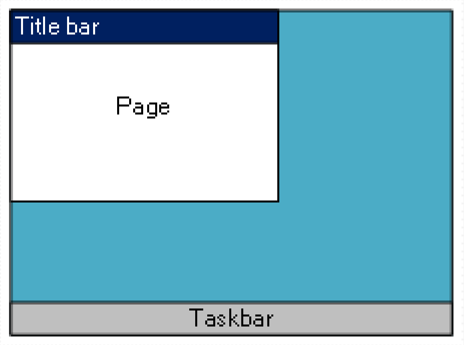
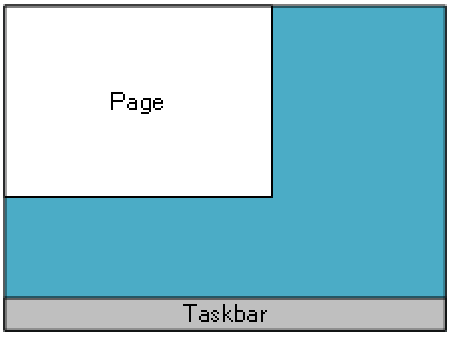
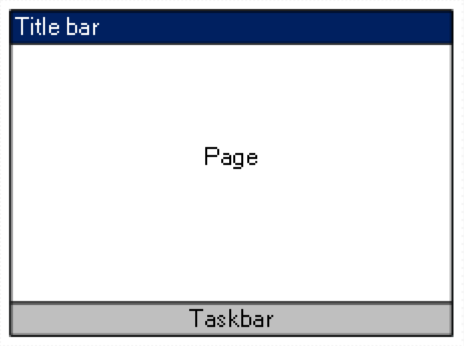
See Also

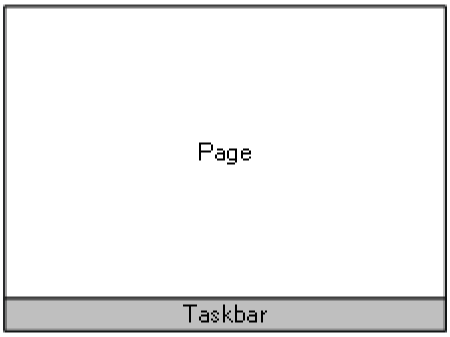
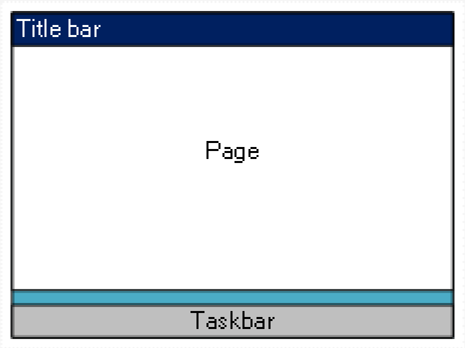
Page size at runtime

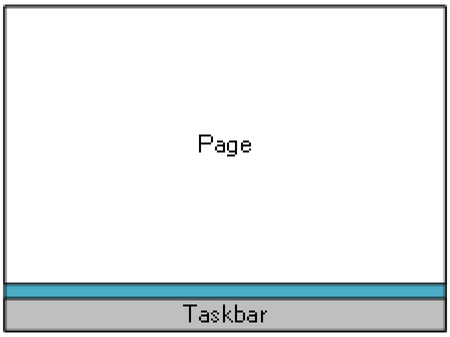
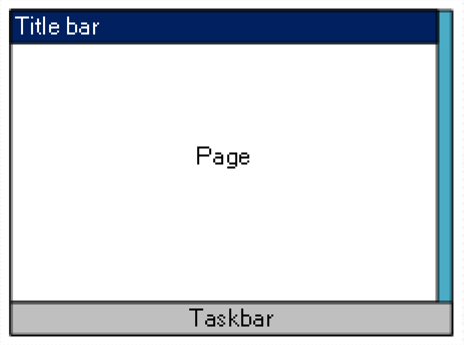
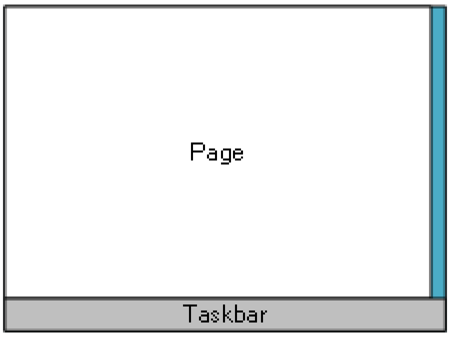
Screen examples

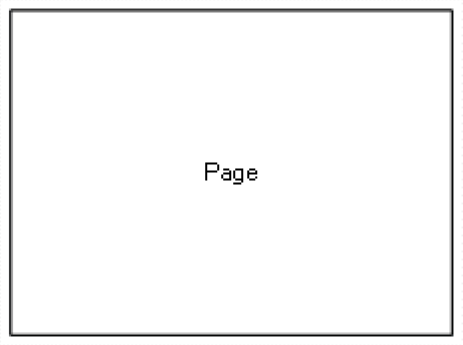
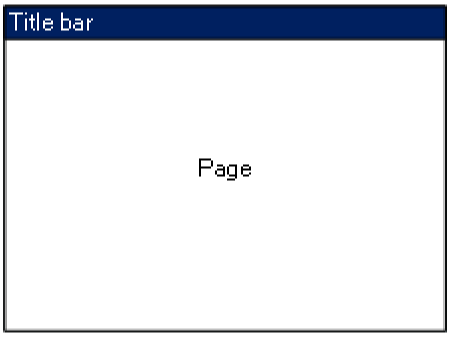
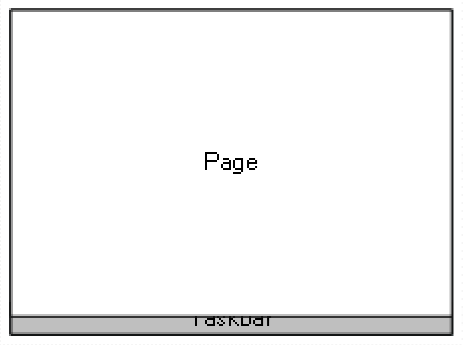
When you draw a graphics page, determine the resolution of the computer you are using to draw the page, and the resolution of the screen that will display the page in your runtime system.

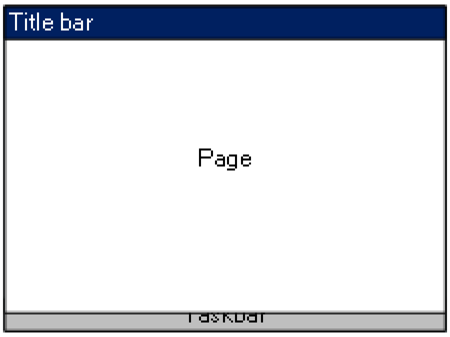
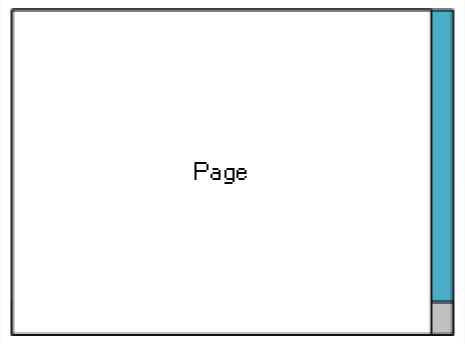
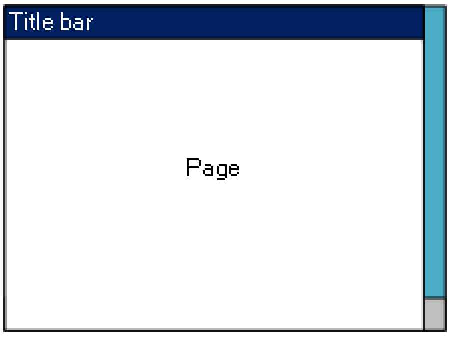
Example	Description	Notes
	[Animator]FullScreen = 0 [Animator]MaximizedWindow = 0 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 0	
	[Animator]FullScreen = 0 [Animator]MaximizedWindow = 0 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1	
	[Animator]FullScreen = 0 [Animator]MaximizedWindow = 0 [Page]DynamicSizing = 0	Page content is not scaled. Scrollbars are enabled if the window is resized and no longer large enough to show the entire page.

Example	Description	Notes
	[Animator]FullScreen = 0 [Animator]MaximizedWindow = 0 [Page]StartupMode = 4	
	[Animator]FullScreen = 0 [Animator]MaximizedWindow = 0 [Page]StartupMode = 16	
	[Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 0 or Aspect ratio = monitor - titlebar - taskbar [Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1 or Page size = monitor - titlebar - taskbar [Animator]FullScreen = 0 [Page]DynamicSizing = 0	"Page size = monitor - titlebar - taskbar" means the aspect ratio of the page is the same as that of the screen area of a monitor after taking out the total area of the titlebar and the taskbar.

Example	Description	Notes
	[Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 0 [Page]StartupMode = 16 or Aspect ratio = monitor - taskbar [Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1 [Page]StartupMode = 16 or Page size = monitor - taskbar [Animator]FullScreen = 0 [Page]DynamicSizing = 0 [Page]StartupMode = 16	
	Aspect ratio <> monitor - titlebar - taskbar [Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1	

Example	Description	Notes
	Aspect ratio <> monitor - taskbar [Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1 [Page]StartupMode = 16	
	Aspect ratio <> monitor - titlebar - taskbar [Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1	
	Aspect ratio <> monitor - taskbar [Animator]FullScreen = 0 [Animator]MaximizedWindow = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1 [Page]StartupMode = 16	

Example	Description	Notes
	[Animator]FullScreen = 1 [Page]DynamicSizing=1 [Page]MaintainAspectRatio = 0 or Aspect ratio = monitor [Animator]FullScreen = 1 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1 or Page size = monitor [Animator]FullScreen = 1 [Page]DynamicSizing = 0	
	[Animator]FullScreen = 2 [Page]DynamicSizing=1 [Page]MaintainAspectRatio = 0 or Aspect ratio = monitor - titlebar [Animator]FullScreen = 2 [Page]DynamicSizing = 1 [Page]MaintainAspectRatio = 1 or Page size = monitor - title-bar [Animator]FullScreen = 2 [Page]DynamicSizing = 0	The window cannot be moved.
	Aspect ratio <> monitor [Animator]FullScreen = 1 [Page]DynamicSizing=1 [Page]MaintainAspectRatio = 1 or Page size <> monitor [Animator]FullScreen = 1 [Page]DynamicSizing=0	

Example	Description	Notes
	Aspect ratio <> monitor - titlebar [Animator]FullScreen = 1 [Page]DynamicSizing=1 [Page]MaintainAspectRatio = 1 or Page size <> monitor - titlebar [Animator]FullScreen = 1 [Page]DynamicSizing=0	The window cannot be moved.
	Aspect ratio <> monitor [Animator]FullScreen = 1 [Page]DynamicSizing=1 [Page]MaintainAspectRatio = 1 or Page size <> monitor [Animator]FullScreen = 1 [Page]DynamicSizing=0	
	Aspect ratio <> monitor - titlebar [Animator]FullScreen = 1 [Page]DynamicSizing=1 [Page]MaintainAspectRatio = 1 or Page size <> monitor - titlebar [Animator]FullScreen = 1 [Page]DynamicSizing=0	The window cannot be moved.

See Also

Page size at runtime

Page size at runtime

By default, a page that is displayed at runtime:

- Appears the same size as when it was saved, unless its parent (the page it was called from) was resized.
- Displays in restored state (you can click **Maximize**, unless viewed in fullscreen mode where maximize is disabled).

You can resize the page by dragging the window frame. Some options for resizing pages may be disabled under certain screen parameter settings. See Screen examples for information.

See Also

Sizing the Page

Securing the window title bar

You can now specify confirmation actions for standard title bar buttons (minimize, maximize, restore, and close) to override the standard behavior for graphic page windows by writing a Cicode function that defines the alternative action, and specifying that Cicode function using the appropriate OnEvent() function event code. The Cicode function needs to return a value.

If your Cicode function returns a zero value, then the Windows message will be passed on to the default message handler so that the window will continue to execute the default behavior.

If your Cicode function returns non-zero, then the Windows message will not be passed on to the default message handler.

In addition, when you call Cicode function Shutdown(), you can decide whether to bypass the confirmation function or not.

See Also

Event Functions
Window Functions
Shutdown()

Defining Page Properties

You can define properties for your graphics pages.

To set up a page:

1. Open the page in the Graphics Builder.
2. Choose File | Properties.
3. Use the Page Properties dialog to define the properties for your graphics pages.

See Also

Page Properties - General

Page Properties - General

Graphics pages have the following general properties:

Window title

The title to be displayed on the page at runtime. A window title can have a maximum size of 64 characters.

The title can be plain text, Super Genie syntax or both e.g. "Pump ?AREA? status".

Description

Enter a description of the page, and its various functions and so on up to a maximum of 250 characters. The description is designed for comments only and does not affect how your system performs, nor does the description appear during runtime.

Previous page

The page that will precede the current page in the runtime browse sequence (maximum 64 characters). Choose an existing page from the menu or type in a page name.

This property is optional. If you do not specify a Previous page, the Page Previous command is inoperative while this page is displayed.

Next page

The page that will follow the current page in the runtime browse sequence (maximum 64 characters). Choose an existing page from the menu or type in a page name.

This property is optional. If you do not specify a Next page, the Page Next command is inoperative while this page is displayed.

[Security] All areas

Select the check box if you want the page to belong to every area (the page can be seen by any operator with view access to at least one area; see Adding user records).

[Security] Area

Enter the area to which this page belongs. Only users with access to this area can view this page. Click the menu to select an area, or type in an area number directly. If you do not specify an area, the page is accessible to every user.

[Page scan time] Default

The Page scan time defines how often this graphics page is updated at runtime. The Page scan time also determines the rate of execution of the While page shown events (i.e. the command(s) which are executed while the page is displayed at runtime).

Select this check box to use the default page scan time (as set using the [Page]ScanTime parameter); otherwise, leave it blank, and enter (or select) another value in the field below. For example, if you enter a page scan time of 200 milliseconds, there will be an attempt to update the page every 200 milliseconds, and any While page shown events are executed every 200 milliseconds.

Note: You can set the default page scan time using the Computer Setup Wizard.

[Logging] Log device

This is the device to which messages are logged for the page's keyboard commands. Click the menu to the right of the field to select a device, or type a device name.

Note: You need to include the *MsgLog* field in the format of the log device for the message to be sent.

[Cluster context] Inherit from caller

Select this check box to make the current page inherit the default cluster context from the page or cicode task that opened this page (for example using the PageDisplay or WinNewAt Cicode functions). If this is checked you cannot select a specific default cluster. See About cluster context for more information.

[Cluster context] Cluster

Specifies a default cluster context for this page.

Click **Apply**, and then click **OK**. To define further properties for the page, select the relevant tabs.

See Also

Defining Page Properties

Page Properties - Appearance

You define the appearance of your graphics pages by using the controls on the **Appearance** tab.

Graphics pages have the following appearance properties:

[Template] Style

The style (appearance) of the graphics page in the runtime system. You can set a default style (to be applied to new pages) using the page defaults of existing pages and templates using the Page Properties.

Most users prefer the Standard style. You can view the pre-defined styles by looking in the Include project under Graphics, Templates.

[Template] Resolution

The default screen resolution of the pages:

Screen Type	Width (pixels)	Height (pixels)
VGA	640	480
SVGA	800	600
XGA	1024	768
SXGA	1280	1024
User	****	****

[Template] Name

The name of the template on which the page is based. Choose a template name from the menu.

Note: If you are looking for a template that you created yourself, check that you entered the correct **Style** and **Resolution** above.

[Template] Show title bar

Determines whether the Windows title bar displays (at the top of the page). The title bar contains the title of the window, maximize, minimize and close buttons (at the right hand end of the title bar), and the control menu button (at the left hand end of the title bar).

To display a page in full screen (without a title bar), the size of the page needs to be the same size as the display (or larger). If the page is smaller than the display, the title bar still displays, even if full screen mode is enabled. Standard templates styles are available for both page sizes.

[View area] Width

The width (in pixels) of the area that the operator can view at runtime. Click the up and down arrows to increase and decrease the width, or type in another value directly.

[View area] Height

The height (in pixels) of the area that the operator can view at runtime. Click the up and down arrows to increase and decrease the height, or type in another value directly.

Page color

The color that will display in the background of the graphics page.

The preview field to the right of this dialog displays a picture of the selected template. Click **Apply**, then click **OK**. To define further properties for the page, click the relevant tabs.

Ignore Quality

This field allows you to override the global [Page]IgnorevalueQuality parameter setting in the Citect.ini file for a particular page.

The options are:

- Default Indication - use the current setting for the [Page]IgnorevalueQuality parameter setting in the Citect.ini file.
- #COM Indication - change the setting to the equivalent of 0 for the [Page]IgnorevalueQuality parameter setting in the Citect.ini file.
- No Indication - change the setting to the equivalent of 1 for the [Page]IgnorevalueQuality parameter setting in the Citect.ini file.
- Background Indication - change the setting to the equivalent of 2 for the [Page]IgnorevalueQuality parameter setting in the Citect.ini file.

See Also

Defining Page Properties

Page properties - Keyboard Commands

The Keyboard Commands property lets you define keyboard commands for the page. A keyboard command is a particular key sequence which executes a command when it is typed in by the operator at runtime.

You can also define a message which will log every time the key sequence is entered.

Operators who do not satisfy the Access requirements specified under **Security** below cannot enter keyboard commands for this page at runtime.

Keyboard commands have the following properties:

Key sequence (32 Chars.)

Enter the key sequences that the operator can enter to execute a command.

You can enter as many key sequences as you like. To add a key sequence, click **Add**, and type in the sequence or select one from the menu. To edit an existing sequence, click the relevant line, and click **Edit**. You can also remove key sequences by clicking **Delete**.

Key sequence command

The commands (set of instructions) to be executed immediately when the selected key sequence is entered. The key sequence command can have a maximum length of 254 characters.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: **Insert Tag**, and **Insert Function**.

[Security] Same area as page

Select this check box to assign the keyboard command to the same area as the page (see Page Properties - General). Only users with access to this area (and any necessary privileges) can issue this command or log the message. If you want to assign this keyboard command to another area, do not select this box: enter another area below.

[Security] Command area

Enter the area to which this keyboard command belongs up to a maximum of 16 characters. Only users with access to this area (and any necessary privileges) can issue this command or log the message. For example, if you enter Area 1 here, operators need to have access to Area 1 to issue this command.

Click an area from the menu or type in an area number.

[Security] No privilege restrictions

Tick this box to disable privilege restrictions; otherwise, leave it blank, and enter another privilege below.

The result of not assigning a privilege restriction differ according to whether you have used areas in your security setup:

- **No Areas**: Every operator has full control of the page.
- **Areas**: An operator will only need view access to the area assigned to this page to have full control over the page (see Adding user records).

[Security] Privilege level

Enter the privilege level that a user needs to possess to issue this command or log the message. For example, if you enter Privilege Level 1 here, operators need to possess Privilege Level 1 to issue this command. You can also combine this restriction with area restrictions (see above). For example, if you assign the keyboard command to Area 5, with Privilege Level 2, the user needs to be set up with Privilege 2 for Area 5 (see Adding user records).

Choose a privilege from the menu or type in an area number.

[Logging] Log Message

A text message sent to the *MsgLog* field of the Log Device when the selected action is performed by the operator at runtime (32 characters maximum). The message can be plain text, Super Genie syntax, or a combination of the two.

If you want to include field data as part of a logged message, you need to insert the field name as part of the device format when you configure the device. For instance, in the **Format** field of the **Devices** form, you could enter {MsgLog,20} {FullName,15}. This would accommodate the logging of messages such as **P2 started by John Smith**.

The log device to which the message is sent is specified through the General page properties.

Click **Apply**, and then click **OK**. Click **Clear Property** to clear property details, and disable the property. To define further properties for the page, click the relevant tabs.

See Also

Defining Page Properties

Page Properties - Events

You use the **Events** tab to assign commands to your graphics pages. These commands can be executed when the page is opened, closed, or whenever the page is open. You can also define different messages to log at the same time.

Page events have the following properties:

Event

There are three events to which commands can be attached. You can select more than one type of event. Unique commands can be attached to each (i.e., you can perform one task when the page is opened, and another when it is closed).

[Event] On page entry

Select this option if you want a command to be executed when the page is initializing. For example, you could execute a command to extract recipe data from a database into a Cicode variable, to be displayed on the page.

> **Note:** Use the "On Page Shown" event if your command uses ActiveX controls. The "On page entry" event may not properly initialize ActiveX controls. The resulting value from a tag write that is perfomed using the "On page entry" event may not be displayed and accessed correctly if the tag is used directly within the page. The "On page shown" event can be used for initializing tags that are to be accessed on the page.

[Event] On page exit

Select this option if you want a command to be executed when the operator exits the page. For example, this command could be used to close a database that was opened at page entry.

Do not call the following functions: `PageGoto()`, `PageNext()`, `PagePrev()`, `PageDisplay()`, or `PageLast()`.

> **Note:** If you shut down the system, "On page exit" functions for the currently open pages do not execute. If a particular page exit code needs to run, call the code before calling the `Shutdown()` function.

[Event] While page is shown

Select this option if you want a command to execute continually for the entire time that the page is open. For example, the **While page is shown** command could be used to perform background calculations for this page.

[Event] On page shown

Select this option if you want a command to execute when a page is first displayed and objects on it (including ActiveX controls) have been initialized. If you want to reference ActiveX controls in your command then use this event instead of "On page entry".

> **Notes:**
> 1. Use the "On Page Shown" event if your command uses ActiveX controls. The "On page entry" event may occur before the ActiveX controls have been initialized.
> 2. Use the "On Page Shown" event if your command requires access to the latest variable tag value. The "On page entry" event may occur before the tag has been resolved.

See also PageInfo Cicode function, type 25.

Command

The commands (set of instructions) to be executed immediately when the selected Event occurs.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options; **Insert Tag**, and **Insert Function**.

Click **Apply**, and then click **OK**. Click **Clear Property** to clear property details, and disable the property. To define further properties for the page, click the relevant tabs.

See Also

Defining Page Properties

Page Properties - Environment

You use the **Environment** tab to define environment information for your graphics pages. You can add, remove, or edit page environment variables for a graphics page, template, or Super Genie.

For example, you can design your loop pages to expand when clicked (a **Tune** >> button) to show tuning parameters. You can use the environment variable to define the size of the expanded window. The DspGetEnv() function would be used as part of the Cicode for the button that expands the window. This means that the Cicode used to expand the window can be generic: you can use the same Cicode each time.

From CitectSCADA v7.20, environment variables are propagated from Super Genie templates to Super Genie pages automatically. Refer to Environment variables in 7.20 for more information.

Variables

The environment variables to be added to the page. When the page is opened during runtime, these variables are created. Their values can then be read using the `DspGetEnv()` function. When the page is closed, the environment variable memory is freed (discarded). For the example above, you would add one variable to define the width of the page, and another to define the height.

To add an environment variable, click **Add**. To edit an existing environment variable, select it, and click **Edit**. (If you click **Add** or **Edit**, a small dialog appears, containing two fields, one for the property and one for its value.) To remove an environment variable, select it, and click **Remove**.

See Also

Defining Page Properties

Page Properties - Associations

An association is the name or number you use when defining a Super Genie substitution, the value or values of which are dynamically generated at runtime.

Use this tab to reference or edit existing associations. In using this tab you can add those associations that are currently in use in a Super Genie substitution and define the appropriate default value and value on error fields. You can also add a description.

Double-click in the row to make it editable, or Click **Add**.

Name:

> **Note**: On upgrading from an earlier version of CitectSCADA existing numbered substitutions are taken from defined Super Genie pages and automatically listed in the drop down box.

For new associations enter the name or number of the association directly in the field provided, or select the name or number of the association from the drop-down box. Only those associations that are 'in-use' as part of a Super Genie substitution are listed.For example, if the substitution in the expression field of a button was'?Speed?' then the association name is Speed.

Name Associations:

- To start with an alphanumeric or "_" character
- To contain only those characters that are allowed in variable tag names (excluding the period character '.'), and not contain any white space.
- Be no more than 253 characters in length

Default (optional)

Enter the default value to be used if the page association has not been performed using the Ass(..) Cicode function at runtime. The default value needs to be either a literal string enclosed in single quotes (e.g. 'a literal value') or a valid tag name.

Value on Error (optional):

Enter the value to be used if no default value is defined and the association is not performed, or if the defined association did not resolve to a valid tag name.

If the substitution on the page is specified in a typed format, the value will be converted to the specified type. If the value can not be converted a hardware alarm will be raised. For example, if the substitution is specified as ?INT 1? and the value on error is set as 'LowLevel', the animation will display zero (0) and a hardware alarm will be raised. Typeless substitutions will display the value on error as specified.

Description (optional):

Enter a description of the page association. The description is useful when maintaining associations.

In Use:

This column can not be edited and simply indicates if the association is in use on the current page in a Super Genie substitution.

When finished click **Add**, the association is then added to the table.

To edit an association select an item and click **Edit** or double-click on the row to make it editable. You can not edit the name of an existing association that is in use. However you can select a new name in an existing row, and when this name is <ENTERED>, a new row of information will be displayed and the existing row of data will disappear.

To delete an association select the row in the table and click **Delete**. You can delete associations in use or not in use. If you delete the object from the page, the status of the association used in the Super Genie substitution will be updated to not in-use.

Click **Ok** or **Apply** to save the association table or click Cancel to discard any changes you may have made.

See Also

Defining Page Properties
Passing Metadata as Super Genie Assoications
Defining Associations for Super Genies
Defining Substitutions for Super Genies

Setting Default Page Settings

You can define settings that new graphics pages will use.

To set the properties to be used for new graphics pages:

1. Choose **File** | **Defaults**.
2. Complete the **Page Defaults** dialog box, then click **OK.**

See Also

Page defaults

Page defaults

You use the Page Defaults dialog box to define the screen resolution and style that graphics pages will use.

You can override these defaults for your pages when you create them or edit them.

[Template] Resolution

Default screen resolution of the standard graphics pages (for example, alarms pages and standard trend pages):

Screen Type	Width (pixels)	Height (pixels)
VGA	640	480

SVGA	800	600
XGA	1024	768
SXGA	1280	1024
User	****	****

[Template] Style

The style (appearance) of the graphics pages in the runtime system. The style you select is the default style for new pages you add to the project. You can change the style of existing pages and templates using the Page Properties.

Most users prefer the Standard style. You can view the pre-defined styles by looking in the Include project under Graphics, Templates.

[Template] Show title bar

Determines whether the Windows title bar displays at the top of each graphics page. The title bar contains the title of the window, maximize, minimize and close buttons (at the right hand end of the title bar), and the control menu button (at the left hand end of the title bar).

To display a page in full screen (without a title bar), the size of the page needs to be the same size as the display (or larger). If the page is smaller than the display, the title bar still displays, even if full screen mode is enabled. Standard templates styles are available for both page sizes

You can override this default for your own pages at the time you create them, or later.

Background color

The color that will display in the background of new graphics pages.

Preview

This dialog also displays a preview of your page with the defaults applied.

Understanding the Drawing Environment

The Citect Graphics Builder is a feature-rich drawing environment that lets you develop a highly functional interface for your projects.

See Also

Grids
Guidelines
Options

Colors
Zooming

Grids

You can use a grid to align and place objects with precision. When the grid is active, any objects or groups of objects that you create, move, or re-size snap to the nearest grid intersection.

To display the grid:

1. Choose **View** | **Grid Setup**.
2. Click the **Display Grid** check box.

To snap to the grid:

- Choose **View** | **Snap to Grid**.

By default, the grid is set to 8 x 8 pixels, with the origin located at the top-left corner of your page.

To change the default grid size and location:

- Choose **View** | **Grid Setup**. The Grid Setup dialog box appears.

See Also

Grid Setup dialog box

Grid Setup dialog box

This dialog box lets you define the origin and pitch (spacing) for Grids. The grid allows you to align and place objects precisely.

Pitch

The horizontal and vertical spacing of the grid points (in pixels). The smallest grid size you can specify is 3 x 3 pixels.

Origin

Specifies the grid origin (base point).

Display Grid

Displays the grid on screen.

Snap to Grid (F8)

Activates the grid. When the grid is active, any object or group of objects that you create, move, or re-size snaps to the nearest grid intersection.

Guidelines

You can use horizontal and vertical guides as a straight-edge, to align and place objects precisely. When an edge or the center of an object gets close to a guide, that edge or center automatically snaps to the guide.

To set up guidelines:

1. Choose **View** | **Guidelines Setup**.
2. To run the guideline horizontally across your page, click **Horizontal**.
3. Enter a position (distance from the top of your page) for the guideline, and click **Set**.
4. To run the guideline vertically down your page, click **Vertical**.
5. Enter a position (distance from the left edge of your page) for the guideline, and click **Set**.
6. Click the **Display Guidelines** check box.
7. Click the **Snap to Guidelines** check box.
8. Click **OK**.

To turn off guidelines:

- Choose **View** | **Snap to Guidelines.**

To move a guideline:

1. Move the cursor over the guideline.
2. Click and hold the left mouse button.
3. Move the guideline to a new location.
4. Release the mouse button.

To create a new guideline with the mouse:

1. Move the cursor over the guideline.
2. Press and hold the Ctrl key.
3. Click and hold the left mouse button.
4. Move the guideline to a new position.
5. Release the mouse button and Ctrl key.

To delete a guideline with the mouse:

- Drag the guideline to the edge of the page.

See Also

Guidelines Setup dialog box

Guidelines Setup dialog box

This dialog box lets you define 32 horizontal and 32 vertical Guidelines. Guidelines act as a straight-edge, allowing you to align and place objects precisely. When an edge or center of an object gets close to a guide, that edge or center automatically snaps to the guide.

Direction

The direction of the guideline (horizontal or vertical).

Display Guidelines

Displays the guidelines on screen.

Snap to Guidelines (F7)

Activates the guidelines. When the guidelines are active, any object or group of objects that you create, move, or re-size snaps to the nearest guideline.

Options

You can define general options for your drawing environment.

To define general options for the drawing environment:

1. Choose **Tools | Options**. The Options dialog box appears.
2. Enter the relevant details.

Option	Description
Display Properties on New	Enables the automatic display of the relevant properties dialog when you add an object to the page.
Display Properties on Copy	Enables the automatic display of the relevant properties dialog when you copy an existing object on the page.
Save Template Warning	Enables the display of an alert message when you modify and then save an existing template. When you modify an existing template, any graphics pages that are associated with the template are not updated until you perform an Update Pages to update each page based on the template.
Modify AN Field	Allows you to modify the number of the animation point (AN) of any object. You cannot change an AN to the same number as an existing AN on the graphics page.
Disable	Disables the display of Genie forms when a new Genie is added to the page

Option	Description
Genie Forms	or an existing Genie is edited. With Genie forms disabled, a form for each native object in the Genie displays.
Display Group Button	Enables the display of a group button on a Genie dialog. The group button displays a form for each native object in the Genie.
Compile Enquiry Message	Enables the "Do you want to compile?" message window when the project has been modified and **Run** is selected. Normally the project is compiled automatically (if the project has been modified) when **Run** is selected.
Fast Runtime Display	Enables the fast display of graphics pages in the runtime system.
List System Pages	Specifies that system pages will be included in: • The list of pages in the Graphics Builder Open and Save dialog boxes. • The Page Properties Previous and Next menus, used for defining a browse sequence for your pages. • The list of files in the Contents area of Citect Explorer. System pages are prefixed with an exclamation mark (!).
Show version 3.xx/4.xx tools	Enables the old (version 3 and version 4) toolbox. This toolbox contains old tools (such as Slider and Bar Graph), which are no longer necessary, as they can be configured using the Object properties.
Fast Update Pages	Affects the operation of Graphic Builder's "Update Pages" tool. If Fast Update Pages is checked, Graphics Builder only updates modified pages. If not checked, every project page is updated.
Transparent Paste	Allows you to specify a color that becomes transparent when a bitmap is pasted on a graphics page. This applies to bitmaps that are pasted from the clipboard, or imported from another application. **Note:** Transparent data bits are not natively supported by other applications. If pasting a bitmap into an external application, transparent bits will appear as the transparent paste color.
ActiveX Automatic Page Saving	Lets you save graphics pages before inserting an ActiveX control. This guards against the loss of data if you insert custom-built ActiveX controls which cause the Graphics Builder to stop operating normally. With Automatic Page Saving enabled, if inserting an unstable ActiveX control causes such an event, you minimize the likelihood of losing work. When you reopen the Graphics Builder, you can normally recover the saved page. The options are: • **Save page before inserting ActiveX controls:** The graphics page

Option	Description
	is automatically saved with the current page name. • **Prompt before saving:** A query will display, asking if you want to save the page before inserting an ActiveX control. • **Do not save automatically:** The graphics page is not saved automatically, and the query is not displayed.

3. Click **OK**.

See Also

Options

Colors

True Color (16.7 million colors) is supported for static and animated objects, including page backgrounds, imported images, symbols, metafiles and bitmaps.

Wherever a particular page or object property has a color value, a current color button will appear.

To choose a different color to the one currently displayed on the button, click on the small black arrow to the right. This launches the Color Picker.

Color Picker

The first 11 rows of the Color Picker show a set of standard colors, including transparent (marked with a black cross). The remaining rows display any user defined colors, referred to as **Color Favorites**. This includes flashing colors, represented by a two color block, divided diagonally.

To select one of the colors displayed on the color picker, click on it.

If the necessary color does not appear, you have the option to create a custom color, or match an existing color from one of your graphics pages.

To match an existing color on a graphics page:

1. Verify that the color you would like to match is present on the page currently displayed in Graphics Builder.
2. From the Color Picker, choose the Color Selector tool (looks like an eyedropper).
3. Use the Color Selector to click on the color you would like to match.
4. The color you have chosen will now appear in the Color Value Display button.

To create a customized color:

1. From the Color Picker, click on the Edit button. This will make the Edit Favorite Color dialog appear.

2. Use the Edit Favorite Color dialog to create the color you would like to use (See Edit Favorite Colors dialog box for details).
3. **Name** the color if necessary.
4. Use the **Add** button to include the color with the Color Favorites displayed on the Color Picker.
5. Click **OK**. The color you have just created will now be selected.

See Also

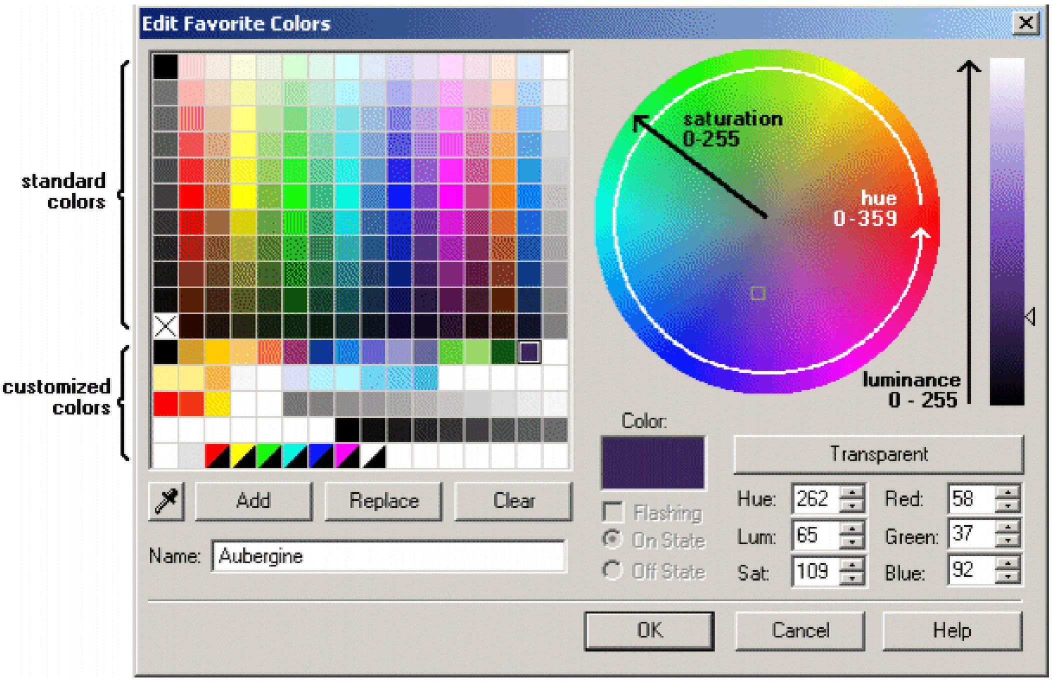
Edit Favorite Colors dialog box
Swap Color dialog box
Adjust colors dialog box

Edit Favorite Colors dialog box

With the Edit Favorite Colors dialog box, you can:

- Make a visual selection of a color you would like to use by clicking on the color wheel.
- Decide the brightness level for a color by clicking on the brightness bar.
- Create a color by entering hue, saturation and luminance values.
- Create a color by entering red, green and blue color values.
- Modify the colors available on the Color Picker by adding, replacing and removing colors on the color grid.

- Name a color, allowing it to be identified on the color grid via a tool tip.



Edit Favorite Colors

Name: Aubergine

☐ Flashing
● On State
○ Off State

Hue: 262
Lum: 65
Sat: 109
Red: 58
Green: 37
Blue: 92

The Edit Favorite Colors dialog contains the following elements:

Colors Grid

Displays a selection of predefined colors. The first 11 rows show a set of standard colors, including transparent (marked with a black cross). The remaining rows display the user-defined colors currently added as favorites. This includes flashing colors, represented by a two color block, divided diagonally.

The colors that appear in this grid represent those that are available from the Color Picker.

Add

Adds the color currently displayed in the **Color** panel to the user-defined favorites in the color grid. If there are no places available on the grid, you will have to use the **Replace** or **Clear** button instead.

Replace

This button allows to you alter a color on the color grid. Select the color you would like to change, adjust its color value, and then hit the **Replace** button. The selected color will be updated to reflect the changes that were made.

Clear

Removes the selected color, leaving an "unused" position on the color grid.

Name

Allows you to associate a name with a predefined color. The name can be viewed as a tool tip in the Color Picker, making it easy to distinguish a specific color among similar shades.

You can associate a name with a newly created color by typing in a name before clicking the **Add** button. You can also apply a name to an existing color by selecting the color, keying in a name and clicking the **Replace** button.

> **Note:** The pre-defined color labels are already defined as color names.

Color

The **Color** panel displays the color created by the current settings applied to the Edit Favorite Color dialog.

The values displayed on the Edit Favorite Color dialog automatically adjust to correctly represent the color currently displayed in this panel. The values in each field are not independent.

Color wheel

The color wheel allows you to visually select a color. It represents the full spectrum of colors in a cyclic layout, with color saturation increasing towards the outside edge. Simply click on the wheel to select a color.

Brightness bar

Allows you to visually select the brightness you would like applied to a color. Click on the bar in the appropriate location to apply a brightness level. The bar represents luminance, as the colors move away from pure black as they progress up the bar to pure white.

Hue

Specifies the hue value for the color currently displayed in the **Color** panel. Hue primarily distinguishes one color from another. The value can be between 0 (zero) and 359, representing degrees around the color wheel. For example, zero is a pure red to the right, 180 is pure cyan on the left.

Sat

Specifies the saturation level for the color currently displayed in the **Color** panel. Saturation level increases the further the selected color moves away from gray scale to a pure primary color. The value can be between 0 (zero) and 255.

Lum

Specifies the luminance of the color currently displayed in the **Color** panel. Luminance represents the brightness of a color, the value increasing the further the color moves away from black towards pure white. The value can be between 0 (zero) and 255.

Note: When you create a color by using HLS values, you may find that the HLS values you specified for a color have changed when you reopen the dialog box. This happens because RGB values are less precise than HLS values, sometimes resulting in several HLS values being assigned the same RGB value. As a result, when the HLS values are generated from the RGB values, some values may change.

Red

Indicates the amount of red used to create the color currently displayed in the **Color** panel. The value can be between 0 (zero) and 255. Adjust this setting if you want to create a color using RGB values.

Green

Indicates the amount of green used to create the color currently displayed in the **Color** panel. The value can be between 0 (zero) and 255. Adjust this setting if you want to create a color using RGB values.

Blue

Indicates the amount of blue used to create the color currently displayed in the **Color** panel. The value can be between 0 (zero) and 255. Adjust this setting if you want to create a color using RGB values.

Transparent

Click this button to select transparent. Wherever transparent is used as a color, the background color, or color behind the transparent object, will be displayed.

Flashing

Check this box if you want to create a flashing color. A flashing color appears as a diagonally divided panel in the color grid. To create a flashing color, you will have to select an **On State** and **Off State** color.

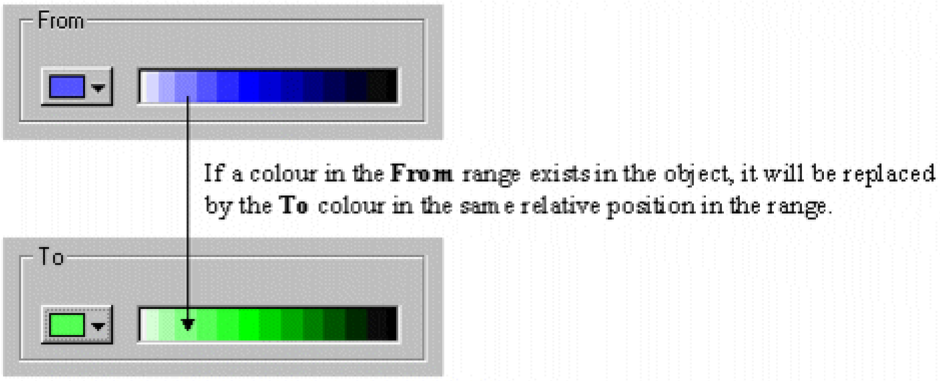
Swap Color dialog box

You use the Swap Color dialog box to swap the colors of an object (or group of objects, or a bitmap) to new colors.

Note: You may find that you get unexpected results when swapping colors in your bitmaps if they contain areas that have either very high or very low luminance values (that is, white or black areas). This is due to the fact that these areas contain almost no color, only luminance. For example, areas that represent specular highlights (the shine on a brightly illuminated steel conveyor belt, for example) will remain colorless no matter how the color for the rest of the bitmap is adjusted. Consequently, before working with the Swap Color dialog box and other color tools, familiarize yourself with the concepts of hue, luminance, and saturation in order to avoid unexpected results with the images on your graphics pages.

From

The current color of the object. If you click the **Swap range** check box, it presents a range of colors in varying degrees of brightness ranging from white to black. Any colors in this range that exist in the object are replaced by the corresponding colors from the **To** range as follows:



If a colour in the **From** range exists in the object, it will be replaced by the **To** colour in the same relative position in the range.

To

The color the original object color will be changed to. If you click the **Swaprange** check box, it presents a range of colors in varying degrees of brightness from white to black. This allows you to swap a whole range of colors at once.

From any color

Specifies to change every color in the object to the new color.

Swap range

Specifies to swap a range of colors. The **From** and **To** boxes indicate the starting colors in the ranges.

Note: You cannot invert colors with **Swap Range** selected. This means, for example, that you could not swap dark red for light green and light red for dark green in one go.

Adjust colors dialog box

The Adjust Colors dialog allows refined color re-mapping with Graphics Builder objects.

Hue (degrees)

The Hue area allows the user to set the color hue range to map to and from. The bars displayed span values of 0 (zero) to 359 degrees, representing the cyclic nature of hue color values. A numeric value for each slider can be keyed in to the field to the right.

- The slider above the **From Hue Range** bar selects the start of the color range that will be mapped.
- The slider below the **From Hue Range** bar selects the end of the color range to be mapped.
- The slider above the **To Hue Range** bar selects the start point for the color range you'll be mapping to. Because the value range is cyclic, the selected area can wrap around to the left side of the bar.

The range of colors that is excluded by your selection is grayed out, allowing for a visual assessment of the selected range.

Lightness (%)

The lightness slider allows you to boost the lightness of colors across a range of (negative) - 100% to 100%. If the slider is increased above zero, colors will tend towards white. If the slider is set below zero, colors will move towards black. A numeric value for the slider can be entered into the field to the right.

The **Selected Hues Only** check box applies the lightness setting to only those colors that will be remapped. Leaving the box unchecked allows the lightness to be adjusted for every color.

Saturation (%)

The saturation slider allows you to boost the saturation of color across a range of (negative) -100% to 100%. If the slider is increased above 0, colors will tend towards primary colors. If the slider is set below zero, colors will move towards grayscale. A numeric value for the slider can be entered into the field to the right.

The **Selected Hues Only** check box applies the saturation setting to only those colors that will be remapped. Leaving the box unchecked allows saturation to be adjusted for every color.

Zooming

Use the Zoom command to view an enlarged view of your drawing. The Zoom command displays a zoom box on screen that magnifies the area around your cursor, enabling you to position or draw objects precisely.

Procedure	Description

To display the Zoom box	Choose **View | Show Zoom**.
To hide the Zoom box	Double-click the control menu box (on the Zoom box) or choose **View | Show Zoom**.

You can change the cursor into crosshairs that extend the full width and length of the drawing area. When you move away from the drawing area, the normal pointer reappears, allowing you to select commands and tools.

Procedure	**Description**
To toggle the cursor between a crosshair cursor and a pointer	Choose **View | Cross Hair Cursor**.
To hide the status bar	Choose **View | Show Status Bar**. Choose the command again to redisplay the status bar.
To hide the Toolbox	Double-click the control menu box (on the Toolbox) or choose **View | Show Tools**. Choose the command again to redisplay the Toolbox.
To hide the toolbar	Choose **View | Show Tool Bar**. Choose the command again to redisplay the tool bar.
To change the speed of the cursor when using the mouse	Choose **View | Mouse Slow Speed**.
To change the speed of the cursor when using the cursor keys	Choose **View | Cursor Keys Slow Speed**. **Hint**: Move the cursor on screen by using the left, right, up, and down cursor keys.

Using libraries

You can store frequently used objects or groups of objects (including bitmap objects) in a library as symbols. You can then paste these symbols onto your page.

After you paste a symbol from the library onto a graphics page, it can be moved, resized, re-shaped, brought to the front, and its properties can be edited, just like any other type of object.

You can define Metadata for a pasted symbol and on the separate animation points that make up that symbol. Each of these animation points will have different ANs. Access to this metadata is obtained through its corresponding AN.

You can paste a symbol from the library to the page:

- As an **unlinked** symbol; the pasted symbol is not updated with changes to the symbol in the library.
- As a **linked** symbol; the symbol on the page is updated when the symbol in the library is changed (to alter the properties of a symbol in the library, open the library and edit it in the library). If you edit the symbol from the page and then change the source symbol in the library, the pasted symbol changes. For example, if you double the size of a pasted symbol, then double the size of the symbol in the library, the pasted symbol doubles again. You can cut the link to the library by using the **Edit | Cut Link** command.

When you save an object in a library, the current properties of that object are saved with it. When you paste it as a symbol to a graphics page, they are used as defaults for the symbol. Pasted symbols have different Appearance properties to those of normal objects: you can only specify a visibility property.

When a symbol is pasted onto a page, the objects that form the symbol cannot be accessed from the page by double-clicking the symbol. To display the properties of these objects, hold the **Control** (CTRL) key down and double-click the specific object. Alternatively, you can select Goto Object from the Tools | the group, and click **OK**. However, if the link to the library is retained, most of these properties are read-only.

A symbol would be useful, for example, if you have defined a command button with a particular security classification, and you need to use it on quite a few graphics pages. You could save it as a symbol, and each time you want to use the button, paste it from the library. Each time it is pasted, it will have the same properties.

Note: There is a comprehensive range of symbols that you can use in your project(s). These symbols are stored in several libraries in the "Include" project. When a library is saved, the first eight characters of the library name needs to be unique to that library.

Copying an object to the library

You can copy an object to the library so that you can use it later on other graphics pages.

To copy an object to the library (i.e. make it a symbol):

1. Click **Select**.
2. Select the object (or group of objects).
3. Choose **Edit** | **Copy to Library**. The Copy To dialog box appears.

Copy To dialog box

This dialog box lets you copy an object (or group of objects) to the library as a symbol.

Feature	Description
Symbol	A name for the symbol.
Library	The project library where the symbol is stored.
Project	The project where the library is stored.
Preview Enable	Displays a thumbnail image of the symbol.

Note: To edit the symbol, select it and click **Edit**. To create a new symbol, click **New**.

New Library dialog box

This dialog box lets you create a new library for the symbol, Genie, or Super Genie (32 characters maximum). Enter a new **Name** for your new library. (The first eight characters of the library name needs to be unique to this library.)

Using symbols

You can create symbols to use on your graphics pages.

To create a new symbol:

1. Click **New**, or choose **File** | **New**.
2. Select **Type: Symbol**

Note: You can also create an object (or group of objects on the page) and then choose **Edit** | **Copy to Library**.

To open an existing symbol:

1. Click **Open**, or choose **File** | **Open**.
2. Select **Type: Symbol.**
3. Select the **Project** where the library is stored.
4. Select the **Library** where the symbol is stored.

5. Select the symbol you want.
6. Click **OK**.

Note: To delete a symbol from the library, select the symbol name and click **Delete**.

To save the current symbol:

1. Click **Save**, or choose **File** | **Save**.
2. Select the Project in which the library is stored.
3. Select the Library in which the symbol is to be stored.
4. Enter a name for the symbol.
5. Click **OK**.

For details on using symbols on graphics page, see Pasted Symbol Objects.

Bitmaps

A bitmap image is a drawing object represented as an array of pixels (or dots), rather than as individual entities. A bitmap is treated as a single object that you can move, copy, and reshape. Because you can edit individual pixels in a bitmap, you can use bitmaps for vignettes and image blending.

You can create bitmaps with the Bitmap Editor, or import bitmaps from any other Windows-based bitmap editor.

In a runtime system, bitmaps are displayed differently to other objects. Bitmaps are mapped directly to the screen (that is, each pixel in the image corresponds to a pixel on the screen). Objects are stored as a series of instructions, and are drawn on the screen in the same order as they were drawn on the graphics page.

The Bitmap Editor

You use the Bitmap Editor to create and edit bitmap images. You can use bitmap images on your graphics pages, and as animated symbols.

The background color in the Bitmap Editor is always transparent, indicated as a white pixel with a black dot at the center. To draw with the background (transparent) color, click (and hold) the right mouse button.

Flashing colors are represented by a diagonally split pixel, indicating the on-state and off-state colors used.

The tool bar of the Bitmap Editor has the following buttons:

✓	Exits the Bitmap Editor and saves editing changes.
✗	Exits the Bitmap Editor and discards changes.
⊕	Zooms in on the image.
⊖	Zooms out on the image.
✒	Selects a color from the image to set as the current color (keyboard shortcut is **Shift+P**). You can also select the current color from the color swatch.
⬚	Displays the **Bitmap Size** dialog box where you can view the image's current dimensions and edit the image's edge.

To resize a bitmap:

1. In Graphics Builder, click the bitmap.
2. Choose **Tools** | **Bitmap Editor**, or press **F9**.
3. Click **Resize**. The Bitmap Size dialog box appears.
4. Select a mode. Click **Grow** to enlarge the image, or **Shrink** to reduce it.
5. For each side of the bitmap, specify how many pixels you want to add or remove, then click **OK**.

To set a color from a bitmap as the current color:

1. In Graphics Builder, click a bitmap.
2. Choose **Tools** | **Bitmap Editor**, or press **F9**.
3. Click **Eye Dropper**, and then click a color in the image. The selected color becomes the current color, and is used when you click elsewhere on the image.

To convert an object (or objects) to a bitmap:

1. Select the object(s).
2. Choose **Tools** | **Convert to Bitmap**.

> **Note:** The Convert to Bitmap operation is only supported in 8-bit (256) color mode.

To invoke the Bitmap editor:

- Choose **Tools** | **Bitmap Editor**.

To paste a bitmap (from another application):

1. Create the image in an external application.
2. Use the external application's copy command to copy the image to your computer's clipboard.
3. Switch to the graphics builder.
4. Choose **Edit** | **Paste**.

You can edit pasted bitmaps by selecting the object and then choosing **Edit** | **Properties**.

To import a graphic:

1. Choose **File** | **Import**. The Import dialog box appears.
2. Select the file you want to import by using the Import dialog box.
3. Click **OK** (or click the file that you want to import and drag it onto a page in Graphics Builder.

You can edit imported bitmaps by selecting the object and then choosing **Edit** | **Properties**.

To import a flashing graphic:

1. Choose **File** | **Import as Flashing**. The Primary Import dialog box appears.
2. Select the first file you want to use for your flashing image.
3. Click **OK.** The Flashing Import dialog box appears.
4. Select the second file you want to use for your flashing image.

Import dialog box

You use the Import dialog box to import a graphic produced with a different application.

If you have selected **Import as Flashing**, two Import dialog boxes appear in sequence, allowing you to choose two images that you want to implement as a flashing symbol. The **Import Primary** dialog allows you to select the initial image used; the **Import Flashing** dialog allows you to choose the second images used.

Look in

The drive and directory where the graphic is stored.

File Name

The name of the graphics file.

Files of Type

The type of graphics file. The following file formats are supported:

- Windows 3.0 bitmaps (*.BMP, *.DIB, *.RLE)
- PCX format bitmaps (*.PCX)

- Text files (*.TXT)
- AutoCAD DXF Files (*.DXF), 2D only. The binary format is also supported.
- Windows metafiles (*.WMF)
- Encapsulated Postscript (*.EPS)
- Fax Image (*.FAX)
- Ventura Image (*.IMG)
- JPEG (*.JPG, *.JIF, *.JFF, *.JGE)
- Photo CD (*.PCD)
- Portable Network Graphic (*.PNG). (PNG files with alpha channels are not supported.)
- Targa (*.TGA)
- Tagged Image Format (*.TIF)
- WordPerfect (*.WPG)

Chapter: 19 Using Objects

Objects are basic drawing entities that you add to your graphics pages. Objects are drawn using the tools in the drawing toolbox, and can be moved, reshaped, or copied after they are drawn. Objects are defined by a set of properties, which are assigned when the object is drawn, or afterwards, by double-clicking (these properties will override any conflicting Cicode Display functions).

Most objects can be assigned keyboard commands and access rights, and can be configured in such a way that they change dynamically at runtime when an expression returns a certain value, or a variable tag changes state. They can even be used as sliders to change the state of a variable.

> **Note:** If an object is part of a group, part of a pasted Genie or symbol, or part of the page's template, you can still access its properties. Simply hold down the **Control** (CTRL) key and double-click the object. Alternatively, choose **Tools** | **Goto Object**, click the object, and then click **OK**. However, if there is still a link to the original Genie/symbol/page template, the object properties are mostly read-only.

Each type of object has its own tool:

	Free Hand Line Objects		Straight Line Objects
□	Rectangle Objects	○	Ellipse Objects
	Polygon Objects		Pipe Objects
A	Text Objects	##	Number Objects
	Button Objects		Symbol Set Objects
	Trend Objects	f(x)	Cicode Objects
	Pasted Symbol Objects		Pasted Genie Objects
	ActiveX Objects		"Process Analyst Objects" in the Process Analyst User Guide

▤	Database Exchange Control Objects		Pelcro Camera Support

See Also

Using groups
Reshaping objects
Using bitmaps
Importing graphics

Using groups

You can group multiple objects. A group has a unique set of properties (much the same set as an object) which determine the runtime behavior of the group as a whole. (The properties of the individual objects in the group remain unchanged.) By defining group properties, you can specify that the entire group changes dynamically under specific runtime conditions (for example, when an expression returns a certain value, or a variable tag changes state).

To edit or view the properties of a group, double-click it. If there are several groups on your page, choose **Tools | Goto Object.** This allows you to see which groups and objects are on your page, making it easier for you to select the object you want to edit. It also allows you to display the properties of the objects (or groups) that make up the group. (You can also edit the properties of an object in the group by holding down the **Control** (CTRL) key and double-clicking the object. Alternatively, select **Goto Object** from the **Tools** menu, select the object, then click **OK**.) This is useful if your page has groups within groups.

A group can be a mix of objects and other groups.

See Also

Reshaping objects

Reshaping objects

Pipe, Polyline, or Polygon objects can be edited to change their shape. Each of these objects consist of a continuous series of lines drawn between structural anchor points called nodes. Nodes are visible when an object is selected. Each node appears as a small square located at specific anchor points along the object. There will be a node located at the start and end of a polyline or pipe, and at every change of direction in an object's shape.

Pipe, Polyline, and Polygon objects can have their shapes changed in many ways. Their nodes can be selected individually or by group and moved to a different position, thus changing the shape of the object. The Pipe, Polyline, and Polygon objects also support node adding and deleting.

Reshaping a line object

Line objects also have nodes, but behave in a more restricted manner than Pipe, Polyline, or Polygon objects.

A straight line can only consist of two nodes, (a start node point and an end node point). These can be individually selected to move the line to a different position, or at least change its direction. The Line object does not support node adding. To achieve the same result as adding a node to a Line object, create a Polyline object instead. Deleting either of the (only two) nodes of a Line object will delete the whole Line object completely.

See Also
Using bitmaps

Using bitmaps

A bitmap image is a special object, represented as an array of pixels or dots, rather than as individual entities. Bitmaps are treated as single objects that you can move, copy, and reshape. You can create and edit bitmaps with the Citect Bitmap Editor. Because you can edit the individual pixels in a bitmap, you can use bitmaps for more 'artistic' images, such as vignettes and image blending.

See Also
Importing graphics

Importing graphics

The Graphics Builder has several file format filters to allow you to import graphics from other applications, such as drafting programs, illustration programs, presentation packages, scanners, etc. After a graphic is imported, you can use the graphics builder to edit the image.

Graphics files can be dragged from a third-party application (such as Windows Explorer), and dropped onto a page in the Graphics Builder.

By default, unavailable colors in an imported bitmap are dithered. To disable this feature, choose **Tools|Options** in the Graphics Builder, and clear the **Dither bitmaps on paste** option. If dithering is disabled, unavailable colors are replaced by the closest match in your color palette.

Object Properties

The properties of an object are defined in the Properties dialog box. When you draw an object, the properties dialog appear, allowing you to define the attributes.

You can also double-click the object (or choose **Edit** | **Properties**) to display the properties dialog.

If an object is part of a group, part of a pasted Genie or symbol, or part of the page's template, you can access the object's properties by pressing **CTRL** and double-clicking the object. Alternatively, choose **Tools** | **Goto Object**, click the object, and then click **OK**.

See Also

Appearance
Movement
Scaling
Fill
Input
Slider
Access
Metadata

Appearance

Click the **Appearance** tab to define the appearance of the object, such as line style, and shadowing. You can also specify when the object will be hidden from the operator (for example when DIGITAL_TAG is OFF).

The check mark to the left of the Appearance tab tells you when an appearance property has been configured. The check marks in the tabs down the right of the dialog indicate which property is configured.

Click the other tabs to define more properties for the object. Most properties work together; for example, an object could possess color fill, movement, and scaling properties simultaneously.

See Also

Object Properties
Movement

Movement

Click the **Movement** tab to move the object vertically or horizontally, or to rotate it, depending on the return of an expression, or the state of a tag.

The check mark to the left of the Movement tab tells you when a Movement property has been configured. The check marks in the tabs down the right of the dialog tell you which property is configured.

Click the other tabs to define more properties for the object. Most properties work together; for example, an object could possess color fill, movement, and scaling properties simultaneously.

See Also

Object Properties
Scaling

Scaling

Click the **Scaling** tab to scale the object both vertically or horizontally, depending on the return of an expression, or the state of a tag.

The check mark to the left of the Scaling tab tells you when a Scaling property has been configured. The check marks in the tabs down the right of the dialog indicate which property is configured.

Click the other tabs to define more properties for the object. Most properties work together, for example, an object could possess color fill, movement, and scaling properties simultaneously.

See Also

Object Properties
Fill

Fill

Click the **Fill** tab to specify the color which is to fill the object, and the level to which the object will be filled. The fill properties can change dynamically, depending on the return of an expression, or the state of a tag etc. (for instance, you could use this tab to visually reflect tank levels).

The check mark to the left of the Fill tab tells you when a Fill property has been configured. The check marks in the tabs down the right of the dialog indicate which property is configured.

Click the other tabs to define more properties for the object. Most properties work together, for example, an object could possess color fill, movement, and scaling properties simultaneously.

See Also

Object Properties
Input

Input

Click the **Input** tab to specify the command to be executed, and the message to be logged when an operator clicks on the object. You can also define keyboard commands for the object, and limit their scope with area and privilege security.

The check mark to the left of the Input tab tells you when an Input property has been configured. The check marks in the tabs down the right of the dialog indicate which property is configured.

Click the other tabs to define other object properties. Most properties work together; for example, an object could possess color fill, movement, and scaling properties simultaneously.

See Also

Object Properties
Slider

Slider

Click the **Slider** tab to use the object as a slider. A variable can be associated with the object, and when the operator moves the object, the value of the variable will change. Objects can be set up to slide vertically and/or horizontally, or they can be rotated.

The check mark to the left of the Slider tab tells you when an Slider property has been configured. The check marks in the tabs down the right of the dialog indicate which property is configured.

Click the other tabs to define other object properties. Most properties work together; for example, an object could possess color fill, movement, and scaling properties simultaneously.

See Also

Object Properties
Access

Access

Click the **Access** tab to assign an area or privilege to the object. Operators without appropriate access rights will not be able to use sliders, object specific keyboard commands etc. It also allows you to disable the object under certain runtime circumstances. This means that the object can be embossed, grayed, or even hidden.

The check mark to the left of the Access tab tells you when an Access property has been configured. The check marks in the tabs down the right of the dialog indicate which property is configured.

Click the other tabs to define more properties for the object. Most properties work together; for example, an object could possess color fill, movement, and scaling properties simultaneously.

Clear Property

Click **Clear Property** to remove the configuration details for this property.

Apply

Click **Apply** to bring your changes into effect. **Apply** allows you to view your changes without closing the Properties dialog.

See Also

Object Properties

Metadata

Metadata is a list of names with corresponding values. The Metadata is attached to an objects animation point. When using any of the DspAnGetMetadata Cicode functions this Metadata is processed, with the field values retrieved, and set at runtime. At runtime Metadata cannot be added or removed.

Metadata can be implemented when configuring the following animation objects-Free Hand Line,Straight Line, Rectangle, Ellipse, Polygon, Pipe, Text, Number, Button, Symbol Set,Trend, Paste Symbol, Active X, Process Analyst (Database Exchange Control, Pelco Camera View Controller, CitectSCADA Web Gate Control)

Note: Metadata can also be assigned to super genie associations. See Passing animation point metadata as Super Genie associations for more information.

To add a Metadata entry:

Click Add or double click on the row to make it editable.

Insert the **Name** of the metadata.

- To start with an alphanumeric or "_" character
- To contain only those characters that are allowed in variable tag names (excluding the period character '.'), and not contain any white space.
- Be no more than 253 characters in length
- To be unique across a specific animation point.
- The name cannot be changed at Runtime.
- The name can contain Genie substitutions

Insert the corresponding **Value** of the specific metadata entry.

- The Value can contain Genie substitutions. E.g. %Level%
- To contain only those characters that are allowed in variable tag names (excluding the period character '.'), and not contain any white space.
- To be no more than 255 characters in length.
- Literal strings are supported and be specified within single quotes e.g.'Literal'.

> **Note:** At runtime Values can be changed, however when the graphics page is closed the Values will revert to the original configuration.

Click Add to save the row. The next row is highlighted and is now editable. Click on the ESC key to cancel this row or continue to enter name | value pairs until finished.

To edit a Metadata entry:

Double-click the cell you want to edit, or select a row and click the edit button.

Modify the Name or Value . Click Apply to bring your changes into effect and then OK to close the graphic object properties dialog.

Refer to Using Metadata for examples on how you can implement metadata in your project.

See Also

Object Properties

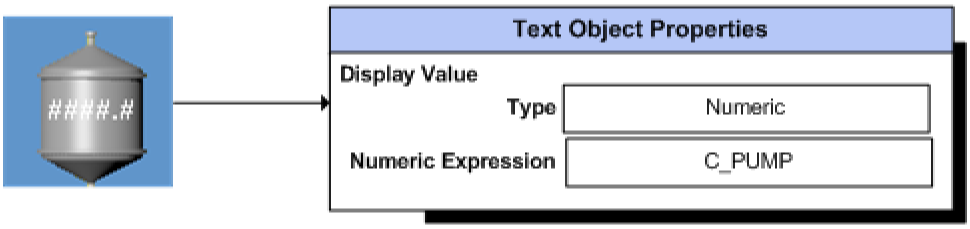
Using Metadata

The following examples provide a guide into how you can implement metadata within your project. In the first example you will set parameters using metadata, and in the second example see how genie substitutions can be used in metadata. The third example shows you how to use Metadata to define a subset of current objects on your graphics page and to use Cicode to call, retrieve and set metadata values from an outside source such as a database.

In the examples a paint factory needs to control the amount of Cyan, Magenta, Yellow, and Key (Black) (CMYK), which is mixed to create different colors in their color chart.

Example 1: Setting Parameters Using Metadata

1. From the toolbox select the 'number' tool and in the Text Properties dialog configure as below. This is the pump that represents the color Cyan.



Text Object Properties

Display Value

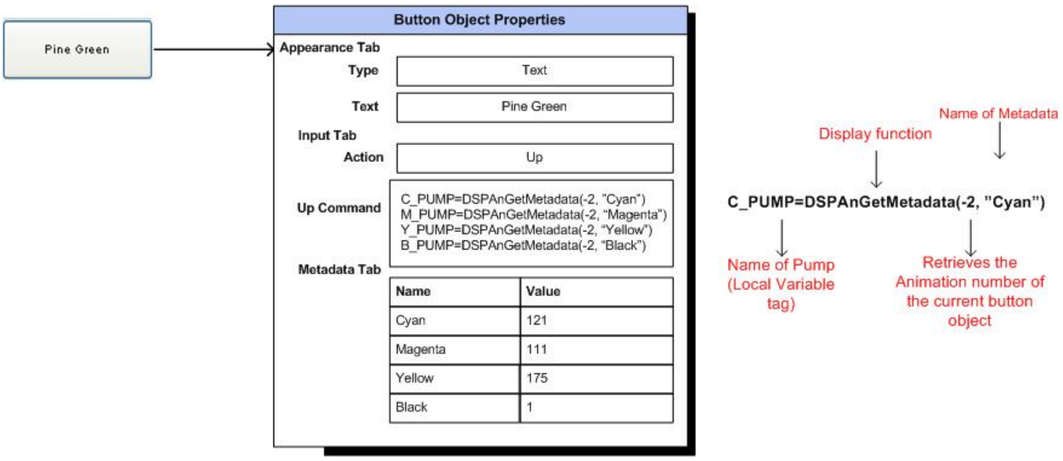
Type: Numeric

Numeric Expression: C_PUMP

2. Repeat to create the remaining pumps M_ PUMP, Y_ PUMP, K_ PUMP

> **Note**:Add these pumps to the example project as Local Variable tags

3. Create a button object. This button when selected at runtime will set the CMYK values for a specific color e.g Pine Green



Pine Green
Display function
Name of Metadata
C_PUMP=DSPAnGetMetadata(-2, "Cyan")
Name of Pump (Local Variable tag)
Retrieves the Animation number of the current button object

Button Object Properties

Appearance Tab

Type: Text

Text: Pine Green

Input Tab

Action: Up

Up Command:

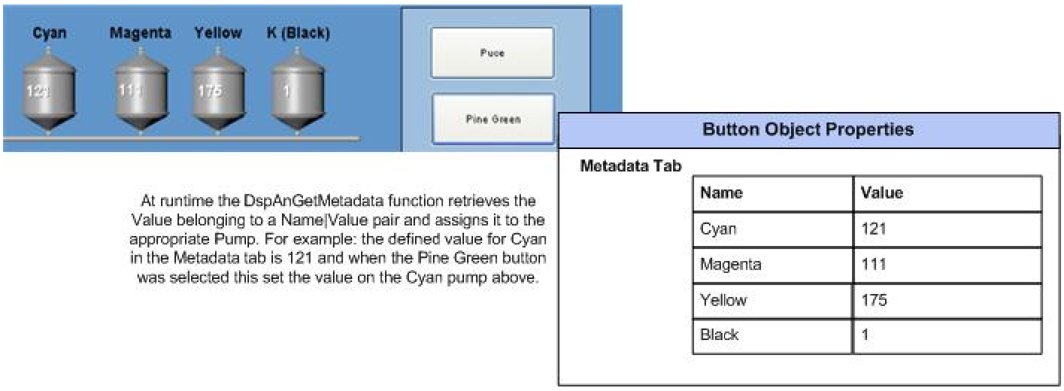
```
C_PUMP=DSPAnGetMetadata(-2, "Cyan")
M_PUMP=DSPAnGetMetadata(-2, "Magenta")
Y_PUMP=DSPAnGetMetadata(-2, "Yellow")
B_PUMP=DSPAnGetMetadata(-2, "Black")
```

Metadata Tab

Name	Value
Cyan	121
Magenta	111
Yellow	175
Black	1

> **Note**: For Display functions DspAnGetMetadataAt, DspAnSetMetadata, DspAnGetMetadata, and DspAnSetMetadataAt -2 is the default value used to retrieve the unique animation number for the button object.

4. At runtime when the 'Pine green' button is selected the values of the metadata defined for Cyan, Magenta, Yellow and Black are retrieved and the relevant pumps set.



At runtime the DspAnGetMetadata function retrieves the Value belonging to a Name|Value pair and assigns it to the appropriate Pump. For example: the defined value for Cyan in the Metadata tab is 121 and when the Pine Green button was selected this set the value on the Cyan pump above.

Button Object Properties

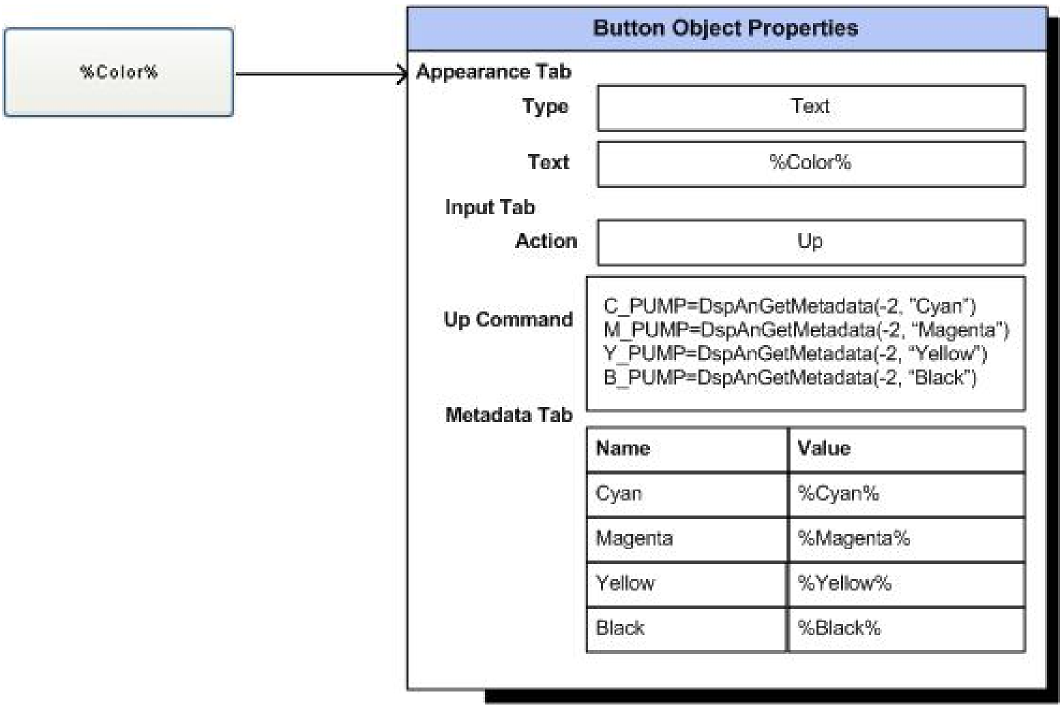
Metadata Tab

Name	Value
Cyan	121
Magenta	111
Yellow	175
Black	1

Example 2: Using Genie Substitutions in Metadata

Instead of creating a new button for every color as you did in the previous example, you can create a genie and save it in the genie library. You can use the genie every time you need to create a new color button.Only needing to configure the name of the button and the CMYK values that belong to it.

1. From File New-> Genie
2. Add a button object onto the page
3. Configure the button – the title of the button and the Value of the Metadata have been defined using genie substitutions e.g %Color% and %Cyan%.



Button Object Properties

Appearance Tab

Type: Text

Text: %Color%

Input Tab

Action: Up

Up Command:

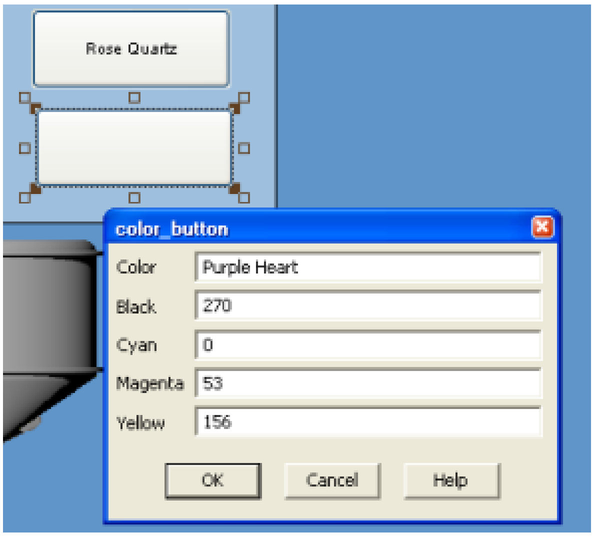
```
C_PUMP=DspAnGetMetadata(-2, "Cyan")
M_PUMP=DspAnGetMetadata(-2, "Magenta")
Y_PUMP=DspAnGetMetadata(-2, "Yellow")
B_PUMP=DspAnGetMetadata(-2, "Black")
```

Metadata Tab

Name	Value
Cyan	%Cyan%
Magenta	%Magenta%
Yellow	%Yellow%
Black	%Black%

4. Save the genie in the appropriate genie library

5. At design time paste the genie onto the graphics page. A dialog will open prompting you for the name of the button, and the numeric values for Cyan, Magenta, Yellow and Black. In this example the name of the color is “Purple Heart”



Color: Purple Heart
Black: 270
Cyan: 0
Magenta: 53
Yellow: 156

6. At runtime when the ‘Purple Heart’ color button is selected the values of the metadata (that were entered in the genie prompt) are retrieved and the relevant pumps set.

Example 3: Defining a subset of Animation points using Metadata

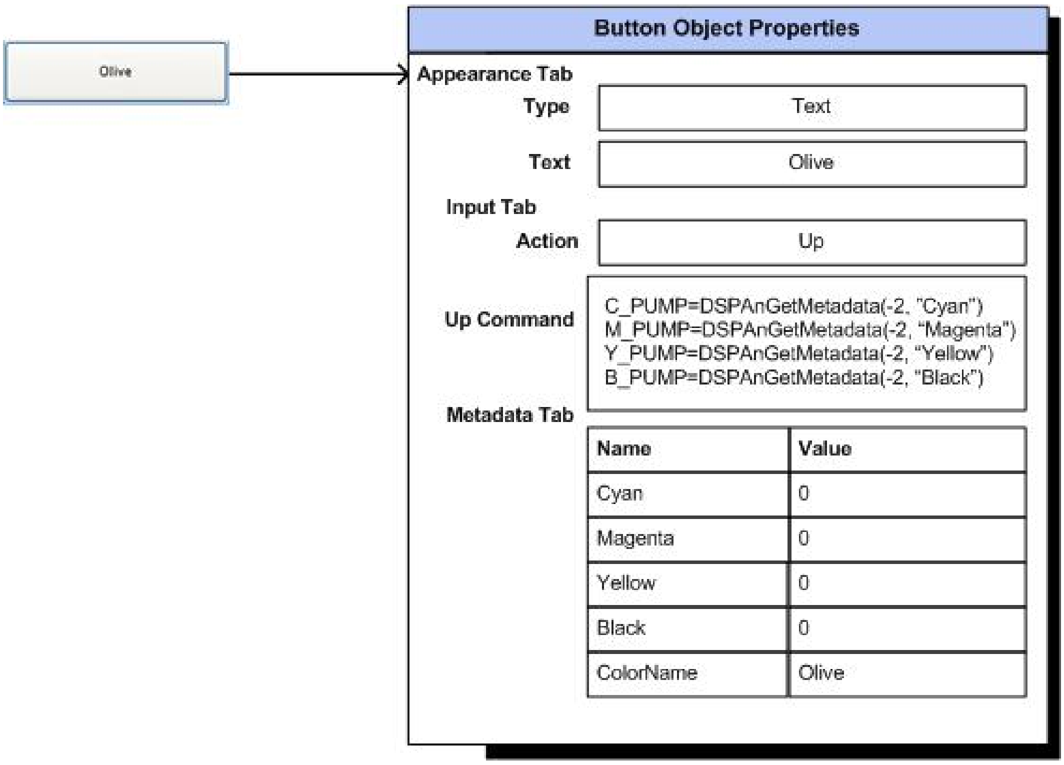
A graphics page could have over four hundred Animation points. Each Animation point belongs to an object, which raises the question how can you define a subset of these objects and perform an operation only on that subset. In this example, metadata is used to define a subset of animation points, and then Cicode is used to operate on this subset collectively using values stored in an outside source.

> **Note**:For the following example it is assumed you are already using an ODBC data source.

This example:

- Defines a subset
- Two cicode functions (created for this example only) – LoadColors(), and LoadColorsForButton()
- Cicode to loop through a database table and match, and retrieve values belonging to the subset defined as “ColorName”.

1. Create four new buttons for the colors – Charcoal, Lime, Olive and Puce.



Button Object Properties

Appearance Tab

Type: Text

Text: Olive

Input Tab

Action: Up

Up Command:

```
C_PUMP=DSPAnGetMetadata(-2, "Cyan")
M_PUMP=DSPAnGetMetadata(-2, “Magenta")
Y_PUMP=DSPAnGetMetadata(-2, “Yellow”)
B_PUMP=DSPAnGetMetadata(-2, "Black”)
```

Metadata Tab

Name	Value
Cyan	0
Magenta	0
Yellow	0
Black	0
ColorName	Olive

2. In the graphics editor create a new button. This button will be used to call the values from the data source using the function LoadColors()
3. At runtime when the LoadColors() button is selected Cicode is triggered that will search for metadata entries named “ColorName” on the current page. In this example the “ColorName” is Olive. From the data source it then loads relevant the CMYK values.
4. When the Olive color button is then selected the values are displayed and the pumps set.

Cicode used in this example:

```
// This function finds animation objects on the current page with
// a metadata entry named "sColorName". This demonstrates how metadata
// values can be used to dynamically identify objects.
FUNCTION LoadColors()
INT nAn;
STRING sColorName;

// Disable hardware errors for this Cicode task. If you don't do
// this then DspAnGetMetadata will generate a hardware error for
// any object that does not have a metadata entry "sColorName".
```

```
ErrSet(1);

nAn = DspGetAnFirst();
WHILE nAn <> -1 DO
// Determine whether the current animation object has a
// metadata entry named "ColorName". If DspAnGetMetadata
// cannot find the named entry, it returns an empty string.
sColorName = DspAnGetMetadata(nAn, "ColorName");
IF sColorName <> "" THEN
LoadColorsForButton(nAn, sColorName);
END
nAn = DspGetAnNext(nAn);
END
END
// This function loads the CMYK metadata values for the specified
// button (nAn) from a database. This demonstrates how metadata values
// can be modified at runtime.
FUNCTION LoadColorsForButton(INT nAn, STRING sColorName)
INT hSQL;
INT nStatus;
STRING sCyan;
STRING sMagenta;
STRING sYellow;
STRING sBlack;

// Load the CMYK values from the database.
hSql = SQLConnect("DSN=Colors");
IF hSQL <> -1 THEN
nStatus = SQLExec(hSQL, "SELECT * FROM Colors WHERE ColorName = '" + sColorName +
"';");
IF nStatus = 0 THEN
IF SQLNext(hSQL) = 0 THEN
sCyan = SQLGetField(hSQL, "Cyan");
sMagenta = SQLGetField(hSQL, "Magenta");
sYellow = SQLGetField(hSQL, "Yellow");
sBlack = SQLGetField(hSQL, "Black");
END
SQLEnd(hSQL);
ELSE
Message("Exec Error", SQLErrMsg(), 48);
RETURN;
END
SQLDisconnect(hSQL);
ELSE
Message("Connect Error", SQLErrMsg(), 48);
RETURN;
END

// Update the metadata values of the specified object.
DspAnSetMetadata(nAn, "Cyan", sCyan);
DspAnSetMetadata(nAn, "Magenta", sMagenta);
DspAnSetMetadata(nAn, "Yellow", sYellow);
DspAnSetMetadata(nAn, "Black", sBlack);
END
```

See Also

Object Properties

Passing Animation Point Metadata as Super Genie Associations

The metadata you define can be assigned to Super Genie associations. The name of the metadata and the name of the association in the Super Genie need to be the same. If the name matches, the value defined for that name | value pair is then inserted into the relevant ‘association’ field in the Super Genie. At runtime the value is dynamically generated and then displayed on the Super Genie page.

When configuring the object that will call the Super Genie, you can use a Cicode function that will perform the associations individually (DspAnGetMetadataAt()) or at once (AssMetadata()).

The table below outlines the possible display results when passing animation point metadata as Super Genie associations:

where - Y = Defined, N= Not Defined, * =N/A, and

VoE - used when Tag value is not resolved.

Default Value is used when the Metadata Value is not defined

Empty String is treated as "Not defined"

Animation Point Metadata Pairs		Associations			Display Results at runtime		
Name	**Value**	**Name**	**Default Value**	**Value on Error**	**Tag Resolution**	**Unresolved Tag**	**Literal**
Y	Y	Y	Y	Y	Tag Value	VoE	Literal
Y	Y	Y	Y	N	Tag Value	#ASS	Literal
Y	Y	Y	N	Y	Tag Value	VoE	Literal
Y	Y	Y	N	N	Tag Value	#ASS	Literal
Y	Y	N	*	*	Tag Value	#ASS	Literal
Y	N	Y	Y	Y	Default Value	VoE	Default Value (literal)
Y	N	Y	Y	N	Default Value	#ASS	N/A
Y	N	Y	N	Y	#ASS	*	*
Y	N	Y	N	N	#ASS	*	*
N	N	N	*	*	#ASS	*	*
N	*	Y	Y	Y	Default Value	VoE	Default Value (literal)

Animation Point Meta-data Pairs		Associations			Display Results at runtime		
N	*	Y	Y	N	Default Value	#ASS	N/A
N	*	Y	N	Y	#ASS	*	*
N	*	Y	N	N	#ASS	*	*
N	*	N	*	*	#ASS	*	*

See Also

Page Properties - Associations

Super Genies

Manipulating Objects

Objects can be manipulated in various ways, such as moving, resizing, and grouping.

See Also

Selecting objects

Moving objects

Resizing objects

Deleting objects

Locking/unlocking objects

Grouping objects

Copying and pasting objects

Changing the Overlap of Objects

Aligning objects

Rotating objects

Mirroring objects

Locate an object

Selecting objects

To select a single object:

1. In Graphics Builder, click **Select**.
2. Click the object. The object's sizing handles appear, and the cursor changes from an arrow to a hand while on the object.

> **Note:** To add other objects to the selection, hold the **Ctrl** key and click each object in turn. To deselect an object from a group selection, while still holding the **Ctrl** key,

click the object again. To select every object in the drawing, use **Select All** from the Edit menu. To deselect every object, click anywhere other than on an object.

To select a group of objects using a marquee box:

1. In Graphics Builder, click **Select**.
2. Click and hold the left mouse button and drag the cursor across the page. This creates a temporary bounding box. Any objects within the box will be selected when you release the mouse button.
3. Release the mouse button.

Note: To add other objects to the selection, or remove objects from the selection, hold the **Ctrl** key and click each object in turn. To quickly select every object in the drawing, you can use **Select All** from the Edit menu**.** To deselect every object, click anywhere other than on an object.

See Also

Manipulating Objects
Moving objects

Moving objects

You can move an object if you want by clicking the object and dragging it to the new location.

If you move an object as soon as you select it, an outline of the object boundary displays as you move it on your page. If you hold the mouse stationary for 1 sec or more before you move the object, the object itself displays as you move it, enabling you to better see the result for a more accurate placement.

See Also

Manipulating Objects
Resizing objects

Resizing objects

You can resize objects simply.

To re-size an object:

1. Select the object, and then move the cursor over a sizing handle. The cursor changes to a two-sided arrow showing the directions that you can drag the handle to resize the object.

2. Click and drag the handle to a new location. The object's bounding box appears as you drag.
3. Release the mouse button.

Select a sizing handle at a corner of the object to change the height and width at the same time. If you hold the **CTRL** key while you move a corner sizing handle, the object maintains its aspect ratio (that is, a square remains a square).

To select an object's node:

1. Select the object. Node selection is only applicable to a Line, Pipe, Polyline, or Polygon object.
2. Position and hold the mouse pointer over the node. The mouse pointer will change to a small cross shape.
3. Click the left mouse button. The color of the selected node will change to the inverse of the background (light on a dark background, dark on a light background).

If you have a node selected and then click another node within the same object, the first node will deselect. To select multiple nodes, hold down the **Ctrl** key and click each node you want to select. (With the Ctrl key held down, you can click a previously selected node to deselect it.) Clicking the same node again toggles the selection of the node alternately on and off. To deselect every node, click anywhere other than on a node.

To move a node of an object:

1. Select the node(s).
2. Position and hold the mouse pointer over the node. The mouse pointer will change to a small cross shape.
3. Click and hold the left mouse button. The cursor changes to a positioning symbol.
4. Drag the selected node(s) to the desired position.
5. Release the mouse button.

Selecting and moving multiple nodes maintains the aspect ratio of the graphic object between the selected nodes.

To add a node to an object:

1. Select the object.
2. Position and hold the mouse pointer directly over the graphic object at the exact point where the new node will be added. The mouse pointer will change to a pointing hand shape.
3. Press **Insert**, or either of the available plus (+) keys.

Depending upon the keyboard you're using, the plus key could be either on the number pad section, or accessed on the main keyboard via the **Shift** key.

To delete a node from an object

1. Select the node(s).
2. Press **Delete** or a minus (-) key.

If no nodes are selected, pressing the **Delete** or minus keys deletes the object.

See Also

Manipulating Objects
Deleting objects

Deleting objects

You can delete unwanted objects.

To delete an object (or a group of objects):

1. Select the object (or group of objects)
2. Choose **Edit|Delete** or press the **Delete** key (or a minus (-) key).

See Also

Manipulating Objects
Locking/unlocking objects

Locking/unlocking objects

On complex drawings (with many objects), selecting a discrete group of objects without including every object (in the selected area) can be difficult (for example when an object is hidden by another object). To avoid unintentionally selecting an object, you can `lock' it in position. When an object is `locked', it cannot be selected, deleted, moved, or edited. Objects are locked only when the **Edit** menu **Break Lock Mode** option is not selected.

To lock an object:

1. Select the object.
2. Choose **Edit | Lock Object**.

To unlock an object:

1. Choose **Edit | Break Lock Mode**.
2. Select the object.
3. Choose **Edit | Unlock Object**.

See Also

Manipulating Objects
Grouping objects

Grouping objects

You can group objects to make them easier to manipulate.

To group objects:

1. Click **Select**.
2. Select the objects to group, and then click **Group** (or choose **Arrange** | **Group Objects**).

To ungroup objects:

1. Click **Select**.
2. Select the objects to group, and then click **Ungroup** (or choose **Arrange** | **Ungroup Objects**).

To change the properties of a group:

1. Click **Select**.
2. Double-click the group. The Properties dialog box appears.
3. Change the properties as necessary.
4. Alternatively, choose **Tools** | **Goto Object**, select the group, and then click **OK**.

See Also

Manipulating Objects
Copying and pasting objects

Copying and pasting objects

You can copy and paste objects onto other graphics pages.

To copy an object to the clipboard:

1. Click **Select**.
2. Select the object (or group of objects).
3. Click **Copy** or choose **Edit** | **Copy**.

To paste an object (or group of objects) from the clipboard:

- Click **Paste** or choose **Edit** | **Paste**.

To cut (remove) an object:

1. Click **Select**.
2. Select the object (or group of objects).
3. Click **Cut** or choose **Edit** | **Cut**.

To paste an object (or group of objects) from the clipboard:

- Click **Paste**, or choose **Edit** | **Paste**.

You can use the clipboard to transfer objects between different graphics pages and from other graphics applications. By default, unavailable colors in a pasted bitmap are dithered. To disable this feature, select **Options** from the **Tools** menu in the Graphics Builder, and remove the tick from the **Dither bitmaps on paste** option.

To cancel your last drawing operation(s):

- Click **Undo**, or choose **Edit** | **Undo**.

You can undo operations performed during the current drawing session apart from edits to bitmaps.

To cancel the Undo (or Redo) the last drawing operation(s):

- Choose **Edit** | **Redo**.

See Also

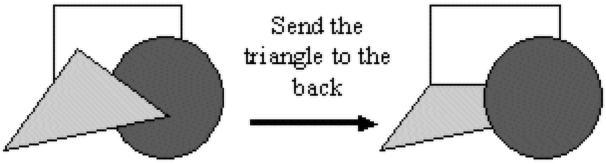
Manipulating Objects
Changing the Overlap of Objects

Changing the Overlap of Objects

In the Graphics Builder, objects can overlap. Where there is an overlap, new objects are always placed on top of existing objects. You can move objects backwards and forwards through this display sequence to change the way they overlap.

To position an object (or group of objects) behind other objects so that objects overlap it:

1. Click **Select**.
2. Select the object (or group of objects).
3. Click **Send to Back** or choose **Arrange** | **Send to Back**.

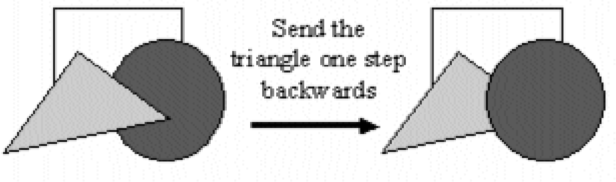


An object can be moved backwards and forwards one step at a time (rather than completely to the back, or completely to the front).

To send an object (or group of objects) one step backwards:

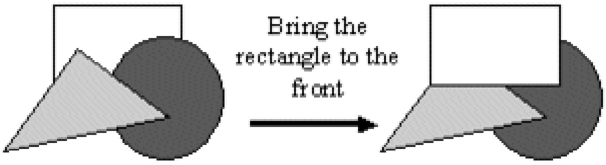
1. Click **Select**.
2. Select the object (or group of objects).

3. Click **Send Backwards** or choose **Arrange** | **Send Backwards**.



To bring an object to the front:

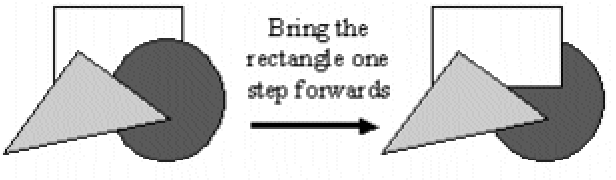
1. Click **Select** tool.
2. Select the object (or group of objects).
3. Click **Bring to Front** or choose **Arrange** | **Bring to Front**.



An object can be moved backwards and forwards one step at a time (rather than completely to the back, or completely to the front).

To bring an object (or group of objects) one step forwards:

1. Click **Select**.
2. Select the object (or group of objects)
3. Click **Bring Forwards** or choose **Arrange** | **Bring Forwards**.



See Also

Manipulating Objects
Aligning objects

Aligning objects

You can precisely align a group of objects vertically, horizontally, or both.

To align objects:

1. Click **Select**.
2. Select the objects.

3. Choose **Arrange** | **Align**. The Align dialog box appears.

Option		Description
Vertical	Top	Aligns the top edges of the selected objects
	Center	Vertically aligns the midpoints of the selected objects
	Bottom	Aligns the bottom edges of the selected objects
	Even	Vertically aligns the midpoints of the selected objects with even spacing
	None	Doesn't change the vertical alignment of the selected objects
Horizontal	Left	Aligns the left edges of the selected objects
	Center	Horizontally aligns the midpoints of the selected objects
	Right	Aligns the right edges of the selected objects
	Even	Horizontally aligns the midpoints of the selected objects with even spacing
	None	Doesn't change the horizontal alignment of the selected objects

See Also

Manipulating Objects

Rotating objects

Rotating objects

You can rotate an object 90° right (clockwise) or 90° left (counter-clockwise).

To rotate an object (or group of objects):

1. Click **Select**.
2. Select the object(s).
3. Choose **Arrange** | **Rotate**.

To rotate a text object:

1. Click **Select**.
2. Select the object(s).
3. Choose **Tools** | **Convert to Bitmap.**
4. Choose **Arrange** | **Rotate**.
5. Select the direction to rotate the object (or group of objects).

The object(s) are rotated 90 degrees in the direction you select. To rotate the object(s) 180 degrees, click the direction button twice.

See Also

Mirroring objects
Manipulating Objects

Mirroring objects

You can mirror an object relative to its horizontal or vertical axis.

To mirror an object (or group of objects) relative to its horizontal or vertical axis:

1. Click **Select**.
2. Select the object(s)
3. Choose **Arrange** | **Mirror**.
4. Choose the axis about which to mirror the object (or group of objects).

See Also

Locate an object
Manipulating Objects

Locate an object

You use the Goto Object dialog box to find, select and access the properties of objects on your current page or on page templates used by the current page. You can select objects, Genies, symbols, and groups on the page (or template) as well as the graphical elements that make up those Genies, symbols, and groups.

To locate an object (or a group, Genie, symbol, or page template) on the current page and display its properties:

1. Choose **Tools** | **Goto Object**.
2. Locate the object (or group, Genie, symbol, or page template) in the tree structure or type the relevant AN in the **Object AN** box.
   Objects that are made up of several objects (or other graphical elements) have a plus sign (+) next to them. Click the + sign to see these component objects.
3. Double click the object in the tree structure, or click **OK** to display its properties.

Chapter: 20 Understanding Object Types

There are many different object types, each with their own unique set of properties. For details of properties common to every object type, see Defining Common Object Properties.

See Also

Free Hand Line Objects
Straight Line Objects
Rectangle Objects
Ellipse Objects
Polygon Objects
Pipe Objects
Text Objects
Number Objects
Button Objects
Symbol Set Objects
Trend Objects
Cicode Objects
Pasted Symbol Objects
Pasted Genie Objects
ActiveX Objects
"Using the Process Analyst" in the Process Analyst User Guide
Database Exchange Control Objects

Free Hand Line Objects

The Free Hand Line tool allows you to draw lines. Lines can be moved, resized, reshaped, brought to the front and so on, and their properties edited just like other types of object.

To draw a freehand line:

1. Click the **Freehand** tool.

2. Move the cursor to where you want the line to start.
3. Click and drag the cursor to draw the line.
   When you release the mouse button, the object properties dialog for the line is displayed.

See Also

Freehand Line Properties - Appearance (General)
Understanding Object Types

Freehand Line Properties - Appearance (General)

Free Hand Line drawings have the following general appearance properties.

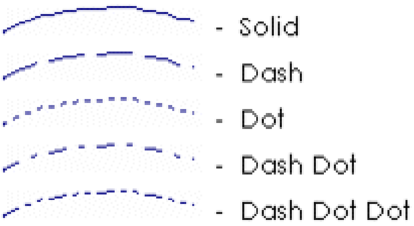
[Line] Width

The width of the line (in pixels). You can change the width by clicking the up and down arrows to the right of the field, or by entering another value in this field.

If you make the line more than 1 pixel wide, it needs to be solid.

[Line] Style

The style of the line. You can choose one of the following line styles:



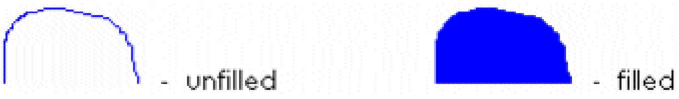
To change the style, choose a style from the menu to the right of this field.

[Line] Color

The color of the line.

[Fill] Filled

The Filled check box determines whether the object will be filled with a color. If you check this box, an invisible line is drawn from one end of your line to the other. Everything between the invisible line and your line will be filled.



[Fill] Color

The color with which the object will be filled. The color that you select as your fill color here is static.

To specify a fill color that changes with runtime conditions, click the **Fill** tab.

If you have enabled the Fill (Color) properties, be aware that the color you select here will override the OFF color for Fill Color (On/Off), the ABC color for Fill Color (Multi-state), Array Color 0 for Fill Color (Array), and the At minimum color for Fill Color (Gradient).

For help on the remaining properties tabs, see Defining Common Object Properties.

Straight Line Objects

The Straight Line tool allows you to draw straight lines. Straight lines can be moved, resized, reshaped, brought to the front and so on, and their properties edited just like any other type of object.

To draw a straight line:

1. Click the **Straight Line** tool.

2. Move the cursor to where you want to start the line.
3. Click and drag to draw the line. (If you hold the **Ctrl** key while drawing the line it is constrained to the vertical or horizontal.
   When you release the mouse button, the object properties dialog for the line is displayed.

See Also

Straight Line Properties - Appearance (General)
Understanding Object Types

Straight Line Properties - Appearance (General)

Straight Lines have the following general appearance properties.

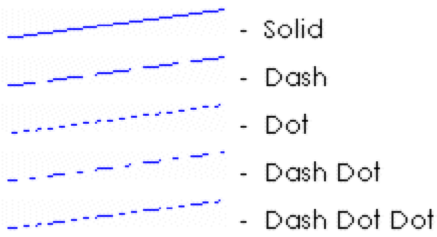
[Line] Width

The width of the line (in pixels). You can change the width by clicking the up and down arrows to the right of the field, or by entering another value in this field.

If you make the line more than 1 pixel wide, it needs to be solid.

[Line] Style

The style of the line. You can choose from the following line styles:



To change the style, choose a style from the menu to the right of this field.

[Line] Color

The color of the line.

For help on the remaining properties tabs, see Defining Common Object Properties.

Rectangle Objects

Use the Rectangle tool to draw rectangles and squares. Rectangles and squares can be moved, resized, reshaped, brought to the front and so on, and their properties edited just like other types of object.

To draw a rectangle:

1. Click the **Rectangle** tool.
2. Move the cursor to where you want the rectangle to start.
3. Click and drag the mouse to the opposite corner of the rectangle and release the mouse button. If you hold the **Shift** key before you start drawing the rectangle, it is drawn from its center outwards.
   When you release the mouse button, the object properties dialog for the rectangle is displayed.

To draw a square:

1. Click the **Rectangle** tool.
2. Click (and hold) the **Ctrl** key.
3. Move the cursor to where you want the square to start and click (and hold) the mouse button.
4. Drag the cursor to the opposite corner of the square and release the mouse button. If you hold the **Shift** key (and the **Ctrl** key) before you start drawing the square, it is drawn from its center outwards.
   When you release the mouse button, the object properties dialog for the square is displayed.

See Also

Rectangle Properties - Appearance (General)
Understanding Object Types

Rectangle Properties - Appearance (General)

Rectangles have the following general appearance properties.

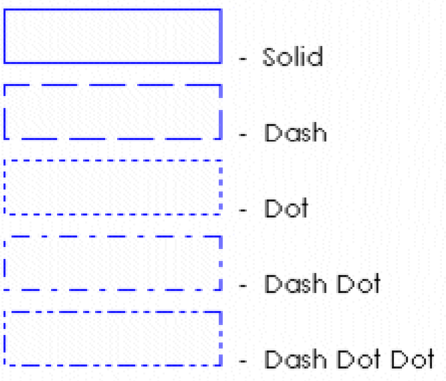
[Line] Width

The width of the outline for the rectangle (in pixels). You can change the width by clicking the up and down arrows to the right of the field, or by entering another value in this field.

If you make the line more than 1 pixel wide, it needs to be solid.

[Line] Style

The outline style of the rectangle. You can choose from the following line styles:



To change the style, choose a style from the menu to the right of this field.

[Line] Color

The outline color of the rectangle.

[Fill] Filled

The Filled check box determines whether the rectangle will be filled with a color.

[Fill] Color

The color with which the rectangle will be filled. The color that you select as your fill color here is static.

To specify a fill color that changes with runtime conditions, click the Fill tab. If you have enabled the Fill (Color) properties, be aware that the color you select here overrides the OFF color for Fill Color (On/Off), the ABC color for Fill Color (Multi-state), Array Color 0 for Fill Color (Array), and the At minimum color for Fill Color (Gradient).

[Object type] Extra line

Adds an extra line (1 pixel width) of lowlight color to the rectangle, if the rectangle is defined as Raised or Lowered (click the 3D Effects tab).

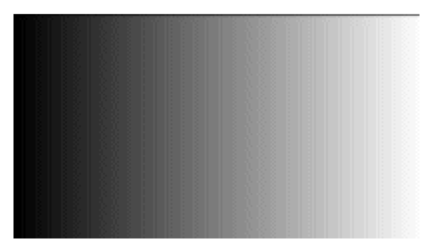
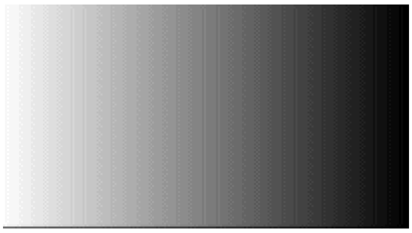
[Gradient] Color

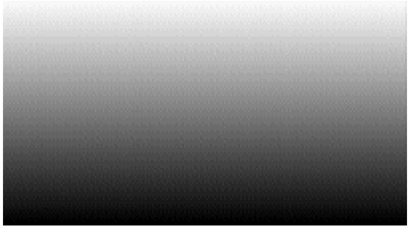
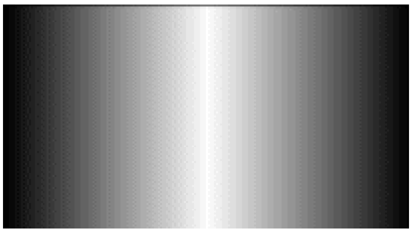
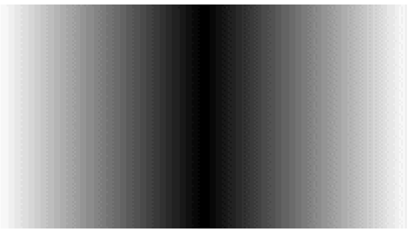
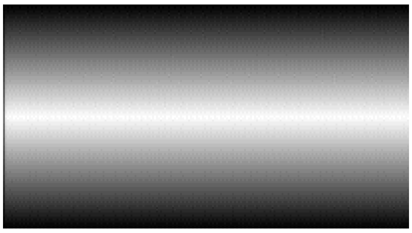
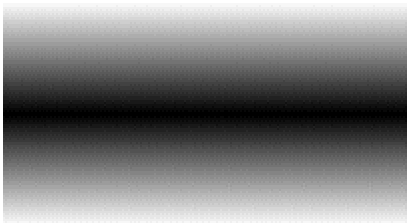
Controls the color of the gradient fill between the fill color and the gradient color. This option is available only when the **Filled** and **Gradient Fill** check boxes are selected. The gradient is updated at runtime to reflect the gradient between the two colors selected. Gradient fills support flashing colors.

Gradients do not rotate with an object; for example, if an object contains a left-to-right gradient fill and is rotated 90 degrees (either at runtime or in Graphics Builder), the gradient is still left to right.

[Gradient] Direction

The direction to be used for the gradient color. Use the table below as a guide to choose the gradient color direction you want.

Example	Gradient Color Direction
	Left to Right
	Right to Left
	Top to Bottom

	Bottom to Top
	Horizontal Gradient to Middle
	Horizontal Gradient from Middle
	Vertical Gradient to Middle
	Vertical Gradient from Middle

[Object type] Border

Adds an extra line (1 pixel width) of black to the perimeter of the rectangle.

[Object type] Corner Radius

Controls the radius of the corners of the rectangle. Enter a value between 0 and 32. The higher the value, the more rounded the corners of the rectangle.

When the radius is greater than 0, the **Extra line** and **Border** options are not available.

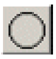
For help on the remaining properties tabs, see Defining Common Object Properties.

Ellipse Objects

You use the Ellipse tool to draw ellipses, circles, arcs, and pie-slices. Ellipse objects can be moved, resized, reshaped, brought to the front and so on, and their properties edited just like other types of object.

To draw an ellipse:

1. Click the **Ellipse** tool.



2. Move the cursor to a corner of the bounding rectangle (marquee) and click (and hold) the mouse button.
3. Drag the cursor to the opposite corner of the bounding rectangle and release the mouse button. If you hold the **Shift** key before you start drawing the ellipse, it is drawn from its center outwards.
   When you release the mouse button, the object properties dialog for the ellipse is displayed.

To draw a circle:

1. Click the **Ellipse** tool.
2. Click (and hold) the **Ctrl** key.
3. Move the cursor to a corner of the bounding rectangle (marquee) and click (and hold) the mouse button.
4. Drag the cursor to the opposite corner of the bounding rectangle and release the mouse button. If you hold the **Shift** key and the **Ctrl** key before you start drawing the circle, it is drawn from its center outwards.
   When you release the mouse button, the object properties dialog for the circle is displayed.

See Also

Ellipse Properties - Appearance (General)
Understanding Object Types

Ellipse Properties - Appearance (General)

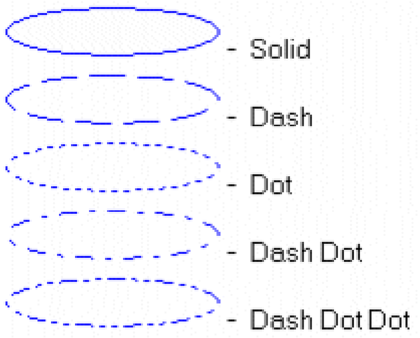
Ellipses have the following general appearance properties:

[Line] Width

The width of the outline of the ellipse (in pixels). You can change the width by clicking the up and down arrows to the right of the field, or by entering another value in this field. If you make the line more than 1 pixel wide, the line style will be solid.

[Line] Style

The outline style of the ellipse. You can choose one of the following line styles:



To change the style, choose a style from the menu to the right of this box.

[Line] Color

The outline color of the ellipse.

[Fill] Filled

The Filled check box determines whether the ellipse will be filled with a color.

[Fill] Color

The color with which the ellipse will be filled. The color that you select as your fill color here is static.

To specify a fill color that changes with runtime conditions, click the Fill tab. If you have enabled the Fill (Color) properties, be aware that the color you select here will override the OFF color for Fill Color (On/Off), the ABC color for Fill Color (Multi-state), Array Color 0 for Fill Color (Array), and the At minimum color for Fill Color (Gradient).

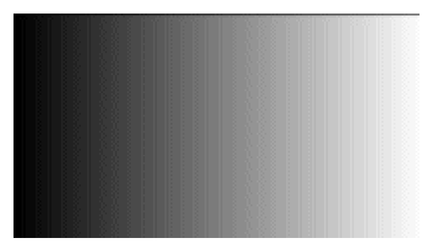
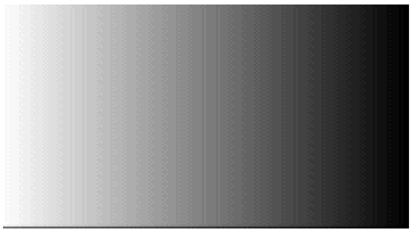
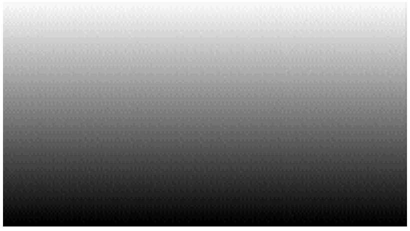
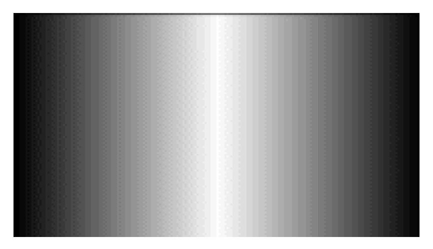
[Gradient] Color

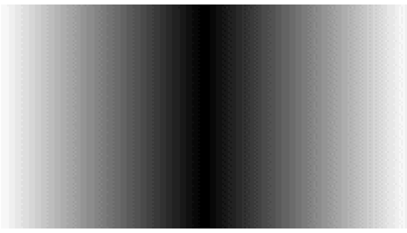
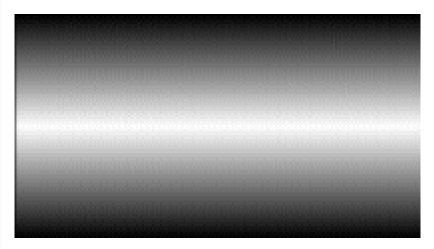
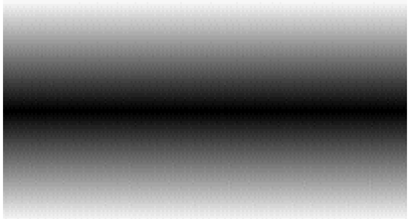
Controls the color of the gradient fill between the fill color and the gradient color. This option is available only when the **Filled** and **Gradient Fill** check boxes are selected. The gradient is updated at runtime to reflect the gradient between the two colors selected. Gradient fills support flashing colors.

Gradients do not rotate with an object; for example, if an object contains a left-to-right gradient fill and is rotated 90 degrees (either at runtime or in Graphics Builder), the gradient is still left to right.

[Gradient] Direction

The direction to be used for the gradient color. Use the table below as a guide to choose the gradient color direction you want.

Example	Gradient Color Direction
	Left to Right
	Right to Left
	Top to Bottom
	Bottom to Top
	Horizontal Gradient to Middle

	Horizontal Gradient from Middle
	Vertical Gradient to Middle
	Vertical Gradient from Middle

[Object type] Ellipse

Select this radio button if you want to the object to be a full ellipse.

For a full ellipse, you do not need to specify Start and End angles.

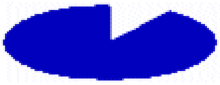
[Object type] Pie-slice

Select this radio button if you want to remove a section from your ellipse (i.e., you want it to resemble a pie-slice).

If you select this option, you can specify a Start angle, and an End angle:

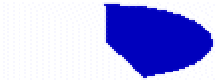
Start angle

The angle (measured clockwise from 0 degrees) of the section to be removed from the ellipse. For example, if you enter a start angle of 50 degrees, your pie-slice would look something like this:

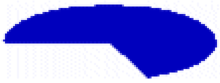


End angle

The angle (measuring clockwise from 0 degrees) of the section of the ellipse which is to remain. For example, if you enter an end angle of 150 degrees, your pie-slice would look something like this:



Start and End angles can be combined for various effects. For example, a Start angle of 270 degrees, and an End angle of 150 degrees would produce the following pie-slice:



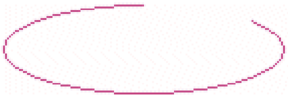
[Object type] Arc

Select this radio button if you want to draw an arc.

If you select this option, you can specify a Start angle, and an End angle:

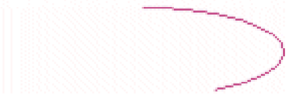
Start angle

The angle (measured clockwise from 0 degrees) defining the segment to be removed from the ellipse, leaving an arc. For example, if you enter a start angle of 50 degrees, your arc would look something like this:

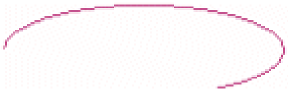


End angle

The angle (measuring clockwise from 0 degrees) defining the segment of the ellipse which is to remain. For example, if you enter an end angle of 150 degrees, your pie-slice would look something like this:



Start and End angles can be combined for various effects. For example, a Start angle of 270 degrees, and an End angle of 150 degrees would produce the following arc:



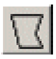
For help with the other properties, see Defining Common Object Properties.

Polygon Objects

Use the Polygon tool to draw polygons and polylines. Polygons can be moved, resized, reshaped, brought to the front and so on, and their properties edited just like other types of object.

To draw a polygon:

1. Click the **Polygon** tool.



2. Move the cursor to where you want the polygon to start and click and hold the mouse button.
3. At the end of the first line segment, release the mouse button.
4. Move the cursor to each point on the polygon in turn, and click the mouse button (clicking and dragging is not necessary after the first segment).
   To draw horizontally or vertically only, hold the **Ctrl** key down when you are drawing the polygon.
5. To complete the polygon, double-click the mouse button.
   When you complete the polygon, the object properties dialog is displayed.

To draw a polyline:

1. Click the **Polygon** tool.
2. Move the cursor to where you want the polyline to start and click and hold the mouse button.
3. At the end of the first line segment, release the mouse button.
4. Move the cursor to each point on the polyline in turn and click the mouse button (clicking and dragging is not necessary after the first segment).
   To draw horizontally or vertically only, hold the **Ctrl** key down when you are drawing the polyline.
5. To complete the polyline, double-click the mouse button.
   When you release the mouse button, the object properties dialog for the circle is displayed.
   Initially, the object will actually be a polygon. To change it to a polyline, in the object's properties dialog define it as [Object type] Open.

See Also

Polygon Properties - Appearance (General)
Understanding Object Types

Polygon Properties - Appearance (General)

Polygons have the following general appearance properties.

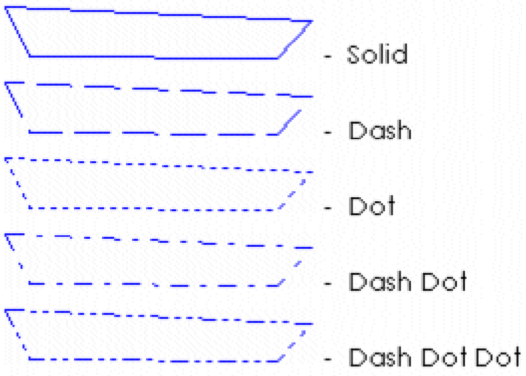
[Line] Width

The width of the outline of the polygon (in pixels). You can change the width by clicking the up and down arrows to the right of the field, or by entering another value in this field.

If you make the line more than 1 pixel wide, it needs to be solid.

[Line] Style

The outline style of the polygon. You can choose any one of the following line styles:



To change the style, choose the style you want from the menu to the right of this field.

[Line] Color

The outline color of the polygon.

[Fill] Filled

The Filled check box, determines whether the polygon will be filled with a color.

[Fill] Color

The color with which the polygon will be filled. The color that you select as your fill color here is static.

To specify a fill color that changes with runtime conditions, click the Fill tab. If you have enabled the Fill (Color) properties, be aware that the color you select here will override the OFF color for Fill Color (On/Off), the ABC color for Fill Color (Multi-state), Array Color 0 for Fill Color (Array), and the At minimum color for Fill Color (Gradient).

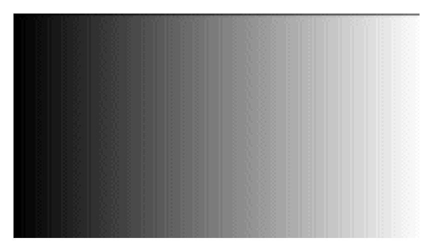
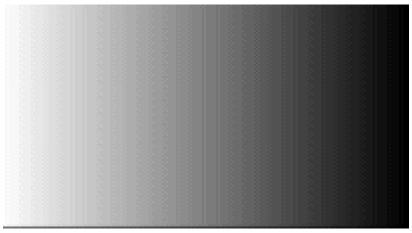
[Gradient] Color

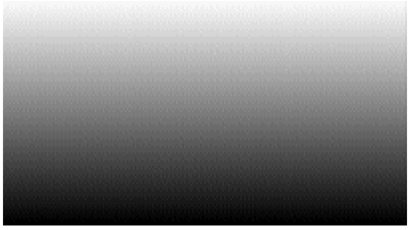
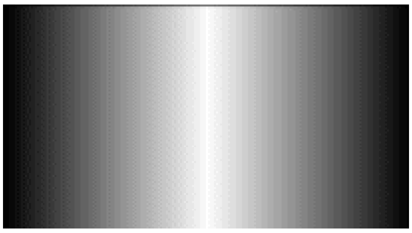
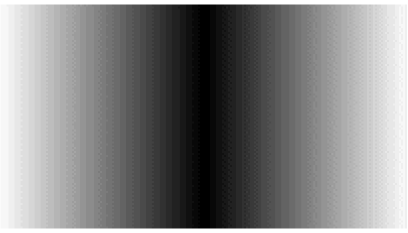
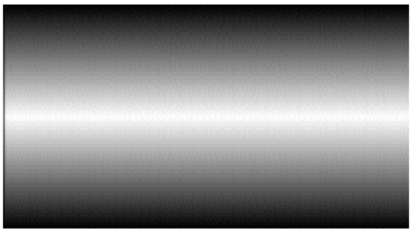
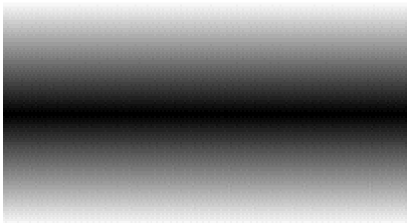
Controls the color of the gradient fill between the fill color and the gradient color. This option is available only when the **Filled** and **Gradient Fill** check boxes are selected. The gradient is updated at runtime to reflect the gradient between the two colors selected. Gradient fills support flashing colors.

Gradients do not rotate with an object; for example, if an object contains a left-to-right gradient fill and is rotated 90 degrees (either at runtime or in Graphics Builder), the gradient is still left to right.

[Gradient] Direction

The direction to be used for the gradient color. Use the table below as a guide to choose the gradient color direction you want.

Example	Gradient Color Direction
	Left to Right
	Right to Left
	Top to Bottom

	Bottom to Top
	Horizontal Gradient to Middle
	Horizontal Gradient from Middle
	Vertical Gradient to Middle
	Vertical Gradient from Middle

[Object type] Open

Defines the object as a polyline (the first point and the last point are *not* joined).

[Object type] Closed

Defines the object as a polygon (the first point and the last point *are* joined).

For help on the remaining properties tabs, see Defining Common Object Properties.

Pipe Objects

Use the Pipe tool to draw pipes with automatic three-dimensional shading. Pipes can be moved, resized, reshaped, brought to the front and so on, and their properties edited just like other types of object.

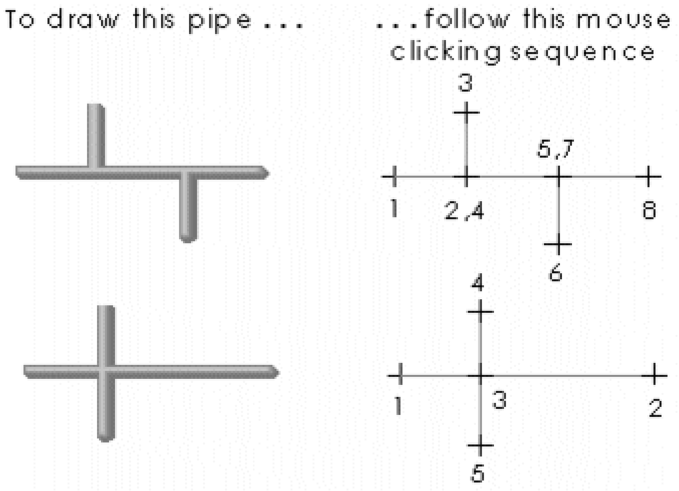
To draw a pipe:

1. Click the **Pipe** tool.
2. Move the cursor to where you want the pipe to start, and click and hold the mouse button.
3. At the end of the first line segment, release the mouse button.
4. Move the cursor to each point on the path in turn and click the mouse button (clicking and dragging is not necessary after the first segment).
5. To complete the pipe, double-click the mouse button.
   When you complete the pipe, the object properties dialog is displayed.

Hint: To draw horizontally or vertically only, hold the **Ctrl** key down when drawing the pipe.

Drawing Complex Pipe Arrangements

Use the Pipe tool to draw complex pipe arrangements (including 'T' pieces and junctions). The illustration below shows some pipes, and the sequence of mouse clicks needed to draw each of them:



Hint: Use the grid to assist in accurate positioning for each click.

See Also

Pipe Properties - Appearance (General)
Understanding Object Types

Pipe Properties - Appearance (General)

Pipes have the following general appearance properties.

[Line] Width

The width of the pipe (in pixels). You can change the width by clicking the up and down arrows to the right of the field, or by entering another value in this field. Every pipe needs to be at least 1 pixel wide.

[Line] Highlight Color

The color of the pipe where it is "in the light"; that is, the brightest color on the pipe.

[Line] Lowlight Color

The color of the pipe where it is "in shadow"; that is, the dullest color on the pipe.

For help on the remaining properties tabs, see Defining Common Object Properties.

Text Objects

Use the Text tool to type text on the page. Text can be moved, resized, reshaped, brought to the front and so on, and their properties edited just like other types of object.

To add text:

1. Click the **Text** tool.

2. Type your text on the keyboard. (Press **Enter** to start a new line.)
3. Move the cursor to where you want to position the text and click the left mouse button.
   When you click the mouse button to place the text, the object properties dialog is displayed.

See Also

Text Properties - Appearance (General)
Understanding Object Types

Text Properties - Appearance (General)

Text has the following general appearance properties

Font

The font used to display the text. Use the scroll bar to the right to view available fonts, or type the font name, or part of it, directly into this field.

Style

Select whether you would like the text to be Regular, **Bold**, **Bold Italic**, or *Italic*.

Size

Define the size of the text (point size). Available sizes might vary according to the selected printer and the selected font.

[Alignment] Left

Select this radio button to align the text to the left of the text box.

[Alignment] Right

Select this radio button to align the text to the right of the text box.

[Alignment] Center

Select this radio button to align the text in the center of the text box.

[Effect] Strikeout

Select this box to make the text will appear with a line through it.

[Effect] Underline

Select this box to underline the text.

Text

This field contains the text that will display on the page. You can enter any keyboard character(s). You can edit the text here, or directly from the graphics page. It is useful to edit text at this field, as you can apply text changes at the same time as you apply other font and color changes.

This text changes automatically depending on the Display Value properties that you define.

Foreground

The color of the text.

> **Note:** There are several radio buttons in Display Value (On/Off, Multi-state and so

on). When selected, these radio buttons change the appearance of the right hand side of the dialog.

See Also

Text Properties - Appearance (Display Value)

Text Properties - Appearance (Display Value)

Text has the following display value appearance properties. Selecting one of the following five options changes the appearance of the right hand side of the dialog.

[Type] On / Off

Changes the text which displays when a particular condition is met, and another when it is not. For example, you could display an alarm message when a particular variable tag is in alarm, and a normal message when it is not.

See Text Properties - Appearance Display Value (On/Off).

[Type] Multi-state

This option is useful when you have several possible conditions, occurring together in different combinations, at different times. Select this option to display different text for each combination.

For example, three digital variable tags (A,B, and C) can each be ON or OFF at any time. You can display a different message for each ON/OFF combination. In other words, you could display a different message for each of the following ON/OFF combinations **A**BC, A**B**C, AB**C**, **AB**C, A**BC**, **A**B**C**, **ABC**, ABC.

Text Properties - Appearance Display Value (Multi-state).

[Type] Array

Allows you to enter an expression which returns an integer. For each integer (from 0-255), you can display different text. For example, you could display a different message for each state of an analog tag.

See Text Properties - Appearance Display Value (Array).

[Type] Numeric

Displays the value of a tag or expression in numeric format (you can specify the format).

See Text Properties - Appearance Display Value (Numeric).

[Type] String

Displays the value of an expression as a string.

See Text Properties - Appearance Display Value (String).

Text Properties - Appearance Display Value (On/Off)

Text has the following On/Off Display Value properties:

ON text when

The text entered in the **ON text** field (below) appears when the condition entered here is true. The text can be a maximum of 128 characters long.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options; **Insert Tag**, and **Insert Function**.

OFF text

The text that will display whenever the condition entered above is false. You can use any keyboard character(s) to a maximum length of 256 characters.

For example, you could display the message **Conveyor 110 Normal** when **CV110_ERROR.On** is false (i.e., there is no alarm at conveyor 110).

ON text

The text that will display whenever the condition entered above is true. You can enter any keyboard character(s) to a maximum length of 256 characters.

For example, you could display the message **Conveyor 110 Alarm** when **CV110_ERROR.On** is true (i.e. there is no alarm at conveyor 110).

Click **Clear Property** to clear property details and disable the property.

See Also

Text Properties - Appearance (Display Value)
Text Properties - Appearance Display Value (Multi-state)
Text Properties - Appearance Display Value (Array)
Text Properties - Appearance Display Value (Numeric)
Text Properties - Appearance Display Value (String)

Text Properties - Appearance Display Value (Multi-state)

Text has the following multi-state display value properties:

Conditions

The conditions you enter here will occur together in different ways, at different times. You can use each different combination to determine the text that will display.

The default number of conditions is 3, but you can add more (to a maximum of 5 conditions, providing 32 combinations), using the **Add** button. You can also delete conditions using the **Delete** button, but you need to always enter a condition in this field. To enter a condition, click the relevant line (A, B, C, etc.), and click **Edit**. To insert a tag or function, click the **Wizard** button. This button displays two options: Insert Tag and Insert Function.

State text

The text that is to display for each combination of the above conditions. You can enter any keyboard character(s).

For example:

To display different messages about the status of a valve, you could fill out the **Conditions** and **State text** fields as follows:

Conditions		**State text**	
A	Open Feedback	ABC	Valve Inoperable
B	Close Feedback	**A**BC	Valve Inoperable
C	Open Output	A**B**C	Valve Closed
		AB**C**	Valve Inoperable
		ABC	Valve Inoperable
		AB**C**	Valve Open
		A**BC**	Valve Inoperable
		ABC	Valve Inoperable

In this example, **Open_Feedback** and **Close_Feedback** are variable tags representing digital inputs on the valve; **Open_Output** is a variable tag representing an output on the valve. So, ABC means **Open_Feedback** is on, and **Close_Feedback** and **Open_Output** are both off. For this combination, the text "Valve Inoperable" will display, because the valve is open when it is meant to be closed. The same type of logic applies to the rest of the states.

Click **Clear Property** to clear property details and disable the property.

See Also

Text Properties - Appearance (Display Value)
Text Properties - Appearance Display Value (On/Off)
Text Properties - Appearance Display Value (Array)
Text Properties - Appearance Display Value (Numeric)
Text Properties - Appearance Display Value (String)

Text Properties - Appearance Display Value (Array)

Text has the following Array Display Value properties:

Array expression

Enter the expression which is to return one or more integers. For each returned integer, a different piece of text is displayed.

If the return value is:

- Less than 0 (zero), it will be set to 0 (zero), and a runtime hardware alarm will be triggered.
- Greater than 255, it will be set to 255, and a runtime hardware alarm will be triggered.
- A real (non-integer) number, it will be rounded off to the nearest integer.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

Array text

The text that is to display for each integer returned by the Array expression entered above. You can enter any keyboard character(s).

For example, to display different messages about the status of a motor, you could fill out the **Array expression** and **Array text** fields as follows:

Array expression	Array text	
MOTOR_STATUS	0	Running
	1	Starting
	2	Stopping
	3	Stopped - Normal
	4	Stopped - Error
	5	Isolated

In this example, MOTOR_STATUS is an analog variable tag representing the status of a motor. When the motor changes state, an integer is returned (0 = Running, 1 = Starting etc.) and the appropriate text displays.

Click **Clear Property** to clear property details and disable the property.

See Also

Text Properties - Appearance (Display Value)
Text Properties - Appearance Display Value (On/Off)
Text Properties - Appearance Display Value (Multi-state)
Text Properties - Appearance Display Value (Numeric)
Text Properties - Appearance Display Value (String)

Text Properties - Appearance Display Value (Numeric)

Text has the following Numeric Display Value properties.

Numeric expression

The value of the expression entered here will be displayed on the graphics page. It will be formatted according to the format selected below.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert tag and Insert Function.

Format

The value returned for the expression entered above will be displayed according to the format you enter here. For example, the analog variable tag **MOTOR_STATUS** returns integers 0-5. If you enter this tag as the Numeric expression above, and enter #.## as your format, the display will alternate between **0.00**, **1.00**, **2.00**, **3.00**, **4.00**, and **5.00**. You can select a format from the drop-down list, or type in your own. If the numeric expression is a single variable, its format is overwritten by the format you enter here.

Click **Clear Property** to clear property details and disable the property.

See Also

Text Properties - Appearance (Display Value)
Text Properties - Appearance Display Value (On/Off)
Text Properties - Appearance Display Value (Multi-state)
Text Properties - Appearance Display Value (Array)
Text Properties - Appearance Display Value (String)

Text Properties - Appearance Display Value (String)

Text has the following String Display Value properties.

String expression

The value of the expression entered here will be displayed as a string on the graphics page.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert tag and Insert Function.

Click **Clear Property** to clear property details, and disable the property. To define further properties for the object, click the relevant tabs.

For help on the remaining properties tabs, see Defining Common Object Properties.

See Also

Text Properties - Appearance (Display Value)
Text Properties - Appearance Display Value (On/Off)
Text Properties - Appearance Display Value (Multi-state)

Text Properties - Appearance Display Value (Array)
Text Properties - Appearance Display Value (Numeric)

Number Objects

Use the Number tool to represent a tag or expression as a number. When you place a number on your page, simply enter the relevant variable tag or expression. Numbers can be moved, resized, brought to the front, and so on, and their properties edited just like other types of object.

(The same functionality is also available through the Text tool.)

To add a number to your graphics page:

1. Click the **Number** tool.

   ##

2. Move the cursor to where you want the number to display and click the mouse button. The Text Properties dialog box appears where you enter the relevant variable tag or expression.
   For help on the various Properties tabs, see Text Properties - Appearance (General) and Defining Common Object Properties.

Button Objects

Use the Button tool to draw buttons on graphics page. You can then assign security rights and attach commands to it.

Buttons can be moved, resized, reshaped, brought to the front, and so on, and their properties edited just like other types of object.

To draw a button:

1. Click the **Button** tool.

   ▭

2. Move the mouse to where you want the button to start and press (and hold) the mouse button.
3. Drag the mouse to where you want the button to finish and release the mouse button. When you release the mouse button, the object properties dialog is displayed.

See Also

Button Properties - Appearance (General)
Understanding Object Types

Button Properties - Appearance (General)

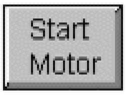
Buttons have the following General Appearance properties.

[Type] Text

Select this option to display text on the button. If you select this option, the **Text** and **Font** fields will display to the right of the dialog.

If either the **Text** or **Symbol** type option is selected, you can use XP style buttons. To configure a button to use the Windows XP style, select the **XP Style** check box. During runtime an XP style button has a blue border when it has keyboard focus, and an orange border when the mouse is on the button. When you place a button on a page, the **XP Style** check box is selected by default.

Pressing '^n' or 'Enter' wraps the text onto the next line. For example, Start^nMotor would display as:



Font

The font used to display the text. Use the scroll bar to the right to view available fonts, or type part of a font name directly into this field.

Note: The Background Color property is no longer supported for button fonts. Any imported buttons from a previous version will have the Background Color set to the default color.

Style

Select whether you would like the text to be Regular, Bold, Bold Italic, or Italic.

Size

Define the size of the text (point size). Available sizes might vary according to the selected printer and the selected font.

Custom Fill Color

Select this check box to enable the Fill Color Up and Fill Color down boxes.

Fill Color Up

The button color when in the Up state.

Fill Color Down

The button color when in the Down state.

Note: Fill Color Up and Down options available for Text and Symbol Types only.

[Alignment] Left

Select this radio button to left-align multi-line text. The aligned text will remain in the center of the button. This has no effect on single lines of text.

[Alignment] Right

Select this radio button to right-align multi-line text. The aligned text will remain in the center of the button. This has no effect on single lines of text.

[Alignment] Center

Select this radio button to center-align multi-line text. The aligned text will remain in the center of the button. This has no effect on single lines of text.

[Effect] Strikeout

Select this box to make the text appear with a line through it.

[Effect] Underline

Select this box to underline the text.

Text

This field contains the text that will display on the page. You can enter any keyboard character(s). You can edit the text here, or directly from the graphics page. It is useful to edit text at this field, as you can apply text changes at the same time as you apply other font and color changes.

This text changes automatically depending on the Display Value properties that you define.

Foreground

The color of the text.

[Type] Symbol

Select this option to display a symbol on the button. If you select this option, the **Set** button will display to the right of the dialog.

Click **Set** to choose the symbol which is to display on the button. A picture of the selected symbol will also display.

[Type] Target

When this option is selected, the button will not have any text or symbols on it, and it will have a transparent face.

Mode

There are three different modes of transparent buttons:

- **BORDER_3D**: The button is drawn with only the 3-D border (transparent face).
- **BORDER**: The button is drawn with only a thin line border.
- **TARGET**: The button is totally transparent. This constitutes a screen target.

For help on the remaining properties tabs, see Defining Common Object Properties.

Symbol Set Objects

The Symbol Set tool allows you to represent changing runtime conditions with changing symbols. By clicking on this tool, then clicking on the graphics page, you can define the symbols which are to display for each condition.

After a symbol set has been added to the page, it can be moved, re-sized, re-shaped, brought to the front etc., and its properties can be edited, just like any other type of object.

To add a Symbol Set:

1. Click the **Symbol** tool.
2. Move the mouse pointer to the desired position on the page, and click with the left mouse button.
   When you click the mouse button, the object properties dialog is displayed.
3. Fill out the relevant properties for the symbol set, and click **OK**.

> **Note:** There are several radio buttons in Symbol Sets Appearance (described below). When selected, these radio buttons change the appearance of the right hand side of the dialog.

[Type] On / Off

Select this radio button to display one symbol when a particular expression is TRUE, and another when it is FALSE. For example, you could display a red symbol when a particular variable tag is in alarm, and a green symbol when it is not.

See Symbol Set Properties - Appearance General (On/Off).

[Type] Multi-state

This option is useful when you have several possible conditions, occurring together in different combinations, at different times. Select this option to display different symbols for each combination.

For example, three digital variable tags (A,B, and C) can each be ON or OFF at any time. You can display a different symbol for each ON/OFF combination. In other words, you could display a different symbol for each of the following ON/OFF combinations ABC, **A**BC, A**B**C, AB**C**, **AB**C, A**BC**, **A**B**C**, **ABC**.

See Symbol Set Properties - Appearance General (Multi-state).

[Type] Array

The Array option allows you to enter an expression which returns an integer. For each unique integer (from 0 to 255), you can display a unique symbol. For example, you could display a different symbol for each threshold of an analog alarm.

See Symbol Set Properties - Appearance General (Array).

[Type] Animated

Select this radio button to display an actual animation (several different symbols in sequence).

When selected, the radio buttons on the dialog box change the appearance of the right hand side of the dialog. These radio buttons are only documented once below.

See Symbol Set Properties - Appearance General (Animated).

See Also

Understanding Object Types

Symbol Set Properties - Appearance General (On/Off)

Symbol Sets have the following general appearance (On/Off) properties:

ON symbol when (128 Chars.)

The symbol entered in the **ON symbol** field (below) will display whenever the condition entered here is TRUE. The symbol entered in the **OFF symbol** field (below) will display whenever the condition entered here is FALSE.

To insert a tag or a function, click the Wizard button to the right of this field. This button displays two options; **Insert tag**, and **Insert Function**.

OFF symbol

The symbol that will display whenever the condition entered above is false. Click **Set** to select a symbol, or **Clear** to clear the current selection.

For example, you could display the OFF symbol when **MIX_RUNNING** is false.

ON symbol

The symbol that will display whenever the condition entered above is true. Click **Set** to select a symbol, or **Clear** to clear the current selection.

For example, you could display the ON symbol when **MIX_RUNNING** is true.

Click **Apply** or **OK** to bring your changes into effect, or **Cancel** to discard them and exit. Click **ClearProperty** to clear property details, and disable the property. To define further properties for the object, click the relevant tabs.

See Also

Symbol Set Objects
Symbol Set Properties - Appearance General (Multi-state)
Symbol Set Properties - Appearance General (Array)
Symbol Set Properties - Appearance General (Animated)
Understanding Object Types

Symbol Set Properties - Appearance General (Multi-state)

Symbol Sets have the following general appearance (Multi-state) properties:

Conditions

The conditions you enter here will occur together in different ways, at different times. You can use each different combination to determine which symbol will display.

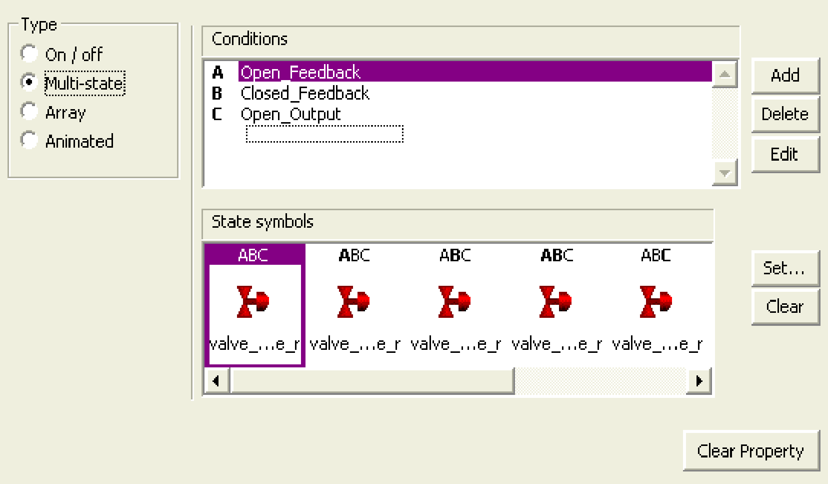
To enter a condition, click the relevant line (A, B, C, etc.), and click **Edit**. You can add more conditions (to a maximum of 5, providing 32 combinations), using the **Add** button. To insert a tag or function, click the **Wizard** button. This button displays two options; **Insert Tag** and **Insert Function**. You can also delete conditions using the **Delete** button, but there needs to always be a condition in this field. Conditions which are left black (instead of deleted) will be evaluated as false at runtime.

State symbols

The symbols that will display for each combination of the above conditions. Click the **Set** button to select a symbol, or **Clear** to clear the current selection.

For example:

To display different symbols each time the status of a valve changes, you could fill out the **Conditions** and **State symbols** fields as follows:



Type

○ On / off

● Multi-state

○ Array

○ Animated

Conditions

A Open_Feedback

B Closed_Feedback

C Open_Output

In this example, **Open_Feedback**, and **Close_Feedback** are variable tags representing digital inputs on the valve, and **Open_Output** is a variable tag representing an output on the valve. So, ABC means **Open_Feedback** is ON, and **Close_Feedback** and **Open_Output** are both OFF. For this combination, the inoperable symbol will display, because the valve is open when it is meant to be closed. The same type of logic applies to the rest of the states.

Click **Clear Property** to clear property details, and disable the property. To define further properties for the object, click the relevant tabs.

See Also

Symbol Set Objects

Symbol Set Properties - Appearance General (On/Off)

Symbol Set Properties - Appearance General (Array)

Symbol Set Properties - Appearance General (Animated)

Understanding Object Types

Symbol Set Properties - Appearance General (Array)

Symbol Sets have the following general appearance (Array) properties:

Array expression

Enter the expression which is to return one or more integers. For each returned integer, a different symbol will be displayed.

If the return value is:

- Less than 0 (zero), it will be set to 0 (zero), and a runtime hardware alarm will be triggered.

- Greater than 255, it will be set to 255, and a runtime hardware alarm will be triggered.
- A real (non-integer) number, it will be rounded off to the nearest integer.

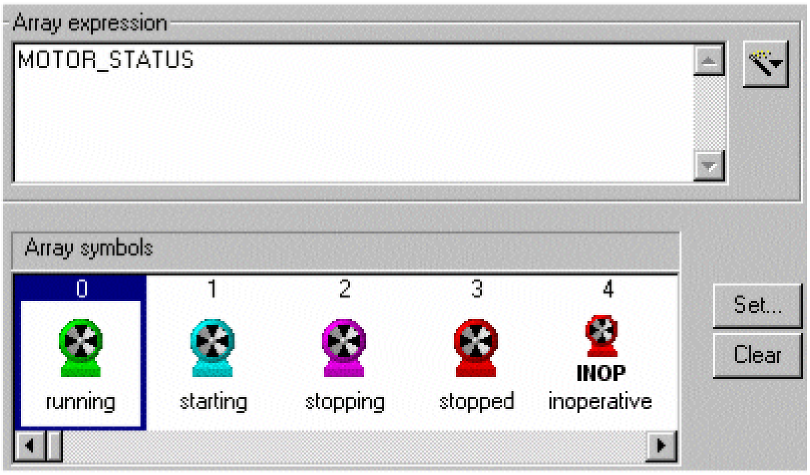
To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options; **Insert tag**, and **Insert Function**.

Array symbol

The symbol that is to display for each integer returned by the Array expression entered above (symbol 0 will be used when the expression returns integer 0, symbol 1 will be used when integer 1 is returned etc.).

Click the **Set** button to select a symbol, or **Clear** to clear the current selection.

For example, to display different symbols illustrating the various states of a motor, you could fill out the **Array expression** and **Array symbol** fields as follows:



Array expression: MOTOR_STATUS

In this example, **MOTOR_STATUS** is an analog variable tag representing the status of a motor. Each time the motor changes state, an integer is returned (0 = Running, 1 = Starting etc.), and the appropriate symbol displays.

Click **Clear Property** to clear property details, and disable the property. To define further properties for the object, click tabs.

See Also

Symbol Set Objects
Symbol Set Properties - Appearance General (On/Off)
Symbol Set Properties - Appearance General (Multi-state)
Symbol Set Properties - Appearance General (Animated)
Understanding Object Types

Symbol Set Properties - Appearance General (Animated)

Symbol Sets have the following general appearance (Animated) properties:

Animate when

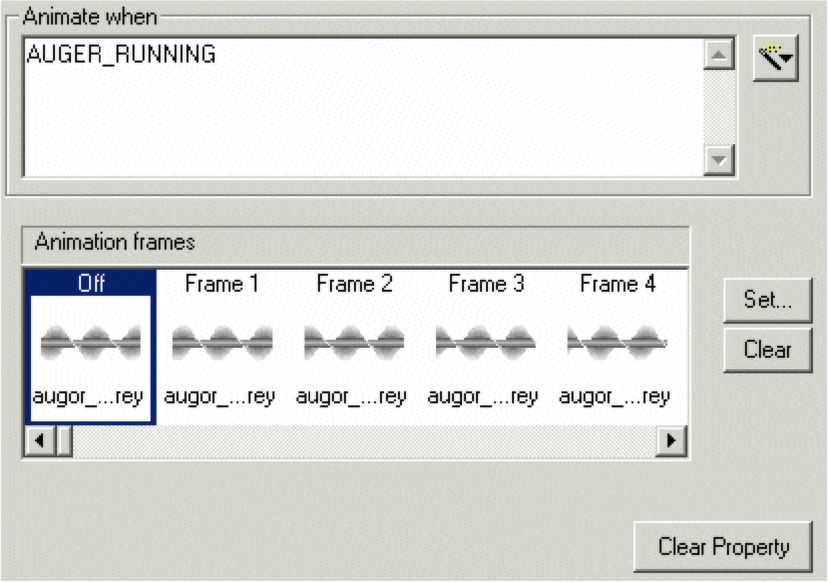
Whenever this expression is true, the animation will run. Whenever the expression is false, the Off frame (below) will display.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options; **Insert tag**, and **Insert Function**.

Animation frames

The symbols from Frame 1 onwards are those that will be used as the animation. They are displayed in sequence when the expression above is TRUE. The frequency at which the symbols are displayed is determined by the [Page]AnmDelay parameter. The symbol in the Off frame will display when the expression above is FALSE.

For example, to animate a running auger, you could fill out the **Animate when** and **Animation frames** fields as follows:



Animate when

AUGER_RUNNING

In this example, **AUGER_RUNNING**, is a variable tag which is TRUE when the auger is running. The symbols in the animation frames (Frame 1 onwards) have been designed so that when displayed in sequence, they animate a running auger. The symbol in the Off animation frame will display when **AUGER_RUNNING** is FALSE.

Click **Apply** or **OK** to bring your changes into effect, or **Cancel** to discard them and exit. Click **Clear Property** to clear property details, and disable the property. To define further properties for the object, click the relevant tabs.

For help on the remaining properties tabs, see Defining Common Object Properties.

See Also

Symbol Set Objects
Symbol Set Properties - Appearance General (On/Off)
Symbol Set Properties - Appearance General (Multi-state)
Symbol Set Properties - Appearance General (Array)
Understanding Object Types

Trend Objects

The Trend tool allows you to add a trend to the graphics page with the mouse (click and drag).

After a trend object is drawn, it can be moved, re-sized, re-shaped, brought to the front etc., and its properties can be edited, just like any other type of object.

To add a trend to a page:

1. Click the **Trend** tool

   or choose **Objects** | **Trend**.
2. Move the mouse to where you want the trend to start and click (and hold) the mouse button.
3. Drag the mouse to the opposite corner of the trend and release the mouse button. The **Trend Properties** dialog appears. Assign the Trend Tags to the pens, choosing appropriate colors.

See Also

Trend properties
Insert Trend dialog box
Understanding Object Types

Trend properties

Trends have the following general appearance properties:

Cluster Name

The name of the cluster that runs the trends being graphed. Each trend graph can only communicate with one cluster, so you cannot mix trends from multiple clusters on a single trend graph.

> **Note:** To mix trends from different clusters on a single trend graph you will need to

use Process Analyst.

If there is only one cluster, or, the client is connected to only one cluster, this property can be left blank and the value of the single connected cluster is inferred.

If the client is connected to more than one cluster, then this field needs to be specified.

Pens

The pens (including color) to be displayed on the graph (31 characters maximum). You can use up to eight pens.

Double-clicking a selected pen or clicking **Edit** allows you to change the trend tag and pen color. To insert a trend tag, click **Wizard** to display theInsert Trend dialog box.

If more than one trend tag is displayed in a trend window and each has a different sample period, the trend with the smallest sample period is used as the general display period.

Note: If the trend object is part of a group, part of a pasted Genie or symbol, or part of the page's template, you can still access its properties: hold down the **Control** (CTRL) key and double-click the object. Alternatively, choose **Goto Object** from the Tools menu, click the object, then click **OK**. Be aware, however, that if it is part of a pasted Genie or symbol, or part of the template, you cannot edit existing pens, only new ones.

If you are configuring an SPC control chart, you need to add a suffix to the trend tag to indicate the type of SPC. There are SPC templates that are easily configured through Genies. Use the Genies rather than defining these trend tags for yourself.The following table lists available SPC types:

SPC Definition	SPC Type
<tag name>.X	Mean of raw data in a subgroup (X - bar)
<tag name>.XCL	Center line of X - bar
<tag name>.XUCL	Upper control limit of X - bar
<tag name>.XLCL	Lower control limit of X - bar
<tag name>.R	Range of raw data in a subgroup (R - bar)
<tag name>.RCL	Center line of R - bar

<tag name>.RUCL	Upper control limit of R - bar
<tag name>.RLCL	Lower control limit of R - bar
<tag name>.S	Standard deviation of raw data in a subgroup (S - bar)
<tag name>.SCL	Center line of S - bar
<tag name>.SUCL	Upper control limit of S - bar
<tag name>.SLCL	Lower control limit of S - bar

where <tag name> is any trend tag, for example:

Pen 1	PIC117_PV.XCL
Pen 2	PIC117_PV.XUCL
Pen 3	PIC117_PV.XLCL
Pen 4	PIC117_PV.X

If you are using the `PageTrend()` function to display this trend page, leave these fields blank.

Display Trend Types as Periodic

When selected, enables trend pens (both periodic and event) to be displayed as periodic. Event and periodic trend data can then be displayed on the same graph. If this box is not selected, event and periodic pens have different styles and needs to be displayed on separate graphs.

> **Note:** This option is set by default in the predefined templates designed for use with periodic trends. It will only need to be enabled for customized templates.

[Samples] Number of samples (5 Chars.)

The number of samples (1-32767) you can display in your trend window without scrolling (i.e., the width of your trend object). The default depends on the number of pixels per sample and your display resolution. The width of a trend object is equal to **Pixels per sample** x **Number of samples**.

> **Note:** For a meaningful trend graph, make **Pixels per sample** x **Number of samples** less than the width of the display. For example, an XGA screen has a width of 1024

> pixels. If you use 10 pixels per sample, 102 samples can be displayed on the screen without scrolling.

[Samples] Pixels per sample (2 Chars.)

The display width of each sample. The width of a trend object is equal to **Pixels per sample** x **Number of samples**. The default is 1 pixel.

Click **Clear Property** to clear property details, and disable the property. To define other properties for the object, click the relevant tabs.

For help on the remaining properties tabs, see Defining Common Object Properties.

Insert Trend dialog box

This dialog box lets you select a trend tag. To insert a trend tag, select the tag name, then click **OK**. The tag is inserted at the location of the cursor.

See Also

Trend Objects
Understanding Object Types

Cicode Objects

The Cicode Object tool allows you to add a Cicode Object to the graphics page with the mouse (click and drag).

A Cicode Object can be any command (such as a function and so on). When the graphics page is displayed at runtime, the command is run continually. Cicode objects can also be assigned a key sequence, allowing you to enter keyboard commands when it is selected at runtime.

After a Cicode object is added, it can be moved and so on and its properties edited, just like other types of object.

To add a Cicode Object to a page:

1. Click the **Cicode Object** tool,

   f(x)

   or choose **Objects** | **Cicode Object**.
2. Move the mouse to where you want to add the object, and click the left mouse button.
3. Define the relevant properties for the object, and click **OK**.

See Also

Cicode Object Properties - Cicode (General)

Understanding Object Types

Cicode Object Properties - Cicode (General)

Cicode Objects have the following General properties:

Command (254 Chars.)

A Cicode command that is continually executed. You can use any Cicode command, built-in Cicode function or user-written function. The command is executed continually (while the page is displayed), for example:

Command	DspSymAnm(25, "Pumps.Slurry1", "Pumps.Slurry2", "Pumps.Slurry3");

The command in this example uses the built-in function DspSymAnm(). The function displays three symbols ("Pumps.Slurry1", "Pumps.Slurry2", "Pumps.Slurry3") continually (at AN 25).

You can also write generic functions by using the Cicode function DspGetAnCur() to get the AN number, for example:

Command	DspSymAnm(DspGetAnCur(), "Pumps.Slurry1", "Pumps.Slurry2", "Pumps.Slurry3");

The command in this example displays three symbols ("Pumps.Slurry1", "Pumps.Slurry2", "Pumps.Slurry3") continually (at the current AN).

If you are using an actual animation, each symbol is displayed at a frequency that is set using the Computer Setup Wizard (also determined by the `[Page] AnmDelay` parameter). To add just an animation point to the page, add a Cicode Object, without a command.

Click **Clear Property** to clear property details, and disable the property. To define other properties for the object, click the relevant tabs.

For help on the remaining properties tabs, see Defining Common Object Properties.

Pasted Symbol Objects

The Paste Symbol tool allows you to insert a symbol from a library onto the graphics page.

After a symbol is pasted using this tool, it can be moved, resized, reshaped, brought to the front and so on, and its properties edited just like any other type of object.

Pasted [library] symbols can be linked to their source, so that changes made to the original are inherited by the pasted symbol.

To display the properties of the objects in the symbol (after pasting), hold the **Control** (CTRL) key down and double-click the specific object. Alternatively, choose **Tools | Goto Object**, click the object, then click **OK**.

To learn more about creating symbols, see Using libraries.

To paste a symbol from the library to the page:

1. Click the **Paste Symbol** tool or choose **Edit | Paste Symbol.**
2. To paste a linked symbol, select the **Linked** check box. To paste an unlinked symbol, deselect the **Linked** check box.

To break the link:

1. Select the symbol.
2. Choose **Edit | Cut Link.**

See Also

Paste Symbol dialog box
Symbol Properties - Appearance (General)

Paste Symbol dialog box

This dialog box lets you paste a symbol from a library to the graphics page (or template).

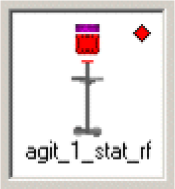
The Paste Symbol dialog box has the following properties:

Symbol

A table of symbols in the project.

To add a symbol to a graphics page, use the scroll bar to locate the thumbnail image of the symbol, then select it and click **OK** (or double-click the thumbnail image). To edit the object in the library, select it and click **Edit**. To create a new symbol, click **New**.

Note: If the symbol has a small diamond-shaped badge next to it, it indicates it is a flashing symbol (see example below.)



Library

The library where the symbol is stored.

Linked

To paste a symbol that maintains the link with its library, check this box. A symbol that is linked will be automatically updated if the symbol in the library is changed.

You can cut the link at any time with the **Cut Link** command from the Edit menu, but you cannot re-link a symbol with the library after the link has been cut.

If you have selected **Paste Symbol as Flashing**, two dialog boxes appear in sequence, allowing you to choose two images that you want to implement as a flashing symbol. The **Primary Select Symbol** dialog allows you to select the initial image used, the **Flashing Select Symbol** dialog the second image. If the bitmaps are different in size, the flashing symbol is scaled to the size of the primary image. If one of the symbols is itself a flashing symbol, only the primary state will be displayed.

Symbol Properties - Appearance (General)

This dialog displays a picture of the selected symbol, name, and path. Click **Set** to change the symbol, or double-click the image. The Select Symbol dialog appears, letting you select a new symbol.

For help on the remaining properties tabs, see Defining Common Object Properties.

To change the properties of an object in a pasted Symbol:

1. Click the **Select** tool.
2. Hold down the Control (CTRL) key and double-click the object.
3. Change the relevant properties in the dialog box. Alternatively, select **Tools** | **Goto Object**, select the object, and then click **OK**.

See Also

Pasted Genie Objects
Understanding Object Types

Pasted Genie Objects

The Paste Genie tool allows you to insert a Genie onto the graphics page.

After a Genie is pasted using this tool, it can be resized, rotated, moved, copied, duplicated, pasted, brought to the front, and so on.

To display the properties of the objects in the Genie (after pasting), hold the **Control** (CTRL) key down and double-click the specific object.

ActiveX Objects

You can incorporate ActiveX objects into your project. This means you can use components that have been developed independently of CitectSCADA.

The ActiveX tool can be used to insert ActiveX objects in graphics pages. After you have selected and positioned an ActiveX object, it can be moved, re-sized, re-shaped, brought to the front etc., and its properties can be edited, just like any other object.

Managing associated data sources

If an ActiveX object has an association with a data source (for example, it stores data to a DBF file), you need to consider the impact of running a project that contains it on a different machine or via one of the Internet clients (Internet Display Client or Web Client).

If the path to the data source is hardcoded to a location on the local machine, the data source will not be found if the project is moved or run remotely. For example, the Database Exchange ActiveX control connects to a recipe.DBF file in the project path. If you restore a project that uses it on a different computer with a different installation path, you will need to recreate the data source to retrieve any recipes.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Whenever possible, avoid programming the literal paths to data sources in your project.
- When a project or a data source is relocated to a different computer, examine the program and program objects and correct any literal paths before placing the project back into service.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

One way to avoid having to recreate data sources is to locate any associated data sources in a central location on a network. For example, if the data source is located on a SQL server, it will be accessible from every machine on the common network.

To insert an ActiveX Control:

1. Click the **ActiveX** tool,

   or choose **Edit | Insert ActiveX Control**.
2. Select an ActiveX Control and click **Insert**.

See Also

ActiveX Object Properties
Tag Association
Object Identification

ActiveX Object Properties

The two properties tabs common to ActiveX objects are the **Tag Association** and **Visibility** tabs which appear vertically on the **Appearance** tab. The content and number of other tabs depends on the individual design of each ActiveX object. This is determined by the amount of flexibility and support its creator has included.

For details on configuring these additional tabs, refer to the documentation provided with the ActiveX object.

See Also

Tag Association

Tag Association

You can create an association between a property of an ActiveX object and a variable tag.

To create an association between a property of an ActiveX object and a variable tag:

1. Double-click the ActiveX object. The Properties dialog box appears.
2. Click the **Tag Association** tab.
3. Select a property from the **Properties** list.
4. Click the **Wizard** | **Insert Tag** button.
5. Select a tag from the list and click **OK**.

See Also

"ActiveX Object Properties" in CitectSCADA User Guide
"Understanding Object Types" in CitectSCADA User Guide

ActiveX Object Properties - Appearance (Tag Association)

ActiveX objects have the following appearance properties:

Properties

The "properties" of an ActiveX object relate to the elements that define the object's functionality and appearance. The available properties for a selected ActiveX object are listed here, as defined by the object's creator.

The check box to the left of each item on the list indicates whether or not a tag has been associated with the Property. If the box is checked, it means an association has already been defined for the property. If you want to clear this tag association, simply uncheck the box, or clear the tag from the "Associate property with tag. . ." field.

Associate property with tag

You can create an association between an ActiveX object property and a variable tag so that changing values in one are reflected in the other. To create an association, you need to first choose a property from the property list. The label **Associate property with tag** will change to display the property name.

Select the variable tag you would like to associate with a property by clicking on the Wizard button and choosing from the list of available tags. Alternatively, the tag name can be typed in to the **Associate property with tag** field.

Note: You can only use variable tag names in this field. Functions, expressions and constants are not supported when defining ActiveX property tag associations.

You can only associate a property with a variable tag if the tag type is compatible with the property. To display a list of compatible tag types, select the property, and click **List Property Types**. A list of compatible data types will display, or a message will inform you that there are no compatible types.

If there are no types compatible with the property, the **Associate Property with tag** and **Update association on** fields will be disabled.

If you specify a tag which might be inappropriate for the selected ActiveX property, the **Type Evaluation dialog** will display an alert. This might happen if:

- The types compatible with the property are different from the tag type.
- The tag type is smaller than the types compatible with the property, meaning that data could be truncated or lost.

Bindable

If an object property is "bindable", it means it can automatically send notification of value changes to CitectSCADA, and acknowledge any value changes from an associated tag. This means both the property and an associated tag will automatically update whenever the value for either changes.

If a property is not bindable, the property/tag association can only be synchronized according to the event selected in the **Update association on Event** field.

Be aware that if an object property is bindable, the **Update association on Event** field will be automatically set to **<Property change notification>** by default. You can change this setting if you want the tag association to be updated by a more specific event.

For properties that are not bindable, you can mimic the behavior of <property change notification> by selecting **After update** for the **Update association on Event** field.

Property status

This indicates the read/write status of the selected property. With ActiveX objects, some properties are read-only, whereas others accept value changes. If a property is marked "read only", it indicates that its value is fixed and can only be read by CitectSCADA. If its status is read/write, can modify the property during runtime via a tag association.

Update association on Event:

This defines when you want a value update to occur for the selected property and its associated tag. Use the menu to the right of the **Event** field to view events that can be used to trigger a tag association update.

The available events that can be used with a particular property are predefined by an object's creator. They typically include user interaction events (for example, mouse clicks), time events (such as a new day or new month), or value changes (such as "after update").

Property documentation

Most ActiveX objects have documentation describing the object's controls and functionality. Some have a separate Help file included, others, simple text prompts to explain each property. This depends on what an object's creator has included.

The Property documentation field displays Help information for a selected property, or give instructions to obtain the Help necessary for a selected property. Usually the following message will appear:

"Click the '?' button to the right to display the Help topic for this property"

The **Help** button displays the ActiveX object Help file (if included), usually with the topic displayed that relates to the selected property. It will also provide information about settings provided on additional tabs the ActiveX object might call up on the properties dialog.

Clear Property

The Clear Property button clears the tag associations for each of the ActiveX object properties. To clear a tag association for a particular object property, clear the check box to the left of the property.

If you accidentally click **Clear Property**, you can restore your tag associations by clicking **Cancel** and reopening the ActiveX properties dialog.

Object Identification

You can identify your ActiveX object.

To identify your ActiveX object:

1. From Graphics Builder, double-click the ActiveX object. The Properties dialog appears.
2. Click the **Access** tab.
3. Click the **Identification** tab.
4. **Assign** your ActiveX object a name in the **Object Name** field.
5. **Assign** your ActiveX object an **Event Class**.
6. Click **OK**.

See Also

Object Properties - Access (Identification)

Object Properties - Access (Identification)

Objects have the following Access Identification properties:

[Identification] Object Name

This field allows you to assign a name to your ActiveX object (254 characters maximum). It identifies the object when using the `ObjectByName()`, `ANByName()`, and `CreateControlObject()` Cicode functions.

The name can be any combination of alpha or numeric characters.

[Identification] Event Class

Allocate a name for the event class of your ActiveX object (16 characters). You can then use this name to create a Cicode function to trap an event.

Don't change the default value if you want to access the ActiveX object using CitectVBA. If you do, CitectVBA can't access the object. If the Event Class is changed, you can reset it to the default value by clicking **Clear Property**.

[Persistence] Persist ActiveX data between page transitions

Select this check box to allow changes made to the control to be persisted between pages. For example, you can select this check box if you want an ActiveX edit control to keep the current text in the control, so that next time the page is entered, the same text is displayed.

> **Note:** An ActiveX control may or may not implement data persistence in the manner you expect. Some controls will not persist certain data, and therefore you will not be able to save and restore that data.

Note: Even when an ActiveX control implements data persistence in an expected manner, be advised that data is only persisted for the current session. If CitectSCADA runtime is shut down and restarted the updated data is not available.

Database Exchange Control Objects

The Database Exchange ActiveX control lets you connect to a data source (for example a database), and extract, display, and edit recipe values. The control is inserted and configured on a graphics page using the Citect Graphics Builder. For more information see the Database Exchange online help.

Chapter: 21 Defining Common Object Properties

This section describes properties that are common to several object types. Some are also common to object groups.

See Also

3D Effects
Visibility
Movement
Scaling
Fill Color
Fill Level
Touch Commands
Keyboard Commands
Sliders
Access

3D Effects

You can apply 3D effects to objects to make them more realistic.

To apply 3D effects to an object:

1. Draw the object/group (or paste a symbol). The properties tab dialog will automatically display, unless you have turned off the Display properties on new option in the Graphics Builder. (For a group, the properties dialog will not display automatically; you need to double-click the group).
2. Click the **Appearance** tab.
3. Click the **3D Effects** tab (to the right of the dialog).
4. The dialog will then display several options to enable you to manipulate your object. To activate any of these options click the option (or the radio bullet to the left of the option).
5. By selecting certain options additional fields will display to enable you to further manipulate your object. Enter further details as necessary, using the Help button for detailed information about each field.
6. Click **OK**.

See Also

Object Properties - Appearance (3D Effects)
Defining Common Object Properties

Object Properties - Appearance (3D Effects)

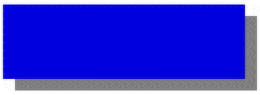
Objects have the following 3D Effects properties.

[Effects] None

Select this option to display the object without any special effects (such as shadowing, embossing and so on).

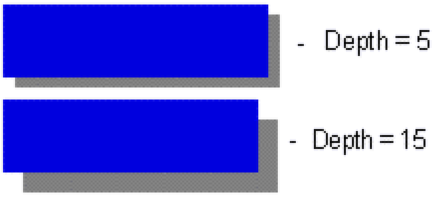
[Effects] Shadowed

Select this option to display the object with a shadow; for example:



Depth

The distance (in pixels) that the shadow extends below and to the right of the object. This option alters the apparent distance between the object and its shadow, for example:

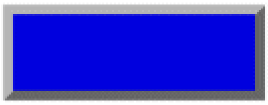


Shadow color

The color of the shadow. The shadow color will not change dynamically with runtime conditions.

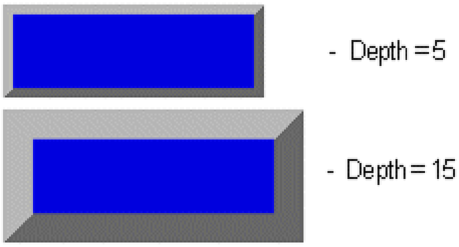
[Effects] Raised

Select this option to display the object as a raised three dimensional solid, for example:



Depth

The distance (in pixels) that the sides of the object extend out from the raised surface. This option alters the apparent distance from the raised surface down to your graphics page, for example:



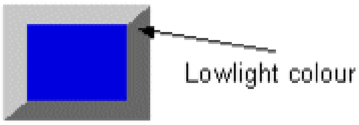
Highlight color

The color of the directly illuminated "edges" of the object.

Highlight colour

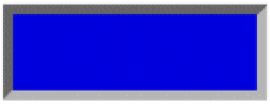
Lowlight color

The color of the "edges" of the object that are in shadow.



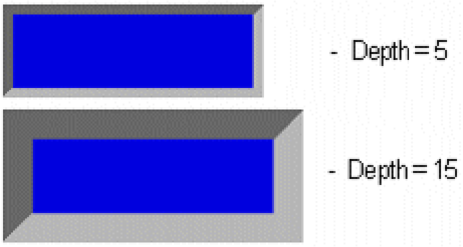
[Effects] Lowered

Select this option to display the object as if it is actually lower than your graphics page, for example:



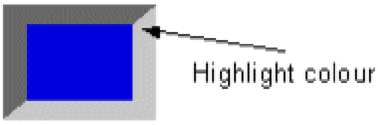
Depth

The distance (in pixels) that the sides of the object extend out from the lowered surface. This option alters the apparent distance from the lowered surface up to your graphics page, for example:



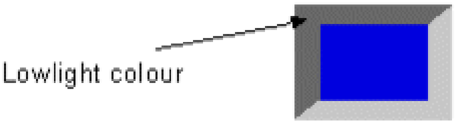
Highlight color

The color of the directly illuminated "edges" of the object.



Lowlight color

The color of the "edges" of the object that are in shadow.



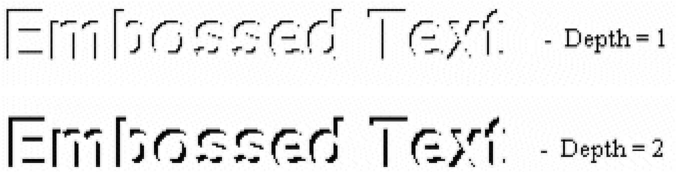
[Effects] Embossed

Select this option to display the object as if it has been embossed on your graphics page, for example:

Embossed Text

Depth

The distance (in pixels) that the embossed surface is lowered. This option alters the apparent distance from the embossed surface up to your graphics page, for example:



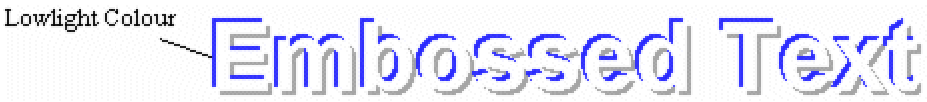
Highlight color

The color of the right and lower edges of the object.

Embossed Text
Highlight Colour

Lowlight color

The color of the left and upper edges of the object.



Click **Apply** or **OK** to save your changes, or **Cancel** to exit. To define further properties for the object, click the relevant tabs.

Visibility

You can determine whether an object is visible or not.

To hide/unhide an object:

1. Double click the object you would like to hide.
2. Select the **Appearance** tab.
3. Select the **Visibility** tab (to the right of the dialog).
4. Click the **Wizard** button to the right of the **Hidden when** field.
5. Select either **Insert Tag** or **Insert Function** depending on which you would like to relate to your object.
6. Enter an expression in the **Hidden when** field. When this expression is true your object will be hidden.
7. Click **OK**.

See Also

Object Properties - Appearance (Visibility)
Defining Common Object Properties

Object Properties - Appearance (Visibility)

Objects and groups have the following visibility properties:

Hidden when

The object/group will be hidden whenever the expression entered in this field is TRUE. Enter an expression of 128 characters or less. For example, if you want the object/group to be hidden for every operator except the superintendent, you could enter the following:

```
NOT GetPriv( _Super, _SectionA )
```

where `_Super` and `_SectionA` are labels.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

> **Note:** If a group is hidden, every object (and other groups) in the group will also be hidden regardless of their individual properties. If the group is visible, its objects will behave according to their individual properties.

Click **Clear Property** to clear property details and disable the property.

Movement

You can control the movement of objects.

To configure an object or group that moves:

1. Draw the object/group (or paste a symbol). The properties tab dialog automatically display, unless you have turned off the **Display properties on new** option in the Graphics Builder. (For a group, the properties dialog will not display automatically; you need to double-click the group.)
2. Click the **Movement** tab.
3. Click the **Horizontal**, **Vertical** or **Rotational** tab (to the right of the dialog).
4. Enter a Movement **expression** (the expression that will move the object/group at run-time).
5. Enter further details as necessary, using the **Help** button for detailed information about each field.
6. Click **OK**.

See Also

Object Properties - Movement (Horizontal)
Object Properties - Movement (Vertical)
Object Properties - Movement (Rotational)
Group and object movement - examples

Object Properties - Movement (Horizontal)

Objects and groups can be moved from side to side during runtime, changing dynamically whenever the value of a particular expression changes. By default, as the value of the expression increases, the object/group will move (in increments) to the right. As the value of the expression decreases, the object/group will move (in increments) to the left.

This property could, for example, be used to display the position of a coal stacker moving along a stockpile.

> **Note:** Horizontal movement cannot be used if the horizontal slider is enabled. A group and its objects can be configured with any movement combination (i.e., a group can move vertically while one of its objects rotates, and so on).

Objects and groups have the following horizontal movement properties:

Movement expression

The value of the expression entered in this field (253 characters maximum) will determine the horizontal movement of the object/group. By default, when the expression returns its minimum value, the object/group will shift hard to the left. When the expression returns its maximum, the object/group will shift hard to the right. For intermediate values, the object/group will move to the appropriate position between the minimum and maximum offset.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Movement expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the movement expression, rather than using the default values. For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Movement expression] Minimum

Enter the minimum value for the expression. When this value is returned by the expression, the object/group will shift to the left, by the **At minimum** offset. You can only enter a value here if you have selected the **Specify** range box.

[Movement expression] Maximum

Enter the maximum value for the expression. When this value is returned by the expression, the object/group will shift to the right, by the **At maximum** offset. You can only enter a value here if you have selected the **Specify** range box.

[Offset] At minimum

The distance (number of pixels from the original object/group center) that the object/group will shift to the left when the **Movement expression** returns its minimum value.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

[Offset] At maximum

The distance (number of pixels from the original object/group center) that the object/group will shift to the right when the **Movement expression** returns its maximum value.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

Note: You can shift the object/group right at minimum and left at maximum, by entering negative distances in the Offset fields, or by swapping the expression limits (in the **Minimum** and **Maximum** fields).

Click **Clear Property** to clear property details, and disable the property.

See Also

Object Properties - Movement (Vertical)
Object Properties - Movement (Rotational)

Object Properties - Movement (Vertical)

Objects and groups can be moved up and down during runtime, changing dynamically whenever the value of a particular expression changes. By default, as the value of the expression increases, the object/group will move up (in increments). As the value of the expression decreases, the object/group will move down (in increments).

This property could be used to display the movement of an elevator.

Note: Vertical Movement cannot be used if the Vertical Slider is enabled. A group and its objects can be configured with any movement combination (i.e. a group can move vertically while one of its objects rotates, and so on).

Objects and groups have the following vertical movement properties:

Movement expression

The value of the expression entered in this field (253 characters maximum) will determine the vertical movement of the object/group. By default, when the expression returns its minimum value, the object/group will shift down to its lowest position. When the expression returns its maximum, the object/group will shift up to its highest position. For intermediate values, the object/group will move to the appropriate position between the minimum and maximum offset.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Movement expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the movement expression, rather than using the default values. For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Movement expression] Minimum

Enter the minimum value for the expression. When this value is returned by the expression, the object/group will shift down, by the **At minimum** offset. You can only enter a value here if you have selected the **Specify** range box.

[Movement expression] Maximum

Enter the maximum value for the expression. When this value is returned by the expression, the object/group will shift up, by the **At maximum** offset. You can only enter a value here if you have selected the **Specify** range box.

[Offset] At maximum

The distance (number of pixels from the original object/group center) that the object/group will shift up when the **Movement expression** returns its maximum value.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

[Offset] At minimum

The distance (number of pixels from the original object/group center) that the object/group will shift down when the **Movement expression** returns its minimum value.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

> **Note:** You can shift the object/group up at minimum and down at maximum, by entering negative distances in the Offset fields, or by swapping the expression limits (in the **Minimum** and **Maximum** fields).

See Also

Object Properties - Movement (Horizontal)
Object Properties - Movement (Rotational)

Object Properties - Movement (Rotational)

Objects and groups can be dynamically rotated during runtime, whenever the value of a particular expression changes. By default, as the value of the expression increases, the object/group will rotate clockwise (in increments). As the value of the expression decreases, the object/group will rotate anti-clockwise (in increments).

This property could be used to display an aerial view of the movement of a circular stacker in a coal mining operation.

> **Note:** Rotational Movement cannot be used if the Rotational Slider is enabled. A

> group and its objects can be configured with any movement combination (i.e. a group can move vertically while one of its objects rotates, and so on).

Objects and groups have the following rotational movement properties:

Angle expression(253 Chars.)

The value of the expression entered in this field will determine the rotation of the object/group. During runtime, when the expression returns its minimum value, the object/group will rotate to its anti-clockwise limit. When the expression returns its maximum, the object/group will rotate to its clockwise limit. For intermediate values, the object/group will rotate to the appropriate position between the minimum and maximum limits.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Angle expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the angle expression, rather than using the default values. For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Angle expression] Minimum

Enter the minimum value for the expression. When this value is returned by the expression, the object/group will rotate anti-clockwise, by the minimum offset. You can only enter a value here if you have selected the **Specify** range box.

[Angle expression] Maximum

Enter the maximum value for the expression. When this value is returned by the expression, the object/group will rotate clockwise, by the maximum offset. You can only enter a value here if you have selected the **Specify** range box.

[Angle] At minimum

The anti-clockwise angle (in degrees relative to 0 degrees) that the object/group will rotate when the **Angle expression** returns its minimum value.

You can change the angle by pressing the up and down arrows to the right of the field, or by entering another value in this field.

[Angle] At maximum

The clockwise angle (in degrees relative to 0 degrees) that the object/group will rotate when the **Angle expression** returns its minimum value.

You can change the angle value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

> **Note:**You can rotate the object/group clockwise at minimum and anti-clockwise at maximum, by entering negative angles in the Angle fields, or by swapping the expression limits (in the **Minimum** and **Maximum** fields).

[Center axis offset] Express

Click this radio button for the quick and easy way of selecting the point about which the object/group will rotate. The express option gives you the choice of 9 points (Top Left, Bottom Right and so on), which are displayed in the picture field to the right of the dialog. To select one, just click it with your mouse.

[Center axis offset] Custom

Click this radio button to define your own center axis. When you select this radio button, two fields will display to the right, allowing you to plot the position of your center axis. Specify the distance to the right in the first field, and the distance down in the second. The Center axis is plotted based on these two values.

For example, if you enter 8 as the horizontal offset, and 13 as the vertical offset, the Center axis will be 8 pixels to the right, and 13 pixels below the center of the object/group.

Enter negative values in the offset distance fields to move the Center axis left instead of right, and up instead of down.

See Also

Object Properties - Movement (Horizontal)
Object Properties - Movement (Vertical)

Group and object movement - examples

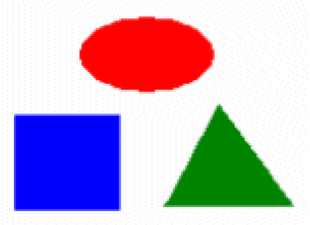
A group and its objects can be configured with any combination of movement (horizontal, vertical, and rotational). The following examples illustrate how some of these combinations work.

Example 1: Rotating the group and moving the object left to right

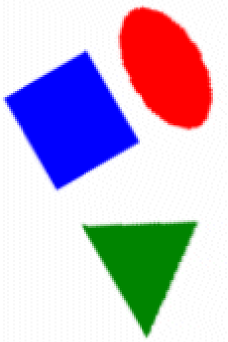
If your group is configured to rotate from 0 degrees to 60 degrees, and one of its objects is configured to move left and right, the object will do both. It will move left and right as per its own properties, and, at the same time, it will rotate as part of the group. Remember, however, that 'left' and 'right' are relative to the original orientation of the group, not the page. As the group rotates, 'horizontal' also rotates. When the group has rotated 15 degrees, 'left' is actually 285 degrees; (not 270 degrees), and 'right' is actually

105 degrees (not 90 degrees). When the group has rotated 50 degrees, 'left' is 320 degrees, and 'right' is 140 degrees, and so on.

Original state of group



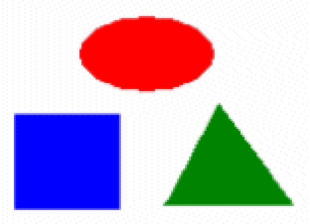
Group rotated right, and ellipse moved left



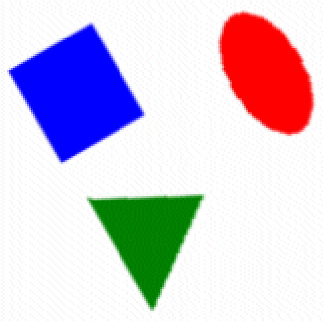
Example 2: Rotating the group and moving the object up and down

If your group is configured to rotate from 0 degrees to 60 degrees, and one of its objects is configured to move up and down, the object will do both. It will move up and down as per its own properties, and, at the same time, it will rotate as part of the group. Remember, however, that 'up' and 'down' are relative to the original orientation of the group, not the page. As the group rotates, 'vertical' also rotates. When the group has rotated 15 degrees, 'up' is actually 15 degrees (not 0 degrees), and 'down' is actually 195 degrees (not 180 degrees). When the group has rotated 50 degrees, 'up' is 50 degrees, and 'down' is 230 degrees, and so on.

Original state of group



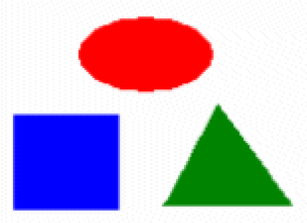
Group rotated right, and ellipse moved up



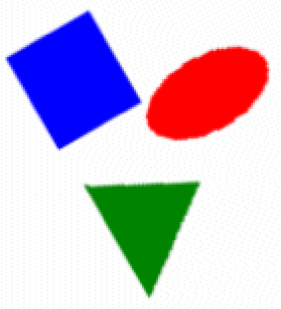
Example 3: Rotating the group clockwise and rotating the object anticlockwise

If your group is configured to rotate from 0 degrees-60 degrees, and one of its objects is configured to rotate from 90 degrees-0 degrees, the object will do both. It will rotate as per its own properties, and, at the same time, it will rotate as part of the group. Remember, however, that the object's rotation is relative to the group, not the page. If the group rotates 60 degrees to the right, and the object rotates 90 degrees to the left, the object has only rotated 30 degrees to the left relative to the page.

Original state of group



Group rotated clockwise, and ellipse rotated anticlockwise



> **Note:** By moving the ellipse as shown in the above examples, you are actually changing the overall size of the group. It is important to remember this as it might affect object fill levels.

See Also

Movement

Defining Common Object Properties

Scaling

You can scale objects to the size you want.

To configure an object or group that changes size:

1. Draw the object (or paste a symbol). The object properties tab dialog will automatically display, unless you have turned off the **Display properties on new** option in the Graphics Builder.
2. Click the **Scaling** tab.
3. Click the **Horizontal** or **Vertical** tab (to the right of the dialog).
4. Enter a **Scaling expression** (the expression that will change the size of the object at runtime).
5. Enter further object property details as necessary, using the **Help** button for detailed information about each field.
6. Click **OK**.

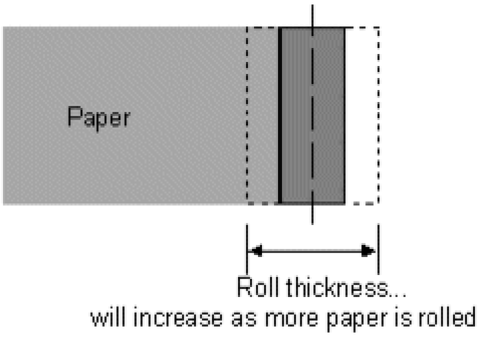
See Also

Object Properties - Scaling (Horizontal)
Object Properties - Scaling (Vertical)
Defining Common Object Properties

Object Properties - Scaling (Horizontal)

The width of an object can be dynamically changed during runtime whenever the value of a particular expression changes. As the value of the expression increases and decreases, the width of the object increases or decreases accordingly as a percentage of the original width; that is, when it was added to the graphics page.

For example, an aerial view of a paper roll (in a paper mill), could display changing roll thickness:



Objects have the following horizontal scaling properties:

Scaling expression

The value of the expression entered in this field (253 characters maximum) will determine the horizontal scaling (width) of the object. By default, when the expression returns its minimum value, the object will display at its minimum width (as defined in the Scaling fields below). When the expression returns its maximum value, the object will display at its maximum width (as defined in the Scaling fields below).

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Scaling expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the scaling expression, rather than using the default values. For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Scaling expression] Minimum

Enter the minimum value for the expression. When this value is returned by the expression, the width of the object will be reduced to its minimum. You can only enter a value here if you have selected the **Specify** range box.

[Scaling expression] Maximum

Enter the maximum value for the expression. When this value is returned by the expression, the width of the object will be increased to its maximum. You can only enter a value here if you have selected the **Specify** range box.

[Scaling] At minimum

The minimum width of the object (as a percentage of its original width). The object will be reduced to this width when the **Scaling expression** returns its minimum value.

You can change the percentage by pressing the up and down arrows to the right of the field, or by entering another value in this field. Percentages of greater than 100% can be entered.

[Scaling] At maximum

The maximum width of the object (as a percentage of its original width). The object will grow to this width when the **Scaling expression** returns its maximum value.

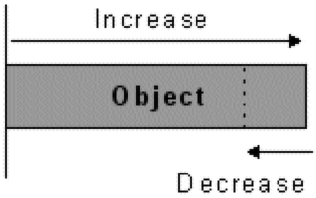
You can change the percentage by pressing the up and down arrows to the right of the field, or by entering another value in this field. Percentages of greater than 100% can be entered.

Note: You can increase the width at minimum, and decrease it at maximum, by swapping the percentages in the Scaling fields (i.e. put the high percentage in the **At minimum** field, and the low in the **At maximum**), or by swapping the expression limits (in the **Minimum** and **Maximum** fields).

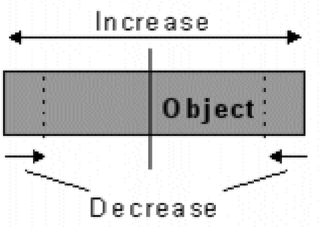
[Center axis offset] Express

Click this radio button for the quick and easy way of selecting one of three of the object's vertical axes (left, center, and right). These axes appear in the picture field to the right of the dialog.

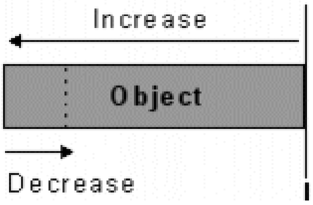
If you choose **Left**, width changes occur to the right of the object. (i.e., the left edge remains anchored):



If you choose **Center**, width changes occur equally to both sides. (i.e., the vertical center axis remains anchored):



If you choose **Right**, width changes occur to the left of the object. (i.e., the right edge remains anchored):



[Center axis offset] Custom

Click this radio button to define your own center axis. A field appears to the right of the dialog allowing you to specify how far from the object center (in pixels) you would like to place the axis. Although this option gives you the option to place the center axis anywhere on the object, once placed, the scaling process works in exactly the same manner as for the Express option (illustrated above).

For example, if you enter 20, the Center axis will be 20 pixels to the right of the object center.

Enter a negative value to move the center axis left instead of right.

> **Note:** If a group and its objects are configured to change size during runtime, the group scaling effects will be combined with the object scaling effects. For example, if a group is configured to double in size at runtime, and one of its objects is configured to halve in size, the object will appear to remain the same size (it halves, then doubles). Remember, however, that the object's position might change as the group gets bigger.

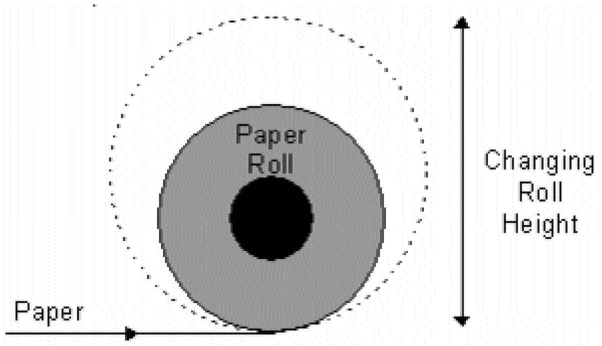
See Also

Object Properties - Scaling (Vertical)

Object Properties - Scaling (Vertical)

You can change the height of an object during runtime. As the value of the expression increases or decreases, the height of the object increases or decreases accordingly as a percentage of the original height; that is, when the object was added to the graphics page.

For example, an elevation of a paper roll (in a paper mill), could display changing roll height (and width):



Objects have the following vertical scaling properties:

Scaling expression

The value of the expression entered in this field (253 characters maximum) will determine the vertical scaling (height) of the object. By default, when the expression returns its minimum value, the object will display at its minimum height (as defined in the Scaling fields below). When the expression returns its maximum value, the object will display at its maximum height (as defined in the Scaling fields below).

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Scaling expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the scaling expression, rather than using the default values. For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Scaling expression] Minimum

Enter the minimum value for the expression. When this value is returned by the expression, the height of the object will be reduced to its minimum. You can only enter a value here if you have selected the **Specify** range box.

[Scaling expression] Maximum

Enter the maximum value for the expression. When this value is returned by the expression, the height of the object will be increased to its maximum. You can only enter a value here if you have selected the **Specify** range box.

[Scaling] At minimum

The minimum height of the object (as a percentage of its original height). The object will be reduced to this height when the **Scaling expression** returns its minimum value. You can change the percentage by pressing the up and down arrows to the right of the field, or by entering another value in this field. Percentages of greater than 100% can be entered.

[Scaling] At maximum

The maximum height of the object (as a percentage of its original height). The object will grow to this height when the **Scaling expression** returns its maximum value.

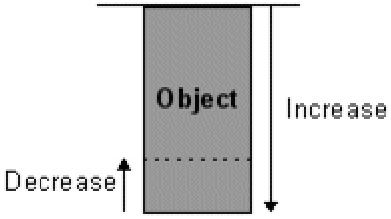
You can change the percentage by pressing the up and down arrows to the right of the field, or by entering another value in this field. Percentages of greater than 100% can be entered.

Note: You can increase the height at minimum, and decrease it at maximum, by swapping the percentages in the Scaling fields (i.e. put the high percentage in the **At minimum** field, and the low in the **At maximum**), or by swapping the expression limits (in the **Minimum** and **Maximum** fields).

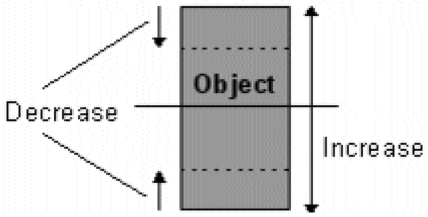
[Center axis offset] Express

Click this radio button to select one of three of the object's horizontal axes (top, middle, and bottom). These axes appear in the picture field to the right of the dialog. Click an axis to select it.

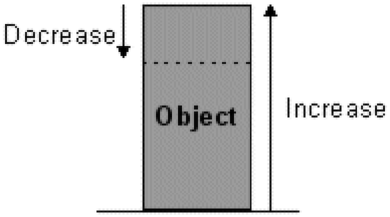
If you choose the top, height changes will occur from the top of the object down. (i.e. the top edge will remain anchored):



If you choose the middle, height changes will occur equally above and below the axis. (i.e. the horizontal center axis will remain anchored):



If you choose the bottom, width changes will occur to the top edge of the object. (i.e. the bottom edge will remain anchored):



[Center axis offset] Custom

Click this radio button to define your own center axis. A field will display to the right of the dialog, allowing you to specify how far from the object center (in pixels) you would like to place the axis. Although this option gives you the freedom to place the center axis anywhere on the object, once placed, the scaling process works in exactly the same manner as for the Express option (illustrated above).

For example, if you enter 20, the Center axis will be 20 pixels below the object center.

Enter a negative value to move the Center axis up instead of down.

Notes:

- If a group and its objects are configured to change size during runtime, the group scaling effects will be combined with the object scaling effects. For example, if a group is configured to double in size at runtime, and one of its object is configured to halve in size, the object will appear to remain the same size (it halves, then doubles). Remember, however, that the object's position might change as the group gets bigger.
- When the radio buttons in Object Properties - Fill Color (On/Off, Multi-state and so on) are selected, they change the appearance of the right-hand side of the dialog.

See Also

Object Properties - Scaling (Horizontal)

Fill Color

You can control the fill color to use for your objects.

To configure an object or group with changing fill color:

1. Draw the object/group (or paste a symbol). The properties tab dialog will automatically display, unless you have turned off the **Display properties on new** option in the Graphics Builder. (For a group, the properties dialog will not display automatically; you need to double-click the group.)
2. Click the **Fill** tab.
3. Click the **Color** tab (to the right of the dialog).
4. Select the type of color change (On/Off, Multi-state and so on).
5. Enter the expression/conditions that will change the object's fill color at runtime.
6. Enter additional object property details as necessary.
7. Click **OK**.

See Also

Object Properties - Fill Color (On/Off)
Object Properties - Fill Color (Multi-state)
Object Properties - Fill Color (Array)
Object Properties - Fill Color (Threshold)

Object Properties - Fill Color (Gradient)

Object Properties - Fill Color (On/Off)

Objects and groups have the following Fill Color (On/Off) properties:

[Type] On / Off

Select this radio button to fill the object/group with one color when a particular expression is TRUE, and another when it is FALSE. For example, you could fill an object/group with red when a particular variable tag is in alarm, and green when it is not.

[Type] Multi-state

This option is useful when you have several possible conditions, occurring together in different combinations, at different times. Select this option to fill the object/group with a different color for each combination.

For example, three digital variable tags (A,B, and C) can each be ON or OFF at any time. You can fill the object/group with a different color for each ON/OFF combination. In other words, you could use a different fill color for each of the following ON/OFF combinations ABC, **A**BC, A**B**C, AB**C**, **AB**C, A**BC**, **A**B**C**, **ABC**.

[Type] Array

The Array option allows you to enter an expression which returns an integer. For each unique integer (from 0 to 255), you can fill the object/group with a different color. For example, you could use a different fill color for each threshold of an analog alarm.

[Type] Threshold

Select this radio button to dynamically change the fill color when an expression reaches a specific value (threshold). For example, you might decide that the fill color will change to red when the speed of a motor is greater than or equal to 4500 rpm, and to white when less than or equal to 100 rpm, but remains gray for every speed in between.

[Type] Gradient

Select this radio button to dynamically graduate the fill color, displaying a different color for each unique value returned by a particular expression. This option allows you to select two colors, to be used as the color limits. The color for each value returned is automatically selected from within the range defined by these limits. The result is a fade from one color to another.

ON color when

The color selected as the **ON color** (below) will be used as the fill color whenever the condition entered here (128 characters maximum) is TRUE. The color selected as the **OFF color** (below) will be used as the fill color whenever this condition is FALSE. For example, you could fill an object/group with blue when **MIX_RUNNING** is TRUE, and white when it is FALSE.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

OFF color

The fill color whenever the condition entered above is FALSE. For example, you could fill the object/group with white when **MIX_RUNNING** is FALSE.

> **Note:** The color that you select here will change any Fill color specified through Appearance (General) properties tab.

ON color

The fill color whenever the condition entered above is TRUE. For example, you could fill the object/group with blue when **MIX_RUNNING** is TRUE.

> **Note:** Group fill color is only applied if the individual objects in the group do not have their own fill colors defined.

See Also

Object Properties - Fill Color (Multi-state)
Object Properties - Fill Color (Array)
Object Properties - Fill Color (Threshold)
Object Properties - Fill Color (Gradient)

Object Properties - Fill Color (Multi-state)

Objects and groups have the following Fill Color (Multi-state) properties:

[Type] On / Off

Select this radio button to fill the object/group with one color when a particular expression is TRUE, and another when it is FALSE. For example, you could fill an object/group with red when a particular variable tag is in alarm, and green when it is not.

[Type] Multi-state

This option is useful when you have several possible conditions, occurring together in different combinations, at different times. Select this option to fill the object/group with a different color for each combination.

For example, three digital variable tags (A,B, and C) can each be ON or OFF at any time. You can fill the object/group with a different color for each ON/OFF combination. In other words, you could use a different fill color for each of the following ON/OFF combinations ABC, **A**BC, A**B**C, AB**C**, **AB**C, A**BC**, **A**B**C**, **ABC**.

[Type] Array

The Array option allows you to enter an expression which returns an integer. For each unique integer (from 0-255), you can fill the object/group with a different color. For example, you could use a different fill color for each threshold of an analog alarm.

[Type] Threshold

Select this radio button to dynamically change the fill color when an expression reaches a specific value (threshold). For example, you might decide that the fill color will change to red when the speed of a motor is greater than or equal to 4500 rpm, and to white when less than or equal to 100 rpm, but remains gray for every speed in between.

[Type] Gradient

Select this radio button to dynamically graduate the fill color, displaying a different color for each unique value returned by a particular expression. This option allows you to select two colors, to be used as the color limits. The color for each value returned is automatically selected from within the range defined by these limits. The result is a fade from one color to another.

Conditions

The conditions you enter here (using a maximum of 128 characters per condition) will occur together in different ways, at different times. You can use each unique combination to force a different fill color for the object/group.

To enter a condition, click the relevant line (A, B, C, and so on), click **Edit**, and type in the condition. You can add more conditions (to a maximum of 5, providing 32 combinations), using the **Add** button. To insert a tag or function, click the **Wizard** button. This button displays two options: Insert Tag and InsertFunction. You can also remove conditions by clicking **Delete**, but you need to always provide at least one condition in this field. Conditions left blank (instead of deleted) are evaluated as false at runtime.

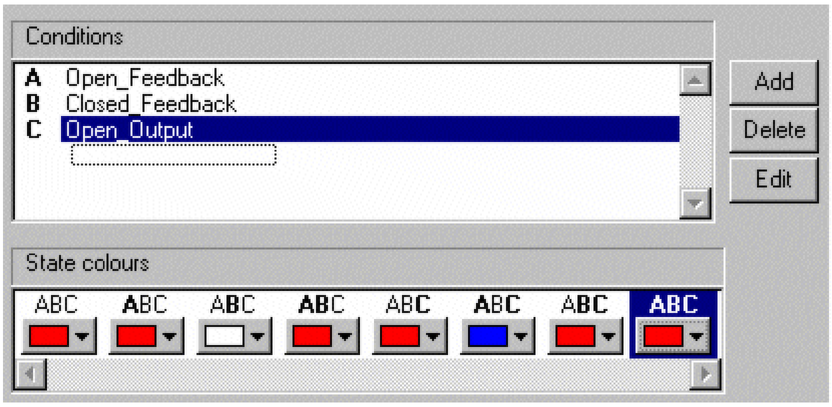
State colors

The fill colors that will be used for each combination of the above conditions.

> **Note:** The color that you select as **ABC** (all conditions false) will change any Fill color specified through Appearance (General) properties tab.

For example:

To fill the object/group with a different color each time the status of a valve changes, you could fill out the **Conditions** and **State symbols** fields as follows:



Conditions

A Open_Feedback
B Closed_Feedback
C Open_Output

In this example, **Open_Feedback**, and **Close_Feedback** are variable tags representing digital inputs on the valve, and **Open_Output** is a variable tag representing an output on the valve. So, ABC means **Open_Feedback** is ON, and **Close_Feedback** and **Open_ Output** are both OFF. For this combination, the red is used as the fill color to indicate that the valve is not responding to commands, because it is open when it is meant to be closed. The same type of logic applies to the rest of the states.

> **Note:** Group fill color is only applied if the individual objects in the group do not have their own fill colors defined.

See Also

Object Properties - Fill Color (On/Off)
Object Properties - Fill Color (Array)
Object Properties - Fill Color (Threshold)
Object Properties - Fill Color (Gradient)

Object Properties - Fill Color (Array)

Objects and groups have the following Fill Color (Array) properties:

[Type] On / Off

Select this radio button to fill the object/group with one color when a particular expression is TRUE, and another when it is FALSE. For example, you could fill an object/group with red when a particular variable tag is in alarm, and green when it is not.

[Type] Multi-state

This option is useful when you have several possible conditions, occurring together in different combinations, at different times. Select this option to fill the object/group with a different color for each combination.

For example, three digital variable tags (A,B, and C) can each be ON or OFF at any time. You can fill the object/group with a different color for each ON/OFF combination. In other words, you could use a different fill color for each of the following ON/OFF combinations ABC, **A**BC, A**B**C, AB**C**, **AB**C, A**BC**, **A**B**C**, **ABC**.

[Type] Array

The Array option allows you to enter an expression which returns an integer. For each unique integer (from 0-255), you can fill the object/group with a different color. For example, you could use a different fill color for each threshold of an analog alarm.

[Type] Threshold

Select this radio button to dynamically change the fill color when an expression reaches a specific value (threshold). For example, you might decide that the fill color will change to red when the speed of a motor is greater than or equal to 4500 rpm, and to white when less than or equal to 100 rpm, but remains gray for every speed in between.

[Type] Gradient

Select this radio button to dynamically graduate the fill color, displaying a different color for each unique value returned by a particular expression. This option allows you to select two colors, to be used as the color limits. The color for each value returned is automatically selected from within the range defined by these limits. The result is a fade from one color to another.

Array expression (128 Chars.)

Enter the expression which is to return an integer. For each value returned, a different color will fill the object/group.

If the return value is:

- Less than 0 (zero), it will be set to 0 (zero), and a runtime hardware alarm will be triggered.
- Greater than 255, it will be set to 255, and a runtime hardware alarm will be triggered.
- A real (non-integer) number, it will be truncated (for example 8.1 and 8.7 would both be truncated to 8).

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

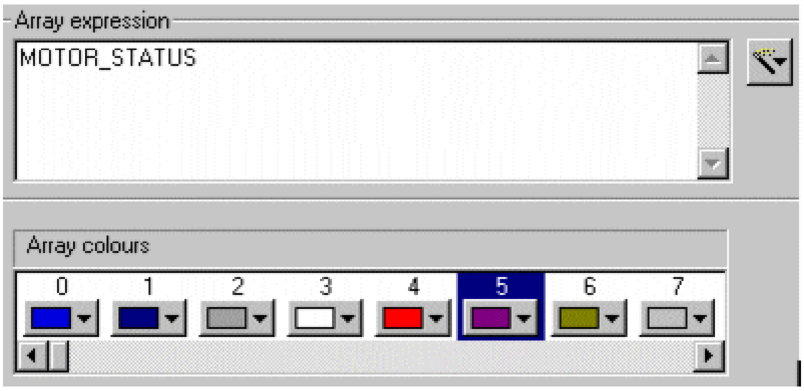
Array colors

The fill colors that will be used for each integer returned by the Array expression entered above (color 0 will be used when the expression returns integer 0, color 1 will be used when integer 1 is returned and so on).

Note: The color that you select for color 0 (zero) will change any Fill color specified

> through Appearance - General tab.

For example, to display different symbols illustrating the various states of a motor, you could fill out the **Array expression** and **Array symbol** fields as follows:



In this example, **MOTOR_STATUS** is an analog variable tag representing the status of a motor. Each time the motor changes state, an integer is returned (0 = Running, 1 = Starting and so on), and the appropriate color fills the object/group. Color 5 onwards have no bearing on the fill color, because the tag only returns 5 unique integers (0-4).

> **Note:** Group fill color is only applied if the individual objects in the group do not have their own fill colors defined.

See Also

Object Properties - Fill Color (On/Off)
Object Properties - Fill Color (Multi-state)
Object Properties - Fill Color (Threshold)
Object Properties - Fill Color (Gradient)

Object Properties - Fill Color (Threshold)

Objects and groups have the following fill color (threshold) properties.

[Type] On / Off

Select this radio button to fill the object/group with one color when a particular expression is TRUE, and another when it is FALSE. For example, you could fill an object/group with red when a particular variable tag is in alarm, and green when it is not.

[Type] Multi-state

This option is useful when you have several possible conditions, occurring together in different combinations, at different times. Select this option to fill the object/group with a different color for each combination.

For example, three digital variable tags (A,B, and C) can each be ON or OFF at any time. You can fill the object/group with a different color for each ON/OFF combination. In other words, you could use a different fill color for each of the following ON/OFF combinations ABC, **A**BC, A**B**C, AB**C**, **AB**C, A**BC**, **A**B**C**, **ABC**.

[Type] Array

The Array option allows you to enter an expression which returns an integer. For each unique integer (from 0-255), you can fill the object/group with a different color. For example, you could use a different fill color for each threshold of an analog alarm.

[Type] Threshold

Select this radio button to dynamically change the fill color when an expression reaches a specific value (threshold). For example, you might decide that the fill color will change to red when the speed of a motor is greater than or equal to 4500 rpm, and to white when less than or equal to 100 rpm, but remains gray for every speed in between.

[Type] Gradient

Select this radio button to dynamically (and smoothly) graduate the fill color, displaying a different color for each unique value returned by a particular expression. This option allows you to select two colors, to be used as the color limits. The color for each value returned is automatically selected from within the range defined by these limits. The result is a smooth fade from one color to another.

Color expression

The value of the expression entered in this field (128 characters maximum) determines the fill color of the object/group. i.e. When the value of this expression reaches a threshold value (as defined below), fill color will change.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Color expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the Color expression, rather than using the default values. (Threshold values are percentages of the range between **Minimum** and **Maximum**.) For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Color expression] Minimum

Enter the minimum value for the expression. In terms of thresholds, the Minimum is **0%**. You can only enter a value here if you have selected the **Specify** range box.

[Color expression] Maximum

Enter the maximum value for the expression. In terms of thresholds, the Maximum is **100%**. You can only enter a value here if you have selected the **Specify** range box.

Thresholds (%)

The thresholds and their associated colors. A threshold is entered as a percentage of the expression range (the range of values that can be returned by the expression). For example, if the expression's minimum is 0 and its Maximum 200, the default thresholds would have the following effects:

Threshold	Associated Color	Meaning
< 5%	Bright Blue	When the expression returns **less than 10**, the color fill will be **Bright Blue**.
< 15%	Blue	When the expression returns **less than 30**, the color fill will be **Blue**.
> 85%	Red	When the expression returns **greater than 170**, the color fill will be **Red**.
> 95%	Bright Red	When the expression returns **greater than 190**, the color fill will be **Bright Red**.

You can add up to 100 threshold color combinations. To add a combination, click **Add** and enter the relevant details. To edit an existing combination, click the relevant line. You can also remove combinations by clicking **Delete**.

Any values not included in a range (for example between 15% and 85% in the example above) produce a static fill color as specified through **Appearance - General** tab.

Note: Group fill color is only applied if the individual objects in the group do not have their own fill colors defined.

See Also

Object Properties - Fill Color (On/Off)
Object Properties - Fill Color (Multi-state)
Object Properties - Fill Color (Array)
Object Properties - Fill Color (Gradient)

Object Properties - Fill Color (Gradient)

Objects and groups have the following Fill Color (Gradient) properties:

[Type] On / Off

Select this radio button to fill the object/group with one color when a particular expression is TRUE, and another when it is FALSE. For example, you could fill an object/group with red when a particular variable tag is in alarm, and green when it is not.

[Type] Multi-state

This option is useful when you have several possible conditions, occurring together in different combinations, at different times. Select this option to fill the object/group with a different color for each combination.

For example, three digital variable tags (A,B, and C) can each be ON or OFF at any time. You can fill the object/group with a different color for each ON/OFF combination. In other words, you could use a different fill color for each of the following ON/OFF combinations ABC, **A**BC, A**B**C, AB**C**, **AB**C, A**BC**, **A**B**C**, **ABC**.

[Type] Array

The Array option allows you to enter an expression which returns an integer. For each unique integer (from 0 to 255), you can fill the object/group with a different color. For example, you could use a different fill color for each threshold of an analog alarm.

[Type] Threshold

Select this radio button to dynamically change the fill color when an expression reaches a specific value (threshold). For example, you might decide that the fill color will change to red when the speed of a motor is greater than or equal to 4500 rpm, and to white when less than or equal to 100 rpm, but remains gray for every speed in between.

[Type] Gradient

Select this radio button to dynamically graduate the fill color, displaying a different color for each unique value returned by a particular expression. This option allows you to select two colors, to be used as the color limits. The color for each value returned is automatically selected from within the range defined by these limits. The result is a fade from one color to another.

Color expression

The value of the expression entered in this field (128 characters maximum) will determine the fill color of the object/group. By default, when the expression returns its minimum value, the fill color will be the **At minimum** color (as defined below). When the expression returns its maximum value, the fill color will be the **At maximum** color (as defined below). When the expression returns a value half-way between its minimum and maximum, a color will be selected from the half-way point of the range defined by the **At minimum** and **At maximum** colors.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Color expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the Color expression, rather than using the default values. For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Color expression] Minimum

Enter the minimum value for the expression. When this value is returned by the expression, the fill color of the object/group will be the **At minimum** color. You can only enter a value here if you have selected the **Specify** range box.

[Color expression] Maximum

Enter the maximum value for the expression. When this value is returned by the expression, the fill color of the object/group will be the **At maximum** color. You can only enter a value here if you have selected the **Specify** range box.

At minimum

The fill color of the object/group when the **Color expression** returns its minimum value.

> **Note:** The color that you select here will change any Fill color specified through **Appearance - General** tab.

At maximum

The fill color of the object/group when the **Color expression** returns its maximum value.

> **Note:** Group fill color is only applied if the individual objects in the group do not have their own fill colors defined.

See Also

Object Properties - Fill Color (On/Off)
Object Properties - Fill Color (Multi-state)
Object Properties - Fill Color (Array)
Object Properties - Fill Color (Threshold)

Fill Level

You can control the fill level shown in your objects.

To configure an object or group with changing fill level:

1. Draw the object/group (or paste a symbol). The properties tab dialog will automatically display, unless you have turned off the **Display properties on new** option

in the Graphics Builder. (For a group, the properties dialog will not display automatically; you need to double-click the group.)

2. Click the **Fill** tab.
3. Click the **Level** tab (to the right of the dialog).
4. Enter a **Level expression** (the expression that will change the fill level of the object/group at runtime).
5. Enter additional properties as necessary.
6. Click **OK**.

See Also

Object Properties - Fill (Level)
Defining Common Object Properties

Object Properties - Fill (Level)

The fill level of an object/group can be changed during runtime, increasing or decreasing dynamically whenever the value of a particular expression changes. As the value of the expression increases and decreases, the fill level will increase and decrease accordingly (as a percentage of the full capacity of the object/group). If the object/group resizes at runtime, the fill level will adjust automatically in order to maintain the correct percentage.

The color that is used is set through either General Appearance, or Color Fill.

This property could be used to display temperature variations. You could even combine the Fill Color and Fill Level properties to produce a thermometer with mercury that rises and changes color with rising temperature.

Objects and groups have the following Fill Level properties:

Level expression

The value of the expression entered in this field (253 characters maximum) will determine the fill level of the object/group. By default, when the expression returns its minimum value, the object/group will be filled to the **At minimum** level. When the expression returns its maximum value, the object/group will be filled to the **At maximum** level. When the expression returns a value half-way between its minimum and maximum, the object/group will be filled to half-way between the **At minimum** and **At maximum** levels.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Level expression] Specify range

Select this box to manually specify **Minimum** and **Maximum** values for the Level expression, rather than using the default values. For an expression containing an analog variable tag, the defaults are the Engineering Zero and Full Scale values from the last variable tag in the expression. If the analog variable tag does not have Engineering Zero and Full Scale values, the defaults are 0 (zero) and 32000. For expressions without tags, the defaults are 0 (zero) and 100.

[Level expression] Minimum

Enter the minimum value for the expression. When this value is returned by the expression, the object/group will fill to the **At minimum** level. You can only enter a value here if you have selected the **Specify** range box.

[Level expression] Maximum

Enter the maximum value for the expression. When this value is returned by the expression, the object/group will fill to the **At maximum** level. You can only enter a value here if you have selected the **Specify** range box.

At minimum

The level to which the object/group will be filled when the **Level expression** returns its minimum value. For example, if you enter 30, the object/group will be 30% full when the expression returns its minimum value.

You can change the percentage by pressing the up and down arrows to the right of the field, or by entering another value in this field.

At maximum

The level to which the object/group will be filled when the **Level expression** returns its maximum value. For example, if you enter 90, the object/group will be 90% full when the expression returns its maximum value.

You can change the percentage by pressing the up and down arrows to the right of the field, or by entering another value in this field.

Fill Direction

The direction in which the color will spread when increasing. There are four options (each represented by an arrow): **Up**, **Down**, **Left**, **Right**. If you choose Up, the object/group will be filled from the bottom up. If you choose Left, the object/group will be filled from right to left, and so on

Background color

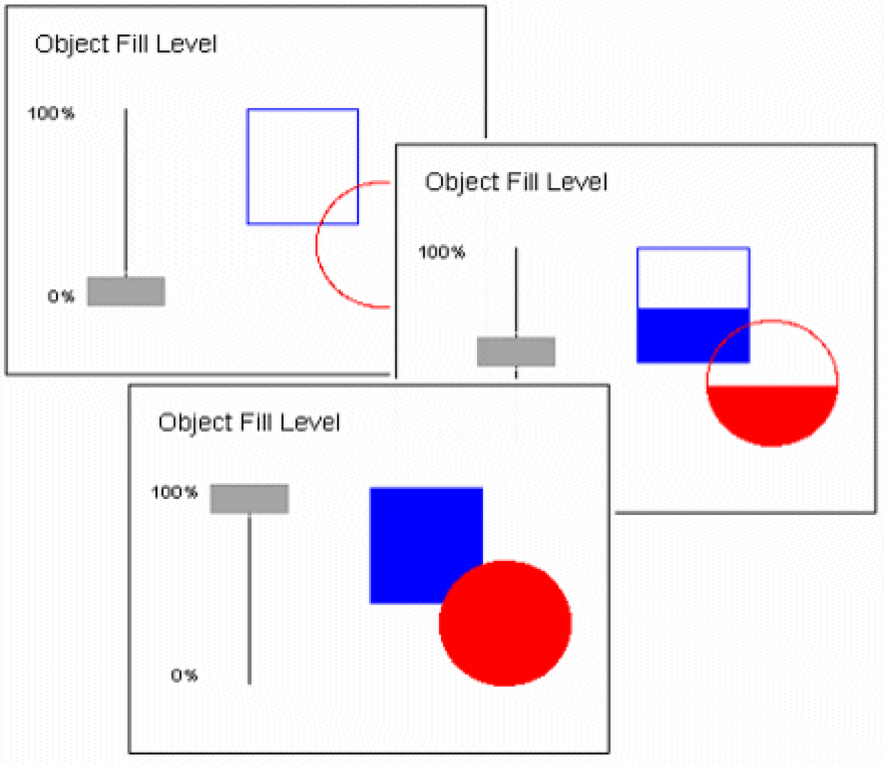
The color of any unfilled part of the object/group (for example, if the object/group is only 90% full, the unfilled 10% will be display using this color). The background is often made transparent. Using transparent, you would see the outline of the object/group, and anything behind the object/group on the page.

Note: If an object in a group is a slider, it might change the overall size of the group when used at runtime. If it does, the fill level of the group will adjust accordingly.

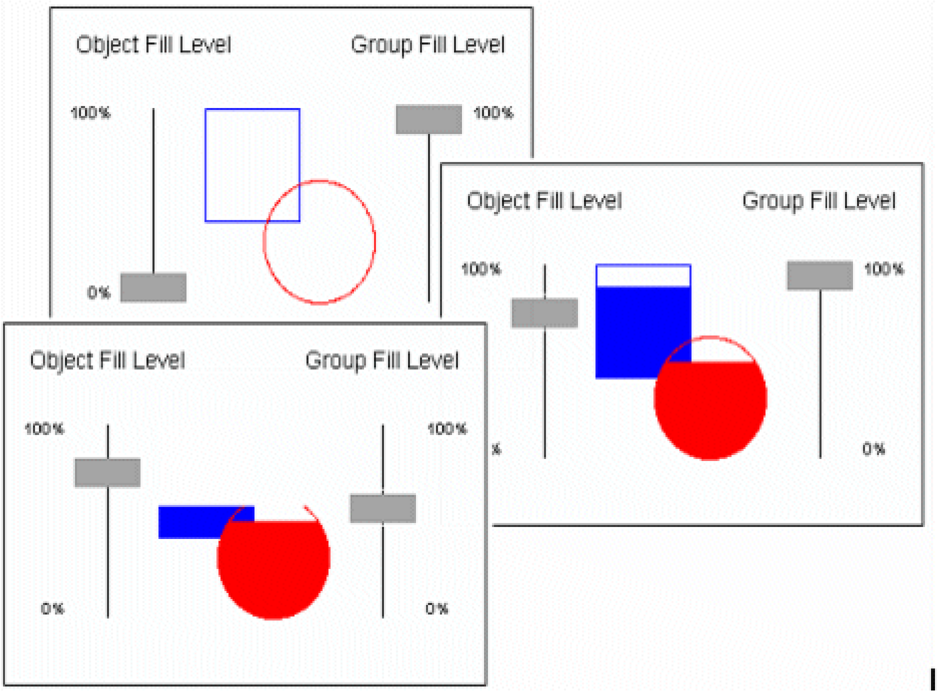
Group and Object Fill Level: Examples

A group and its objects can be configured with different fill levels. The group fill level, however, is usually thought of as a reveal of the objects in the group. Group fill level and object fill level operate independently of each other; the group fill level just determines how much of the objects display.

Example 1: The fill level of the objects:



Example 2: Group the objects and configure a fill level for the group as well



In this example, the objects' fill levels can still be adjusted normally. The group's fill level determines how much of the objects you can see (and how much will be obscured by the groups background color; white, in this case).

Touch Commands

You can assign touch commands to an object or group.

To assign a touch command to an object or group:

1. Draw the object/group (or paste a symbol). The properties tab dialog will automatically display, unless you have turned off the **Display properties on new** option in the Graphics Builder. (For a group, the properties dialog will not display automatically; you need to double-click the group.)
2. Click the **Input** tab.
3. Click the **Touch** tab (to the right of the dialog).
4. Enter a command in the command field (the command that will be executed when the object/group is touched at runtime).
5. Enter further details as necessary, using the **Help** button for detailed information about each field.
6. Click **OK**.

See Also

Object Properties - Input (Touch)
Defining Common Object Properties

Object Properties - Input (Touch)

The touch property lets you assign commands to the object/group. These commands are then executed when the object/group is touched at runtime (i.e., an operator clicks on the object/group). You can also define messages which will log at these times.

For example, a drive can be jogged by starting it when the mouse button is depressed and stopping it when the mouse button released; variables can be incremented while the mouse button is held, and so on. At the same time, it could log the time and date, and the name of the operator.

Operators who do not satisfy the access requirements cannot touch the object/group at runtime.

Objects and groups have the following input (touch) properties.

Action

There are three actions to which commands can be attached. You can select more than one type of action. Unique commands and log messages can be attached to each action (i.e. you can perform one task on the down action, and another on the up action, and log a separate message for each).

[Action] Up

Select this option if you want a command to be executed (and a unique message to be logged) when the operator positions the mouse pointer over the object/group, and clicks *and releases* the left mouse button.

As with standard Windows buttons, if the operator moves the cursor away from the object/group before releasing the mouse button, the command isn't executed (unless you also select the **Down** option).

[Action] Down

Select this option if you want a command to be executed (and a unique message to be logged) when the operator positions the mouse pointer over the object/group, and clicks the left mouse button. The command will execute as soon as the mouse button is clicked.

[Action] Repeat

Select this option if you want a command to execute continually (and a unique message to log continually) whenever the operator has the mouse pointer positioned over the object/group, and is holding the left mouse button depressed. If the operator moves the mouse pointer away from the object/group without releasing the mouse button, the command will stop executing, but will start again as soon as the mouse pointer is re-positioned over the object/group. The only exception is when you also have the Down option selected, in which case, the command will execute continually even if the mouse pointer has been moved away from the object/group.

To set the delay which precedes the first execution of the command (and the first log of the message), and the delay between each repeat.

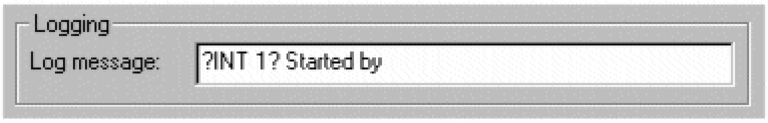
Up/Down/Repeat command

The commands (set of instructions) to be executed immediately when the selected action is performed. The command(s) can be a maximum of 253 characters long.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Logging] Log Message

The text message sent to the *MsgLog* field of the Log Device when the selected action is performed by the operator at runtime. The message is plain text, and can include tag name substitutions using Genie or Super Genie syntax. When using Super Genie syntax, the data type needs to be specified. The name of the tag will then be included in the text. The log message can be a maximum of 32 characters long.



Logging

Log message: ?INT 1? Started by

To include field data as part of a logged message, insert the field name as part of the device format when configuring the device. For instance, in the **Format** field of the **Devices** form, you could enter {MsgLog,20} {FullName,15}. This would accommodate the logging of messages such as **P2 started by John Smith**.

The log device to which the message is sent is specified through the General Access tab.

Note: If the object is part of a Genie or symbol, this property can be defined after the Genie/symbol is pasted onto a page. (Hold down the **Control** (CTRL) key and double-click the object.) If you define it before pasting (i.e. you define it for the original in the library), you cannot edit it after. Similarly, if the object is part of a template, it can be defined after a page has been created using that template (again, with Control + double-click). If you define it for the actual template, you cannot edit it for pages based on the template.

Repeat rate

This option sets the delay which precedes the first execution of the command(s), and the delay between each subsequent repeat of the command(s).

You can change the rate by pressing the up and down arrows to the right of the field, or by entering another value in this field.

> **Note:** If you define a touch command for an object in a group, the group's touch command will not work.

Keyboard Commands

You can assign a keyboard command to an object or group.

To assign a keyboard command to an object or group:

1. Draw the object/group (or paste a symbol). The object properties tab dialog will automatically display, unless you have turned off the **Display properties on new** option in the Graphics Builder. (For a group, the properties dialog will not display automatically; you need to double-click the group.)
2. Click the **Input** tab.
3. Click the **Keyboard Commands** tab (to the right of the dialog).
4. Enter the key sequence.
5. Enter a command in the command field (the command that will be executed when the key sequence above is entered by the operator at runtime).
6. Enter additional details as necessary.
7. Click **OK**.

See Also

Object Properties - Input (Keyboard Commands)
Defining Common Object Properties

Object Properties - Input (Keyboard Commands)

The keyboard commands property lets you assign keyboard commands to the object/group. A keyboard command is a particular key sequence which executes a command when it is typed in by the operator at runtime. To execute an object/group keyboard command, the operator positions the cursor over the object/group and enters the key sequence using the keyboard.

You can also define a message which will log every time the key sequence is entered.

For example, the operator could change the water level in a tank by placing the mouse over the symbol representing the tank, and typing in the new level. At the same time, a message could be logged, listing the time and date, and the name of the operator.

Operators who do not satisfy the access requirements specified under **Security** below cannot enter keyboard commands for this object/group at runtime.

For a detailed explanation of keyboard command input see the topic Getting Runtime Operator Input in the Cicode Programming Reference.

Objects and groups have the following keyboard commands input properties:

Key sequence

Enter the key sequences that the operator can enter to execute a command. For example, you might define the key sequence ### **Enter**. During runtime, this key sequence would allow you to type in any three digit number, and click **Enter** to change variable tag values from mimic pages and so on

You can enter as many key sequences as you like. To add a key sequence, click **Add** and type in the sequence or select one from the menu. To edit an existing sequence, click the relevant line and click **Edit**. You can also remove key sequences by clicking **Delete**.

Key sequence command

The commands (set of instructions) to be executed immediately when the selected key sequence is entered. The commands can be a maximum of 253 characters long.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: Insert Tag and Insert Function.

[Security] Same area as object/group

Select this box to assign the keyboard command to the same area as the object/group. Only users with access to this area (and any necessary privileges) will be able to issue this command or log the message. If you want to assign this keyboard command to another area, do not select this box; instead, enter another area below.

[Security] Command area

Enter the area to which this keyboard command belongs. Only users with access to this area (and any necessary privileges) will be able to issue this command or log the message. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to issue this command.

Click the menu to the right of this field to select an area, or type in an area number directly.

Note: If the object is part of a Genie or symbol, this property can be defined after the Genie/symbol is pasted onto a page (**Ctrl** + double-click). Similarly, if the object is part of a template, this property can be defined after a page has been created using that template (**Ctrl** + double-click).

You can leave this field blank by selecting the **Same privilege as object/group** box.

[Security] Same privilege as object/group

Select this box to assign the keyboard command the same privilege as the object/group. Only users with this privilege level will be able to issue this command, or log the message. If you want to assign this keyboard command a different privilege, do not select this box; instead, enter another privilege below.

[Security] Privilege level

Enter the privilege level that a user needs to possess to be able to issue this command or log the message. For example, if you enter Privilege Level 1 here, operators need to possess Privilege Level 1 to be able to issue this command. You can also combine this restriction with area restrictions. For example, if you assign the keyboard command to Area 5, with Privilege Level 2, the user needs to be set up with Privilege 2 for Area 5.

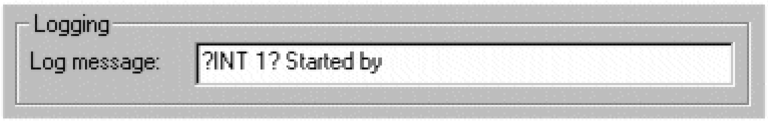
Click the menu to the right of this field to select a privilege, or type in an area number directly.

Note: If the object is part of a Genie or symbol, this property can be defined after the Genie/symbol is pasted onto a page (**Ctrl** + double-click). Similarly, if the object is part of a template, this property can be defined after a page has been created using that template (**Ctrl** + double-click).

You can leave this field blank by selecting the **Same privilege as object/group** box.

[Logging] Log Message

The text message sent to the *MsgLog* field of the Log Device when the selected action is performed by the operator at runtime. The message is plain text, and can include tag name substitutions using Genie or Super Genie syntax. When using Super Genie syntax, the data type needs to be specified. The name of the tag will then be included in the text. The message can be a maximum of 32 characters long.



Log message: ?INT 1? Started by

If you want to include field data as part of a logged message, you need to insert the field name as part of the device format when you configure the device. For instance, in the **Format** field of the **Devices** form, you could enter {MsgLog,20} {FullName,15}. This would accommodate the logging of messages such as **P2 started by John Smith**.

The log device to which the message is sent is specified through the General Access tab.

> **Note:** If a group and one of its objects are both assigned a keyboard command with the same key sequence, the object's command will take precedence (i.e., the group's command will not execute).

Sliders

You can create sliders to have on your graphics pages.

To configure a slider:

1. Draw the object/group (or paste a symbol). The properties tab dialog will automatically display, unless you have turned off the **Display properties on new** option in the Graphics Builder. (For a group, the properties dialog will not display automatically; you need to double-click the group.)
2. Click the **Slider** tab.
3. Click the **Horizontal**, **Vertical** or **Rotational** tab (to the right of the dialog).
4. Enter the **Tag** to link to the slider.
5. Enter any additional details.
6. Click **OK**.

See Also

Object Properties - Slider (Horizontal)
Object Properties - Slider (Vertical)
Object Properties - Slider (Rotational)

Object Properties - Slider (Horizontal)

Objects and groups can be linked to variable tags in such a way that horizontal sliding of the object/group changes the value of the tag. As the slider moves to the right, the variable tag increases in value. As the slider moves to the left, the variable tag decreases in value. The slider also moves automatically to reflect the changing values of the tag.

> **Note:** The horizontal slider cannot be used if the rotational slider is enabled, or if horizontal movement is enabled.

Objects and groups have the following horizontal slider properties.

Tag

The value of the tag entered in this field (79 characters maximum) will change when the slider is moved left or right. You can define two slider limits on your graphics page. The object/group will not slide beyond these two points. During runtime, when the slider reaches its left-hand limit (**Offset At minimum**), the tag value changes to its minimum limit. When the slider reaches its right-hand limit (**Offset At maximum**), the tag value changes to its maximum limit.

To insert a tag, click the **Wizard** button to the right of this field.

[Tag] Continuous update of tag

Select this box if you want the variable tag to be updated continuously while the slider is being moved. If you do not select this box, the tag will only be updated when the slider has been released (i.e., it has been moved, and the operator has released the mouse button).

[Offset] At minimum

The distance (number of pixels from the original object/group center) that the object/group can slide to the left. When it reaches the point defined by this distance, the tag value changes to its minimum limit.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

[Offset] At maximum

The distance (number of pixels from the original object/group center) that the object/group can slide to the right. When it reaches the point defined by this distance, the tag value changes to its maximum limit.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

You can increase the value of the tag with a left-slide, and decrease it with a right-slide, by entering negative distances in the Offset fields.

Note: If an object in a group is a slider, it might change the overall size of the group when used at runtime. If it does, the fill level of the group will adjust accordingly. If a group and one of its objects are both defined as sliders, and they slide in the same direction or one is rotational, the object will take precedence (i.e. only the object will operate as a slider).

See Also

Object Properties - Slider (Vertical)
Object Properties - Slider (Rotational)

Object Properties - Slider (Vertical)

You can link objects and groups to variable tags in such a way that vertical sliding of the object/group changes the value of the tag. As the slider moves to the up, the variable tag increases in value. As the slider moves to the down, the variable tag decreases in value. The slider also moves automatically to reflect the changing values of the tag.

> **Note:** The vertical slider cannot be used if the rotational slider is enabled, or if vertical movement is enabled.

Objects and groups have the following vertical slider properties:

Tag

The value of the tag entered in this field (79 characters maximum) will change when the slider is moved up or down. You can define two slider limits on your graphics page. The object/group will not slide beyond these two points. During runtime, when the slider reaches its upper limit (**Offset At maximum**), the tag value changes to its maximum limit. When the slider reaches its lower limit (**Offset At minimum**), the tag value changes to its minimum limit.

To insert a tag, click the **Wizard** button to the right of this field.

[Tag] Continuous update of tag

Select this box if you want the variable tag to be updated continuously while the slider is being moved. If you do not select this box, the tag will only be updated when the slider has been released (i.e., it has been moved, and the operator has released the mouse button).

[Offset] At maximum

The distance (number of pixels from the original object/group center) that the object/group can slide up. When it reaches the point defined by this distance, the **Tag** value changes to its maximum limit.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

[Offset] At minimum

The distance (number of pixels from the original object/group center) that the object/group can slide down. When it reaches the point defined by this distance, the tag value changes to its minimum limit.

You can change the offset value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

You can increase the value of the tag with a down-slide, and decrease it with an up-slide, by entering negative distances in the Offset fields.

> **Note:** If an object in a group is a slider, it might change the overall size of the group when used at runtime. If it does, the fill level of the group will adjust accordingly. If a group and one of its objects are both defined as sliders, the object will take precedence (i.e. only the object will operate as a slider).

See Also

Object Properties - Slider (Horizontal)
Object Properties - Slider (Rotational)

Object Properties - Slider (Rotational)

Objects and groups can be linked to variable tags in such a way that rotational sliding of the object/group changes the value of the tag. As the slider rotates clockwise, the variable tag increases in value. As the slider rotates anti-clockwise, the variable tag decreases in value. The slider also moves automatically to reflect the changing values of the tag.

> **Note:** The rotational slider cannot be used if either of the other sliders is enabled, or if rotational movement is enabled.

Objects and groups have the following rotational slider properties.

Tag

The value of the tag entered in this field (79 characters maximum) will change as the slider is rotated. You can define two slider limits on your graphics page. The object/group will not rotate beyond these two points. During runtime, when the slider reaches its anti-clockwise limit (**Offset At minimum**), the tag value changes to its minimum limit. When the slider reaches its clockwise limit (**Offset At maximum**), the tag value changes to its maximum limit.

To insert a tag, click the **Wizard** button to the right of this field.

[Tag] Continuous update of tag

Select this box if you want the variable tag to be updated continuously while the slider is being moved. If you do not select this box, the tag will only be updated when the slider has been released (i.e. it has been moved, and the operator has released the mouse button).

[Angle] At minimum

Enter an anti-clockwise angle (in degrees relative to 0 degrees). The slider cannot rotate anti-clockwise beyond this limit. When it reaches this limit, the **Tag** value changes to its minimum limit.

You can change the angle value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

[Angle] At maximum

Enter an clockwise angle (in degrees relative to 0 degrees). The slider cannot rotate clockwise beyond this limit. When it reaches this limit, the **Tag** value changes to its maximum limit.

You can change the angle value by pressing the up and down arrows to the right of the field, or by entering another value in this field.

> **Note:**You can increase the value of the tag with an anti-clockwise slide, and decrease it with a clockwise-slide, by entering negative distances in the Angle fields.

[Center axis offset] Express

Click this radio button for the quick and easy way of selecting the point about which the object/group will rotate. The express option gives you the choice of 9 points (Top Left, Bottom Right and so on), which are displayed in the picture field on the dialog. To select one, just click it with your mouse.

[Center axis offset] Custom

Click this radio button to define your own center axis. When you select this radio button, two fields will display to the right, allowing you to plot the position of your center axis. Specify the distance to the right in the first field, and the distance down in the second. The Center axis is plotted based on these two values.

For example, if you enter 8 as the horizontal offset and 13 as the vertical, the center axis will be 8 pixels to the right and 13 pixels below the center of the object/group.

Enter negative values in the offset distance fields to move the center axis left instead of right, and up instead of down.

> **Note:**If a group and one of its objects are both defined as sliders, the object will take precedence (i.e., only the object will operate as a slider).

See Also

Object Properties - Slider (Horizontal)
Object Properties - Slider (Vertical)

Access

You can determine the kind of access you want to have to your objects.

- General Access to Objects
- Disable Access to Objects

General Access to Objects

Use the General tab to define the general access characteristics of objects.

To define general access properties for objects:

1. Double-click the object.
2. Click the **Access** tab.
3. Click the **General** tab.
4. Enter details as necessary, then click **OK**.

See Also

Object Properties - Access (General)
Defining Common Object Properties

Object Properties - Access (General)

Use the Object Properties dialog to set up general identification, security, logging, and privilege parameters for your ActiveX object.

Objects and groups have the following general access properties.

[Identification] Object/Group AN

Displays the Animation number of the object/group. The AN uniquely identifies the object/group, and can be used in Cicode functions and so on.

> **Note:** If the object is part of a Genie or symbol, the following properties can be completed after the Genie/Symbol is pasted onto a page (**Ctrl** + double-click). Similarly, if the object is part of a template, the properties can be defined after a page has been created using that template (**Ctrl** + double-click).

[Identification] Description

Enter a description of the object/group, and its various functions and so on. This field is purely for the entry of information which you consider beneficial to the smooth running and maintenance of your system. It will not affect the way the system runs, and it will not display during runtime.

[Identification] Tool tip

Enter a short, meaningful description (48 characters maximum) of the object/group. During runtime, this description appears when the operator moves the cursor onto the object/group. The message can be plain text, Super Genie syntax or both. When using Super Genie syntax, the data type needs to be included. The name of the tag will then be included in the text.

If an object in a group has a tool tip, this object's tool tip will be displayed, and not the group's tool tip. If the object does not have a tool tip, the group tool tip will display. In this instance, if the object is a member of a group, and that group is part of another group, the tool tip for the first group will display.

If you place an object behind a group, its tool tip will not display. Remember, however, that group boundaries are rectangular, no matter what shape is formed by the objects in the group. This means that 'blank' spaces between objects in a group are actually part of the group. Even if you can see the individual object, if it is behind the group, its tool tip will not display.

[Security] Same area as page

Select this box to assign the object/group to the same area as the page on which it has been drawn; otherwise, leave it blank, and enter another area in the **Object/Group area** field (below).

[Security] Object/Group area

Enter the area to which this object/group belongs. Users without access to this area (and any necessary privileges) will not be able to make full use of the object/group. They will not be able to use touch command, keyboard commands, movement, sliders and so on. (In order to avoid confusion for such operators, it is sometimes a good idea to disable the object/group when it is unavailable due to lack of security rights. Disabled objects/groups can be grayed, hidden or embossed.) For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to make use of this object/group.

Click the menu to the right of this field to select an area, or type in an area number directly.

[Security] No privilege restrictions

Select this box to disable privilege restrictions; otherwise, leave it blank, and enter another privilege below. The implications of not assigning a privilege restriction depend upon whether you have used areas in your security setup:

- **No Areas**: Every operator has full control of the object/group.
- **Areas**: An operator will only need view access to control the object/group if it does not have privilege restrictions.

[Security] Privilege level

Enter the privilege level necessary for an operator to use this object/group. Operators without the necessary privileges will not be able to make full use of the object/group. They will not be able to use touch command, keyboard commands, movement, sliders and so on. (To avoid confusion for such operators, disable the object/group when it is unavailable due to lack of security rights. Disabled objects/groups can be grayed, hidden or embossed.) For example, if you enter Privilege Level 1 here, operators need to possess Privilege Level 1 to use of this object/group. You can also combine this restriction with area restrictions. For example, if you assign the object/group to Area 5, with Privilege Level 2, the user needs to have Privilege 2 for Area 5.

Click the menu to the right of this field to select a privilege, or type in an privilege number directly.

> **Note:** If an object is part of a group, users need to have access to the group in order to have access to the object.

[Logging] Log device

This is the device to which messages will be logged for the object/group's keyboard and touch commands. Click the menu to the right of the field to select a device, or type a device name.

> **Note:** You need to include the *MsgLog* field in the format of the log device for the message to be sent.

See Also

Disable Access to Objects

Disable Access to Objects

If you need to, you can disable access to objects.

To disable access to an object:

1. Double-click the object.
2. Click the **Access** tab.
3. Click the **Disable** tab.

4. Specify the condition which will disable the object as well as the appearance of the object when disabled.
5. Click **OK**.

See Also

Object Properties - Access (Disable)
Defining Common Object Properties

Object Properties - Access (Disable)

Objects and groups have the following access (disable) properties

Disable when

The object/group will be disabled whenever the expression entered here (128 characters maximum) is true. If the object/group is disabled, the operator will not be able to use any form of input, such as sliders, touch commands, keyboard commands and so on.

To insert a tag or a function, click the **Wizard** button to the right of this field. This button displays two options: **Insert Tag** and **Insert Function**.

There are three ways of indicating a disabled object/group: Embossed, Grayed, and Hidden.

[Disable when] Disable on insufficient area or privilege

The object/group will be disabled for operators whose area and privilege rights do not satisfy the requirements defined in the Access.

Disable style:

- **Embossed** - When disabled, the object/group will look as if it has been embossed on the graphics page.
- **Grayed** - When disabled, the object/group will be grayed out (all color detail will be removed).
- **Hidden** - When disabled, the object/group will be entirely hidden from view.

Note: If a group is disabled, objects in that group are also disabled.

Chapter: 22 Defining Commands and Controls

Commands allow your operators to interact with the runtime system. You can define three types of direct command controls:

- Touch commands that your operators issue by clicking specific graphics object (displayed on a graphics page).
- Keyboard commands that your operators issue by typing instructions on the keyboard.
- Slider controls that your operators use to change the values of analog variables.

You can assign privileges to commands and controls, and send a message to the command log each time an operator issues a command.

Touch commands

You can assign Touch commands to the objects that you create on your graphics pages. Touch commands allow the operator to send commands to the runtime system, by clicking (with the mouse or similar) on an object on the graphics page. (For buttons, the command can be executed by highlighting the button with the cursor keys on the keyboard and pressing Enter.)

You can define several commands for an object, one command to execute when the operator clicks on the object, another for when the operator releases the mouse button, and another to operate continuously while the operator holds the mouse button down. For example, a drive can be jogged by starting it when the mouse button is depressed and stopping it when the mouse button released; variables can be incremented while the mouse button is held, and so on.

You can also associate a tool tip (Help text) with an object; if the operator holds the mouse pointer over the object, the tool tip will display in a pop-up window.

> **Note:** You can define a disable condition for any object on a page (including buttons). When the condition is active, the object is grayed and the operator cannot select it.

See Also

Touch Commands

Keyboard commands

Keyboard commands consist of a key sequence that an operator enters on the keyboard, and an instruction (or series of instructions) that executes when the key sequence is entered.

You can define keyboard commands that operate:

- For any graphics page displayed on the computer screen (system keyboard commands).
- Only when a specific graphics page is displayed (page-keyboard commands).
- Only when an operator positions the mouse pointer on an object on a graphics page. You can associate tool tips with any object; if the operator holds the mouse pointer over the object, the tool tip displays in a pop-up box.

Object keyboard commands have precedence over page keyboard commands (which have precedence over system (global) keyboard commands). If you define a keyboard command for an object that is identical to a page keyboard command, the object keyboard command executes when key sequence is entered, but the page keyboard command is ignored.

See Also

Keyboard Commands
System Keyboard Commands
Keyboard Keys
Keyboards
Defining Key Sequences for Commands

Slider controls

Slider controls allow an operator to change the value of an analog variable by dragging an object on the graphics page. Sliders also move automatically to reflect the value of the variable tag.

See Also

Sliders

System Keyboard Commands

You can define system keyboard commands.

To configure a system keyboard command:

1. If you plan to use any special keys, you need to first define your keys.
2. In the Project Editor, choose **System | Keyboard Commands**.
3. Complete the properties in the **System Keyboard Commands** form that appears. You need to enter a key sequence and a command.
4. Click **Add** to append a record you have created, or **Replace** to modify a record.

See Also

System keyboard command properties
Defining Commands and Controls

System keyboard command properties

System keyboard commands have the following properties:

Key Sequence (32 Chars.)

The key sequence for the command.

Command (253 Chars.)

The commands (set of instructions) to execute when an operator enters the key sequence.

Privilege (16 Chars.)

The privilege necessary by an operator to issue the command.

Comment (48 Chars.)

Any useful comment.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Area (16 Chars.)

The area to which the command belongs. Only operators who belong to this area will be able to issue this command. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to issue this command.

Message Log (32 Chars.)

A text message sent to the *MsgLog* field of the Log Device when the selected action is performed by the operator at runtime. The message needs to be plain text:

If you want to include field data as part of a logged message, you need to insert the field name as part of the device format when you configure the device. For instance, in the **Format** field of the **Devices** form, you could enter {MsgLog,20} {FullName,15}. This would accommodate the logging of messages such as **P2 started by John Smith**.

The log device to which the message is sent is specified in the Log Device field below.

Log Device (16 Chars.)

The device to which the **Message Log** is sent when the command is issued.

> **Note:** You need to include the *MsgLog* field in the format of the log device for the message to be sent.

Keyboard Keys

You can define keyboard keys to make it easier to issue commands.

To define a keyboard key:

1. Choose **System** | **Keyboard Keys**. The Keyboard Keys dialog box appears.
2. Enter the **Key Name** and **Key Code** (and **Comment** if applicable).
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Keyboard keys properties
Defining Commands and Controls

Keyboard keys properties

Keyboard Keys have the following properties:

Key Name

The name assigned to the key. Enter a value of 16 characters or less. You can define a meaningful name for any key on your keyboard, and use these keys in any keyboard command. For example, you can define the F1 key as the Login key. When the Login key is subsequently referenced in the system, it refers to the F1 key.

You can assign a key name to any key on the keyboard (including the alphanumeric keys A-Z, a-z, and 0-9). However, after a key is defined as a command key it can only be used as a command key. If you do assign a definition to an alphanumeric key (for example the character A), then that key can not be used as a data key. Each time it is pressed, the definition for the key is recognized and not the keyboard character itself. Keyboard key definitions are usually only used with non-alphanumeric characters (for example the function keys).

Key Code

The code assigned to the key name. Enter a value of 16 characters or less. The Key Code is what links your Key Name to the actual key. You can specify the key code either as a hexadecimal value or use the standard label already associated with the key.

Comment

Any useful comment. Enter a value of 48 characters or less.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Echo

Determines if the key is echoed on the screen (when the key is used). Enter a value of 8 characters or less.

Echo	TRUE

(Display (echo) the **Key Name** when the key is pressed. The key name is displayed on the graphics page in the keyboard entry line - AN1)

Echo	FALSE

(Do not display (echo) the **Key Name** when the key is pressed)

This property is optional. If you do not specify Echo, Echo defaults to True.

Keyboard Type (16 Chars.)

The type of keyboard. This field is only necessary if you have more than one type of keyboard on the same system.

If you do have more than one type of keyboard, use Keyboard Type 1 for your first type of keyboard, use a separate number for each type (1, 2, 3, etc.), and define keys for each keyboard. You can use the default (Type 0) for common keys.

Keyboards

You can use non-standard keyboards (as well as multiple keyboards) to control Citect-SCADA. You can also define key names.

Using non-standard keyboards

You can use many types of keyboards with your runtime system. The most common keyboards are IBM compatible keyboards; this type of keyboard is used as a default. Many industrial keyboards do not conform with this standard; if you use a non-standard keyboard, you need to define each of the keyboards in the database.

Using multiple keyboards

Sometimes you might need to use keyboards of different types; for example, an IBM compatible keyboard in your control room and a sealed-membrane keyboard on the plant floor. If you use more than one type of keyboard, you need to define every key for each keyboard and assign each keyboard type to the respective computer.

You need to set the keyboard type for each computer, with the `[Keyboard]Type` parameter.

> **Note:** If you define commands that use mouse buttons, test that a double-click command cannot be mistaken for a single-click command. A double-click is an extension of a single click; a single click message will be received before a double-click message.

Defining key names

You can refer to a keyboard key by a meaningful name, rather than by the key itself. For example, you can refer to the F1 key as the "Help" key and the F2 key as the "Login" key. When you use the key in a command, you can use the name you have defined.

You can assign a key name to any key on the keyboard (including the alphanumeric keys A - Z, a -z, and 0 - 9). However, after a key is defined as a command key it can only be used as a command key. If you do assign a definition to an alphanumeric key (for example the character A), that key can not be used as a data key. Each time it is pressed, the definition for the key is recognized and not the keyboard character itself. Keyboard key definitions are usually only used with non-alphanumeric characters (for example the function keys).

See Also

Defining Key Sequences for Commands
Defining Commands and Controls

Defining Key Sequences for Commands

To define a command, you need to specify the key sequence that the operator types to issue the command. You can specify a single key for the key sequence, for example, the function key F2:

Key Sequence	F2

Alternatively, you can specify several keys that needs to be typed in sequence, for example, the function key F2 followed by the Enter key:

Key Sequence	F2 Enter

Note: If you use more than one key for the sequence, you need to separate each key with a space.

You will want to avoid the ambiguity in keyboard commands that can occur if you define separate commands that use a common key. For example, if you define a key sequence for one command as F3, and the key sequence for a second command as F3 F4, then when F3 is pressed, the first command would execute immediately; the second command could not execute. To avoid overlapping keyboard commands, add a **delimiter** to common keyboard commands, for example:

Key Sequence	F3 Enter
Command	SP1 = 50;

Key Sequence	F3 F4 Enter
Command	SP1 = 100;

These commands do not execute until the operator types the delimiter (the Enter key).

See Also

Using a hot key
Using variable data input
Passing multiple arguments
Passing keyboard arguments to functions

Using a hot key

A command defined with a 'hot' key executes immediately the key is pressed. You can only define a single key in the key sequence, and only use a 'hot' key to define commands that act on the current keyboard buffer (for example the Backspace and Delete keys). To define a 'hot' key, prefix the key sequence with an asterisk (*) as in the following example:

Key Sequence	*Backspace
Command	KeyBs()

At run time, this command operates in exactly the same way as the Backspace key on a standard computer keyboard, to correct typing errors. (Each time the Backspace key is pressed, the last key in the command line is removed.)

Note: You can only define a 'hot' key command as a system (global) command.

See Also

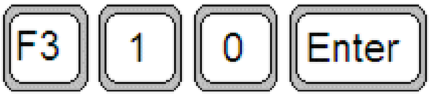
Using variable data input

Using variable data input

You can configure a keyboard command to accept variable data at run time. When the system is running, an operator can enter a value with the command, and the value is passed as an argument (or arguments) into the Cicode command. You can therefore define a single keyboard command that your operators use with different values. For example, you could define the following command to set the variable SP1 at run time:

Key Sequence	F3 ### Enter
Command	SP1 = Arg1;

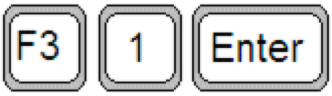
In this example, an operator can set the variable SP1 to a new value by first pressing the F3 function key, entering the new value, and pressing the Enter key, for example:



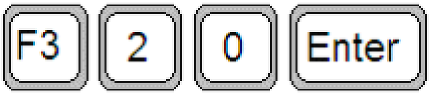
sets the value of SP1 to 10.

Each `#' character (in the Key Sequence) represents one keyboard character that an operator can enter in the command. In the above example, the operator can enter up to three keyboard characters when issuing the command. (The number of # characters determines the maximum number of characters that an operator can enter for the argument; if the operator enters more than three characters, an "Invalid Command" alert message displays.)

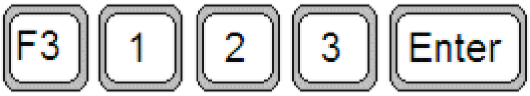
The command in the above example could be issued as follows:



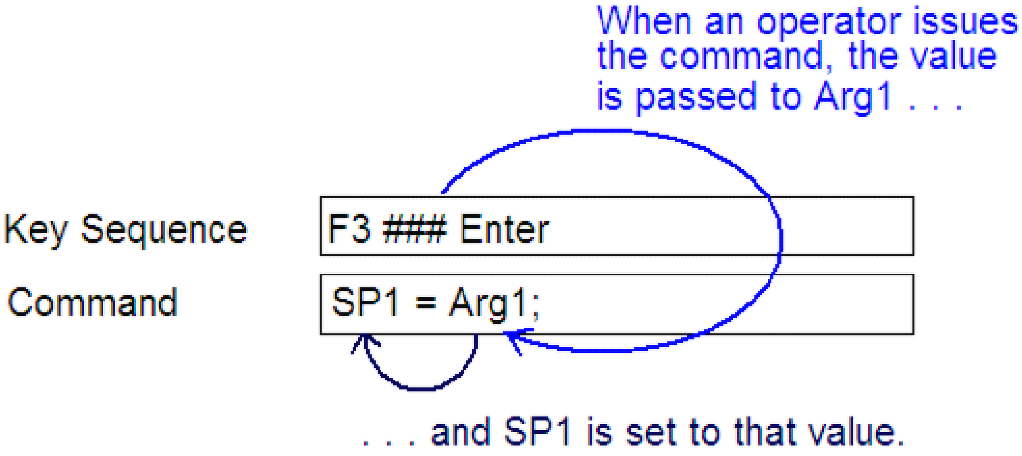
or



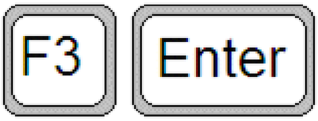
or



When the command is issued (the operator presses the Enter key), the value is passed to the command at Arg1.


When an operator issues the command, the value is passed to Arg1 . . .
Key Sequence
F3 ### Enter
Command
SP1 = Arg1;
. . . and SP1 is set to that value.

Note: If an operator does not enter any data (i.e., the key sequence:



is used), the value of Arg1 is zero, and the variable is set to zero. To prevent this from happening, use the **ArgValue1** label, for example:

```
Command SP1 = ArgValue1;
```

The **ArgValue1** label checks for illegal input; if the input is invalid, the value of the variable is not changed. You can also use the StrToValue() function.

Be aware that the ArgValue1 label and the StrToValue() function halts the command. Any instructions following either the ArgValue1 label or the StrToValue() function do not execute.

See Also

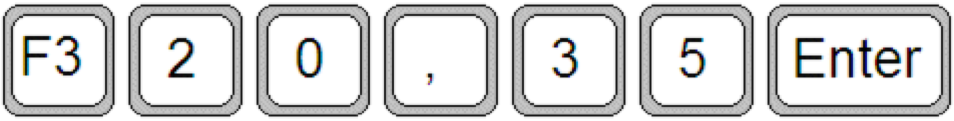
Passing multiple arguments

Passing multiple arguments

You can pass multiple arguments into a Cicode command by separating each argument with a comma (,). Each argument is passed into the command in sequence, and is referred to in the command field as **Arg1, Arg2, Arg3,** and so on. For example:

Key Sequence	F3 ###, ### Enter
Command	SP1 = Arg1; SP2 = Arg2;

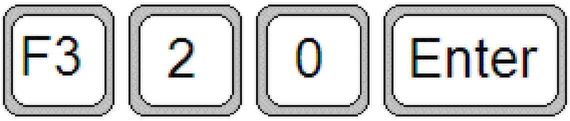
In this case, an operator can set two variables with the same command, for example:



sets the variables SP1 to 20 and SP2 to 35.

You can use up to eight arguments. However, avoid passing many arguments.

> **Note:** There is no way to check if the input for each argument is valid. If an operator does not enter any data for one of the arguments (i.e. the key sequence:



is used), the value of Arg2 is zero, and the second variable is set to zero. Do not use multiple arguments in a command if invalid input might generate undesired behavior in the project or in the larger system.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use keystroke sequences to pass multiple arguments to Cicode commands unless possible input combinations have been tested and determined to be safe.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

See Also

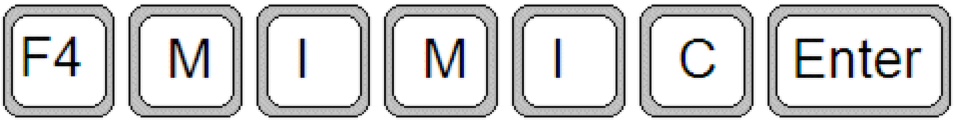
Passing keyboard arguments to functions

Passing keyboard arguments to functions

You can also pass arguments directly to functions at run time, as in the following example:

Key Sequence	F4 ######## Enter
Command	PageDisplay(Arg1);

In this example, an operator can select any graphics page (defined in the project) with a single command, for example:



selects the "Mimic" page.

Note: Keyboard arguments are passed as string values. If the command (or function) requires a numeric value, Cicode converts the string to a numeric value before it is used.

If you use variable data, the operator can only enter alphanumeric characters (A - Z, a-z, and 0 - 9) for the data. Do not use variable data input as the last item in a key sequence, as ambiguity could occur; always use a delimiter.

Configuring Page Menus

You can configure a menu structure and use it in your graphical pages. Using the built-in menu configuration dialog in Project Editor, you can define menu items in a hierarchical tree structure for your project. You can optionally assign individual menu items to a particular page, or make it available to every page. Menu items configured in included projects will be merged with the ones configured in the main project into a single menu tree structure when you compile your main project. This allows extra menu items to be automatically added to your project depending on what projects are included.

At runtime, two menu trees are created. The first menu tree merges the configured menu items for the generic pages and page specific items. You can access this menu tree using the Cicode function MenuGetGenericNode to get the page node for the generic pages in the menu tree and MenuGetPageNode to get the specified page node in the menu tree. The second menu tree automatically merges the menu items for the generic pages and the items specified for the current page. You can access this menu tree using Cicode function MenuGetWindowNode to get the root node of this menu tree. Through the combination of Cicode and graphical objects, you can create a custom menu system that suits your project.

Note: A set of page templates known as Tab Style Templates is provided with the product. These templates already contain a tabbed menu system. If your pages are based on these templates, the menu items configured in the menu configuration database will be displayed on the page automatically at runtime. If no menu is defined in the project, a default menu will be created at runtime to provide access to pages in the project. The default menu functionality can be changed using [Page]AddDefaultMenu parameter.

Note: For existing users of the CSV_Include project, the menu definition would have been defined in a file called Menu.dbf in your project folder. You can migrate the menu definition in this file to the built-in menu configuration database using the migration tool, although the CSV_Include project will continue to use the Menu.dbf file for its page templates. This step is useful if you plan to migrate your project from CSV_Include project to the newer set of built-in templates.

To configure menus for Tab Style templates, see Creating Custom Menus for Tab Style templates.

To configure menus for CSV templates, see Creating Custom Menus for CSV templates.

See Also

Menu Configuration Properties
Menu Cicode Functions

Menu Configuration Properties

The Menu Configuration dialog has the following properties:

Order

The relative position of a menu entry with respective to the other menu entries in the same hierarchy level. This affects the order that the menu entries are returned through the menu Cicode functions. If this field is left blank, it will take the default value of 0. The relevant menu entries will be returned at the start as a result. If more than one entry of the same order exists, they are displayed in the same order as they are defined in the database.

Level 1-4

Gives the menu hierarchy path of the entry. String length: 64 characters. Up to 4 levels of menu hierarchy are supported. Menu entries that have the same upper levels are considered to be under the same hierarchical branch.

Menu Command

The Cicode expression to be executed when the menu item is selected. String of 256 characters.

Symbol

The symbol to associate with the menu entry. The String length:128 characters. A symbol needs to be already defined in the project / included project, and specified in the format of <library name>.<symbol name>.

Comment

A comment about the menu entry. String of 128 characters.

Page

The page on which this entry will exist. If blank, the entry exists on every page.

Hidden when

Cicode expression to determine when the menu entry is hidden on the page. String of 256 characters.

Disabled when

Cicode expression to determine when the menu entry is disabled on the page. String of 256 characters.

Disabled style

A number to indicate how the disabled entry is displayed in the menu. It is up to the user to decide what the numbers mean.

Width

The width of the menu entry on the page. It is up to the user to decide what units to be used.

Checked

Whether the menu entry is initialized with a checked status.

Privilege

The user privilege of (0-8) necessary to select the menu entry.

Area

The user area (0-255) necessary to select the menu entry.

> **Note**:At runtime, you can use the Menu family of Cicode functions to access the information entered in the Menu Configuration dialog.

See Also

Defining Page Menus for Tab Templates
Menu Functions

Displaying Tags

You can apply visual cues to a tag displayed on a graphics page, providing greater emphasis to the data presented. For example, you could highlight a bad quality tag value by presenting it in a different color or style.

This type of activity is enabled using the [Page]IgnoreValueQuality parameter, which can be used to display a graphical representation of data quality.

[Page]IgnoreValueQuality can be used in tandem with a number of [Page] parameters to provide different visual outcomes based on the state of a tag. For example, you could change the text background color for a tag according to its current state using the following parameters:

- [Page]BadTextBackgroundColor
- [Page]ErrorTextBackgroundColor
- [Page]UncertainTextBackgroundColor
- [Page]OverideTextBackgroundColor
- [Page]ControlInhibitTextBackgroundColor

You can also use the parameter [Page]EnableQualityToolTip to enable tool tips that present quality and timestamp data when the cursor is held over a tag.

The Tag Extension Parameters page in the CitectSCADA Example Project provides an interactive example of how these parameters can be implemented.

See Also

Tag Extensions

Chapter: 23 Configuring and Processing Alarms

Protecting your plant equipment is a key role of CitectSCADA. The alarm facility constantly monitors equipment data and alerts operators when undesirable events occur, such as a device that is not responding to input commands.

Two types of alarms are supported:

- **Hardware alarms**. Diagnostic routines are continually run to check peripheral equipment, such as I/O Devices. Undesirable conditions and events are reported automatically to the operator. This facility is fully integrated within the product; no configuration is necessary.
- **Configured alarms**. Unlike hardware alarms, you need to configure the alarms that report undesirable conditions in your plant (for example, when a tank level is too high or when a motor overheats).

See Also

Configured alarms
Using alarm delay
Using custom alarm filters
Alarm Categories
Formatting an Alarm Display
Using Alarm Properties as Tags
Handling Alarms at Runtime

Configured alarms

You can use seven types of configured alarms:

- Digital Alarms
- Multi-digital Alarms
- Time-stamped Alarms
- Analog Alarms
- Advanced Alarms
- Time-stamped Digital Alarms
- Time-stamped Analog Alarms

You can process each alarm individually, or assign each of your alarms to separate categories, where they can be prioritized. You can then display, acknowledge, reset, and log alarms in a particular category.

You can also customize the order in which alarms will be displayed on the alarm summary page using the SummarySort and SummarySortMode parameters. (This order will override the alarm category priority order.)

To help operators with alarms, you can create graphics pages that contain information about the alarms (such as the action an operator needs to perform to correct the situation). You can display these pages automatically when the alarm occurs, or only when an operator uses the Alarm Help keyboard command.

Alarm properties can also be used anywhere a normal variable tag can be used. For example the status of an alarm could be used to change the color of a symbol on a graphics page.

See Also

Using alarm delay
Alarm Categories

Using alarm delay

The Alarm Delay property enables you to configure digital, analog, and advanced alarms so that they do not activate unless their triggering conditions remain true for a specified period.

For analog alarms, this means the analog variable need to fall within a specified range for the duration of the alarm delay period before an alarm will activate. In the case of digital and advanced alarms, the triggering condition of the digital variable or Cicode expression need to remain true for the delay period.

An example of where alarm delay could be useful is in monitoring temperature. By setting a delay period, you can filter out momentary alarms caused by the temperature moving in and out of different ranges.

See Also

Using custom alarm filters
Alarm Categories

Using custom alarm filters

You can define keywords for your alarm tags that you can then use to perform customized queries on your alarm data.

The Alarm Properties dialog allows you to define up to eight custom filters for each of the alarms in your system, allowing you to generate queries that filter your alarm data for specific information.

For example, you will want to prioritize any alarms that monitor equipment and conditions with the potential to cause a fire. You could set Custom Filter 1 for each relevant tag to "fire hazard". You could then call a Cicode function that requests alarms with a "custom filter 1" field equal to "fire hazard". The end result would be a list of alarms notifying an operator of any potentially flammable circumstances.

You could also set aside Custom Filter 2 to define the type of equipment the alarm is associated with, and label each alarm accordingly (for example "pump", "conveyor", etc.). Queries could then be created to list the alarms related to a particular type of machinery; for example, alarms associated with pumps.

Implementing queries that use custom alarm filters

Implementing a query based on the custom alarm filters you have defined requires two steps:

1. Create a Cicode function that performs the query you want to implement. The format of the query function you need to create is:
   ```
   INT FUNCTION Query (INT nRecordID, INT nVersion, STRING sInfo)
   ```
   where:

nRecordID	The value to be used in calls to AlarmGetFieldRec; for example AlarmGetFieldRec(nRID, "CUSTOM1", nVer). (See the help for AlarmGetFieldRec for details on the fields available.)
nVersion	The version of the record; this will increase each time a record is changed.
sInfo	A user defined string to be used to control the logic in the Query function.

   A 0 or 1 is expected as a return value, with 1 determining that the record will be displayed.
2. Set this query by calling it via the function AlarmSetQuery. Verify that you've defined the custom filters appropriately for your alarm tags before proceeding.

Example

You want to create a query that calls current alarms for the conveyors in your plant. This could be drawn from the alarm tags you have identified as being associated with a conveyor, by applying the keyword "conveyor" to the field **Custom Filter 1**.

The query function you would create would be as follows:

```
INT
FUNCTION CheckCustom1(INT nRid, INT nVer, STRING sValue)
```

```
        STRING sCustom;
        // Get the information in CUSTOM1
        sCustom = AlarmGetFieldRec(nRid, "CUSTOM1", nVer);
        IF sCustom = sValue THEN
                // Lets display this
                RETURN 1;
        ELSE
                // Skip over this
                RETURN 0;
        END
END
```

Say you then want to create a button on a graphics page that lists current conveyor alarms. You would implement this by applying the AlarmSetQuery function to the button.

```
AlarmSetQuery(0, "CheckCustom1", "conveyor");
```

You could also create a reset button that clears the current displayed list by cancelling the query:

```
AlarmSetQuery(-1, "", "")
```

You have the choice of calling a specific Cicode function, for example, "Customfeld1", with the argument "pumpcontrol", or using a more generic approach with the function "Checkfield" with argument "CUSTOM1= pumpcontrol". In this case, the Cicode function parses the passed string and checks the field specified.

Efficiency considerations

Cicode takes longer to implement than predefined filters. To avoid unnecessary processing of alarms that have already been checked, smart custom filters are enabled by default to minimize the processing necessary for large alarm record counts.

Smart custom filters mean that the first time a query is run, each alarm record will be checked by your function. Subsequent screen displays or record changes will then only call the function for changed or new records.

For most queries, this is how you would want your system to behave. However, there maybe special queries where you wish to turn this behavior off and check each alarm record.

An example might be if you were to request changed or new alarms in the last 10 seconds. In this case the "smart" filtering will not return the desired alarms because some records will not have changed, yet they may be included or excluded from your query.

Two mechanisms exist to control this:

- If you set the global ini parameter [ALARM]EnableSmartCustomFilters to 0, it will disable smart filters for every querie.
- On a per query basis, set an optional 4th argument to 1 or TRUE, for example:

```
AlarmSetQuery(0, "MyQueryTime", "10", TRUE);
```

This, when set to TRUE, forces the query to run. The default value is 0 or FALSE (allow for smart queries).

See Also

Alarm Categories

Alarm Categories

Each alarm in your system can be assigned to a category, and each category can be processed as a group. For each category, you can set alarm display details (font and page type), logging details (printer or data file), and the action to be taken when an alarm in the category is triggered (for example, activating an audible alarm).

Each category can have an associated priority. The alarm priorities can be used to order alarm displays, providing useful filtering for the operator.

You can also customize the order in which alarms will be displayed on the alarm summary page using the SummarySort and SummarySortMode parameters. (This order will override the alarm category priority order.)

You can configure up to 16376 alarm categories. If you do not specify a category for an alarm, the alarm has the same attributes as alarm category 0. If you do not define an alarm category 0, a default is used for the category.

To define an alarm category:

1. Choose **System** | **Alarm Categories**. The Alarm Categories dialog box appears.
2. Enter the alarm category properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.
   Use the Alarm Categories dialog box to configure your alarms.

See Also

Alarm Category Properties

Alarm Category Properties

Alarm Categories have the following properties:

Category Number

The alarm category (0-16375). Enter a category of 16 characters or less.

Category 254 is reserved for user-created alarm summary entries. Category 255 is reserved for hardware alarms.

Priority

The priority which will apply to alarms assigned to this alarm category (0- 255). Alarm priority governs the order in which alarms are displayed, acknowledged, enabled and so on. Enter a priority of 16 characters or less.

Priority 1 is the highest priority, and priority 255 is the lowest. For example, if alarms with priorities 1 to 8 were displayed, priority 1 alarms would be displayed first in their time/date order, then priority 2 alarms, then priority 3, and so on up to priority 8.

Priority 0 (zero) is the default priority and categories have priority zero (as well as the value entered in this field). Priority 0 is used to reference priorities. For example, to change the display parameters of an alarm list, so that alarms of priorities are displayed, **AlarmSetInfo** would be used with ***Type*** = 7, ***Value*** = 0.

Note: When priority 0 is used to display alarms of priorities, priority 0 only alarms will display first, followed by priority 1 alarms, then priority 2 etc. You can also customize the order in which alarms will be displayed on the alarm summary page using the SummarySort and SummarySortMode parameters. (This order will override the alarm category priority order.)

Display on Alarm Page

Defines whether or not alarms of this category will be displayed on the Alarm Page. The default for this field is TRUE. Enter a value of 6 characters or less.

Display on Summary Page

Defines whether or not alarms of this category will be displayed on the Summary Page. The default for this field is TRUE. Enter a value of 6 characters or less.

Unacknowledged: Alarm Off Font

The font to display alarms that are no longer active (but have not been acknowledged). This property is optional. If you do not specify a font, the font defaults to 10 pt. BROWN. Enter a value of 16 characters or less.

Acknowledged: Alarm Off Font

The font used to display alarms that are no longer active and have been acknowledged. This property is optional. If you do not specify a font, the font defaults to 10 pt. WHITE. Enter a value of 16 characters or less.

Unacknowledged: Alarm On Font

The font used to display an expression that has not been acknowledged. This property is optional. If you do not specify a font, the font defaults to 10 pt. YELLOW. Enter a value of 16 characters or less.

Acknowledged: Alarm On Font

The font used to display an active alarm that has been acknowledged. This property is optional. If you do not specify a font, the font defaults to 10 pt. CYAN. Enter a value of 16 characters or less.

Disabled Font

The font used to display disabled alarms. This property is optional. If you do not specify a font, the font defaults to 10 pt. WHITE. Enter a value of 16 characters or less.

ON Action

A Cicode command that is executed when an alarm of this Category is triggered. For example:

Alarm ON Action	STOP_PROCESS = 1;

The digital variable STOP_PROCESS is set to ON when an alarm in this category is triggered.

> **Note:** Do not put a Cicode blocking function in this field. The alarm system executes ON, OFF, or ACK actions within the polling loop. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not put a Cicode blocking function in this field. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

A special case of this command occurs when the Alarm ON Action is self-referring, with a form such as TAG1 = TAG1 + 1. This command will not work properly since tags are not reread before processing the Alarm On action (for performance reasons). This particular command will therefore initially set the value of TAG1 to 1 rather than incrementing it.

To correctly run a command of this type in the Alarm ON Action, use TaskNew() to run your own Cicode function to perform the tag command:

Alarm ON Action	TaskNew("MyFunc","Data",5);

OFF Action

A Cicode command that is executed when an alarm of this Category is reset. For example:

Alarm OFF Action	ENABLE_PROCESS = 1;

The digital variable ENABLE_PROCESS is set to ON when an alarm in this category is reset.

Note: Do not put a Cicode blocking function in this field. The alarm system executes ON, OFF, or ACK actions within the polling loop. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not put a Cicode blocking function in this field. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

ACK Action

A Cicode command that is executed when an alarm of this Category is acknowledged .

Note: Do not put a Cicode blocking function in this field. The alarm system executes ON, OFF, or ACK actions within the polling loop. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not put a Cicode blocking function in this field. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Alarm Format

The screen display format for alarms in this category. The Alarm Display format specifies how the data (for alarms in the category) are displayed on the alarms page (on the screen only). Each alarm displays on the alarms page in a single line, for example:

12:32:21RFP3 Raw Feed pump 3 Overload

The Display Format property is optional. If you do not specify a Alarm Display format, the format defaults to:

Format	{Time,12}^t {Tag,10}^t {Name,20}^t {Desc,32}

See Alarm display fields for the formatting details for each field type.

When alarms are displayed using variable width fonts (such as Arial or Helvetica), the alarm fields may not align properly across different rows. This problem can be avoided by using a field separator in the alarm format configuration instead of just a space. Tab characters, denoted by either "^t" (horizontal tab) or "^v" (vertical tab), between the alarm fields will act as alignment points in your alarm display.

You can change this default Alarm Display Format for alarms by setting the [Alarm]DefDspFmt parameter.

Note: If an alarm value is longer than the field it is to be displayed in, it will be truncated or replaced with the #OVR ("overflow of format width") alert message. When the alarm is logged to a device (i.e. printed or written to a file or database), the format specified for the logging device overrides the display format.

Summary Format

The summary display format for alarms in this category. The Summary Display format specifies how the alarm summary displays on the alarms summary page (on the screen only).

The display format is defined exactly as the Alarms Display Format. However, you can also use additional data fields.

This property is optional. If you do not specify a Summary Display format, the format defaults to:

Format	{Name,20}^t {OnTime,8}^t {OffTime,8}^t {DeltaTime,8}^t {Comment,22}

See Alarm summary fields for formatting details for each field type.

When alarms are displayed using variable width fonts (such as Arial or Helvetica), the alarm fields may not align properly across different rows. This problem can be avoided by using a field separator in the alarm format configuration instead of just a space. Tab characters, denoted by either "^t" (horizontal tab) or "^v" (vertical tab), between the alarm fields will act as alignment points in your alarm display.

You can change the default Summary Display Format for alarms by setting the [Alarm]DefSumFmt parameter.

> **Note:** When the alarm is logged to a summary device (i.e. printed or written to a file or database), the format specified for the logging device overrides the display format.

Alarm Acquisition Error

An acquisition error is an error related to acquiring data from I/O sub-system for the underlying tag and/or expression, for example, Device Offline, Tag Unknown, etc. (Integer length 6).

All acquisition errors are stored for each alarm record separately within the alarm server as a new AcqError field. They are not logged and no hardware alarm is raised from them. The AcqError field stores the error as the Citect error code (for example, CT_ ERROR_NO_ERROR). This field is made accessible through the alarm browse functionality.

If there are multiple acquisition errors for a single alarm record, then the first of these is stored within the system.

Summary Device

The device where the alarm summary is sent.

An alarm is logged to the summary device when it has gone off, and has been displayed for a longer period than the time specified in the [Alarm] SummaryTimeout parameter. When the alarm is logged to the device, it is removed from the alarm summary page.

When the alarm is printed, or written to a file or device, the format specified in the device overrides the display format.

This property is optional. If you do not specify a Summary Device, alarm summaries are not logged.

Log Device

The device where the alarm state changes are sent.

An alarm entry is made in the log device each time an alarm changes state (for example, on, off, acknowledged, enabled, and disabled).

When the alarm is printed, or written to a file or device, the format specified in the device overrides the display format.

This property is optional. If you do not specify a log device, alarm state changes are not logged.

Log Alarm Transitions: ON

Logs the alarm details when the alarm becomes active. The default for this field is TRUE.

Log Alarm Transitions: OFF

Logs the alarm details when the alarm becomes inactive. The default for this field is TRUE.

Log Alarm Transitions: ACK

Logs the alarm details when the alarm is acknowledged. The default for this field is TRUE.

Comment

Any useful comment.

Digital Alarms

You can configure digital alarms to activate based on the state of one or two digital variables. The alarm becomes active when the triggering condition spans the duration of a specified delay period.

The variables configured for a digital alarm are polled at the rate set by the Citect.ini parameter [Alarm]ScanTime. If an alarm state changes, notification will occur the next time the variables are polled. Be aware that the time associated with the alarm state will represent the time the variable was polled, not the actual time the alarm condition occurred.

To configure a digital alarm:

1. Choose **Alarm | Digital Alarms**. The Digital Alarms dialog box is displayed.
2. Complete the properties. See Digital Alarm Properties.
3. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

See Also

Digital Alarm Properties

Digital Alarm Properties

Digital Alarms have the following properties:

Alarm Tag

The name of the alarm. The name needs to be unique to the cluster. Alarm Tag names need to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

Note: Where Cluster Name is left blank, the name needs to be unique to defined clusters.

Cluster Name

The name of the cluster that runs this alarm. If the Cluster Name is not set, then this alarm will run on defined clusters.

Alarm Name

The name of the physical device associated with the alarm. This property is optional, it is only used when details of the alarm are displayed on the screen or logged to a device.

Alarm Desc

The description of the alarm. This can include variable data. This property is optional, it is only used when details of the alarm are displayed on the screen or logged to a device.

Var Tag A/Var Tag B

The digital variables (tags) that trigger the alarm. You can configure digital alarms to activate based on the state of one or two digital variables.

If you only use one variable to trigger the alarm, use the **Var Tag A** field. For example:

Var Tag A	RFP3_TOL

When the state of the variable RFP3_TOL changes to ON (1), the alarm is triggered.

Alternatively, you can define the alarm to trigger when the state of the variable changes to OFF (0), by preceding the digital address with the logical operator NOT, for example:

Var Tag A	NOT RFP3_TOL

In this case, the alarm is triggered when the state of the variable MCOL304 changes to OFF (0).

You can also configure digital alarms to activate based on the state of two digital variables, for example:

Var Tag A	RFP3_TOL
Var Tag B	NOT MCOL304

In this case, the alarm is triggered when the state of both variables changes to the active state: when the state of the variable RFP3_TOL changes to ON (1), and when the state of the variable MCOL304 changes to OFF (0).

Note: If you leave the **Var Tag B** property blank, only Var Tag A triggers the alarm.

Category

The alarm category number or label. This property is optional. If you do not specify a category, the alarm defaults to Category 0.

Delay (hh:mm:ss)

The alarm delay period.

A digital alarm becomes active when the state of the triggering condition remains true for the duration of the delay period. The active alarm has an ON time of when the state became true.

This property is optional. If you do not specify a delay period, the alarm is active as soon as it is triggered by the digital tag(s).

Note: The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Help

The name of the graphics page that displays when the AlarmHelp() function is called by a user-defined command. This property is optional. If you don't specify a help page, no action occurs when the AlarmHelp() function is called.

Comment

Any useful comment.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to acknowledge or disable the alarm.

Note: If you assign an acknowledgment privilege to an alarm, do not assign a privilege to the command(s) that acknowledge the alarm. If you assign a different privilege to the commands, an operator needs to have both privileges to acknowledge the command. More importantly, the area defined here may be ignored.

Area

The area to which the alarm belongs. If an operator does not have access to an area, the alarm is not visible on the alarm display. For example, if you enter **Area 1** here, operators need to have access to Area 1 (plus any necessary privileges) to acknowledge or disable this alarm.

Note: The area and privilege fields defined here needs to be designed to work in conjunction. A privilege defined on a button (say) will ignore the alarm defined area.

Custom Filter1...Custom Filter8

A user-defined string for filtering active alarms (maximum 64 characters).

Used in a custom Cicode query function as search criteria, the custom alarm filter enables operators to identify and display a subset of active alarms.

Notes:

- Custom filters are visible only when the Digital Alarms form is open in Extended mode.
- The fields are not case-sensitive and can contain 'A'..'Z', 'a'..'z', '0'..'9', and the underscore '_'.
- A custom filter cannot start with a digit.

Paging

A read/write property that indicates whether the alarm will be paged. When the value is 1 (TRUE) the alarm will be paged. The default value is 0 (FALSE). See Alarm Paging Properties. This property can be read using alarm tag browsing and read or modified when tag properties are enabled using the tag name "myCluster.myAlarm.paging".

Paging Group

A read only text string that indicates the paging group to which the alarm belongs. Maximum length is 80 characters. See your third-party paging system documentation for information on how to use this Paging Group string. This property can be read using alarm tag browsing or when tag properties are enabled read using the tagname "myCluster.myAlarm.paginggroup". For example, assign the value of PagingGroup to a variable:

```
myString = myCluster.Alarm_1.paginggroup
```

See Also

Using custom alarm filters

Multi-digital Alarms

Multi-digital alarms use the output from three digital variables (for example: tags A, B, and C) to define eight states. The states represent every possible combination of true/false values the variables can have.

The tag values in each state are represented in the order tag C, tag B, tag A. A true value is represented by the tag letter, and 0 (zero) represents false.

The eight states are as follows:

- **State 000 -** All 3 tags are false.
- **State 00A -** Tags C and B are false and Tag A is true.
- **State 0B0 -** Tags C and A are false and Tag B is true.
- **State 0BA -** Tag C is false and Tags B and A are true.
- **State C00** - Tag C is true and Tags B and A are false.
- **State C0A -** Tags C and A are true and Tag B is false.
- **State CB0 -** Tags C and B are true and Tag A is false.
- **State CBA -** All 3 tags are true.

When configuring the multi-digital alarm properties, you can set which states will trigger an alarm, and specify Cicode functions to be called when alarms become active and inactive.

The variables configured for a multi-digital alarm are polled at the rate set by the Citect.ini parameter [Alarm]ScanTime. If an alarm state changes, notification will occur the next time the variables are polled. The time associated with the alarm state will represent the time the variable was polled, not the actual time the alarm condition occurred.

Examples

In the following example, tag C is left blank, and the variables BIT_12 and BIT_1 are specified for tags A and B. With state 0BA specified to activate an alarm, when tags A and B change to ON (1) the alarm will activate.

Var Tag A	BIT_12

Var Tag B	BIT_1
Var Tag C	

In this example, variables are specified for the three tags. If state CBA is specified to activate an alarm, when the three variables change to ON (1) the alarm will activate.

Var Tag A	RFP3_TOL
Var Tag B	BIT_1
Var Tag C	MCOL_304

To configure a multi-digital alarm:

1. Choose **Alarm** | **Multi-digital Alarms**. The Multi-digital Alarms dialog box appears.
2. Enter the alarm properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Multi-digital Alarm Properties

Multi-digital Alarm Properties

Multi-digital Alarms have the following properties.

Alarm Tag

The name of the alarm. The name needs to be unique to the cluster. Alarm Tag names need to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

> **Note:** Where Cluster Name is left blank, the name needs to be unique to every defined cluster.

Cluster Name

The name of the cluster that runs this alarm. If the Cluster Name is not set, then this alarm will run on every defined cluster.

Alarm Name

The name of the physical device associated with the alarm.This property is optional, it is only used when details of the alarm are displayed on the screen or logged to a device.

Alarm Desc

The description of the alarm. This can include variable data. This property is optional, it is only used when details of the alarm are displayed on the screen or logged to a device.

Var Tag A/Var Tag B/Var Tag C

The digital variables used to define eight states. Each state represents a different combination of tag values.

In the following example, the digital variables RFP3_TOL, BIT_1, and MCOL304 are specified for Tags A, B, and C.

Var Tag A	RFP3_TOL
Var Tag B	BIT_1
Var Tag C	MCOL_304

States and Descriptions

The following eight states represent any possible tag value combinations (8 characters maximum for description and 1 character for state). The tags are represented in the order Tag C, Tag B, Tag A.

- **State 000 -** All 3 tags are false.
- **State 00A -** Tags C and B are false and Tag A is true.
- **State 0B0 -** Tags C and A are false and Tag B is true.
- **State 0BA -** Tag C is false and Tags B and A are true.
- **State C00 -** Tag C is true and Tags B and A are false.
- **State C0A -** Tags C and A are true and Tag B is false.
- **State CB0 -** Tags C and B are true and Tag A is false.
- **State CBA -** All 3 tags are true.

For each state, there are two fields on the Multi-Digital Alarms dialog. In the first field you can enter a description (for example Healthy or Stopped), with a maximum of eight characters.

In the second field, you indicate whether the state will trigger an alarm. A value of 1 indicates an alarm state, 0 indicates no alarm will be triggered.

Realarm (1 char.)

Indicates what happens when there is a transition from one alarm state to another. A value of 1 in this field causes a new time and State Description {State_Desc,n} to be recorded when the states change. With a value of 0, only the time and State Description of the first alarm state is recorded in the Alarm Summary.

ON Function (254 Chars.)

A Cicode function that is executed when a Multi-Digital Alarm becomes active, for example

ON Function	STOP_PROCESS = 1;

The digital variable STOP_PROCESS is set to ON when the alarm is triggered.

Note: Do not put a Cicode blocking function in this field. The alarm system executes ON or OFF actions within the polling loop. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not put a Cicode blocking function in this field. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

A special case of this command occurs when the Alarm ON Function is self-referring, with a form such as TAG1 = TAG1 + 1. This command will not work properly since tags are not reread before processing the Alarm On function (for performance reasons). This particular command will therefore initially set the value of TAG1 to 1 rather than incrementing it.

To correctly run a command of this type in the Alarm ON Function, use TaskNew() to run your own Cicode function to perform the tag command:

ON Function	TaskNew("MyFunc","Data",5);

OFF Function (254 Chars.)

A Cicode function that is executed when a Multi-Digital Alarm becomes inactive. For example,

OFF Function	ENABLE_PROCESS = 1;

The digital variable ENABLE_PROCESS is set to ON when an alarm in this category is reset.

Note: Do not put a Cicode blocking function in this field. The alarm system executes ON or OFF actions within the polling loop. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

Do not put a Cicode blocking function in this field. A blocking function will affect the polling of alarms, and may result in slow or delayed alarm processing.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Category

The alarm category number or label. This property is optional. If you do not specify a category, the alarm defaults to Category 0.

Help

The name of the graphics page that displays when the AlarmHelp() function is called by a user-defined command. This property is optional. If you don’t specify a help page, no action occurs when the AlarmHelp() function is called.

Privilege

The privilege necessary by an operator to acknowledge or disable the alarm.

Note: If you assign an acknowledgment privilege to an alarm, do not assign a privilege to the command(s) that acknowledge the alarm. If you do assign a different privilege to the commands, then an operator needs to have both privileges to acknowledge the command.

Area

The area to which the alarm belongs. If an operator does not have access to an area, the alarm is not visible on the alarm display. For example, if you enter Area 1 here, operators needs to have access to Area 1 (plus any necessary privileges) to acknowledge or disable this alarm.

Comment

Any useful comment.

Suppression

The number of the Suppression Group to which the alarm belongs. This is an integer value between 0–65535. Alarms in the same group display the same value in this field.

This property is used in conjunction with Suppression Level.

> **Note:** To assign a name to a Suppression Group, define the name as a label with an integer value.

Level (Suppression Level)

The level of an alarm within its Suppression Group (integer value). This is a value between 0 and 255, where a lower level represents a higher priority.

This property enables an active alarm to suppress lower priority alarms within the same Suppression Group. When this occurs, only the higher priority (lower level) alarms are displayed. Alarms with lower priorities (higher levels) will only activate and display when the higher priority (lower level) alarms become inactive.

If two alarms of different priorities in the same Suppression Group are triggered at the same time, both will display in the alarm list. This is because at the time they activated, the higher priority alarm was not already active and so could not suppress the lower priority alarm.

If your intent is for higher priority alarms to activate before lower priority alarms in spite of other factors, store the higher priority alarms closer to the beginning of the Alarms database. The database is scanned from beginning to end for triggered alarms, and if higher priority alarms are higher in the database, they will activate first and be able to suppress any lower priority alarms within the Suppression Group.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Custom Filter1...Custom Filter8

A user-defined string for filtering active alarms (maximum 64 characters).

Used in a custom Cicode query function as search criteria, the custom alarm filter enables operators to identify and display a sub-set of active alarms.

Notes:

- The custom filters are visible only when the Digital Alarms form is open in Extended mode.
- The fields are not case sensitive, and can contain 'A'..'Z', 'a'..'z', '0'..'9', and the underscore '_'.
- A custom filter cannot start with a digit.

Paging

A read/write property that indicates whether the alarm will be paged. When the value is 1 (TRUE) the alarm will be paged. The default value is 0 (FALSE). See Alarm Paging Properties. This property can be read using alarm tag browsing and read or modified when tag properties are enabled using the tag name "myCluster.myAlarm.paging".

Paging Group

A read only text string that indicates the paging group to which the alarm belongs. Maximum length is 80 characters. See your third-party paging system documentation for information on how to use this Paging Group string. This property can be read using alarm tag browsing or when tag properties are enabled read using the tagname "myCluster.myAlarm.paginggroup". For example, assign the value of PagingGroup to a variable:

```
myString = myCluster.Alarm_1.paginggroup
```

See Also

Using custom alarm filters

Time-stamped Alarms

Time-stamped alarms are similar to digital alarms, except that a counter is used to provide an accurate timestamp of when a triggering condition occurs, rather than just the time the variable was polled. Time-stamped alarms can only be associated with a single digital variable.

An alarm's variables are polled at the rate set by [Alarm]ScanTime, however, the timer value is used to define the time associated with a change of state.

You can use one of three types of counter or timer to record the triggering of time-stamped alarms:

- **Continuous counter**. A continuous counter is read in the unit to determine the sequence in which the alarms are triggered. The alarms are sorted based on the value of the counter when the alarm was triggered (the exact time is not recorded).
- **Millisecond counter**. If your unit supports a millisecond counter, you can program a counter (in the unit) to count (in milliseconds) for 24 hours, and then reset (at midnight). The value of this timer variable (in the unit) is read to determine the exact time when the alarm was triggered.
- **LONGBCD timer**. Using a LONGBCD timer, you can log the exact time when a time-stamped alarm becomes active. This variable is read, along with the alarm tag when the alarm activates.

To configure a time-stamped alarm:

1. Choose **Alarms** | **Time-stamped Alarms**. The Time-stamped Alarms dialog box appears.
2. Enter the alarm properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Time-stamped Alarm Properties

Time-stamped Alarm Properties

Time-stamped Alarms have the following properties:

Alarm Tag

The name of the alarm. The name needs to be unique to the cluster. Alarm Tag names need to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

Note: Where Cluster Name is left blank, the name needs to be unique for every defined cluster.

Cluster Name

The name of the cluster that runs this alarm. If the Cluster Name is not set, then this alarm will run on every defined cluster.

Alarm Name

The name of the physical device associated with the alarm. This property is optional, it is only used when details of the alarm are displayed on the screen or logged to a device.

Alarm Desc

The description of the alarm. This can include variable data. This property is optional, it is only used when details of the alarm are displayed on the screen or logged to a device.

Variable Tag

The digital variable (tag) that triggers the alarm.

Timer

The variable tag or Cicode expression that represents the counter (or millisecond timer) configured in the I/O Device. The counter needs to be configured and maintained by the program in the I/O Device; it is read only when the alarm is triggered.

You can use one of three types of counter or timer to record the triggering of time-stamped alarms:

- **Continuous counter**. A continuous counter is read in the unit to determine the sequence in which the alarms are triggered. The alarms are sorted based on the value of the counter when the alarm was triggered (the exact time is not recorded). You need to program the counter (in the unit) to count continually to its limit, reset, and again count to its limit.
- **Millisecond counter**. If your unit supports a millisecond counter, you can program a counter (in the unit) to count (in milliseconds) for 24 hours, and then reset (at midnight). The value of this timer variable (in the unit) is read to determine the exact time when the alarm was triggered.
- **LONGBCD timer**. Using a LONGBCD timer, you can log the exact time when a Time-stamped alarm becomes active. This variable is read, along with the alarm tag when the alarm activates. You need to program the LONGBCD variable in the following format with the range specified as hexadecimal numbers, excluding numbers containing alphabetic characters:

BYTE	MEANING	RANGE
1st	Hours	00-23 (most significant byte)
2nd	Minutes	00-59
3rd	Seconds	00-59
4th	100th/sec	00-99 (least significant byte)

- This field can also be used to handshake with the PLC code: The PLC is informed that it has read the timer register and now the PLC can overwrite the last value. For example, with the following code saved in a Cicode file:

```
INT
FUNCTION
AlarmTimerReset(INT iTimer, STRING sTimerTrigger)
TagWrite(sTimerTrigger, 0); //Reset the trigger
RETURN iTimer; //Return the timer value to the alarm system
END
```

You could configure a time-stamped alarm as follows:

Variable Tag	AlmTrigger1
Timer	AlarmTimer Reset(AlmTimer1, "AlmTrigger1")

where **AlmTimer1** is the PLC register that stores the alarm time, and **AlmTrigger1** is the alarm trigger bit.

When AlmTrigger1 is set to one (1), the alarm is triggered, and the Cicode function is called. On calling the function, the AlmTimer1 register is read. The function resets the trigger bit (handshaking), and the value of AlmTimer1 is returned to the alarm system.

Note:AlarmTimerReset is a USER Cicode function. This cicode function does not exist as an internal CitectSCADA function.

Category

The alarm category number or label. This property is optional. If you do not specify a category, the alarm defaults to Category 0.

Help

The name of the graphics page that displays when the AlarmHelp() function is called by a user-defined command. This property is optional. If you don't specify a help page, no action occurs when the AlarmHelp() function is called.

Comment

Any useful comment (maximum 48 characters).

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to acknowledge or disable the alarm.

Note: If you assign an acknowledgment privilege to an alarm, do not assign a privilege to the command(s) that acknowledge the alarm. If you do assign a different privilege to the commands, an operator need to have both privileges to acknowledge the command.

Area

Area to which the alarm belongs (maximum 16 characters). If an operator does not have access to an area, the alarm is not visible on the alarm display. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to acknowledge or disable this alarm.

Custom Filter1...Custom Filter8

A user-defined string for filtering active alarms (maximum 64 characters).

Used in a custom Cicode query function as search criteria, the custom alarm filter enables operators to identify and display a subset of active alarms.

Notes:

- Custom filters are visible only when the Digital Alarms form is open in Extended mode.
- The fields are not case-sensitive and can contain 'A'..'Z', 'a'..'z', '0'..'9', and the underscore '_'.
- A custom filter cannot start with a digit.

Paging

A read/write property that indicates whether the alarm will be paged. When the value is 1 (TRUE) the alarm will be paged. The default value is 0 (FALSE). See Alarm Paging Properties. This property can be read using alarm tag browsing and read or modified when tag properties are enabled using the tag name "myCluster.myAlarm.paging".

Paging Group

A read only text string that indicates the paging group to which the alarm belongs. Maximum length is 80 characters. See your third-party paging system documentation for information on how to use this Paging Group string. This property can be read using alarm tag browsing or when tag properties are enabled read using the tagname "myCluster.myAlarm.paginggroup". For example, assign the value of PagingGroup to a variable:

```
myString = myCluster.Alarm_1.paginggroup
```

See Also

Using custom alarm filters

Analog Alarms

Analog alarms are triggered when an analog variable changes beyond one or more specific limits. Each alarm can be configured as any combination of the following types.

- **high** and **high high** alarms - where the value reaches an atypical high
- **low** and **low low** alarms - where the value reaches an atypical low
- **deviation** alarm - where the values moves away from a predefined set point
- **rate of change** alarm - where a dramatic value change occurs within a specified period of time

The variables configured for an analog alarm are polled at the rate set by the Citect.ini parameter [Alarm]ScanTime. If an alarm state triggers, notification will occur the next time the variables are polled. Be aware that the time associated with the alarm state will represent the time the variable was polled, not the actual time the alarm condition occurred.

To configure an analog alarm:

1. Choose **Alarms** | **Analog Alarms**. The Analog Alarms dialog box appears.
2. Enter the alarm properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Analog Alarm Properties

Analog Alarm Properties

Analog Alarms have the following properties:

Alarm Tag

The name of the alarm. The name needs to be unique to the cluster. Alarm Tag names needs to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

> **Note:** Where Cluster Name is left blank, the name needs to be unique for defined clusters.

Cluster Name

The name of the cluster that runs this alarm. If the Cluster Name is not set, then this alarm will run on defined clusters.

Alarm Name

The name of the physical device associated with the alarm.This is an optional properties. It is only used when details of the alarm are displayed on the screen or logged to a device.

Variable Tag

The analog variable (tag) that triggers the alarm.

Setpoint

An analog variable tag or base value that determines if a deviation alarm is to be triggered. This property is optional. If you do not specify a setpoint, it will default to 0 (zero).

High High

The value used as the triggering condition for a high high alarm. The high high alarm becomes active when the value of the variable tag exceeds this value for the duration of the high high delay period. The active alarm has an ON time of when the tag exceeded the high high value.

Because a high alarm needs to precede a high high alarm, when the high high alarm is triggered it replaces the high alarm. If you want an analog alarm to display more than one state on the alarm page at the same time, configure a separate alarm for each state. (Each alarm would monitor the same tag.)

High High Delay

The delay period for High High Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the high high alarm will be activated as soon as the tag exceeds the high high value.

> **Note:** The delay period needs to be entered in the format HH:MM:SS (hours:minutes:seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

High

The value used as the triggering condition for a high alarm. The high alarm becomes active when the value of the variable tag exceeds this value for the duration of the high delay period. The active alarm has an ON time of when the tag exceeded the high value.

High Delay

The delay period for high alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the high alarm will be activated as soon as the tag exceeds the high value.

> **Note:** The delay period needs to be entered in the format HH:MM:SS (hours:minutes:seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

When a tag value increases from high to high high within the high delay period, the delay timer is reset. The high high alarm is only activated if the value remains in the high high range for the delay period.

When the value increases from high to high high after the high delay period has expired, a high alarm is activated and then the delay period for the high high alarm begins.

If the tag value exceeds the high high value and then falls below it before the high high delay period expires, at the time it falls, the high alarm is triggered immediately. It has an ON time of when the tag value exceeded the high high value.

These points also apply to tag values traveling between Low and Low Low ranges.

Low

The value used as the triggering condition for a Low Alarm. A Low Alarm becomes active when the value of the Variable Tag drops below this value and remains there for the duration of the Low Delay period. The active alarm has an ON time of when the tag fell below the Low value.

Low Delay

The delay period for Low Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the Low Alarm is activated as soon as the tag drops below the Low value.

Note: The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Low Low

The value used as the triggering condition for a Low Low Alarm. A Low Low Alarm becomes active when the value of the Variable Tag drops below this value and remains there for the duration of the Low Low Delay period. The active alarm has an ON time of when the tag fell below the Low Low value.

Because a Low Alarm needs to precede a Low Low Alarm, when the Low Low Alarm is triggered it replaces the Low Alarm. If you want an analog alarm to display more than one state on the alarm page at the same time, configure a separate alarm for each state. (Each alarm would monitor the same tag.)

Low Low Delay

The delay period for Low Low Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the Low Low Alarm is activated as soon as the tag drops below the Low Low value.

Note: The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Deviation

The value used as the triggering condition for a Deviation Alarm. A Deviation Alarm is activated when the value of the Variable Tag remains outside the deviation range (determined by the Setpoint) for the duration of the Deviation Delay period.

This property is optional. If you do not specify a deviation, no Deviation Alarm is activated.

Deviation Delay

The delay period for Deviation Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the Deviation Alarm is activated as soon as the Variable Tag falls outside the deviation range.

> **Note:** The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Rate

By dividing this value by the alarm period, the "maximum rate" at which the value of the variable tag can change is determined. At each Scan Time, the value of the tag is checked. If its rate of change is greater than the maximum rate, a Rate of Change Alarm is triggered.

For example, to verify that a tank does not fill too quickly, you might configure a rate of change alarm, using a Rate of **300** liters, an **[Alarm]Period** of **60** seconds, and an **[Alarm]ScanTime** of **1** second. This means that the maximum allowable rate of change for the tank level is **5** l/sec (300 liters / 60 seconds). The actual rate of change at each ScanTime is calculated. that is, Every second, it checks the current level of the tank and compares it to the level recorded a second earlier. If the actual rate of change is, say, 8 l/sec, a Rate of Change Alarm is triggered immediately.

The variable tags deadband % needs to be less than the alarms rate divided by the engineering scale of the variable tag. Otherwise, the rate of change alarm will only go off when the change in the variable tag exceeds the deadband value.

This property is optional. If you do not specify a value, no Rate of Change Alarm is activated.

Deadband

The value that **Variable Tag** needs to return to before the Alarm becomes inactive.

Format

The display format of the value (of the variable) when it is displayed on a graphics page, written to a file or passed to a function (that expects a string).

This property is optional. If you do not specify a format, the format defaults to the format specified for Variable tag.

Category

The alarm category number or label. This property is optional. If you do not specify a category, the alarm defaults to Category 0.

Help

The name of the graphics page that displays when the AlarmHelp() function is called by a user-defined command. This property is optional. If you don't specify a help page, no action occurs when the AlarmHelp() function is called.

Comment

Any useful comment.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to acknowledge or disable the alarm.

> **Note:** If you assign an acknowledgment privilege to an alarm, do not assign a privilege to the command(s) that acknowledge the alarm. If you do assign a different privilege to the commands, then an operator needs to have both privileges to acknowledge the command.

Area

The area to which the alarm belongs. If an operator does not have access to an area, the alarm is not visible on the alarm display. For example, if you enter Area 1 here, operators needs to have access to Area 1 (plus any necessary privileges) to acknowledge or disable this alarm.

Custom Filter1...Custom Filter8

A user-defined string for filtering active alarms (maximum 64 characters).

Used in a custom Cicode query function as search criteria, the custom alarm filter enables operators to identify and display a sub-set of active alarms.

Notes:

- The custom filters are visible only when the Digital Alarms form is open in Extended mode.
- The fields are not case-sensitive and can contain 'A'..'Z', 'a'..'z', '0'..'9', and the underscore '_'.
- A custom filter cannot start with a digit.

Paging

A read/write property that indicates whether the alarm will be paged. When the value is 1 (TRUE) the alarm will be paged. The default value is 0 (FALSE). See Alarm Paging Properties. This property can be read using alarm tag browsing and read or modified when tag properties are enabled using the tag name "myCluster.myAlarm.paging".

Paging Group

A read only text string that indicates the paging group to which the alarm belongs. Maximum length is 80 characters. See your third-party paging system documentation for information on how to use this Paging Group string. This property can be read using alarm tag browsing or when tag properties are enabled read using the tagname "myCluster.myAlarm.paginggroup". For example, assign the value of PagingGroup to a variable:

```
myString = myCluster.Alarm_1.paginggroup
```

See Also

Using custom alarm filters

Advanced Alarms

An advanced alarm becomes active when the result of the Cicode expression changes. The expression is polled at the rate set by the Citect.ini parameter [Alarm]ScanTime and tests for a change in outcome. If one occurs, an alarm state notification will occur.

To configure an advanced alarm:

1. From the **System** | **Advanced Alarms**. The Advanced Alarms dialog box appears.
2. Enter the alarm properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Advanced Alarm Properties

Advanced Alarm Properties

Advanced Alarms have the following properties.

Alarm Tag

The name of the alarm. The name needs to be unique to the cluster. Alarm Tag names need to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

Note: Where Cluster Name is left blank, the name needs to be unique for defined clusters.

Cluster Name

The name of the cluster that runs this alarm. If the Cluster Name is not set, then this alarm will run on ever defined clusters.

Alarm Name

The name of the physical device associated with the alarm. This property is optional. It is only used when details of the alarm are displayed on the screen or logged to a device.

Alarm Desc

The description of the alarm. This can include variable data. This property is optional. It is only used when details of the alarm are displayed on the screen or logged to a device.

Expression

The Cicode expression that triggers the alarm. Whenever the result of the expression changes to TRUE, the alarm is triggered.

Category

The alarm category number or label.

This property is optional. If you do not specify a category, the alarm defaults to Category 0.

Delay

The delay period for the Advanced Alarm.

An Advanced Alarm becomes active when the result of the Cicode expression triggering the alarm remains TRUE for the duration of the delay period. The active alarm has an ON time of when the expression returned TRUE.

This property is optional. If you do not specify a value, the alarm becomes active as soon as the triggering expression becomes true.

Note: The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00::00) and 24 hours (24:00:00).

Help

The name of the graphics page that displays when the AlarmHelp() function is called by a user-defined command. This property is optional. If you don't specify a help page, no action occurs when the AlarmHelp() function is called.

Comment

Any useful comment.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege needed by an operator to acknowledge or disable the alarm.

> **Note:** If you assign an acknowledgment privilege to an alarm, do not assign a privilege to the command(s) that acknowledge the alarm. If you do assign a different privilege to the commands, then an operator needs to have both privileges to acknowledge the command.

Area

The area to which the alarm belongs. If an operator does not have access to an area, the alarm is not visible on the alarm display. For example, if you enter Area 1 here, operators needs to have access to Area 1 (plus any required privileges) to acknowledge or disable this alarm.

Custom Filter1...Custom Filter8

A user-defined string for filtering active alarms.

Used in a custom Cicode query function as search criteria, the custom alarm filter expression enable operators to identify and display a sub-set of active alarms.

> **Note**:
> **1)** The custom filters are visible only when the Digital Alarms form is open in Extended mode.
> **2)** The fields are not case sensitive, and can contain 'A'..'Z', 'a'..'z', '0'..'9', and the underscore '_'.
> **3)** A custom filter cannot start with a digit.

Paging

A read/write property that indicates whether the alarm will be paged. When the value is 1 (TRUE) the alarm will be paged. The default value is 0 (FALSE). See Alarm Paging Properties. This property can be read using alarm tag browsing and read or modified when tag properties are enabled using the tag name "myCluster.myAlarm.paging".

Paging Group

A read only text string that indicates the paging group to which the alarm belongs. Maximum length is 80 characters. See your third-party paging system documentation for information on how to use this Paging Group string. This property can be read using alarm tag browsing or when tag properties are enabled read using the tagname "myCluster.myAlarm.paginggroup". For example, assign the value of PagingGroup to a variable:

```
myString = myCluster.Alarm_1.paginggroup
```

See Also

Using custom alarm filters

Time-stamped Digital Alarms

Time-stamped digital and time-stamped analog alarms differ to other alarm types, as they do not rely on the polling of variables to determine alarm conditions. They operate via a process where the Alarm Server is notified of any value changes to a specified variable using the AlarmNotifyVarChange Cicode function.

The Alarm Server uses this information to update alarms that monitor the variable. This process allows an accurate timestamp to be associated with an alarm condition.

This process updates the **Var Tag A** and **Var Tag B** properties for time-stamped digital alarms.

Events trends can be used in conjunction with time-stamped digital alarms to provide millisecond accuracy for both trend and alarm data. See TrnEventSetTable and TrnEventSetTableMS Cicode functions.

To configure a time-stamped digital alarm:

1. Choose **Alarm** | **Time-Stamped Digital Alarms**. The Time-Stamped Digital Alarms dialog box appears.
2. Complete the properties in the form that appears.
3. Click the **Add** button to append a new record, or **Replace** if you have modified a record.

> **Note:** For time-stamped digital alarms to function correctly, you need to configure the Cicode function AlarmNotifyVarChange, as it notifies the alarm server of any value changes for the associated variable. It also passes the timestamp with the notification message. The AlarmServer maintains a queue for notification messages and runs a separate task to check this queue.

See Also

Time-stamped Digital Alarm Properties

Time-stamped Digital Alarm Properties

Time-stamped Digital Alarms have the following properties:

Alarm Tag

The name of the alarm. The name needs to be unique to the cluster. Alarm Tag names need to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

> **Note:** Where Cluster Name is left blank, the name needs to be unique to evry defined cluster.

Cluster Name

The name of the cluster that runs this alarm. If the Cluster Name is not set, then this alarm will run on every defined cluster.

Alarm Name

The name of the physical device associated with the alarm. This property is optional. It is only used when details of the alarm are displayed on the screen or logged to a device.

Alarm Desc

The description of the alarm. This can include variable data. This property is optional. It is only used when details of the alarm are displayed on the screen or logged to a device.

Var Tag A/Var Tag B

The digital variables (tags) that trigger the alarm. You can configure time-stamped digital alarms to activate based on the state of one or two digital variables.

If you only use one variable to trigger the alarm, use the **Var Tag A** field. For example:

Var Tag A	RFP3_TOL

When the state of the variable RFP3_TOL changes to ON (1), the alarm is triggered.

Alternatively, you can define the alarm to trigger when the state of the variable changes to OFF (0), by preceding the digital address with the logical operator NOT, for example:

Var Tag A	NOT RFP3_TOL

In this case, the alarm is triggered when the state of the variable MCOL304 changes to OFF (0).

You can also configure digital alarms to activate based on the state of two digital variables, for example:

Var Tag A	RFP3_TOL
Var Tag B	NOT MCOL304

In this case, the alarm is triggered when the state of both variables changes to the active state: when the state of the variable RFP3_TOL changes to ON (1), and when the state of the variable MCOL304 changes to OFF (0).

Note: If you leave the **Var Tag B** property blank, only Var Tag A triggers the alarm.

Category

The alarm category number or label. This property is optional. If you do not specify a category, the alarm defaults to Category 0.

Delay (hh:mm:ss)

The alarm delay period

A time-stamped digital alarm becomes active when the state of the triggering condition remains true for the duration of the delay period. The active alarm has an ON time of when the state became true.

This property is optional. If you do not specify a delay period, the alarm is active as soon as it is triggered by the digital tag(s).

Note: The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Help

The name of the graphics page that displays when the AlarmHelp() function is called by a user-defined command. This property is optional. If you don’t specify a help page, no action occurs when the AlarmHelp() function is called.

Comment

Any useful comment.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to acknowledge or disable the alarm.

> **Note:** If you assign an acknowledgment privilege to an alarm, do not assign a privilege to the command(s) that acknowledge the alarm. If you assign a different privilege to the commands, an operator need to have both privileges to acknowledge the command. More importantly, the area defined here may be ignored.

Area

The area to which the alarm belongs. If an operator does not have access to an area, the alarm is not visible on the alarm display. For example, if you enter **Area 1** here, operators need to have access to Area 1 (plus any necessary privileges) to acknowledge or disable this alarm.

> **Note:** The area and privilege fields defined here needs to be designed to work in conjunction. A privilege defined on a button (say) will ignore the alarm defined area.

Custom Filter1...Custom Filter8

A user-defined string for filtering active alarms (maximum 64 characters).

Used in a custom Cicode query function as search criteria, the custom alarm filter enables operators to identify and display a sub-set of active alarms.

Notes:

- The custom filters are visible only when the Digital Alarms form is open in Extended mode.
- The fields are not case sensitive, and can contain 'A'..'Z', 'a'..'z', '0'..'9', and the underscore '_'.
- A custom filter cannot start with a digit.

Paging

A read/write property that indicates whether the alarm will be paged. When the value is 1 (TRUE) the alarm will be paged. The default value is 0 (FALSE). See Alarm Paging Properties. This property can be read using alarm tag browsing and read or modified when tag properties are enabled using the tag name "myCluster.myAlarm.paging".

Paging Group

A read only text string that indicates the paging group to which the alarm belongs. Maximum length is 80 characters. See your third-party paging system documentation for information on how to use this Paging Group string. This property can be read using alarm tag browsing or when tag properties are enabled read using the tagname "myCluster.myAlarm.paginggroup". For example, assign the value of PagingGroup to a variable:

```
myString = myCluster.Alarm_1.paginggroup
```

See Also

Using custom alarm filters

Time-stamped Analog Alarms

Time-stamped digital and time-stamped analog alarms differ to other alarm types, as they do not rely on the polling of variables to determine alarm conditions. They operate via a process where the Alarm Server is notified of any value changes to a specified variable using the Cicode function AlarmNotifyVarChange.

The Alarm Server will then use this information to update the alarms that monitor the variable. This process allows for an accurate timestamp to be associated with an alarm condition.

This process is used to update the **Variable Tag** and **Setpoint** properties for time-stamped analog alarms.

Events trends can be used in conjunction with time-stamped analog alarms to provide millisecond accuracy for both trend and alarm data. See TrnEventSetTable and TrnEventSetTableMS.

To configure an analog time-stamped alarm:

1. Choose **Alarms** | **Analog Time-stamped Alarms**. The Analog Time-stamped Alarms dialog box appears.
2. Enter the alarm properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

> **Note:** For time-stamped analog alarms to function correctly, you need to configure the Cicode function AlarmNotifyVarChange, as it notifies the alarm server of any value changes for the associated variable. It also passes the timestamp with the notification message. The AlarmServer maintains a queue for notification messages and runs a separate task to check this queue.

See Also

Time-stamped Analog Alarm Properties

Time-stamped Analog Alarm Properties

Time-stamped Analog Alarms have the following properties:

Alarm Tag

The name of the alarm. The name needs to be unique to the cluster. Alarm Tag names need to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

> **Note:** Where Cluster Name is left blank, the name needs to be unique to every defined cluster.

Cluster Name

The name of the cluster that runs this alarm. If the Cluster Name is not set, then this alarm will run on every defined cluster.

Alarm Name

The name of the physical device associated with the alarm. This property is optional. It is only used when details of the alarm are displayed on the screen or logged to a device.

Variable Tag

The analog variable (tag) that triggers the alarm.

Setpoint

An analog variable tag or base value that determines if a deviation alarm is to be triggered. This property is optional. If you do not specify a setpoint, it will default to 0 (zero).

High High

The value used as the triggering condition for a high high alarm. The high high alarm becomes active when the value of the variable tag exceeds this value for the duration of the high high delay period. The active alarm has an ON time of when the tag exceeded the high high value.

Because a high alarm needs to precede a high high alarm, when the high high alarm is triggered it replaces the high alarm. If you want an analog alarm to display more than one state on the alarm page at the same time, configure a separate alarm for each state. (Each alarm would monitor the same tag.)

High High Delay

The delay period for High High Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the high high alarm will be activated as soon as the tag exceeds the high high value.

Note: The delay period needs to be entered in the format HH:MM:SS (hours:minutes:seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

High

The value used as the triggering condition for a high alarm. The high alarm becomes active when the value of the variable tag exceeds this value for the duration of the high delay period. The active alarm has an ON time of when the tag exceeded the high value.

High Delay

The delay period for high alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the high alarm will be activated as soon as the tag exceeds the high value.

Note: The delay period needs to be entered in the format HH:MM:SS (hours:minutes:seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

When a tag value increases from high to high high within the high delay period, the delay timer is reset. The high high alarm is only activated if the value remains in the high high range for the delay period.

When the value increases from high to high high after the high delay period has expired, a high alarm is activated and then the delay period for the high high alarm begins.

If the tag value exceeds the high high value and then falls below it before the high high delay period expires, at the time it falls, the high alarm is triggered immediately. It has an ON time of when the tag value exceeded the high high value.

These points also apply to tag values travelling between Low and Low Low ranges.

Low

The value used as the triggering condition for a Low Alarm. A Low Alarm becomes active when the value of the Variable Tag drops below this value and remains there for the duration of the Low Delay period. The active alarm has an ON time of when the tag fell below the Low value.

Low Delay

The delay period for Low Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the Low Alarm is activated as soon as the tag drops below the Low value.

> **Note:** The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Low Low

The value used as the triggering condition for a Low Low Alarm. A Low Low Alarm becomes active when the value of the Variable Tag drops below this value and remains there for the duration of the Low Low Delay period. The active alarm has an ON time of when the tag fell below the Low Low value.

Because a Low Alarm needs to precede a Low Low Alarm, when the Low Low Alarm is triggered it replaces the Low Alarm. If you want an analog alarm to display more than one state on the alarm page at the same time, configure a separate alarm for each state. (Each alarm would monitor the same tag.)

Low Low Delay

The delay period for Low Low Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the Low Low Alarm is activated as soon as the tag drops below the Low Low value.

> **Note:** The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Deviation

The value used as the triggering condition for a Deviation Alarm. A Deviation Alarm is activated when the value of the Variable Tag remains outside the deviation range (determined by the Setpoint) for the duration of the Deviation Delay period.

This property is optional. If you do not specify a deviation, no Deviation Alarm is activated.

Deviation Delay

The delay period for Deviation Alarms. The alarm will only activate if its triggering condition is met for the duration of this period.

This property is optional. If you do not set a value, the Deviation Alarm is activated as soon as the Variable Tag falls outside the deviation range.

> **Note:** The delay period needs to be entered in the format HH:MM:SS (Hours:Minutes:Seconds). The value needs to be between 0 seconds (00:00:00) and 24 hours (24:00:00).

Rate

By dividing this value by the alarm period, the "maximum rate" at which the value of the variable tag can change is determined. At each Scan Time, the value of the tag is checked. If its rate of change is greater than the maximum rate, a Rate of Change Alarm is triggered.

For example, to minimize the chance that a tank will fill too quickly you might configure a rate of change alarm, using a Rate of 300 liters, an `[Alarm]Period` of 60 seconds, and an `[Alarm]ScanTime` of 1 second. This means that the maximum allowable rate of change for the tank level is 5 l/sec (300 liters/60 seconds). The actual rate of change is calculated at each ScanTime; that is, every second it checks the current level of the tank and compares it to the level recorded a second earlier. If the actual rate of change is, say, 8 l/sec, a Rate of Change Alarm is triggered immediately.

This property is optional. If you do not specify a value, no Rate of Change Alarm is activated.

Deadband

The value that **Variable Tag** need to return to before the Alarm becomes inactive.

Format

The display format of the value (of the variable) when it is displayed on a graphics page, written to a file or passed to a function (that expects a string).

This property is optional. If you do not specify a format, the format defaults to the format specified for Variable tag.

Category

The alarm category number or label. This property is optional. If you do not specify a category, the alarm defaults to Category 0.

Help

The name of the graphics page that displays when the AlarmHelp() function is called by a user-defined command. This property is optional. If you don’t specify a help page, no action occurs when the AlarmHelp() function is called.

Comment

Any useful comment.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to acknowledge or disable the alarm.

> **Note:** If you assign an acknowledgment privilege to an alarm, do not assign a privilege to the command(s) that acknowledge the alarm. If you do assign a different privilege to the commands, then an operator needs to have both privileges to acknowledge the command.

Area

The area to which the alarm belongs. If an operator does not have access to an area, the alarm is not visible on the alarm display. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to acknowledge or disable this alarm.

Custom Filter1...Custom Filter8

A user-defined string for filtering active alarms (maximum 64 characters).

Used in a custom Cicode query function as search criteria, the custom alarm filter enables operators to identify and display a sub-set of active alarms.

> **Note:**
> - The custom filters are visible only when the Digital Alarms form is open in Extended mode.
> - The fields are not case sensitive, and can contain 'A'..'Z', 'a'..'z', '0'..'9', and the underscore '_'.
> - A custom filter cannot start with a digit.

Paging

A read/write property that indicates whether the alarm will be paged. When the value is 1 (TRUE) the alarm will be paged. The default value is 0 (FALSE). See Alarm Paging Properties. This property can be read using alarm tag browsing and read or modified when tag properties are enabled using the tag name "myCluster.myAlarm.paging".

Paging Group

A read only text string that indicates the paging group to which the alarm belongs. Maximum length is 80 characters. See your third-party paging system documentation for information on how to use this Paging Group string. This property can be read using alarm tag browsing or when tag properties are enabled read using the tagname "myCluster.myAlarm.paginggroup". For example, assign the value of PagingGroup to a variable:

```
myString = myCluster.Alarm_1.paginggroup
```

See Also
Using custom alarm filters

Formatting an Alarm Display

The display format specifies how alarms are displayed on screen for the alarms and alarm summary pages. For details on how to change the order of alarms listed on the alarm summary page, see Changing the Order of the Alarm Summary Display.

See Also
Including data
Including fixed text
Displaying lists and tables
Variable data in alarm messages
Alarm display fields
Alarm summary fields

Including CitectSCADA data

Include data in an alarm display by specifying the field name and width for each field to display. You need to enclose each field in braces {} and use the following syntax:

```
{<field name>, [width[, justification]]}
```

For example:

Format	{Tag,8} {Name,32}

In this case, data displays in two fields: Tag, with 8 characters; and Name, with 32 characters. The width specifier is optional; if you do not use it, the width of the field is determined by the number of characters between the braces.

Format	Name of Alarm:{Name }

In this case, Name is followed by four spaces; the data {Name} displays with 8 characters.

Note: The screen resolution of your computer determines the total number of characters (and therefore the number of fields) that can be displayed on the alarms page.

See Also
Including fixed text

Including fixed text

You can include fixed text by specifying the text exactly as it will display; for example:

Format	Name of Alarm:

Any spaces that you use in a text string are also included in the display.

See Also

Displaying lists and tables

Displaying lists and tables

To set the justification of the text in each field, use a justification specifier. You can use three justification characters, L (Left), R (Right), and N (None); for example:

Format	Name of Alarm:{Name,32,R} {Tag,8,L}

The justification specifier is optional; if it is omitted, the field is left justified. If you use a justification specifier, you need to also use the width specifier.

To display field text in columns, use the tab character (^t); for example:

Format	{Tag,8}^t{Name,32}^t{Desc,8}

This format aligns the tag, name and description fields of the alarm when using a proportional font to display the alarms.

See Also

Variable data in alarm messages

Variable data in alarm messages

The **Alarm Desc** field of digital, advanced and time-stamped alarms can be used to display variable data. An expression (variable tag, function etc.) can be embedded into the text of the **Alarm Desc** field. This expression is evaluated when the alarm is tripped, returning the value of any variable tags at the point in time when the alarm was generated.

Enclosing the expression in braces separates the variable data from the static text. For example:

Alarm Desc	Line Broken Alarm at Line Speed {LineSpeed1}

When *LineSpeed1* is a variable tag, this expression will produce the following output on the alarm display or alarm log:

Line Broken Alarm at Line Speed 1234

The following alarm entry uses an expression instead of a tag:

Alarm Desc	High Level at Total Capacity {Tank1+Tank2+Offset()}

When *Tank1* and *Tank2* are variable tags, and *Offset* is a Cicode function, this expression produces the following output:

High Level at Total Capacity 4985 liters

> **Note:** The result is formatted according to the formatting specified for the first variable tag in the expression. Standard variable formatting specifiers can be used to define the format for the numeric variable, over-riding the default format specified in Variable Tags.

See Also

Alarm display fields

Alarm display fields

You can use any of the fields listed below, or the Alarm Summary Fields, to format an alarm display (see Alarm Categories) and an alarm log device (see Formatting an Alarm Display):

Field Name	Description
{Tag,n}	Alarm Tag **Note:** If the **Tag** field is configured to support long names (up to 79 characters), it might cause overlap in an alarm display. Use a smaller display font if long names are expected.
{TagEx,n}	Alarm Tag with Cluster Name prefix **Note:** If the **TagEx** field is configured to support long names (up to 79 characters), it might cause overlap in an alarm display. Use a smaller display font if long names are expected.
{AlarmType,n}	Alarm type (string), not localized. Values are: Digital, Analog, Advanced, Multi-Digital, Argyle Analog, Time Stamped, Time Stamped Digital, Time Stamped Analog.

Field Name	Description
{TypeNum,n}	Alarm type number (use AlarmType to get string value instead). Values are: -1 Invalid 0 Digital 1 Analog 2 Advanced 3 Multi-Digital 4 ArgAna 5 User Event 6 timestamped 7 hardware 8 timestamped digital 9 timestamped analog
{AlmComment,n}	The text entered into the Comment field of the alarm properties dialog.
{Cluster,n}	Cluster Name
{CUSTOM1,n} {CUSTOM2,n} {CUSTOM3,n} {CUSTOM4,n} {CUSTOM5,n} {CUSTOM6,n} {CUSTOM7,n} {CUSTOM8,n}	Alarm custom fields as configured.
{Name,n}	Alarm Name **Note:** If the **Name** field is configured to support long names (up to 79 characters), it might cause overlap in an alarm display. Use a smaller display font if long names are expected.
{Native_Name,n}	Alarm Name in the expression **Note:** If the **Native_Name** field is configured to support long names (up to 79 characters), it might cause overlap in an alarm display. Use a smaller display font if long names are expected.
{Desc,n}	Alarm Description
{Native_Desc,n}	Alarm Description in the native language
{Category,n}	Alarm Category
{Help,n}	Help Page

Field Name	Description
{Area,n}	Area
{Priv,n}	Privilege
{Priority,n}	Alarm category's priority
{Type,n}	The type of alarm or condition: ACKNOWLEDGED CLEARED DISABLED UNACKNOWLEDGED
{LocalTimeDate,n}	Alarm date and time in the form: "yyyy-mm-dd hh:mm:ss[.ttt]"
{Time,n}	The time at which the alarm changed state (hh:mm:ss). (Set the [Alarm]SetTimeOnAck parameter to use this field for the time the alarm is acknowledged.)
{Date,n}	The date on which the alarm changed state (dd:mm:yyyy). Be aware that you can change the format used via the parameter [ALARM]ExtendedDate.
{DateExt,n}	The date on which the alarm changed state in extended format.
{State,n}	The current state of the alarm. This field may be used for Alarm Display Only. It is not applicable to Alarm Summary. ON OFF
{Millisec,n}	Adds milliseconds to the {Time,n} field
{High,n}	High Alarm trigger value
{HighHigh,n}	High High Alarm trigger value
{Low,n}	Low Alarm trigger value
{LowLow,n}	Low Low Alarm trigger value
{Rate,n}	Rate of change trigger value
{Deviation,n}	Deviation Alarm trigger value
{Deadband,n}	Deadband

Field Name	Description
{Format,n}	Display format of the Variable Tag
{Value,n}	The current value of the analog variable
{State,n}	The current state of the alarm. This field may be used for Alarm Display Only. It is not applicable to Alarm Summary. DEVIATION RATE LOW LOWLOW HIGH HIGHHIGH CLEARED
{ErrDesc,n}	Text string associated with a protocol (communication) error. This field is only associated with hardware errors and contains extra information associated with whatever error is detected (for example if the error is associated with a device, the device name is returned; if the error is associated with a Cicode function, the function name is returned; if the error is associated with an I/O Device, the I/O Device's alert message is returned).
{ErrPage,n}	The page, device, etc. associated with the alarm.
{LogState,n}	The last state that the alarm passed through. (This is useful when logging alarms to a device.)
{State_desc, n}	The configured description (for example healthy or stopped) of a particular state. This description is entered when configuring the Multi-Digital Alarm Properties
{Paging,n}	Indicates whether the alarm has to be paged. When the value is TRUE the alarm will be paged. The default value is FALSE. See Alarm Paging Properties.
{PagingGroup, n}	Indicates the paging group to which the alarm belongs. Maximum length is 80 characters.
{AcqDesc,n}	Textual representation of Alarm Acquisition Error.
{AcqError, n}	Numeric representation of Alarm Acquisition Error.

Where n specifies the display field size.

Notes:
- Any of the above fields can be displayed for any type of alarm. Where not applicable for a particular alarm type, zero or an empty string will be displayed.
- If an alarm value is longer than the field it is to be displayed in (n), it will be truncated or replaced with the #OVR ("overflow of format width") alert message.
- For summary pages use {SumState}. To log the state to a device, use {LogState}. State is the current state of the alarm, SumState is the state of the alarm when it occurred, and Log State is the state of the alarm at the transition.

See Also

Alarm summary fields

Alarm summary fields

You can use any fields listed below (or a combination) to format an alarm summary display and an alarm summary device.

Format the alarm summary for an entire category of alarms by specifying field names in the **Summary Format** field of the Alarm Category Properties dialog box.

You can also use the `[Alarm]DefSumFmt` parameter to format the alarm summary, particularly if your alarm summary formats are to be the same.

Field Name	Description
{UserName,n}	The name of the user (User Name) who was logged on and performed some action on the alarm (for example acknowledging the alarm or disabling the alarm, etc.). When the alarm is first activated, the user name is set to "system" (because the operator did not trip the alarm).
{FullName,n}	The full name of the user (Full Name) who was logged on and performed some action on the alarm (for example acknowledging the alarm or disabling the alarm, etc.). When the alarm is first activated, the full name is set to "system" (because the operator did not trip the alarm).
{UserDesc,n}	The text related to the user event
{OnDate,n}	The date when alarm was activated
{OnDateExt,n}	The date (in extended format) when the alarm was activated (dd/mm/yyyy)
{OffDate,n}	The date when the alarm returned to its normal state
{OffDateExt,n}	The date (in extended format) when the alarm returned to its normal state (dd/mm/yyyy)

Field Name	Description
{OnTime,n}	The time when the alarm was activated
{OffTime,n}	The time when the alarm returned to its normal state
{DeltaTime,n}	The time difference between OnDate/OnTime and OffDate/OffTime, in seconds
{OnMilli,n}	Adds milliseconds to the time the alarm was activated.
{OffMilli,n}	Adds milliseconds to the time the alarm returned to its normal state.
{AckTime,n}	The time when the alarm was acknowledged
{AckDate,n}	The date when the alarm was acknowledged
{AckDateExt,n}	The date (in extended format) when the alarm was acknowledged (dd/mm/yyyy)
{SumState,n}	Describes the state of the alarm when it occurred
{SumDesc,n}	A description of the alarm summary
{SumType,n}	Type of alarm summary (similar to alarm "Type"). Values are ACKNOWLEDGED, CLEARED, DISABLED, UNACKNOWLEDGED
{Native_SumDesc,n}	A description of the alarm summary, in the native language
{Comment,n}	A comment the operator adds to an Alarm Summary entry during runtime. The comment is specified using the AlarmComment() function.
{Native_Comment,n}	Native language comments the operator adds to an Alarm Summary entry during runtime.
Where n specifies the display field size.	

Note: You can also include in your Alarm Summary any alarm display field other than **State.**

See Also

Changing the Order of the Alarm Summary Display

Changing the Order of the Alarm Summary Display

You can customize the order in which alarms are displayed on the alarm summary page by using the `SummarySort` and `SummarySortMode` parameters. The `SummarySort` parameter allows you to display alarms according to `OnTime`, `OffTime`, and `AckTime`. `SummarySortMode` determines if the alarms will be arranged in ascending or descending order. (The order set using these parameters will override the alarm category priority order.)

See Also

Formatting an Alarm Display

Using Alarm Properties as Tags

Alarm properties can be used wherever variable tags can be used (except in alarm descriptions). For instance, you can provide the operator with a visual indication when the alarm **CV110_ERROR** is active. When it is active, **CV110_ERROR.On** will be TRUE; when it is inactive, **CV110_ERROR.On** will be FALSE. For example, **CV110_ERROR.On** could be entered as the fill color expression in a graphics object. When the conveyor becomes inoperative, the graphics object will change color.

To use an alarm property as a tag, it needs to be formatted as follows: Alarm tag (for example **CV100_STOP**) followed by a full stop (.) then the property (for example **Category**). The completed alarm property would then be **CV100_STOP.Category**.

> **Note:** If you intend to use time-stamped digital or time-stamped analog alarm properties as variable tags, you will want to verify that they are configured correctly with the necessary data being pushed to the relevant variables via the Cicode function AlarmNotifyVarChange.

See Time-stamped Digital Alarms and Time-stamped Analog Alarms for more details on how these alarms operate.

Changes in Alarm Reference Syntax

In versions prior to 7.10 you could refer to an alarm's properties by dot notation ex. “Alarm1.On”. Such references could appear in both Cicode and on graphic pages and enabled you to see and control alarm states.

In CitectSCADA v7.20 an alarm’s properties carries not only value, as in 7.10, but also items similar to Tag Extensions items for example. “Alarm1.On.T”. The alarm reference syntax is now:

[Cluster.]Alarm.AlarmProperty[.Item]where

Cluster	The optional cluster name.
Alarm	The alarm name.
AlarmProperty	The alarm property name. This part of the reference is not optional as it is for tags. There are different available properties for different categories of alarms as in 7.10.
Item	The optional Item name. If the item name is not specified, the value of the alarm property is referenced. Available items are the same as for tags: Value referenced by "V", Value Timestamp "VT", Quality "Q", Quality Timestamp "QT" and overall Timestamp "T".

See Also

Supported alarm properties
Writing to alarm properties
Setting up alarms
Tag Extensions

Supported alarm properties

The following properties can be used for every alarm type. Remember, the return value relates to the description. For example, for a digital, if 1 is returned, that means the description is TRUE, whereas 0 (zero) means it is FALSE.

Property	Description	Return Type
.On*	Alarm active	Digital
.Ack	Alarm acknowledged	Digital
.Category	Alarm category	Integer
.ComBreak	For Multi-Digital alarms, the property is set to 1 if the device cannot read data from the underlying tag at start-up for a time greater than [Alarm]ArgyleTagValueTimeout value. The property is set to 0 and re-alarms the corresponding alarm, when the alarm server receives valid data from the device. For every Disabled and Display Disabled alarms (except Time Stamped Digital and Time Stamped Analog alarms) the property is set to 1, when they are being Enabled. The property is set to 0 and re-alarms the corresponding alarm, when the alarm server receives valid data from the device.	Digital
.Custom1 .Custom2 .Custom3 .Custom4 .Custom5 .Custom6	Custom Field.	String(-64 bytes)

.Custom7 .Custom8			
.Disabled	Alarm disabled (see note below)		Digital
.Millisec	The milliseconds part of the time the alarm was triggered		Long
.Name	Alarm name		String (80 bytes)
.Paging	Alarm paged		Boolean
.PagingGroup	Paging group alarm belongs to.		String (80 bytes)
.Priority	Alarm priority		Integer
.State	Alarm's state value. An alarm state value is a combination of state enumeration and action bit masks described below: State enumerations		Short
	Alarm OFF state or state 000 for Multi-Digital alarm	0	
	Alarm ON state or state 00A for Multi-Digital alarm	1	
	state 0B0 for Multi-Digital alarm	2	
	State 0BA for Multi-Digital alarm	3	
	State C00 for Multi-Digital alarm	4	
	State C0A for Multi-Digital alarm	5	
	State CB0 for Multi-Digital alarm	6	
	State CBA for Multi-Digital alarm	7	
	Analogue deviation from set point High	8	
	Analogue deviation from set point low	9	

	Analogue rate of change alarm state	10	
	Analogue low limit alarm state	11	
	Analogue high limit alarm state	12	
	Analogue low low limit alarm state	13	
	Analogue High High limit alarm state	14	
	Alarm action masks		
	Unable to get the status of underlying tag at start-up	32	
	Alarm cleared bit mask	64	
	Alarm acknowledged but held	128	
	Alarm unacknowledged bit mask	256	
	Alarm disabled bit mask	512	
	Argyle type alarm bit mask	1024	
	Argyle type alarm ON bit mask	2048	
	Analogue alarm threshold value changed	4096	
	User generated event	8192	
	Event already logged	16384	
	Disable the alarm display	32768	
	For Example: A digital alarm called "AlmDigital1" that is ON and unacknowledged, the value of the state property will be: AlmDigital1.State = 1 (ON state) + 256 (Unacknowledged alarm) = 257		
.Tag	Alarm tag		String (80 bytes)
.Time	32-bit value of the time the alarm was triggered		Long

* The .On property for Analog alarms is true if any alarms associated with the alarm tag are active.

Note: Once an alarm is disabled, it cannot be re-enabled unless you use the function `AlarmEnable()` or `AlarmEnableRec()`

For digital alarms, time-stamped digital alarms, and advanced alarms, the following properties can also be used:

Property	Description	Return Type
.Desc	Alarm description	String (128 bytes)
.Delay	Alarm delay	Long

Note: Desc exists for every alarm type but will not return meaningful data for analog or time-stamped analog alarms. Desc returns an empty string that indicates whether the read succeeded; the write indicates that the tag was resolved and that the write request was sent.

For analog alarms and time-stamped analog alarms, the following properties can also be used:

Property	Description	Return Type
.DevDelay	Deviation delay	Long
.DeadBand	Deadband	Real
.Deviation	Deviation	Real
.HighHigh	High High	Real
.High	High	Real
.LowLow	Low Low	Real
.Low	Low	Real
.HHDelay	High High delay	Long
.HDelay	High delay	Long
.LDelay	Low delay	Long
.LLDelay	Low Low delay	Long
.Rate	Rate	Real
.Setpoint	Setpoint	Real

.Value	Alarm tag Value	Real

For the digital properties below, only one can be true at any point in time for each alarm. They are arranged in order of priority, from lowest to highest.

.DVL	Deviation alarm triggered (Low)	Digital
.DVH	Deviation alarm triggered (High)	Digital
.R	Rate of Change alarm triggered	Digital
.L	Low alarm triggered	Digital
.H	High alarm triggered	Digital
.LL	Low Low alarm triggered	Digital
.HH	High High alarm triggered	Digital

Note: DVL and DVH are only evaluated if Deviation > 0. R is only evaluated if Rate > 0.

Some alarm properties return configuration data. If the user has not defined this information, the following defaults are provided:

Property	**Default**
.Setpoint	0
.HighHigh	3.4e+38
.High	3.4e+38
.LowLow	-3.4e+38
.Low	-3.4e+38
.Rate	0
.Deviation	0

.Deadband	0
.Category	0
.Priority	0

See Also

Writing to alarm properties
Setting up alarms

Writing to alarm properties

If you have the necessary access rights, you can write to the following alarm properties. (Remember, the value you write to the property relates to the description. For example, if you set a digital alarm property to 1, you are making the description TRUE. If you set it to 0 (zero), you are making it FALSE.)

Property	Description	Input Type
.Ack	Alarm acknowledged (once acknowledged cannot be "unacknowledged")	Digital
.Deadband	Alarm Deadband	Real
.Deviation	Deviation from setpoint	Real
.Disabled	Alarm disabled	Digital
.HighHigh	High High	Real
.High	High	Real
.LowLow	Low Low	Real
.Low	Low	Real
.HHDelay	High High delay	Long
.HDelay	High delay	Long
.LDelay	Low delay	Long

.LLDelay	Low Low delay	Long
.DevDelay	Deviation delay	Long
.Custom1 .Custom2 .Custom3 .Custom4 .Custom5 .Custom6 .Custom7 .Custom8	Custom field	String
.Paging	Alarm paged	Boolean
.PagingGroup	Paging group alarm belongs to.	String

Analog alarm thresholds can also be changed using the `AlarmSetThreshold()` function.

Note the following:

- The alarm tag needs to be unique.
- The alarms databases in included projects on the alarms server and the Control Client computer needs to be identical.

See Also

Supported alarm properties
Setting up alarms

Setting up alarm properties

To use alarm properties, you need to enable them on the Alarm Server.

1. In the Project Editor, choose **Servers** | **Alarm Servers**.
2. Press F2 to display the extended form.
3. Specify the port that the Alarm Server will listen on.

See Also

Alarm Server Definitions
Supported alarm properties
Writing to alarm properties

Handling Alarms at Runtime

When an alarm is triggered, it becomes active. The active state of a digital alarm is ON, while the active state of an analog alarm varies, depending on the type of alarm (for instance HIGH, LOW , RATE, and so on). When an operator acknowledges the alarm, its state changes to ACKNOWLEDGED. When the alarm is reset (when the conditions that caused the alarm have been rectified), its state changes to OFF.

Alarms are displayed on the standard alarm display page. To acknowledge an alarm, an operator either selects the alarm with the mouse and clicks the left mouse button, or moves the cursor onto the alarm and presses the **Enter** key. Alternatively, the operator can acknowledge every alarm by clicking **Alarm Ack**. When an alarm is acknowledged, its display color (on screen) changes. Acknowledged alarms remain on screen until their state changes to OFF.

If an alarm does not appear to be operating as designed, or is deemed unnecessary, an operator can disable it. Disabled alarms are ignored until they are re-enabled. (You need to define a command that uses the `AlarmDisable()` function to disable alarms.)

To maintain a history of alarm activity, an event log is kept of alarms. This log stores the time when each alarm was activated, acknowledged, and reset. You can display alarms from the event log (including disabled alarms) on the alarm summary page.

Create a page called **Summary** based on the AlarmSummary template. call the page "Summary" so that the alarm summary button (on pages such as the menu page) operates correctly (the `PageSummary()` function can only open a page called "Summary").

Operators can add comments to any alarm in the summary log. (You need to define a command that uses the `AlarmComment()` function to add comments to an alarm.)

Note: If you have many alarms on the alarm page or alarm summary page, use **Page Up** and **Page Down** to scroll through the list.

To create an alarm page:

1. In **Citect Explorer**, double-click **Create New Page** in the **Graphics|Pages** folder.

- or -

2. In the **Graphics Builder**, choose **File | New** and then click **Page**.
3. Select the alarm template you want to use. Use the **Alarm** template to create a page to display configurable alarms, the **Summary** template for summary alarms, the **Disabled** template for disabled alarms, and the **Hardware** template for hardware alarms.
4. Choose **File|Save**.

5. Specify a name in the page title field. Make the new page name match the template name. For example, call the new hardware alarm page **Hardware**.
6. Click **OK**.

> **Note:** You can also create your own (non-standard) alarm pages. The easiest way to do this is by copying and modifying the standard alarm templates.

To display an alarm page at runtime:

1. Create an alarm (or hardware alarm) page in your project if you have not already done so. Call the page **Alarm** for a configurable alarm page, and **Hardware** for a hardware alarm page.
2. Create a new keyboard command or a button, to call the page at runtime. You can also add a touch command to an existing screen object.
3. In the command field, enter **PageAlarm()** to display the configurable alarms page, or **PageHardware()** to display the hardware alarms page.
4. Configure other properties as necessary.
5. Click **Add** to append a new record, or **Replace** to modify an existing record.

> **Note:** If using the standard page templates, you don't usually need to create a command to display the page: the commands are already built in.

To display a customized alarm page (with a non-standard name), use the `PageDisplay()` function to display the page, followed by the `AlarmSetInfo()` function as necessary.

Using System Fonts

Alarm categories and Cicode functions allow you to use predefined fonts, known as system fonts, to display text. You can also configure your own fonts; for details see Configuring Custom Color Fonts.

> **Note:** If an animation is associated with an I/O Device that does not initialize properly at startup or goes offline while the system is running, the associated animation is grayed on the relevant graphics pages (because the values are invalid). You can disable this feature with the `[Page]ComBreak` parameter. See Handling Communication Errors in Reports.

To define a system font:

1. In the Project Editor or the Graphics Builder, choose **System | Fonts**. The Fonts dialog box appears.

2. Enter your font properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Fonts properties

Fonts properties

Use the Fonts dialog box to define your font properties. Fonts have the following properties:

Font Name

The name of the font. Unlike background text, strings, and numbers (which can use any standard Windows font), you need to define a font for animated text.

Font Type

Any text font supported by Windows (31 characters maximum). Choose a font type from the menu.

You can also specify bold, italic, and underlined text. To specify any of these options, append the appropriate specifier to the Font Type, for example:

Font Type	Courier,B

Specifies bold characters.

Font Type	Helv,I

Specifies italic characters.

Font Type	TmsRmn,U

Specifies underlined text.

You can also specify multiple options, for example:

Font Type	Courier,B,I,U

Specifies bold, italic, and underlined characters.

Note: To use a font that is not displayed in the menu, you need to install the font on your computer before you can use it in your system. To view, install, or remove

Windows fonts, use the Windows Control Panel (Fonts Option). Refer to your Windows documentation for details. (If your system uses a network, every computer needs to have the font installed.)

Pixel Size

The size of the displayed text. You can specify text fonts in pixels or points.

To specify a point size, enter a negative number in the **Pixel Size** field; for example:

Font Size	-10

specifies a ten-point font size. Be aware that you can only specify a point size as a whole number (integer).

If you have not installed the Font Type (or Pixel Size) on your system, a default font and/or size is used that most closely resembles the font and/or size you have specified.

If you use a point size, the size remains constant across screen resolutions. On low-resolution screens, the font appears larger than on high-resolution screens, which might cause misalignment of animation. Only use a point size to display text on computer screens of the same resolution.

Foreground Color

The foreground color of the displayed text (i.e., the color of the text characters). You can use a predefined color label (accessible via the menu), a user-defined custom label, or the RGB encoded number generated by the function `MakeCitectColour` (see the *Cicode Reference Guide*).

Note: Do not confuse predefined labels with the color Name feature associated with the Color Picker. You cannot use color names with this dialog; doing so generates a compile error.

Foreground Flash

The secondary color applied to the font if using flashing color for your text characters. You can use a predefined color label (accessible via the menu), a user-defined custom label, or the RGB encoded number generated by the function `MakeCitectColour` (see the *Cicode Reference Guide*).

If you do not specify a color, the text remains solid.

Background Color

The background color of the displayed text. You can use a predefined color label (accessible via the menu), a user-defined custom label, or the RGB encoded number generated by the function `MakeCitectColour` (see the *Cicode Reference Guide*).

This property is optional. If you do not specify a background color, it defaults to transparent.

Background Flash

The secondary color applied to the background if using flashing color. You can use a predefined color label (accessible via the menu), a user-defined custom label, or the RGB encoded number generated by the function `MakeCitectColour` (see the *Cicode Reference Guide*).

If you do not specify a color, the text remains solid.

Comment

Any useful comment (48 characters maximum).

Configuring Custom Color Fonts

From CitectSCADA version 6.0 onwards, colors are referenced by using hex codes for red, green and blue (RGB) values. You can view RGB values (in decimal) of a selected color by choosing **Tools | Edit Favorite Colors** in Graphics Builder.

If, for example, the color you want to use has the values R = 128, G = 170, B = 213, you can convert these to their hex equivalents (R = 0x80, G = 0xAA, B = 0xD5).

If you define a label for your color, you can use decimal or hex values. For example:

- **Label Name**: Plate_Blue
- **Expression**: 0x80AAD5 (or 8432341)

Once you have defined your label, you can create fonts to use in your project for alarm categories, Cicode functions, and button objects. For example, use the Fonts dialog box to define a font with the following properties:

- **Font Name**: Plate_Font
- **Font Type**: Arial
- **Pixel Size**: 12
- **Foreground Color**: Plate_Blue

For more information, refer to Knowledge base article Q4024.

See Also

Using System Fonts

Chapter: 24 Configuring Events

You can use an event to trigger an action, such as a command or set of commands. For example, an operator can be notified when a process is complete, or a series of instructions can be executed when a process reaches a certain stage.

Events needs to be enabled for events to run. Use the *Computer Setup Wizard* (Custom setup) to enable Events. If using a network, you can process events on any computer (or every computer).

> **Note:** The Events system is not redundant. That is, it is not possible to assign primary and standby Events Servers. If you require a redundant event reporting system, use reports instead. See Reporting Information.

To define an event:

1. Choose **System** | **Events**. The Events dialog box appears.
2. Enter the event properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record

> **Note:** The Events Server needs to be enabled for events to work.

See Also

Events Properties
Running Events

Events Properties

Events have the following properties:

Name

For a single computer system, specify **GLOBAL** for the event name:

Name	GLOBAL

If you are using a network and want to run an event on every computer, specify **GLOBAL** for the event name. If you want to run an event only on specific computers, specify an event name and use the Computer Setup Wizard (Custom setup) to specify which computer(s) will run the event. The event name does not have to be unique: you can specify many events with the same name.

Enter a value of 16 characters max.

> **Note:** Make event names agree with the Tag name syntax. The use of any other characters such as spaces will result in a compiler error. Instead of a space, use an underscore character (_).

Cluster Name

The default cluster to be used when the event action and event trigger Cicode is run. Any tags in these Cicode expressions will resolve to this default cluster if they don't have their cluster specified inside the expression. This field can be left blank in single cluster systems or if every tag inside the expressions are declared in the <*ClusterName*>.<*TagName*> format.

Time

The time of day to synchronize the **Period** in hh:mm:ss (hours:minutes:seconds). If you do not specify a time, **Period** is synchronized at 00:00:00 (that is, midnight). Enter a value of 32 characters maximum.

Period

The period to check for the event, in hh:mm:ss (hours:minutes:seconds). Enter a value of 32 characters maximum. Alternatively you can specify:

- A weekly period by entering the day of the week to check for the event, for example, Monday, Tuesday, Wednesday, etc.
- A monthly period by entering the day of the month to check for the event, for example, 1st, 2nd, 3rd, 4th, 5th, etc.
- A yearly period by entering the day and the month to check for the event, for example, 1st January, 25th February, and so on. The day and month needs to be separated by a space.

If you do not specify a period or time, the period defaults to one second. If you do not specify a period, but do specify the time, the period defaults to one day.

Trigger

The Cicode expression (or Variable tag) which is used to determine whether the event Action is executed. This expression is checked every one second. Enter a value of 254 characters maximum.

Action

The commands to execute. Enter a value of 64 characters maximum.These commands will execute in the following circumstances:

- When the specified **Time** and **Period** occurs, and the **Trigger** condition is TRUE or blank.
- When the **Trigger** becomes TRUE, and the **Time** and **Period** field are blank. The Trigger needs to become FALSE and TRUE again for the action to re-execute.

Comment

Any useful comment. Enter a value of 48 characters maximum.

Running Events

The Event Server needs to be enabled for events to work. You can run an event automatically:

- At a specified time and period.
- When a trigger condition becomes TRUE.
- When a trigger condition is TRUE at a specified time and period.

See Also

Specifying times and periods
Using triggers

Specifying times and periods

The Period determines when the event is run. You can specify the period in hh:mm:ss (hours:minutes:seconds), for example:

Period	Comment
1:00:00	Run the event every hour
6:00:00	Run the event every six hours
72:00:00	Run the event every three days
Monday	Run the event each Monday
15th	Run the event on the 15th of each month
25th June	Run the event on the 25th of June

You can also specify the time of day to synchronize the event, for example:

Period	Comment
6:00:00	Synchronize the event at 6:00 am
12:00:00	Synchronize the event at 12:00 midday

The Time synchronizes the time of day to run the event and, with the Period, determines when the event is run; for example:

Time	6:00:00
Period	1:00:00

In this example, the event is run every hour, on the hour. If you start your runtime system at 7:25am, your event is run at 8:00am, and then every hour after that.

See Also

Using triggers

Using triggers

You can use any Cicode expression (or variable tag) as a trigger for an event. If the result of the expression (in the **Trigger** field) becomes TRUE, and if the **Time** and **Period** fields are blank, the event is run. For example:

Time	
Period	
Trigger	RCC1_SPEED<10 AND RCC1_MC

This event is only run when the expression (Trigger) becomes TRUE, that is, when the digital tag RCC1_MC is ON and the analog tag RCC1_SPEED is less than 10. The expression needs to become FALSE and then TRUE again before the event is run again.

If you use the Time and/or Period fields, the Trigger is checked at the Time and/or Period specified, for example:

Time	6:00:00
Period	1:00:00
Trigger	RCC1_SPEED<10 AND RCC1_MC

This event is run each hour, but only if the expression (Trigger) is TRUE (that is, if the digital tag RCC1_MC is ON and the analog tag RCC1_SPEED is less than 10).

See Also

Running Events

Chapter: 25 Using Accumulators

Accumulators track incremental runtime data, such as motor run hours, power consumption, and downtime. You set a trigger (for example, motor on) to increment three counters:

- The number of times the accumulator is triggered (for example the number of starts for the motor).
- The run time, in steps of 1 second.
- A totalized value, by an increment you define (for example the current).

The accumulated data is stored as variable tags in an I/O Device. Variable tags are read at startup and updated regularly while the trigger is active. You can monitor and display accumulated data by animating, trending, or logging the variable tags.

> **Note:** You can control (re-read or reset) any accumulator at runtime by using the `AccControl()` Cicode function.

To configure an accumulator:

1. Choose **System** | **Accumulators**. The Accumulators dialog box appears.
2. Enter the accumulator properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

You use the Accumulator Properties dialog box to configure your accumulators.

See Also

Accumulator Properties

Accumulator Properties

Accumulators have the following properties:

Name

The name of the accumulator. Enter a value of 79 characters or less.

Cluster Name

The name of the cluster that this accumulator runs on. If the Cluster Name is not set, then this accumulator will run on each defined cluster.

Trigger

The Cicode expression (or variable tag) to trigger the accumulator. If the result of the expression (in this field) is TRUE, accumulation starts. If the result of the expression (in this field) becomes FALSE, accumulation stops. Enter a value of 254 characters or less.

The frequency with which the trigger is checked is controlled by the [Accumulator]WatchTime parameter.

Run Time

The variable (tag) that contains the run time (in seconds). Enter a value of 254 characters or less. On startup, this value is read. Then the local copy of this variable is incremented (while **Trigger** is TRUE) and the variable is written back to the I/O Device at a frequency determined by the [Accumulator]UpdateTime parameter.

No. of Starts

The variable (tag) that contains the number of starts (that is, the number of times the trigger changes from FALSE to TRUE). Enter a value of 254 characters or less. On startup, this value is read. Then the local copy of this variable is incremented and the variable is written back to the I/O Device at a frequency determined by the [Accumulator]UpdateTime parameter.

Totalizer Inc

Any Cicode expression (or variable tag) to add to (increment) the **Totalizer** variable while the **Trigger** condition is TRUE. Enter a value of 254 characters or less.

Totalizer

The variable (tag) that contains the totalized value. Enter a value of 254 characters or less.

On startup, this value is read. Each time the trigger is checked and while the trigger is TRUE, the value in the **Totalizer Inc** field is added to the local copy of this **Totalizer** variable. The new **Totalizer** variable is written back to the I/O Device at a frequency determined by the [Accumulator]UpdateTime parameter.

For example, if you configure an accumulator for a motor, and **Totalizer Inc** is the current (amperage) used by the motor, then **Run Time** will contain the time (in seconds) that the motor has run, **No. of Starts** will contain the number of times the motor was started, and **Totalizer** will contain the total current used by the motor. The average current used by the motor is **Totalizer/Run Time.**

Comment

Any useful comment. Enter a value of 48 characters or less.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege needed by an operator to perform operations on the accumulator (by using accumulator functions). Enter a value of 16 characters or less.

Area

The area to which the accumulator belongs. Only users with access to this area (and any required privileges) will be able to perform operations on the accumulator. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any required privileges) to perform operations on the accumulator. Enter a value of 16 characters or less.

Notes:

- The run time, number of starts, and totalized value are stored in variables in your I/O Device. This allows easy access to these variables and, because they are saved in the I/O Device, their values are saved if the project is shutdown. You can increase system performance by storing these variables in a Disk I/O Device. (If you store these variables in your external I/O Devices, communications bandwidth will be consumed updating the variables.
- If using a network, you can use a redundant Disk I/O Device to secure these variables.
- The accumulator server runs as part of the Reports Server. If you have a redundant Reports Server, you need to use a primary/standby configuration to stop the accumulators running on both Reports Servers. Use the Computer Setup Wizard to define the Reports Servers.
- You can control (re-read or reset) any accumulator at runtime by using the `AccControl()` Cicode function.

Chapter: 26 Logging and Trending Data

Since CitectSCADA version 6.0, the Process Analyst (an built-in trend visualization tool) has superseded the functionality of trend graphs. However, trend graphs are still supported. Be aware that to use SPC trends, use trend graphs, since SPC is not supported by the Process Analyst.

For details, see the Process Analyst Help.

See Also

Trending Data
Trend Graphs
Printing Trend Data
Exporting Trend Data
Using Trend History Files
Using Path Substitution
Debugging Trending

Trending Data

The trend system can help you better understand your plant and equipment's performance. Trending can be used for dynamic visual analysis (trend and SPC graphs), production records, or for regularly recording the status of equipment for efficiency and preventive maintenance.

Using trend tags, you can specify the data you want to collect from your I/O Device variables. This information can be logged at regular intervals (periodic trend), or only when an event occurs (event trend). Event trends are used for trending data that is not time-based, for example, for a product as it comes off an assembly line. Trend data is usually saved on disk for analysis or displayed on a trend graph.

The trend system is based on real-time samples. The trend system expects a return of one data point each time it samples the data. Although gaps in the data can be filled, you will want to verify that your field device can return data values at the rate you specify (especially if you are using sample periods of less than 100 ms).

Any amount of data can be collected and stored. The only restriction on the amount of data that you can store is the size of the hard disk on your computer (an efficient data storage method is used to optimize the use of storage space on your computer's hard disk). For long term storage, you can archive the data to disk or tape (without disrupting your runtime system).

> **Note:** If you are trending data across a network (distributed processing), it is recommended that you enable time synchronization using the Time Synchronization configuration application. This verifies that every trend is synchronized to the same time.

You might also consider staggering your trend sample requests using the [Trend]StaggerRequestSubgroups parameter.

See Also

Configuring trend tags

Configuring trend tags

You use the Trend Tags dialog box to configure your trend tags.

To configure a trend tag:

1. Choose **Tags** | **Trend Tags**. The Trend Tags dialog box appears. (Press **F2** to view the extended Trend Tag form.)
2. Enter your trend tag properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Trend Tag Properties

Trend Tag Properties

Trend Tag Name

The name assigned to the trend data (79 characters maximum). If the trend tag is logging a particular variable, use a 16-character name that resembles the 32-character name of the related variable tag. This will mean an association between the two is easily recognizable. The name needs to be unique to the cluster. Trend Tag names need to adhere to the Tag name syntax. If the name is not unique or is not syntactically correct it may not be recognized. If you have many tags, use a naming convention (see Using structured tag names). This makes it easier to find and debug your tags.

> **Note:** Where Cluster Name is left blank, the name needs to be unique to every defined cluster.

Note: Trend tag names have to be unique and not identical to any SPC tag names within the cluster(s) that run this trend. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

If using the File Name property, ensure that the trend tag uses a unique name. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Cluster Name

The name of the cluster that runs this trend. If the Cluster Name is not set, then this trend will run on every defined cluster.

Expression

The logged value of the trend tag (254 characters maximum). You can log individual variables by using a variable tag. For example:

Expression	LT131
Comment	Logs the Variable Tag LT131

The value of the process variable LT131 is logged.

You can also log any Cicode expression or function, for example:

Expression	LT131/COUNTER
Comment	Logs Variable Tag LT131 divided by the Variable Tag COUNTER

Note: When a variable tag is used in the expression field of a trend tag property, the **Eng Zero Scale** and **Eng Full Scale** fields of that variable tag needs to be set appropriately, or data will be lost because the trend logs negative values as invalid.

Trigger

The Cicode expression (or variable tag) that triggers data logging (254 characters maximum). For example:

Trigger	LT131<50

In this example, logging occurs when the value of the variable tag (LT131) falls below 50.

For a periodic trend, data is logged only while the value of the trigger is TRUE. (The trend graph will still scroll, but will display <GATED> where the trigger is FALSE.) In the above example, data is logged continuously while the value of LT131 remains less than 50. Logging ceases when the value rises to (or above) 50. Logging does not occur again until the value of LT131 falls below 50.

You do not have to specify a trigger for a periodic trend. If you do not specify a trigger for a periodic trend, logging occurs continuously.

For an event trend, data is logged once when the value of the trigger changes from FALSE to TRUE. In the above example, one sample is logged when the value of LT131 first becomes less than 50. Another sample is not logged until the value of LT131 rises to (or above 50) and again falls below 50.

Sample Period

The sampling period of the data. You can either enter a period of your own, or choose one from the menu.

Enter sampling periods of greater than one second in hh:mm:ss (hours:minutes:seconds) format. If you enter a single digit, without the colon (:), it will be considered a second. For example, if you enter **2**, it will be interpreted as 2 seconds.

Sampling periods of less than one second needs to be entered as decimals. For example, to enter a period of 200 milliseconds, you would enter **0.2**.

If the sample period is less than one second, then one second needs to be divisible by the period (to give an integer). For example, a sample period of 0.05 is valid, because 1/0.05 = 20, whereas a sample period of 0.3 is not valid because 1/0.3 = 3.333... .

Note:
- Your I/O Device needs to be capable of providing data at the specified rate, otherwise gaps will appear in the trend data and/or the hardware alarm **Trend has missed samples** will be evoked. You can fill gaps in the file and graph using the [Trend]GapFillTime parameter. Gaps in the graph only can be filled using the TrnSet-DisplayMode() function.
- If trends with a sample period of less than a second are shared by several clients across a network (distributed processing), enable time synchronization using the Time Synchronization configuration application This verifies that trends are synchronized with each other.

The **Trigger** is checked each sample period. If the Trigger is TRUE (or has just changed from FALSE to TRUE, in the case of event trends), the value of the **Expression** is logged.

Examples

Sample Period	Comment
30	Logs data every 30 seconds
10:00	Logs data every 10 minutes
10:00:00	Logs data every 10 hours
2:30:00	Logs data every 2 and a half hours

The sampling period of the fastest trend on the page is taken as the default value for the display period of the page.

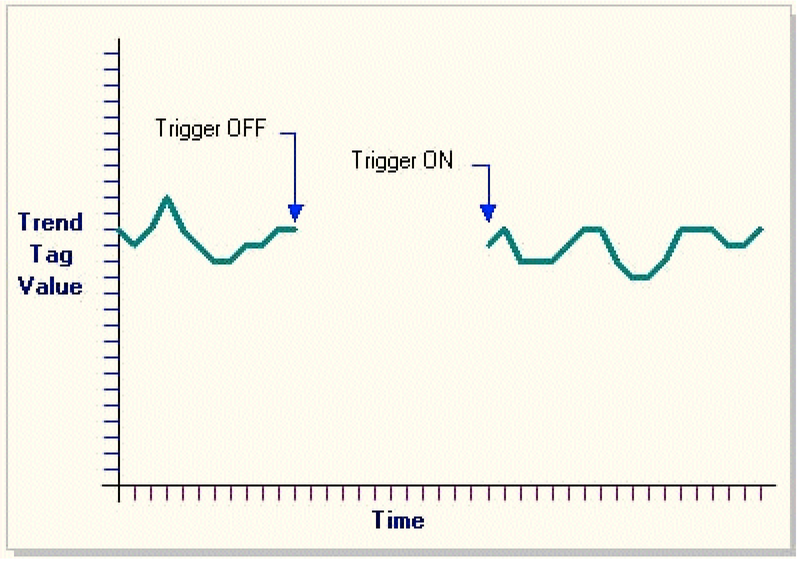
This property is optional. If you do not specify a sample period, the sampling period defaults to 10 seconds.

> **Note:** If you edit this property in an existing project, delete the associated trend files before running the new runtime system (For location of the trend files, see File Name).

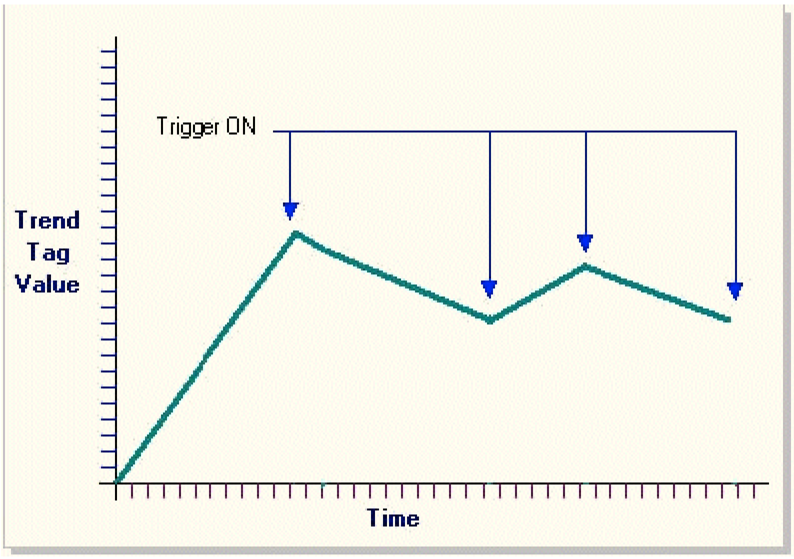
Type

The type of trend (32 characters maximum):

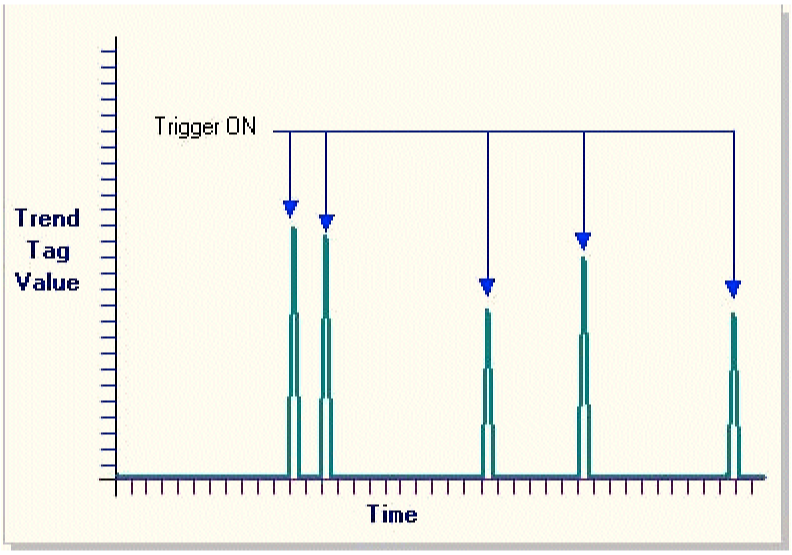
- **TRN_PERIODIC** - A trend that is sampled continuously at a specified period. You can also define a trigger to stop and start the trend (when a specified condition occurs in the plant).



- **TRN_EVENT** - A trend that is sampled once each time the value of the trigger changes from FALSE to TRUE. Interpolation occurs between each data point to create a continuous graph.



- **TRN_PERIODIC_EVENT** - A trend that is sampled once each time the value of the trigger changes from FALSE to TRUE. No interpolation occurs between these points, thus providing a graph which spikes at each data point.



Comment

Any useful comment (48 characters maximum).

File Name

The file where the data is to be stored (253 characters maximum). Specify the full path or use path substitution.

When data is collected from your plant floor, it is stored in a file on the hard disk of your computer which is then used to display a trend or SPC graph (a separate file is used for each trend tag).

By default, CitectSCADA stores the file in the [DATA] directory on the hard disk where you installed CitectSCADA. The default name of the file is the trend tag name. However, you can specify an alternate file name like this:

File Name	`[DATA]:TANK131`

where [DATA] specifies the disk and path for the data. Use path substitution to make your project more 'portable'.

Notes:

- You can no longer store trend files in the `bin`, `runtime`, `backup` or `user` directories or any subdirectories of these. If you have existing Version 3.xx or 4.xx projects that use these directories to store trend files, the path for these will have to be changed to the Data directory.
- The trend system will buffer the acquired data before saving it to a file. The [Trend]CacheSize parameters determine the buffer sizes for returned data.
- The File Name property is optional. If you do not specify a file name, the file name defaults to [DATA]:<Name> on the hard disk where you installed CitectSCADA. Where <Name> is the trend Tag Name.
  If you use the File Name property, ensure that no other SPC tags or trend tags use the same filename. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

If using the File Name property, ensure that the trend tag uses a unique file name. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

- Do not use a file extension when specifying a file name. If you edit this property (change the file name or path) in an existing project, existing trend data is ignored.

Storage Method

Select **Scaled** or **Floating Point** (64 characters). Scaled is a 2-byte data storage method; floating point uses 8 bytes.

Floating point storage has a dramatically expanded data range in comparison to scaled storage, allowing values to have far greater resolution. However, you need to consider that it also uses a lot more disk space. Use scaled where compatibility with pre-V5.31 trend history files is necessary.

If you do not specify a storage method, it is set to **Scaled** by default.

Note: If you edit this property in an existing project, you need to delete the associated trend files - before you run the new runtime system. (For location of the trend files, see the File Name.)

No. Files

The number of history files stored on your hard disk (for this tag) (4 characters maximum). The maximum number of files you can specify per trend tag is 999. Performance and storage will be severely impacted by having a large number of history files per trend.

If you do not specify the number of files, 2 history files are stored on your hard disk.

Note: If you edit this property in an existing project, delete the associated trend files - before you run the new runtime system. (For location of the trend files, see the File Name.)

Time

The time of day to synchronize the beginning of the history file, in hh:mm:ss (32 characters maximum). If you do not specify a time, the file is synchronized at 0:00:00 (i.e. midnight). The time needs to be specified in Greenwich Mean Time, not the local time zone.

Note: If you edit this property in an existing project, delete the associated trend files - before you run the new runtime system. (For location of the trend files, see the File Name.)

Period

The period of the history file, in hh:mm:ss (32 characters maximum). Alternatively, you can:

- Specify a weekly period by entering the day of the week on which to start the history file, for example Monday, Tuesday, Wednesday, etc.

- Specify a monthly period by entering the day of the month on which to start the history file, for example 1st, 2nd, 3rd, 4th, 5th, etc.
- Specify a yearly period by entering the day and the month on which to start the history file, for example 1st January, 25th February, etc. The day and month needs to be separated by a space.
  If you do not specify a period, the period defaults to Sunday (weekly).
  When deciding on a period setting, be aware that the performance of a trend viewer (be it the existing CitectSCADAclient or Process Analyst) may be impacted by the size of a trend file. This is particularly true when displaying event-based trend data.

> **Note:** If you edit this property in an existing project, delete the associated trend files before you run the new runtime system. (For location of the trend files, see File Name.)

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to display the trend data on a trend.

Area

The area to which the trend data belongs.

Eng Units

The engineering units of the variable/expression being logged (8 characters maximum). The engineering units are used by the trend scales and trend cursor displays.

Format

The format of the variable/expression being logged (11 characters maximum). The format is used by the trend scales and trend cursor displays.

This property is optional. If you do not specify a format, the format defaults to ####.#.

Deadband

The value that **Trend Tag** need to return to before the Trend becomes inactive.

Trend Graphs

Trend graphs visually represent past and current activity of plant-floor data, building a picture over time of how a variable (such as product output, level, temperature, and so on) changes or how a device or process is performing. You can monitor current activity as it happens and scroll back through time to view trend history.

As the values of variables change over time or as events happen, the graph moves across the page. The latest values are displayed by default. You can scroll back through historical data to display past values of the variable (or process).

You can trend any single variable or Cicode expression. You can display any number of trends on the screen simultaneously, even if they have different sample periods. You can also display up to eight trend tags (pens) in any trend window.

A trend graph can only communicate with one cluster, therefore you cannot mix trends from multiple clusters on a single trend graph. To graph trends from multiple clusters you will need to use multiple trend graphs, or, use the Process Analyst which has no such restrictions.

Historical data collection continues even when the display is not active. You can switch between pages without affecting trend graphs. Trend data acquisition and storage of data (in trend history files) continues even when the display is not active.

You can use the following standard trends:

- A single full page trend, where one trend window displays on a graphics page.
- A double full page trend, where two trend windows display on a graphics page.
- A zoom trend with two trend windows and added functionality for zooming.
- A pop-up trend that you can 'pop up' anywhere (in a separate window) on your computer screen.
- User-defined trends that you can position anywhere on any graphics page.

Note: Variable tags can also be visually trended using an SPC Control Chart. Statistical Process Control (SPC) is a facility that enables you to control the quality of materials, manufactured products, services, etc. This quality control is achieved by collecting, arranging, analyzing, and testing sampled data in a manner that detects lack of uniformity or quality.

See Also

Creating trend pages

Creating trend pages

You can use any of the predefined trend templates for your trend pages, or use a predefined template to produce your own trend templates. You can draw a trend background (such as gridlines) on your trends.

To configure a trend page:

1. Click **New Page**, or choose **File** | **New.**
2. Select **Type: Page**.
3. Choose the **Resolution** (size) of the trend page.
4. Choose a trend **Template** for the trend page:
   - **Singletrend** - One trend on the page
   - **Doubletrend** - Two trends on the page
   - **Eventtrend** - One event trend on the page
   - **Zoomtrend** - Two trends on the page (one window for zooming)
   - **Poptrend** - A single trend on the page (for display in a pop-up window)
5. Click **OK**.

To create multiple trend pages, you can:

- Create a trend page for each set of trends to display in the runtime system.
- Create a single trend page and use the PageTrend() function to display trends as necessary. With this function, you can display the trends in the system with a single trend page
- Create the trend page with the Graphics Builder, and set the pen names to blank. You then display that page by calling this function and passing the necessary trend tags (up to 8). Call the PageTrend() function from a menu of trend pages.

See Also

Trend interpolation

Trend interpolation

Trend interpolation is used to define the appearance of a trend graph when the incoming samples fall out of synchronization with the display period or when samples are missed.

For example, a particular trend might be sampled five times between each update of the trend graph. As only one value can be displayed for each update, a single value needs to be used that appropriately represents the five samples; and that could be the highest value, the lowest value, or an average.

To define how CitectSCADA calculates the value to use, you set a particular trend interpolator display method.

The following table shows the available interpolator display methods, grouped into *condense methods* (where the display period is longer than the sample period) and *stretch methods* (where the display period is less than or equal to the sample period).

Condense methods	Stretch methods
Average *(default)* - this displays the average of the samples within the previous display period	**Step** *(default)* - This method simply displays the value of the most recent sample.
Minimum- This displays the lowest value that occurred during the previous display period.	**Ratio** - This method uses the ratio of sample times and values immediately before and after the requested time to interpolate a "straight line" value.
Maximum - This displays the highest value that occurred during the previous display period.	**Raw Data** - This method displays the actual raw values.

The interpolation display method is set via TrnSetDisplayMode() function. You can also use the [Trend]GapFillMode parameter, but it will interpolate values within the actual trend file as well as on the trend graph.

Printing Trend Data

You can print trend data using the following functions:

Function	Purpose
TrnPrint	Prints a trend that is displayed on the screen.
TrnPlot	Prints a plot of one or more trend tags.
TrnComparePlot	Prints two trends (one overlaid on the other), each of up to four trend tags.
WinPrint	Prints the active window

The standard trend templates have buttons that call these functions to print data.

When you print using the TrnPrint function, the Plot Setup dialog box appears. Use this dialog box to:

- Specify the title of the trend.
- Add a comment which is displayed beneath the title.
- Specify whether the trend is going to print in black and white, or in color. The selection that you make here will become the setting for the [General]PrinterColorMode parameter.
- Define your printer setup. The printer that you select here will be set as the default printer at the [General]TrnPrinter parameter.
- Specify whether or not the form displays the next time the function is used. This check box sets the [General]DisablePlotSetupForm parameter.

See Also

Exporting Trend Data

Exporting Trend Data

You can export trend data to reports and databases with the following functions:

Function	Purpose
TrnGetTable	Retrieves trend information and stores it in a Cicode array
TrnExportClip	Copies trend data to the clipboard
TrnExportCSV	Copies trend data to a CSV file
TrnExportDBF	Copies trend data to a DBF file

The standard trend templates have buttons that call these functions to export data.

> **Note:** You can also select part of your trend graph (click and drag) and copy the underlying values to the Windows clipboard. You can then paste them into an Excel spreadsheet. (If you are pasting millisecond values, you will need to create a custom format for the TIME column to display these values correctly. To do this, select the column and select **Format | Cells**. In the **Number** tab, select **Custom** for Category, and type **h:mm:ss.000 AM/PM**.)

See Also

Using Trend History Files

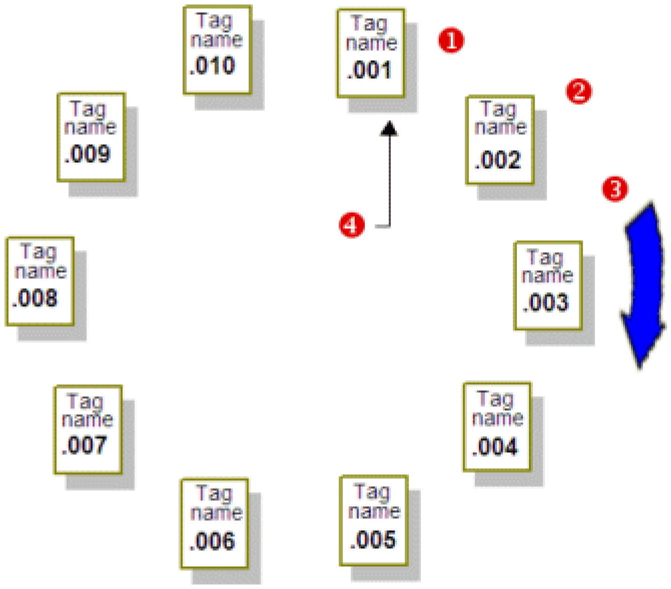
Using Trend History Files

When CitectSCADA starts up for the first time, it creates the trend files necessary by each trend tag in the runtime system. (You can change this default using the [Trend}AllFiles parameter.)

A system of rotational history files is used to store the trend data. Data is stored in several files rather than in a single large file.

By default, there are 2 files (for each trend tag). You can change the default by specifying the number of files to use, for example:

No. Files	Comment
10	Use ten files for the data, as in the diagram below.



1. When Citect begins logging, data is written to the first file.
2. At midnight the following Sunday, Citect writes to the second file.
3. At midnight the following Sunday, Citect writes to the third file, and so on.
4. After week 10, the first file is overwritten with new data.

The maximum number of files you can specify per trend tag is 270.

You can also specify the period between files, i.e., when a new history file is used, for example:

Period	Comment
1:00:00	Use a new file each hour
6:00:00	Use a new file every six hours
72:00:00	Use a new file every three days
Monday	Use a new file each week beginning on Monday
15th	Use a new file every month beginning on the 15th of each month
25th June	Use a new file every year beginning on the 25th of June

You can also specify the time of day to synchronize the start of the history file; for example:

Time	Comment
6:00:00	Synchronize the file at 6:00 am
12:00:00	Synchronize the file at 12:00 midday
18:30:00	Synchronize the file at 6:30 pm

See Also

Storage method

Storage method

You can select the storage method to use for trend tags and SPC tags. You are given a choice of either **Scaled** or **Floating Point.** Scaled represents a 2-byte data storage method; floating point uses 8 bytes.

Floating point storage has a dramatically expanded data range in comparison to Scaled storage, allowing values to be more precise, but it also uses more disk space. Use scaled where compatibility with pre-V5.31 trend history files is necessary.

You can set the necessary storage method via the Trend Tag or SPC Tag properties form (press the **F2** key to view the extended form). The storage method is set to Scaled by default.

See Also

Calculating disk storage

Calculating disk storage

Equations allow you to calculate the total disk space necessary to store a trend across a specified period of time.

> **Note:** The storage method used for a trend (**Scaled** or **Floating Point**) affects the number of bytes necessary for each sample, so in order to calculate the disk space necessary correctly it is important to base your calculations on the appropriate formula.

To find out which storage method a particular trend is using, refer to the extended Trend Tag Properties dialog. (By default, the **Scaled** storage method is used.)

See Also

Scaled
Floating Point
Reconfiguring history files

Scaled

Each data sample requires two bytes of storage. You can therefore calculate the total disk storage necessary for each trend by using the following formula:

$$\text{Bytes necessary for each trend} = 464 \text{ x No. Files} + 176 + \left(\frac{\text{File Period (secs) x (No. Files) x 2}}{\text{Sample Period (secs)}} \right)$$

For example, if a trend record produces one sample every ten seconds for one week, and you are using five data files (five weeks), the number of bytes necessary is:

$$\text{Bytes necessary} = 464 \text{ x } 5 + 176 + \left(\frac{(7 \text{ x } 24 \text{ x } 60 \text{ x } 60) \text{ x } 5 \text{ x } 2}{10} \right)$$

$$\text{Bytes necessary} = 607{,}296 \text{ bytes}$$

See Also

Floating Point
Reconfiguring history files
Calculating disk storage

Floating point

Each data sample requires eight bytes of storage. This alters the equation to:

$$\text{Bytes necessary for each trend} = 704 \text{ x No. Files} + 160 + \left(\frac{\text{File Period (secs) x (No. Files) x 8}}{\text{Sample Period (secs)}} \right)$$

The number of bytes necessary then becomes:

$$\text{Bytes necessary} = 704 \times 5 + 160 + \left(\frac{(7 \times 24 \times 60 \times 60) \times 5 \times 8}{10} \right)$$

Bytes necessary = 2,422,976 bytes

Note: The calculations above do not take into account the space necessary to store the history file for each trend. This is because these files remain at a set size and therefore do not significantly impact the amount of disk space necessary.

Note: For efficient trends storage, use Windows file compression. By using this method you often can reduce your files to 10% of their original size; the actual amount of compression varies depending on the rate of change of the data.

See Also

Calculating disk storage

Reconfiguring history files

Reconfiguring history files

If you change the configuration of your trend history files (in an existing project), or you change the configuration of a trend tag that affects the number, time, or period of the trend files, you need to delete the existing trend files - before you run the new system.

If you change the path of your trend history files (in an existing project), existing trend data is ignored.

Note: You need to not delete history files (that CitectSCADA creates) from your hard disk while your system is running, as the trend server will attempt to recreate the files and this may cause system performance issues.

Using Path Substitution

Instead of specifying the full path to data files in your system, you can use path substitution.

With path substitution, you define a name that is a substitution for the full directory path. You can then use the substitution name in the following format:

File Name	[SUBSTITUTION]:<filename>

For example, if you decide to store a trend data file called MYFILE in a directory called C:\CITECT\DATA\MYTRENDS, you can specify the full path to the file, for example:

File Name	C:\CITECT\DATA\MYTRENDS\MYFILE

or define a path substitution (for example MYDATA) and specify the path as:

File Name	[MYDATA]:MYFILE

Path substitution provides greater control of data storage. You can change the location of data files by changing the definition of the data path - instead of locating and changing each occurrence of the data path.

See Also

Default path definitions

Default path definitions

CitectSCADA has the following predefined path substitutions:

Path Name	Platform	Default Directory
[Bin]	Pre-Vista and Vista	C:/Program Files/Citect/CitectSCADA 7.20/Bin
[User]	Pre-Vista	Documents and Settings/All Users/Application Data/Citect/CitectSCADA 7.20/User
	Vista	ProgramData/Citect/CitectSCADA 7.20/User
[Data]	Pre-Vista	Documents and Settings/All Users/Application Data/Citect/CitectSCADA 7.20/Data
	Vista	ProgramData/Citect/CitectSCADA 7.20/Data
[Run]	Pre-Vista and Vista	The current project directory
[Copy]	Pre-Vista and Vista	The current copy project directory
[Back]	Pre-Vista and Vista	The current backup project directory

Debugging Trending

The TrendDebug `Citect.ini` parameter is provided to help you while logging and trending data.

Example:

```
[Trend]
TrendDebug=n
```

where *n* can be the combination of the following debug options:

- 1 - Logs the client, server, and redundancy message types and also the samples being written in the Trends Server from normal acquisition.
- 2 - Logs detailed information about the currently active backfill process, including the redundant samples written to the archive.
- 4 - Logs detailed information for the TrendSetTable functions.
- 7 - Logs trend activities.
- 8 - Logs the summary information only for the currently active backfill process.
- 16 - Logs to syslog.dat the client trend data.

These settings can be added together to have combinations of logging levels. For example

```
[Trend]
TrendDebug=6
```

logs the detailed backfill process and `TrendSetTable` functions.

These settings are read dynamically, meaning that you can change these settings while CitectSCADA is running and the changes will take effect from that point onwards.

Chapter: 27 Understanding Statistical Process Control

Statistical quality control (SQC) helps track and improve product or service quality. Statistical process control (SPC) is the primary tool of SQC and encompasses the collection, arrangement, and interpretation of process variables associated with a product, in regard to uniformity of quality.

Every process is subject to variation and therefore there is always room to improve process consistency and quality. To respond to this, adopt a continuous improvement strategy. By repeating the following cycle, a strategy of prevention can be implemented. This is the key to using SPC effectively. Consider the following steps:

- Analyze the process
- What do we know about the variability of this process?
- Is this process statistically controllable (predictable)?
- Is the process capable of meeting the set requirements?
- What considerations are most important to meeting process quality criteria?
- Maintain (control) the process
- Improve the process

SPC encompasses the concepts of variation, statistical control, and process capability, and typically uses the tools XRS control chart, capability chart, and the Pareto chart.

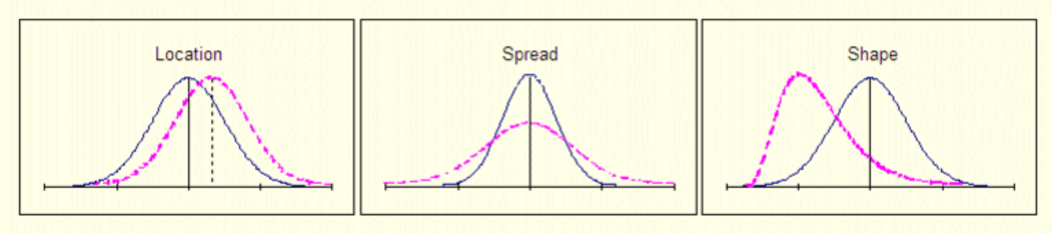
See Also

Process Variation
Statistical Control
Process Capability
XRS Control Charts
Capability Charts
Configuring capability charts
Pareto Charts
Using Statistical Process Control (SPC) with CitectSCADA

Process Variation

To use SPC effectively, understand the concept of *variation*. When a product characteristic is measured repeatedly, each measurement is likely to differ from the last. This is because the process contains sources of variability.

When the data is grouped into a frequency histogram, it will tend to form a pattern. The pattern is referred to as a probability distribution and is characterized in three ways:



> **Note:** Most SPC techniques assume that the collected data has a normal distribution.

Variation is generally categorized into one of two types:

- **Common**: refers to variation that is predictable and repeatable over time. The distribution characteristics will be stable. Common variation could be due to consistent process inaccuracy or similar.

Statistics indicate that common variations account for about 85% of departures from process quality requirements. Usually these departures require solution at the management level.

- **Special**: refers to variation that is not consistently present. When special variation occurs it will tend to change the distribution characteristics. The distribution is not stable over time.

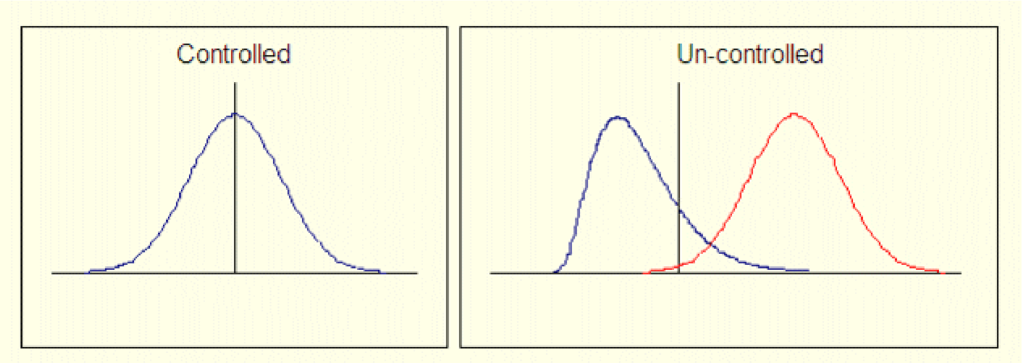
Statistics indicate that special variations account for about 15% of departures from process quality requirements. Typically these departures require local action (equipment repair and so on) for solution.

See Also

Process Variation

Statistical Control

A process is said to be in statistical control when the only sources of variation are from common causes. A statistically controllable process is desirable because it is predictable, while a statistically uncontrollable process will yield unpredictable distributions.



Even though a process might not be statistically controllable, it might still meet requirements. Conversely, a process which is controllable might not meet requirements. This issue is clarified by considering Process Capability.

See Also

Process Variation

Process Capability

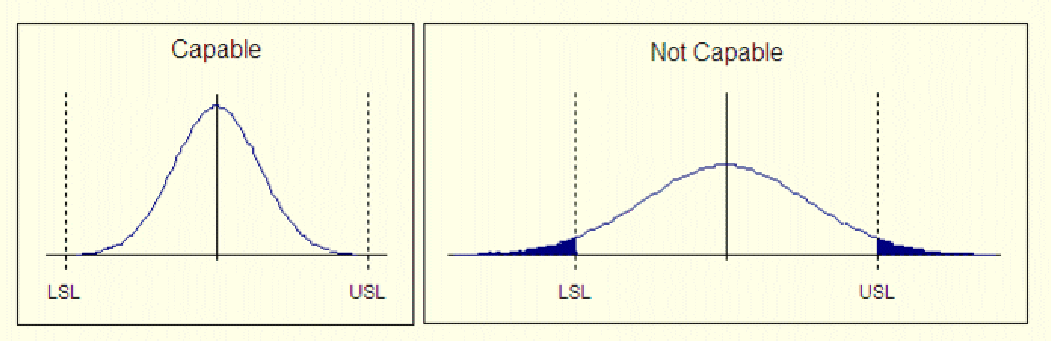
A process is said to be capable when the percentage of samples outside the specification limits is less than a predefined value. The following assumptions need to also be true:

- The process is statistically stable (only common causes of variation exist)
- The individual measurements conform to a normal distribution
- Measurement variation (due to the measuring instrument) is small

The specification limits are a reflection of the customers requirements and are selectable. The percentage of samples that need to lie within the specification limits is calculated from the standard deviation (sigma)"3-sigmas" on either side of the mean.

Note: The "3-sigma" term refers to the boundaries which are located 3 standard deviations on either side of the center. For a normal distribution 99.74% of the samples are expected to fall within this boundary.

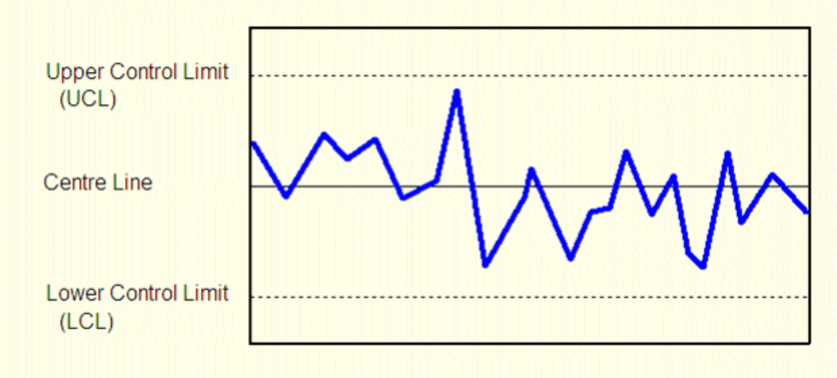
Ultimately capability determines whether the process is statistically able to meet the specification or not. Reducing the effects of common variation will make your process more capable, as shown here:



See Also
XRS Control Charts

XRS Control Charts

XRS charts are the primary tool of SPC and convey information about variation and controllability. The charts are trend graphs that individually show mean (X bar), range (R), and standard deviation (s or sigma). Each point on the graph represents sub-grouped data, not individual samples.



CL, UCL and LCL

Each graph has center, upper control limit (UCL), and lower control limit (LCL) lines drawn. The control limits are estimates of where the "3-sigma" limits would lie for an approximately normal distribution. In practice the limits are calculated from the measured X double bar, R bar, and s bar, and table values which are based on the size of the subgroup used.

These lines are used as a guide for analysis as they are based on the natural variability of the process. These limits are not specification limits.

Interpreting the chart

Control charts can provide valuable information about process variation. By watching for particular patterns and events, conclusions can be drawn as to how the process is controlling.

A (normal) process that is statistically under control will tend to distribute data according to a normal distribution. The data will appear randomly, but centered around the center line. Data that appears near the control limits may be infrequent but expected. Presence of data that appears outside of the control limits indicates the process is statistically uncontrolled and action should be taken to address the cause. Similarly, runs of constant data or patterns indicate instability.

The following list of patterns and events are considered to be cause for alarm:

- Points beyond the control limits
- Several consecutive points on only one side of the average
- Several consecutive points that are consistently increasing or decreasing
- Substantially more than 2/3 of plotted points lie close the average
- Substantially less than 2/3 of plotted points lie close to the average

The presence of these types of patterns and events has different meanings, depending on which type of chart is being analyzed. By using each of the three control charts together (X bar, R and s) together, a better understanding of readings is gained.

See Also

Capability Charts

Capability Charts

The process capability chart is the frequency histogram and distribution of the process mean data and is used to analyze process capability. This chart is drawn with center line, lower specification limit and upper specification limit indicators in place. Interpreting capability charts is typically made with the Cp, Cpk, kurtosis and skewness indices.

USL and LSL

Upper specification limit (USL) and lower specification limit (LSL) specify the requirements of the product, and so are customer driven. Make the target value the midpoint between USL and LSL.

Cp index

The inherent process capability (Cp) is the ratio of tolerance to 6-sigma. Essentially this index indicates whether the distribution would fit inside the USL and LSL limits. Its meaning is defined as follows:

- *Cp > 1.0* - Indicates the process variation will fit within the specified limits (USL and LSL) and therefore, is capable.
- *Cp < 1.0* - Indicates the process is not capable.

Cpk Index

The process capability based on the worst case (Cpk) is similar Cp. This index, however, indicates where the mean lies in relation to the USL and LSL limits. It is used to mathematically clarify Cp. Its meaning is defined as follows:

- *Cpk < 0* - Indicates the process mean is outside the specified limits (USL and LSL)
- *Cpk = 0* - Indicates the process mean is equal to one of the specified limits.
- *Cpk > 0* - Indicates the process mean is within the specified limits.
- *Cpk = 1.0* - Indicates that one side of the 6-sigma limits falls on a specification limit.
- *Cpk > 1.0* - Indicates that the 6-sigma limits fall completely within the specified limits.

See Also

Pareto Charts

Pareto Charts

A Pareto chart is a tool which is used to analyze the relative frequency of deviations from specifications or success criteria. The Pareto chart is a frequency histogram which is ordered from highest to lowest. Unlike control and capability charts, this chart uses no statistical guides.

Pareto charts emphasize Pareto's empirical law that any assortment of events consists of a few major and many minor elements. In the context of SQC, it is important to select the few vital major opportunities for improvement from the many trivial minor ones. The Pareto chart is particularly useful for Cost-to-fix versus Deviation-frequency analysis.

In addition to the histogram, typically a cumulative percentage is also given. From top to bottom, the percentage represents the ratio of the sum of evry value to this point, to the sum of every value in the chart.

Using Statistical Process Control (SPC)

CitectSCADA displays Statistical Process Control information on three types of chart; XRS Control Chart, Process Capability Chart, and Pareto Chart.

To configure an SPC chart:

1. Click **New Page** or choose **File** | **New**.
2. Select **Type: Page**.
3. Choose the **Resolution** (size) of the SPC page.
4. Choose an SPC **Template** for the SPC page:
   - **SPCXRSChart** - An XRS control chart
   - **SPCCpk** - A capability (Cpk) chart combined with an Mean chart

   > **Note:** Pareto charts are configured slightly differently and hence, are not included here.

5. Click **OK**.
6. Double-click the graph display.
7. Enter the variable tag in the Genie pop-up.
8. Click **OK**.
9. Save the page.

See Also

SPC Tags
SPC Control Charts

SPC Tags

SPC tags specify data that is to be collected for use in SPC operations. Once data is defined it can be dynamically analyzed (as SPC graphs and alarms) at run-time. SPC tags are similar to trend tags.

Like trends, CitectSCADA can collect and store any amount of SPC data. The only restriction on the amount of data that you can store is the size of the hard disk on your computer. (CitectSCADA uses an efficient data storage method to optimize the use of storage space on your computer's hard disk.) For long-term storage, you can archive the data to disk or tape (without disrupting your runtime system). You can also log data at regular intervals (periodic trend), or only when an event occurs (event trend), in the same manner as Trend Tags.

> **Note:** SPC does not support Periodic-Event trends, which is a combination of the properties of Periodic and Event trends. In addition, if you use event-based SPC tags, your display screen refresh rate may be slower than desired or practical for your application.

When you define an SPC tag you will want to be sure to fill in the upper specification limit (USL) and lower specification limit (LSL) if you intend to perform a capability analysis. These values have to accurately represent the users requirements, and the target value lie midway between the two. If these fields are left blank the capability analysis will be meaningless.

To configure an SPC tag:

1. Choose **Tags** | **SPC Tags**. The SPC Tags dialog box appears.
2. Enter the SPC tags properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

SPC tag properties

SPC tag properties

Use the SPC Tags dialog box to configure the SPC tag properties. Statistical Process Control (SPC) Tags have the following properties:

SPC Tag Name

The name assigned to the SPC data. If you are logging a variable, use the same name for the SPC tag that you used for the variable tag.

> **Note:** Make SPC tag names unique and not identical to any trend tag names. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

If using the File Name property, ensure that the SPC tag uses a unique name. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Cluster Name

The name of the cluster the SPC tag runs in.

If the Cluster Name is not set, then CitectSCADA considers this SPC tag to run on every defined cluster.

Expression

The logged value of the SPC tag. Enter a value of 254 characters or less. You can log individual variables by using a Variable Tag, for example:

Expression	PT104
Comment	Logs the Variable Tag PT104

The value of the process variable PT104 is logged. Variable PT104 has to be defined as a variable tag. You can also log any Cicode expression or function, for example:

Expression	PT104/COUNTER
Comment	Logs Variable Tag PT104 divided by the Variable Tag COUNTER

Trigger

The Cicode expression (or variable tag) that triggers data logging. Enter a value of 254 characters or less. For example:

Trigger	PT104<500

In this example, logging occurs when the value of the variable tag (PT104) falls below 500.

For a periodic SPC trend, data is logged only while the value of the trigger is TRUE. In the above example, data is logged continuously while the value of PT104 remains less than 500. Logging ceases when the value rises to (or above) 500. Logging does not occur again until the value of PT104 falls below 500.

You do not have to specify a trigger for a periodic SPC trend. If you do not specify a trigger for a periodic SPC trend, then logging will occur continuously.

For an event SPC trend,data is logged once when the value of the trigger changes from FALSE to TRUE. In the above example, one sample is logged when the value of PT104 first becomes less than 500. Another sample is not logged until the value of PT104 rises to (or above 500) and again falls below 500.

Sample Period

The sampling period of the data, in hh:mm:ss (hours:minutes:seconds). CitectSCADA checks the **Trigger** each sample period. If the Trigger is TRUE (or has just changed from FALSE to TRUE, in the case of event SPC trends), CitectSCADA will log the value of the **Expression.**

Examples

Sample Period	Comment
30	Logs data every 30 seconds
10:00	Logs data every 10 minutes
10:00:00	Logs data every 10 hours
2:30:00	Logs data every 2 and a half hours

This property is optional. If you do not specify a sample period, the sampling period will default to 10 seconds.

Note: If you edit this property in an existing project, delete the associated trend files before you run the new runtime system.

Type

The type of SPC trend:

1. TRN_PERIODIC
2. TRN_EVENT

Note: SPC does not support Periodic-Event trends, which is a combination of the properties of Periodic and Event trends. In addition, sif you use event-based SPC tags, your display screen refresh rate may be slower than desired or practical for your application.

Lower Spec Limit

The Lower Specification Limit (LSL). This value is used as the lower limit to determine process capability. When used in conjunction with the USL it provides a tolerance for your process.

If you are unfamiliar with process capability and capability indices, ask for expert opinion. Rather than leave this blank (at least) attempt an estimate. Enter a value that you think is the lowest acceptable value of this tag. If you leave this field blank only your capability analysis will be affected.

Upper Spec Limit

The Upper Specification Limit (USL). This value is used as the upper limit to determine process capability. When used in conjunction with the LSL it provides a tolerance for your process.

If you are unfamiliar with Process Capability and capability indices, ask for expert opinion. Rather than leave this blank (at least) attempt an estimate - Enter a value that you think is the highest acceptable value of this tag. If you do leave this field blank only your capability analysis will be affected.

Comment

Any useful comment.

File Name

The file where the data is to be stored. You need to specify the full path or use path substitution.

When CitectSCADA collects data from your plant floor, it stores the data in a file on the hard disk of your computer. When it subsequently uses the data to display an SPC trend, it reads the data from this file. (CitectSCADA uses a separate file for each SPC tag.)

By default, CitectSCADA stores the file in the [DATA] directory on the hard disk where you installed CitectSCADA. The default name of the file is the SPC tag name. However, you can specify an alternate file name like this:

File Name	`[DATA]:TANK131`

where [DATA] specifies the disk and path for the data. Use path substitution to make your project more 'portable'.

The File Name property is optional. If you do not specify a file name, the file name defaults to [DATA]:<Name> on the hard disk where you installed CitectSCADA. Where <Name> is the SPC Tag Name.

Note: If you use the File Name property, ensure that no other SPC tags or trend tags use the same filename. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

If using the File Name property, ensure that the SPC tag uses a unique name. Two tags accessing the same file can result in system errors which may include lost or corrupted trend/SPC data.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Note: Do not use a file extension when specifying a file name. If you edit this property (change the file name or path) in an existing project, existing SPC data is ignored.

Storage Method

Select either **Scaled** or **Floating Point** as the storage method for the SPC data (64 characters). The key difference between these two options is that Scaled is a two-byte data storage method, whereas Floating Point uses eight bytes.

Floating Point storage has a dramatically expanded data range in comparison to Scaled storage, allowing values to have far greater resolution. However, you need to consider that it also uses a lot more disk space. Scaled should be used where compatibility with pre-V5.31 trend history files is necessary.

If you do not specify a storage method, it is set to **Scaled** by default.

Note: If you edit this property in an existing project, you need to delete the associated trend files - before you run the new runtime system.

No. Files

The number of history files stored on your hard disk (for this tag).

If you do not specify the number of files, 2 history files are stored on your hard disk. The maximum number of files you can specify is 270.

Note: If you edit this property in an existing project, delete the associated SPC trend files - before you run the new runtime system.

Time

The time of day to synchronize the beginning of the history file, in hh:mm:ss (hours:minutes:seconds). If you do not specify a time, the file is synchronized at 0:00:00 (i.e. midnight).

> **Note:** If you edit this property in an existing project, delete the associated SPC trend files before you run the new runtime system.

Period

The period of the history file, in hh:mm:ss (hours:minutes:seconds). Alternatively, you can:

- Specify a weekly period by entering the day of the week on which to start the history file, for example Monday, Tuesday, Wednesday, etc.
- Specify a monthly period by entering the day of the month on which to start the history file, for example 1st, 2nd, 3rd, 4th, 5th, etc.
- Specify a yearly period by entering the day and the month on which to start the history file, for example 1st January, 25th February, etc. The day and month needs to be separated by a space.
  If you do not specify a period, the period defaults to Sunday (weekly).

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

If you edit an SPC tag's Period property, delete associated SPC trend files before placing the system back in service.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

> **Note:** If you edit this property in an existing project, delete the associated SPC trend files before you run the new runtime system.

Subgroup Size

The size of each subgroup. The default value for this value is 5. Valid values are 1 - 25 inclusive.

Standard Deviation

The calculation override for process standard deviation (s bar). If a value is specified here it will be used in every SPC calculation, instead of the value calculated by CitectSCADA. This will affect the calculation of control limits which are normally a function of the collected samples of data.

Do not use this field unless you are experienced in SPC.

X double bar

The calculation override for process mean (X double bar). If a value is specified here it will be used in every SPC calculation, instead of the value calculated by CitectSCADA. This will affect the calculation of control limits which are normally a function of the collected samples of data.

Do not use this field unless you are experienced in SPC.

Range

The calculation override for process range (R bar). If a value is specified here it will be used in every SPC calculation, instead of the value calculated by CitectSCADA. This will affect the calculation of control limits which are normally a function of the collected samples of data.

Do not use this field unless you are experienced in SPC.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to display the SPC data on an SPC page.

Area

The area to which the SPC data belongs. Only users with access to this area (and any necessary privileges) will be able to display the SPC data on an SPC page. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to display the SPC data.

Eng Units

The engineering units of the variable/expression being logged. The engineering units are used by the SPC trend scales and SPC trend cursor displays.

Format

The format of the variable/expression being logged. The format is used by the SPC trend scales and SPC trend cursor displays.

This property is optional. If you do not specify a format, the format defaults to ####.#.

Deadband

The value that **SPC Tag** needs to return to before the SPC data becomes inactive.

SPC Control Charts

CitectSCADA uses the following types of control charts:

- XRS control chart
- Capability charts
- Pareto Charts

See Also

Control Chart Line Constants

XRS control chart

The XRS charts display trends of subgroup means (X bar), ranges (R) and standard deviations (s). The XRS chart operates similarly to a standard trend, but with additional SPC extra features. Each subgroup displays as a single node on the graph and consecutive nodes are linked by a line.

Each control chart has a central line and two control limits-upper and lower (UCL and LCL). CitectSCADA automatically calculates these SPC values at run-time. If you want to override the UCL and LCL you can do so by entering the Process Mean, Range, and Standard Deviation fields in SPC Tags.

See Also

Configuring XRS charts

Configuring XRS charts

Genies simplify the task of adding a new SPC page. To create a new chart:

- Define the SPC Tags.
- Create the page using an XRS template.

> **Note:** If you want to develop your own XRS template, the method is to copy and modify and existing template.

Capability charts

The process capability chart is a frequency histogram and distribution of the sample data currently displayed (on the Mean chart). CitectSCADA automatically takes the data being trended, builds a distribution, adds the LSL and USL. It also calculates the Cp, Cpk, kurtosis, and skewness indices.

The process capability is defined in relation to the upper and lower specification limits (USL and LSL) for a given SPC Tag. These values are defined in SPC Tags and accurately represent the users requirements.

See Also

Configuring capability charts

Configuring capability charts

Genies simplify the task of adding a new SPC page. To create a new chart:

1. Define the SPC Tags and specify the LSL and USL.
2. Create the page using a Capability (Cpk) template.

Pareto Charts

Pareto analysis is a technique used to identify the relative importance of problems and conditions. The Pareto chart is a frequency histogram ordered from highest to lowest-CitectSCADA automatically orders the bars at run-time. The data for each bar in the histogram represents one CitectSCADA variable - as defined in Variable Tags. Do not use SPC tags.

> **Note:** Typically the frequency in a histogram is of integer type, though you can use floating point types if you want. Negative values are not valid.

See Also

Configuring Pareto charts

Configuring Pareto charts

Genies simplify the task of adding a new Pareto chart.

To create a new chart:

1. Define the Variable Tags (Pareto charts do not use SPC Tags).
2. Create the page using a Pareto template.

To configure a Pareto chart:

1. Click New Page, or choose **File** | **New**.
2. Select **Type: Page**.
3. Choose the **Resolution** (size) of the SPC page.
4. Choose the **SPCParetoTemplate** for the SPC chart.
5. Click **OK**.
6. Double-click the display (where prompted by the template).
7. Enter the variable tags in the **Tag Name** fields. (Use Variable tags here, not necessarily SPC tags.)
8. Enter the variable descriptions in the **Tag Description** boxes.
9. Click **OK**.
10. Save the page.

SPC Alarms

CitectSCADAautomatically monitors several special kinds of conditions that are specific to SPC data. When specific patterns or events occur to an SPC tag, it will set the appropriate alarm. Typically these alarms are related to, and used in conjunction with, the XRS control charts.

SPC alarms are configured differently to standard digital alarms to provide for this extra functionality. SPC alarms needs to be configured using the Advanced Alarms form. You use the SPCAlarms Cicode function to check for the condition of the alarms:

> **Note:** An SPC alarm can only be defined on an Alarm Server in the same cluster as the Trends Server that contains the SPC tag (though the variable tag referenced in the SPC tag can be on a different IOServer cluster).

Complete the Advanced Alarm form as shown here:

Advanced Alarms	
Alarm Tag	Feed_Above_UCL
Alarm Desc	Un-controlled variation
Expression	SPCAlarms("Feed_SPC", XAboveUCL)
Comment	Several samples are above UCL

The SPC (Trends) Server checks for any specified alarm conditions. When one is detected, it informs the Alarm Server that an alarm has occurred. Be aware of the number of subgroups displayed on your SPC charts, and the number used in SPC alarm calculations (as set by the [SPC]AlarmBufferSize parameter). If these two values differ, SPC alarms might not correlate with your SPC charts.

The following list shows the alarms types that are valid:

Name	Description
XFreak	Single point greatly differs (2 sigma) from the center line.
XOutsideCL	Process mean outside either of the control limits (UCL or LCL).
XAboveUCL	Process mean above the upper control limit (UCL).
XBelowLCL	Process mean below the lower control limit (LCL).
XOutsideWL	Process mean outside the alert limits which are 67% of the UCL and LCL.
XGradualUp	Process mean is gradually drifting up to a new level indicated by several consecutive points above the mean.
XGradualDown	Process mean is gradually drifting down to a new level indicated by several consecutive points below the mean.
XUpTrend	Several points continuously increasing in value.
XDownTrend	Several points continuously decreasing in value.
XErratic	Large fluctuations that are greater than the control limits.
XStratification	Artificial constancy. Several consecutive points are close to (within 1 sigma of) the center line.
XMixture	Several consecutive points are far from (outside 1 sigma of) the center line.
ROutsideCL	Process range outside either of the control limits (UCL or LCL).
RAboveUCL	Process range above the upper control limit (UCL).
RBelowLCL	Process range below the lower control limit (LCL).

Note: The above alarms rely on ***n*** number of consecutive points to generate the alarm. The value of ***n*** can be set for each type of alarm through SPC parameters.

See Also

SPC Formulas and Constants

SPC Formulas and Constants

The SPC calculations are based on the samples collected in subgroups. Each subgroup will have the same number of samples, typically 4. The subgroup size for each SPC tag is set at the **SPC Tags** properties form.

The number of samples in each subgroup can range from 1 to 25 inclusive.

When the number of samples in each subgroup is 1

Subgroup Mean ($\bar{X}$):

is the value of the single sample in the group, and is defined by:

$$\bar{X}_i = X_i$$

Where X_i is the single sample value in the subgroup.

Moving Range (MR):

is the difference between successive sample values, and is defined by:

$$MR_i = X_i - X_{i-1} \qquad i \geq 2$$

Where X_i is the current sample value and X_{i-1} is the previous sample value. The number of moving ranges in the process is always one less than the number of subgroups.

Subgroup Standard Deviation (s):

is a measure of absolute variation or dispersion. It describes how much the sample values differ from their mean, and is estimated by:

$$s_i = \frac{MR_i}{D2} \qquad i \geq 2$$

The number of subgroup standard deviations in the process is always one less than the number of subgroups.

Process Average ($\bar{\bar{X}}$):

$$\overline{\overline{X}} = \frac{\overline{X}_1 + \overline{X}_2 + \text{K} + \overline{X}_m}{m}$$

Where $\overline{X}_1$, $\overline{X}_2$, and $\overline{X}_m$ are the subgroup means, and m is the total number of subgroups in the process.

Process Range ($\overline{R}$):

$$\overline{R} = \frac{MR_2 + MR_3 + \text{K} + MR_m}{m - 1}$$

Where MR_2, MR_3, and MR_m are the subgroup moving ranges, and m is the total number of subgroups in the process.

Process Standard Deviation ($\overline{s}$):

$$\overline{s} = \frac{\overline{R}}{D2}$$

When the number of samples in each subgroup is greater than 1

Subgroup Mean ($\overline{X}$):

is the average (not median or center) of the samples in the group, and is defined by:

$$\overline{X} = \frac{X_1 + X_2 + \ldots + X_n}{n}$$

Where X_1, X_2, and X_n are the sample values in the subgroup, and n is the total number of samples in the subgroup.

Subgroup Range (R):

is the difference between the highest and lowest samples in the group, and is defined by:

$$R = X_{max} - X_{min}$$

Where X_{max} is the maximum sample value and X_{min} is the minimum sample value in the group.

Subgroup Standard Deviation (s):

is a measure of absolute variation or dispersion. It describes how much the sample values differ from their mean, and is defined by:

$$s = \sqrt{\frac{\sum(X_i - \overline{X})^2}{n-1}}$$

Where X's are the sample values in the group, $\overline{X}$ is the group average, and n is the number of samples in the group.

Process Average ($\overline{\overline{X}}$):

$$\overline{\overline{X}} = \frac{\overline{X}_1 + \overline{X}_2 + \ldots + \overline{X}_m}{m}$$

Where $\overline{X}_1$, $\overline{X}_2$, and $\overline{X}_m$ are the subgroup averages, and m is the total number of subgroups in the process.

Process Range ($\overline{R}$):

$$\overline{R} = \frac{R_1 + R_2 + \ldots + R_m}{m}$$

Where R_1, R_2, and R_m are the subgroup ranges, and m is the total number of subgroups in the process.

Process Standard Deviation ($\overline{s}$):

$$\overline{s} = \frac{s_1 + s_2 + \ldots + s_m}{m}$$

Where s_1, s_2 and s_m are the group standard deviations, and m is the total number of groups in the process.

Average Control Limits (UCL_x and LCL_x):

Specify the approximated 3-sigma boundaries. For a normal distribution 99.74% of the samples will fall within this boundary.

$$UCL_x = \overline{\overline{X}} + A_2 * \overline{R}$$
$$LCL_x = \overline{\overline{X}} - A_2 * \overline{R}$$

Where $\overline{R}$ is the Process Range and A_2 is a constant (given in the Control Chart Line Constants table).

Range Control Limits (UCL_R and LCL_R):

Specify the approximated 3-sigma boundaries. For a normal distribution 99.74% of the samples will fall within this boundary.

$$UCL_R = D_4 * \bar{R}$$
$$LCL_R = D_3 * \bar{R}$$

Where $\bar{R}$ is the Process Range and D_3 and D_4 are constants (given in the Control Chart Line Constants table).

Standard Deviation Control Limits (UCL$_s$ and LCL$_s$):

Specify the approximated 3-sigma boundaries. For a normal distribution 99.74% of the samples will fall within this boundary.

$$UCL_s = B_4 * \bar{s}$$
$$LCL_s = B_3 * \bar{s}$$

Where $\bar{s}$ is the Process Standard Deviation and B_3 and B_4 are constants given in the Control Chart Line Constants table).

Process Capability (Cp):

Is the capability of a process to meet a specific tolerance. A process is considered capable when the percentage of samples of a variable for that process that fall within the upper and lower specification limits is greater than a specified value.

The inherent process capability is defined as:

$$Cp = \frac{(USL - LSL)}{6\bar{s}}$$

- *Cp > 1.0* - Indicates the process variation is within the specified limits (USL and LSL) and therefore, is capable.
- *Cp < 1.0* - Indicates the process is not capable.

The process capability based on worst case data is defined as:

$$Cpk = \frac{\min((USL - \bar{\bar{X}}),(\bar{\bar{X}} - LSL))}{3\bar{s}}$$

- *Cpk < 0* - Indicates the process mean is outside the specified limits (USL and LSL)
- *Cpk = 0* - Indicates the process mean is equal to one of the specified limits.
- *Cpk > 0* - Indicates the process mean is within the specified limits.
- *Cpk = 1.0* - Indicates that one side of the 6-sigma limits falls on a specification limit.
- *Cpk > 1.0* - Indicates that the 6-sigma limits fall completely within the specified limits.

Skewness (Sk):

Is the degree of asymmetry of a frequency distribution (usually in relation to a normal distribution).

$$Sk = \frac{\sum (X_i - \bar{\bar{X}})^3}{N\bar{s}^3}$$

where N is the number of samples for the entire process (i.e. Subgroup Size * number of Subgroups).

- Skewness > 0 - Indicates that the histogram's mean (and tail) is pushed to the right.
- Skewness < 0 - Indicates that the histogram's mean (and tail) is pushed to the left.

Kurtosis (Ku)**:**

Is the degree of peakedness of a frequency distribution (usually in relation to a normal distribution).

$$Ku = \frac{\sum (X_i - \bar{\bar{X}})^4}{N\bar{s}^4}$$

where N is the number of samples for the entire process (i.e. Subgroup Size * number of Subgroups).

- Kurtosis < 3 - Indicates a thin distribution with a relatively high peak.
- Kurtosis > 3 - Indicates a distribution that is wide and flat topped.

Control Chart Line Constants

The table below shows the control chart line constants:

Samples in	**Averages**	**Ranges**			**Standard Deviations**	
Group	A2	D2*	D3	D4	B3	B4
1	2.660	1.128	0	3.267	0	3.267
2	1.880		0	3.267	0	3.267
3	1.023		0	2.574	0	2.568

4	0.729	0	2.282	0	2.266
5	0.577	0	2.114	0	2.089
6	0.483	0	2.004	0.030	1.970
7	0.419	0.076	1.924	0.118	1.882
8	0.373	0.136	1.864	0.185	1.815
9	0.337	0.184	1.816	0.239	1.761
10	0.308	0.223	1.777	0.284	1.716
10	0.308	0.223	1.777	0.284	1.716
11	0.285	0.256	1.744	0.321	1.679
12	0.266	0.283	1.717	0.354	1.646
13	0.249	0.307	1.693	0.382	1.618
14	0.235	0.328	1.672	0.406	1.594
15	0.223			0.428	1.572

		0.347	1.653		
16	0.212	0.363	1.637	0.448	1.552
17	0.203	0.378	1.622	0.466	1.534
18	0.194	0.391	1.608	0.482	1.518
19	0.187	0.403	1.597	0.497	1.503
20	0.180	0.415	1.585	0.510	1.490
21	0.173	0.425	1.575	0.523	1.477
22	0.167	0.434	1.566	0.534	1.466
23	0.162	0.443	1.557	0.545	1.455
24	0.157	0.451	1.548	0.555	1.445
25	0.153	0.459	1.541	0.565	1.435

* D2 is only used for estimating standard deviation when there is one sample per sub-group.

Reference ANSI Z1.1-1985, Z1.2-1985 & Z1.3-1985: American National Standard, *Guide for Quality Control Charts, Control Chart Method of Analyzing Data, Control Chart Method of Controlling Quality During Production.*

Hints

Double-click the chart area on an SPC page to display the SPC Genie and change the SPC variables.

The tools and menu items in these procedures automatically open the CitectSCADA form for you. Move the cursor till it changes to a hand to find these tools or options.

Chapter: 28 Reporting Information

You can request regular reports on the status of your plant, and reports that provide information about special conditions in your plant. Reports can be run on a request basis, at specified times, or when certain events occur (such as a change of state in a bit address). Output from a report is controlled by a device. A report can be printed when it runs or saved to disk for printing later. You can use a text editor or word processor to view, edit, or print the report, or you can display it in CitectSCADA as part of a page.

Reports can also include Cicode statements that execute when the report runs.

Reports are configured in two stages:

- Report properties
- Report format file

If report data is associated with an I/O Device that does not initialize properly at startup or goes offline while CitectSCADA is running, the associated data is not written to the report (because the values would be invalid). An error code is written instead.

See Also

Configuring reports
Running Reports
Report Format File
Handling Communication Errors in Reports

Configuring reports

To design, configure and use a report:

1. Configure a device for output of the report (for example, if you want to save a report to a file when it is run, set up an ASCII_DEV device).
2. Configure the report properties.
3. Edit the report format file. Remember that for an RTF report, the report format file needs to be saved in RTF format (that is, with an .RTF file extension).
4. Define your PC as a reports server using the **Computer Setup Wizard**.

To configure report properties:

1. Choose **System** | **Reports** to display the Reports dialog box.
2. Enter the reports properties. Click **Edit** to edit the report format file.

Use the Reports dialog box to configure your reports.

See Also

Reports dialog box

Reports dialog box

The Reports form has the following properties:

Name

The name of the report. The name can be a maximum of 79 characters. It can consist of any character other than the semi-colon (;) or single quote ('). The name needs to be unique to the cluster.

> **Note:** Where Cluster Name is left blank, the name needs to be unique to every defined cluster.

Cluster Name

The name of the cluster that runs this report. If the Cluster Name is not set, then Citect-SCADA considers this report to run on every defined cluster.

Time

The time of day to synchronize the report, in hh:mm:ss (hours:minutes:seconds). If you do not specify a time, the report is synchronized at 0:00:00 (i.e., midnight). Enter a value of 32 characters or less.

Period

The period of the report, in hh:mm:ss (hours:minutes:seconds). Enter a value of 32 characters or less. Alternatively you can:

- Specify a weekly period by entering the day of the week when the report is to start, for example, Monday, Tuesday, Wednesday, etc.
- Specify a monthly period by entering the day of the month when the report is to start, for example, 1st, 2nd, 3rd, 4th, 5th, etc.
- Specify a yearly period by entering the day and the month when the report is to start, for example, 1st January, 25th February, etc. The day and month needs to be separated by a space.

If you do not specify a period, the report is run daily.

Trigger

Any Cicode expression (or Variable tag) to trigger the report. Enter a value of 254 characters or less. If the result of the expression (in this field) is TRUE, and the **Time** and **Period** fields are blank, the report is run. The report is only run when the expression becomes TRUE, and it needs to become FALSE then TRUE again before the report is re-run.

Report Format File

The name of the report format file. Enter a value of 253 characters or less. If you do not specify a file extension, it defaults to .RPT. Any valid file name can be used; however, you cannot use a Path Substitution in this field. If you specify a filename without a path, the file saves into the directory predefined as Run. The report is assumed (by the Citect-SCADA compiler) to be ASCII unless an RTF extension is used.

Note: The file name of your report format file can be up to 64 characters long, or 253 characters including the path. It can consist of any characters other than the single quote ('), and the semi-colon (;).

Output Device

The device where the report will be sent. Enter a value of 16 characters or less.

For RTF reports that are to be saved as a file, select a device of type ASCII_DEV here. Due to the differing natures of their content; however, it is not recommended that the same ASCII device be used for logging both RTF and non-RTF reports.

Note: If two or more reports are running at the same time and are sending their output to the same printer, the output of each report can become mixed. You need to use semaphores to control the access to the printer in each report. See the SemOpen Cicode function. If the report only contains Cicode statements (and has no output data), this property is optional.

Comment

Any useful comment. Enter a value of 48 characters or less.

Extended forms fields

The following fields are implemented with extended forms (press **F2**).

Privilege

The privilege necessary by an operator to run this report if the report is a command-driven report. Enter a value of 16 characters or less.

If the report is time-driven or event-driven, this property is ignored.

Note: If you assign an acknowledgment privilege to a report, do not assign a privilege to the command(s) that run the report. If you do assign a different privilege to the commands, an operator needs to have both privileges to run the report.

Area

The area to which this report belongs. Enter a value of 16 characters or less. Only users with access to this area (and any necessary privileges) will be able to run this report. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to run this report.

Running Reports

You can run a report by the following methods:

- Automatically when CitectSCADA starts up
- Automatically at a specified time and period
- Automatically when an event is triggered
- By using a command
- A combination of the above

See Also

Running a report on startup
Specifying times and periods
Using triggers
Using commands

Running a report on startup

You can run a report on startup. CitectSCADA searches for a report called "Startup" when it starts up, and if CitectSCADA locates this report, it is run automatically. You can change the name of the default report with the *Computer Setup Wizard*.

See Also

Specifying times and periods

Specifying times and periods

The period determines when the report is run. You can specify the period in hh:mm:ss (hours:minutes:seconds), for example:

Period	Comment
1:00:00	Run the report every hour
6:00:00	Run the report every six hours
72:00:00	Run the report every three days
Monday	Run the report each Monday
15th	Run the report on the 15th of each month
25th June	Run the report on the 25th of June

You can also specify the time of day to synchronize the report, for example:

Time	Comment
6:00:00	Synchronize the report at 6:00 am
12:00:00	Synchronize the report at 12:00 midday

The time synchronizes the time of day to run the report and, with the Period, determines when the report is run, for example:

Time	Period
6:00:00	1:00:00

In this example, the report is run every hour, on the hour. If you start your runtime system at 7:25am, your report is run at 8:00am, and then every hour after that.

See Also

Using triggers

Using triggers

You can use any Cicode expression (or variable tag) as a trigger for a report. If the result of the expression (in the **Trigger** field) becomes TRUE, and if the **Time** and **Period** fields are blank, the report is run. For example:

Time	
Period	
Trigger	RCC1_SPEED<10 AND RCC1_MC

This report is only run when the expression (Trigger) becomes TRUE, i.e., when the digital tag RCC1_MC is ON and the analog tag RCC1_SPEED is less than 10. The expression needs to become FALSE and then TRUE again before the report is run again.

If you use the **Time** and/or **Period** fields, the trigger is checked at the time and/or period specified, for example:

Time	6:00:00
Period	1:00:00
Trigger	RCC1_SPEED<10 AND RCC1_MC

This report is run each hour, but only if the expression (Trigger) is TRUE (i.e., if the digital tag RCC1_MC is ON and the analog tag RCC1_SPEED is less than 10).

See Also

Using commands

Using commands

If the **Time**, **Period**, and **Trigger** fields are blank, the report can only be run by a command that calls the Cicode `Report()` function.

Report Format File

The report format file specifies how data is formatted in a report. You can use fixed text, Cicode expressions, and database variables in any report.

You use a text editor that is supported by Windows to create (and modify) the report format file. If your report format file is in RTF (Rich Text Format), use Microsoft Wordpad.

Including fixed text

You can include fixed text in the report, specifying the text exactly as you want it to appear (for example, Name of Report, Description, and so on).

Including OLE (RTF files only)

Objects can be linked to or embedded within an RTF report format file; however, such objects will not be displayed or printed from CitectSCADA.

Using fonts (ASCII format only)

If your format file is in ASCII format, you can use any text font supported by Windows in the report. To specify a font, use the `PrintFont()` function. RTF format files do not require this function, as they use the formatting features of the host word processor.

Including Cicode expressions and variables

You can include Cicode expressions and variables by enclosing them (and optional format specifications) in braces {} - for example:

```
{TIME(1)  }
{PV12:####.##}
{PV12:4.2}
```

The size of each field (number of characters) is determined by either the format specification, or by the number of characters between the braces. In the above example, the variable PV12 is formatted with four characters before the decimal point and two characters after.

You do not have to include the format, for example:

```
{PV12}
```

Here, the variable is formatted using only four characters (the number of characters between the braces).

The following rules apply when logging a report to a **database device**:

- The format (for the report field) do not specify a field size greater than the size of the relevant field specified in the device.
- No spaces are allowed between each field specification, for example:

```
{TIME(1) }{PV12:####.##}{PV12:4.2}
```

Including blocks of Cicode

You can include a block of Cicode, using the following format:

```
{CICODE}
Statements;
{END}
```

The block of Cicode is delimited by the commands `{CICODE}` and `{END}`. After the `{END}` command, the report switches back into WYSIWYG mode. If the entire report is Cicode or the last section is Cicode, the {END} command is not necessary.

A block of Cicode does not send any output to the device unless you use either the `Print()` or `PrintLn()` functions. If you use one of these functions, the argument is printed to the device.

Cicode variables (ASCII format only)

You can also declare variables for use within your Cicode block. This is not available in RTF or HTML format files. Declare any variables at the beginning of the file (i.e. before any report format or Cicode). Add a `{CICODE}` block first; for example:

```
{CICODE}
INT nVar1;
STRING sVar2;
Statements;
{END}

Remainder of report
```

Including comments

You can include comments by using the comment character `!' enclosed in braces - for example:

```
{!This is a Comment}
```

A comment in the body of a report differs from a comment in a Cicode block: a comment in a Cicode block does not require braces.

Including other report elements

The following table describes other elements you can include in reports.

To...	Do this...
Issue a form feed	Use a form feed specifier: {FF}
Include a plot	Use the Plot functions.
Include trend data	Use the TrnGetTable() function.
Include trend graphs	Use the TrnPlot() function.

See Also

Report example

Report example

The following is an example of a report format file (for a printer or ASCII file device):

```
-----------------------------------
SHIFT REPORT
-----------------------------------
{Time(1) }  {Date(2) }
Shift Production {Shift_Prod:###.##} tons
Total Production {Total_Prod} tons
{! The following Cicode displays "Shift Report Complete" on the
screen}
{CICODE}
```

```
Print("End of Report")
Shift_Prod = 0;  ! Reset the Shift production tonnage

Prompt("Shift Report Complete");
{END}
{FF}
```

This report produces the following output to the device and displays "Shift Report Complete" on the graphics page.

```
-----------------------------------
SHIFT REPORT
-----------------------------------
6:00am    12/3/92
Shift Production 352.45 tons
Total Production 15728 tons
End of Report
```

To edit a report format file:

1. From the Reports properties form, select the relevant report, and click **Edit**. Alternatively, click **Edit Report**.
2. Select the report to edit
3. Click **Edit**.

> **Note:** If the report format file exists, it is loaded into the editor for you to edit. If the file does not exist, CitectSCADA creates a new file.

To change the report format file editor:

1. Click the **Options** tool, or choose **File | Options**.
2. Enter the name of the new **Editor**.
3. Click **OK**.

Handling Communication Errors in Reports

You can handle errors in communication with I/O Devices (for example, an I/O Device does not initialize properly at startup or goes offline while CitectSCADA is running) in two ways:

- You can write communication errors and invalid data to the report as error codes.
- You can disable the running of reports that are triggered from an I/O Device, if communication with the I/O device has been lost.

See Also

Reporting errors in I/O Device data
Suppressing reports

Reporting errors in I/O Device data

If a communication error is detected (with an I/O Device) or if the data is invalid, one of the following alert messages are written to the report (instead of the value):

Error	Meaning
#ASS	The value is incorrectly associated (with a substitution string or Genie).
#COM	Communication with the I/O Device has been lost
#DIV/0	An attempt was made to divide a number by 0 (zero)
#ERR	An uncommon error has occurred. (Use the IsError function to find the occurrence.)
#MEM	Out of memory or more than 64 kb bytes of memory requested.
#PEND	Data from this device is pending an initial update to display a value.
#RANGE	The value returned is out of range
#STACK	The value has caused a stack overflow
#WAIT	Data from this scheduled device is not available as it has not reached its scheduled interval and has no cache value.

For example:

Report Format:

```
{PV_1} {SP_1} {OP_1}
```

If the above report is run when the value of PV_1 is out of range (for example 101.5), SP_1 is 42.35 and OP_1 is 60.0, the output of the report is:

Report Output:

```
#RANGE 42.35 60.0
```

When reports are written to a database device, you might sometimes want to disable the alert messages and write the values to the report (even if the values are invalid). Use the ERR_FORMAT_OFF command to disable alert messages and write data as values.

For example:

Report Format:

```
{ERR_FORMAT_OFF}
{PV_1} {SP_1} {OP_1}
If the above report is run when the value of PV_1 is out of range (for example
101.5)
, SP_1 is 42.35 and OP_1 is 60.0, the output of the report is:
```

Report Output:

```
42.35 60.0
```

To re-enable the alert messages, use the ERR_FORMAT_ON specifier.

Note: If an I/O Device goes offline and you have disabled communication errors, the value printed into the report is either 0 (zero) or the last value read from the I/O Device when the report was last run. In either case, the value is invalid.

See Also

Suppressing reports

Suppressing reports

You can suppress reports that are triggered from I/O Devices if a communication error is detected by using the `[Report]ComBreak` parameter. For example, you might configure a report to be run every hour when a bit is on. The I/O Device associated with that bit goes offline. If the `[Report]ComBreak` parameter is 0, the report does not run. If the parameter is 1, and if the latest valid value that was read from that bit was 1, the report is run.

This parameter only applies to the trigger of the report, not to the data in the report.

Chapter: 29 Using Labels

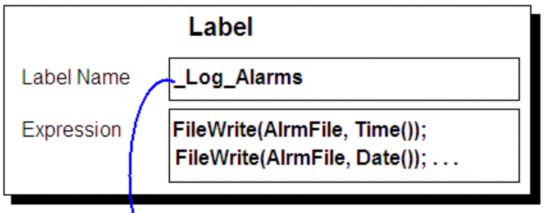
Labels allow you to use a series of commands (or expressions) in your system without having to repeat them each time they are used. When you compile the project, the commands (or expressions) defined in the label are substituted in every occurrence of the label.

Labels are similar to macros used in programming languages such as Basic or 'C'.

You often use the same combination of statements in different commands. For example, when an operator acknowledges an alarm, you might want to log the details to a file. Such a command would require several statements:

```
Command        FileWrite(AlrmFile, Time()); FileWrite(AlrmFile, Date()); . . .
```

Instead of entering the same statements when necessary in a command, you can define a label, and then use the label instead of the statements. When you compile your project, each occurrence of the label is resolved; that is, the expression in the label is substituted for the label name. For example:



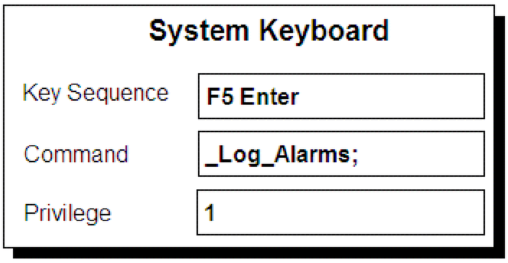
Label

Label Name: _Log_Alarms

Expression: FileWrite(AlrmFile, Time());
FileWrite(AlrmFile, Date()); . . .

Notice the use of an underscore to identify the label.

Once defined, a label can be used as a statement in a command, for example:



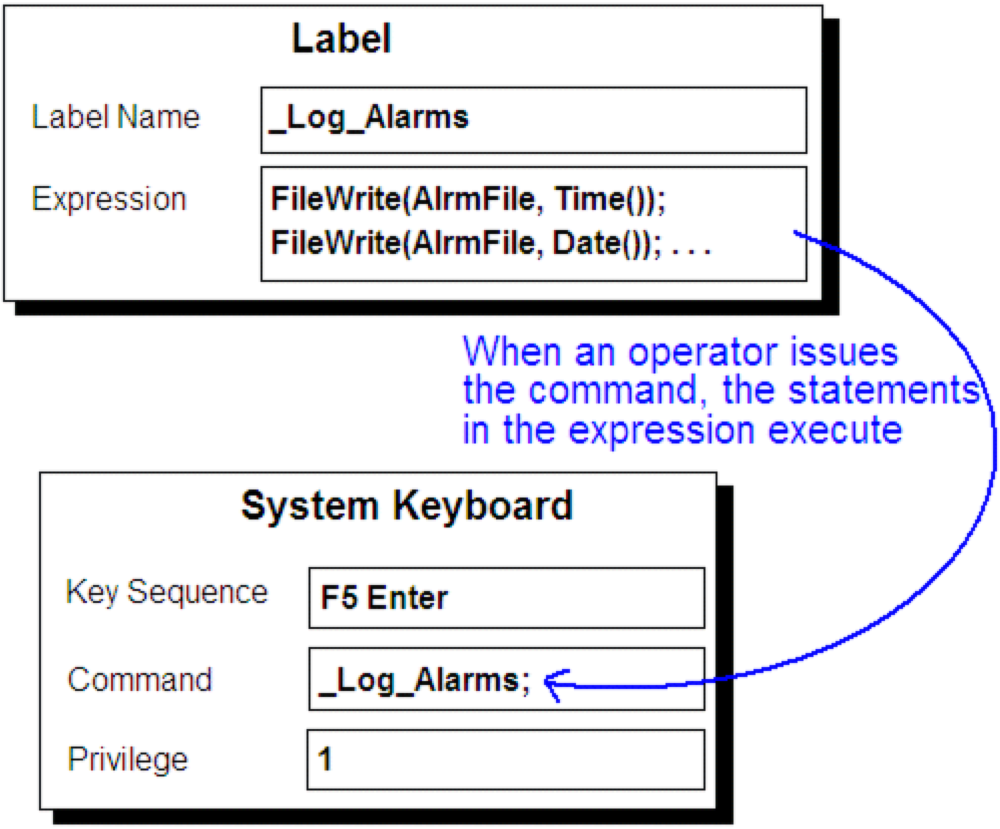
System Keyboard

Key Sequence: F5 Enter

Command: _Log_Alarms;

Privilege: 1

When an operator issues this command, the expression defined in the label is substituted in the command.



Label

Label Name: _Log_Alarms

Expression: FileWrite(AlrmFile, Time());
FileWrite(AlrmFile, Date()); . . .

When an operator issues the command, the statements in the expression execute

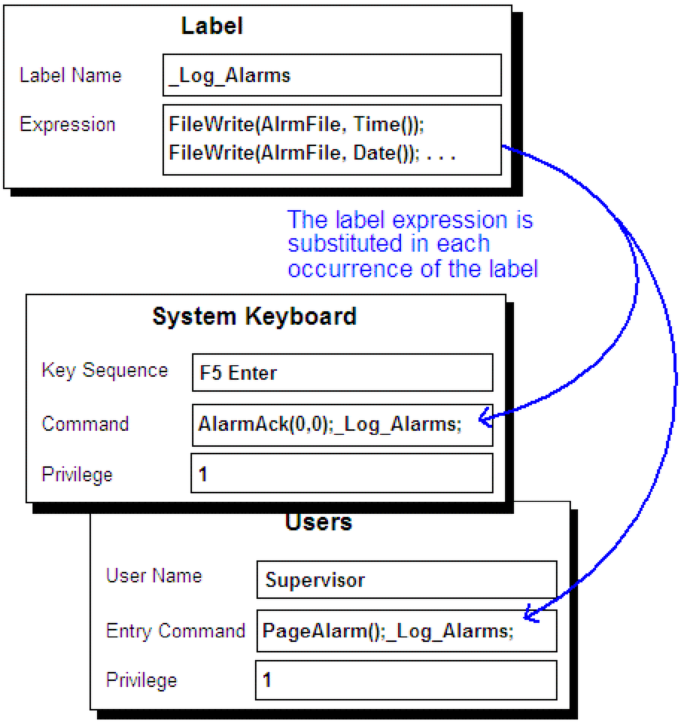
System Keyboard

Key Sequence: F5 Enter

Command: _Log_Alarms;

Privilege: 1

You can also use the label in combination with other statements, for example:



Label

Label Name: _Log_Alarms

Expression: FileWrite(AlrmFile, Time()); FileWrite(AlrmFile, Date()); . . .

The label expression is substituted in each occurrence of the label

System Keyboard

Key Sequence: F5 Enter

Command: AlarmAck(0,0);_Log_Alarms;

Privilege: 1

Users

User Name: Supervisor

Entry Command: PageAlarm();_Log_Alarms;

Privilege: 1

The main advantage of a label is that it is a global definition, recognized throughout the CitectSCADA system. If you want to change something (in the above example you might change the file name or the way the data is logged), you only need to change it in one place - in the label definition. Other occurrences of the label name will reflect the changes.

See Also

Using Arguments in Labels
Converting Values into Strings
Substituting Strings
Defining Labels

Using Arguments in Labels

You can define labels that accept arguments enclosed in parentheses (). The following example shows a label that increments a variable by a specific value:

Label Name	Inc(X, STEP)
Expression	X = X + STEP

Here, "X" is the variable to be incremented and "STEP" determines the amount of the increment. You can then use this label in a command, as in the following example:

Key Sequence	FastInc
Command	Inc(SP12, 10);

An operator can use this command to increment the value of SP12 by 10.

Specifying default values

You can specify a default value for an argument when you define a label, for example:

Label Name	Inc(X, STEP = 10)
Expression	X = X + STEP

When you subsequently use this label without any arguments in a command, the default value is used, for example:

Key Sequence	FastInc
Command	Inc(SP12);

See Also

Converting Values into Strings

Converting Values into Strings

Sometimes, you need to convert a value into a string before it can be used. In the following example, the value of a tag is converted before it is used in the DspStr() function.

Label Name	ShowVariable(TAG)
Expression	DspStr(25, "BigFont", #TAG + "=" + TAG:##.#);

In the above example, only one argument (TAG) is passed to a function that actually requires three arguments (AN, font and message). When you use this label in a command, the function uses AN 25 and the message displays in "BigFont". Only the third argument (the actual message) varies.

The third argument passed to the function is:

```
...#TAG+"="+TAG:##.#
```

#TAG indicates that the name of the tag (and not its value) is displayed.

TAG:##.# indicates that the value of TAG is converted to a string and displayed. It is formatted with two numbers before the decimal point and one number following the decimal point.

You can use the above label in a command such as:

Command	ShowVariable(SP12);

When you use this command in your runtime system, the command displays "SP12=<**value**>", where **value** is the actual value of SP12 at the time (for example **SP12=42.0**).

See Also

Substituting Strings

Substituting Strings

You can pass a string substitution as an argument in a label, for when several variables have part of the variable name in common; for example:

Label Name	SPDev(TAG)
Expression	Prompt("Deviation=" + "IntToStr(CP##TAG## - SP##TAG##));

In the above example, TAG is the common portion of the variable name, and is substituted at each occurrence in the expression. To display the difference between two variables CP123 and SP123, you would specify SPDev(123) in a command, for example:

Command	SPDev(123);

You cannot use a substitution within a string. In the following example, the DESC Parameter (a text description) will not be substituted as it is between quotation marks:

```
Prompt("Deviation for ##DESC##=" + "IntToStr(CP##TAG## -
SP##TAG##))
```

See Also

Defining Labels

Defining Labels

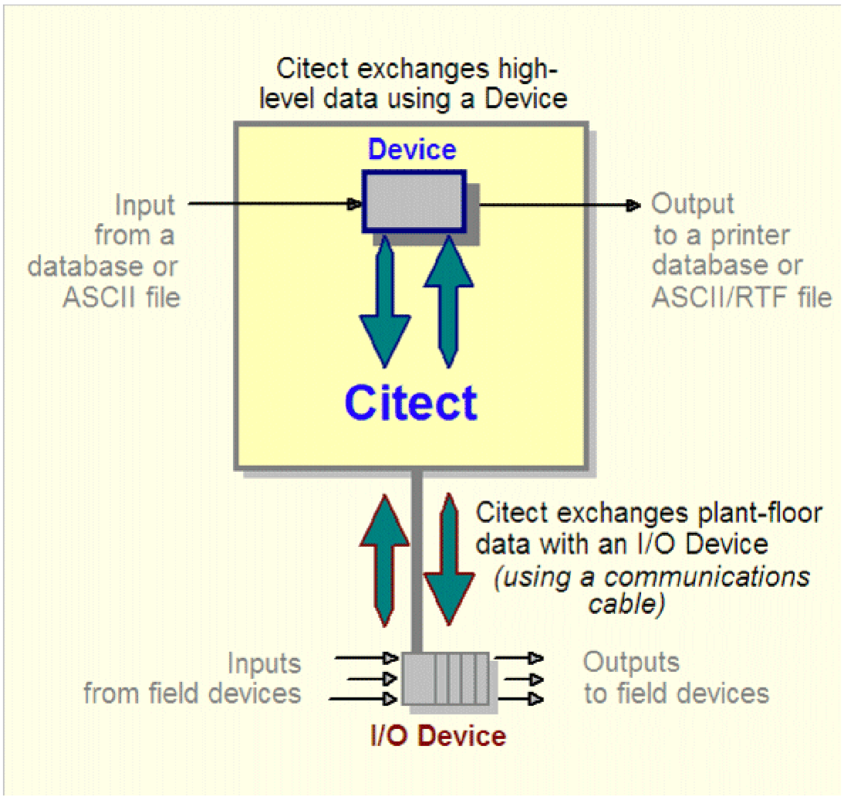
You can define labels to use in your system.

To define a label:

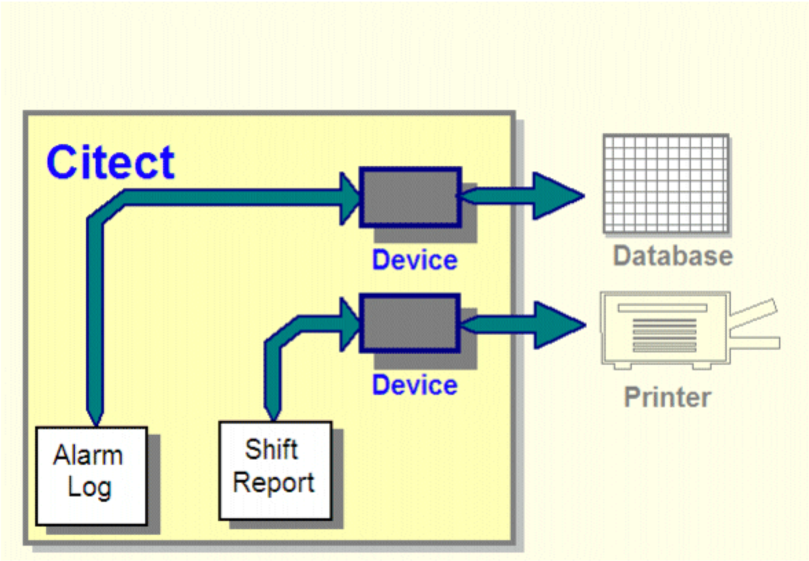
1. Choose **System | Labels.** The Labels dialog box will display.
2. Enter a **Label Name** of 128 characters or less. Whenever this name is used (i.e,. in Cicode or a field), CitectSCADA automatically substitutes the expression below.
3. Enter an **Expression** to be substituted for the label (maximum length is 254 characters). You can use a label to substitute a name for an entity or Cicode expression; for instance, when you use the entity (or Cicode expression) in several database records.
4. Add a **Comment** (of 48 characters or less).
5. Click **Add** to append a new record, or **Replace** to modify an existing record.

Chapter: 30 Using Devices

A device transfers high-level data (such as a report, command log or alarm log) between CitectSCADA and other elements (such as a printer, database, RTF file, or ASCII file) in your CitectSCADA system. Devices are similar to I/O Devices in that they both allow CitectSCADA to exchange data with other components in your control and monitoring system.


Citect exchanges high-level data using a Device
Device
Input from a database or ASCII file
Output to a printer database or ASCII/RTF file
Citect
Citect exchanges plant-floor data with an I/O Device
(using a communications cable)
Inputs from field devices
Outputs to field devices
I/O Device

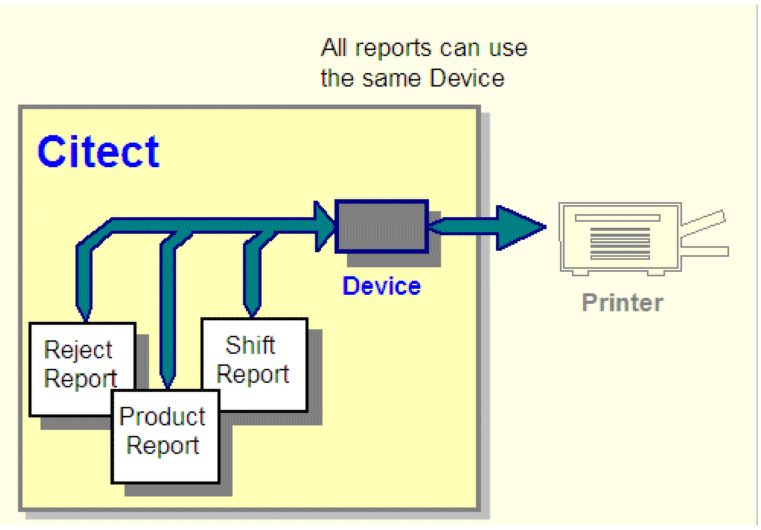
You can use devices for various purposes; for example, to send the output of a report to a printer, or write data to a database.



Using a device you can write data to:

- RTF files
- ASCII files
- dBASE databases
- SQL databases (through ODBC-compliant drivers)
- Printers (connected to your CitectSCADA computer or network)

You can configure any number of devices; however, a device is a common resource. You can, for example, configure a single device that sends the output of your CitectSCADA reports to a printer (when they are requested).



See Also

[Using groups of devices]

[Configuring Devices]

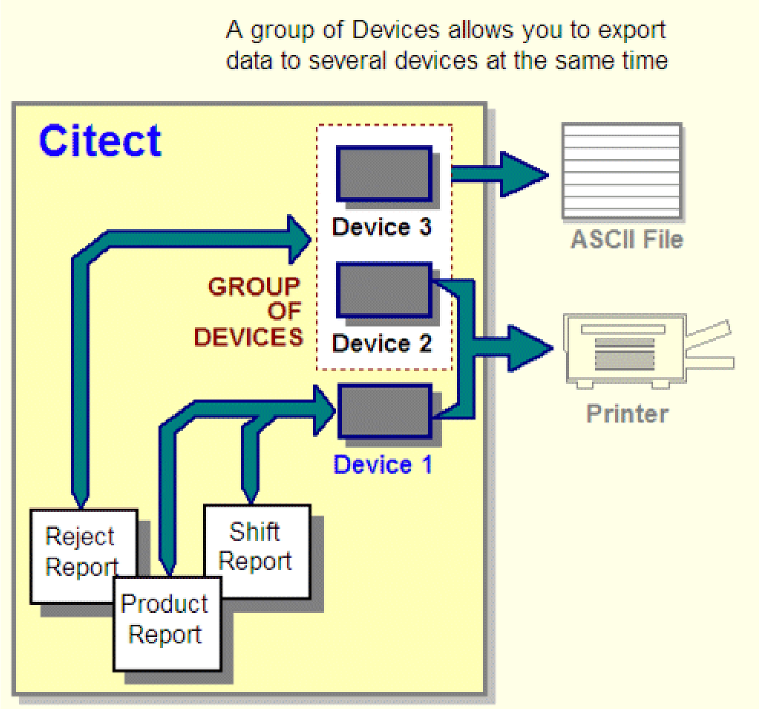
[Formatting Data in the Device]

[Using Device History Files]

[About Print Management]

Using groups of devices

You can add flexibility to your system by using a group of devices. A group of devices allows you to export the same data to two (or more) locations.



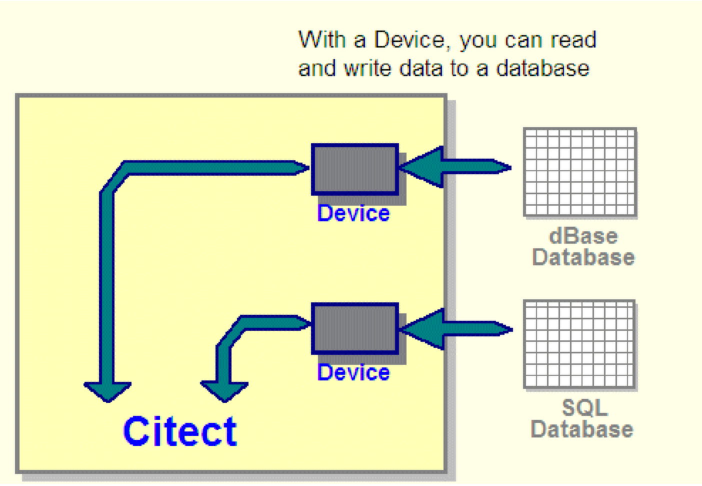
See Also

[Using devices to read data]

Using devices to read data

Using a device (and Cicode functions), you can also read data from:

- ASCII files
- dBASE databases
- SQL databases



Note: When you read from a group of devices, data is only read from the first device in the group.

See Also

Configuring Devices

Configuring Devices

You need to configure your devices before you can use them with your CitectSCADA system.

To configure a device:

1. Choose **System** | **Devices**. The Devices dialog box appears.
2. Complete the Devices dialog box using the description of the text boxes below.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

Devices have the following properties:

Name

The name of the device. The device name can be the name of a group of devices, or a label for a device. Enter a value of 16 characters or less.

See Also

Using groups of devices
Using Labels
"Predefined Devices" in the CitectSCADA Technical Reference

Format

Specifies how the data is formatted in the device. The format is determined by the type of Device, and the data that is sent to the device. Enter a value of 120 characters or less. See Using Command Fields for information on available fields.

If you are logging alarms or command messages, you need to specify a format, or no data is written to the device.

Note: The log device for a command is specified wherever the command is defined. The log device for an alarm is specified at the Alarm Categories form.

When producing reports, the format is ignored. (The format defined for the report is used to write the report to the device.)

See Also

Formatting Data in the Device
Alarm display fields
Alarm summary fields
Using Command Fields

Header

Additional information for the device. Enter a value of 120 characters or less.

Printer devices

The header is printed on each page. A new page is created each time the form length is reached. The [Device]FormLength parameter is used to set the form length.

ASCII file devices

Not applicable.

dBASE database devices

Contains the field name used to index the database, for example:

Header	{Name}

Note: Index Key fields needs to not exceed 100 characters.

SQL database devices

The connection string for the particular database type.

Note: CitectSCADA database devices only support STRING data types. If you use another database editor to modify your database, you will want to verify that fields are in string format.

File Name

The file name of the device. Enter a value of 253 characters or less.

Printer devices

The printer port or UNC name, for example:

File Name	LPT1:
File Name	COM2:
File Name	\\PrintServer\BubbleJet1

When you specify a printer port, you need to include the colon character (:), otherwise CitectSCADA tries to write to a file (device) with a name similar to the printer port (i.e. LPT1 or COM2).

Note: When using a UNC name in Windows 95, the printer needs to be in the Printers section of the Control Panel.

ASCII file devices and dBASE database devices

The name of the active file, for example:

File Name	ALARMLOG.TXT
File Name	[DATA]:ALARMLOG.TXT

This property is optional. If you do not specify a file name, **File Name** defaults to `\CitectSCADA 7.10\bin\<Name>` on the hard disk where you installed CitectSCADA. <Name> is the first eight characters of the device name. If you use this property, verify that no other devices have the same first eight characters in the device name.

SQL database devices

The database table, for example:

File Name	LOGFILE
File Name	REPTBL

Type

The type of device. Enter a value of 16 characters or less.

Device Type	**Device Description**
ASCII_DEV	ASCII file*****

PRINTER_DEV	Printer
DBASE_DEV	dBASE file
SQL_DEV	SQL database

* When defining RTF report properties, an ASCII device would be selected if the report was to be saved as a file.

This property is optional. If you do not specify a type, the device **Type** is ASCII_DEV unless:

The file name is a printer device (LPT1: to LPT4: or COM1: to COM4: or a UNC name), where **Type** is PRINTER_DEV, or

The file name extension is .DBF, where **Type** is dBASE_DEV.

See Also

About Print Management

No. Files

The number of history files. Enter a value of 4 characters or less.

By default, CitectSCADA creates a single data file for each device. (This data file is called <filename.TXT> or <filename.DBF>, depending whether the device is an ASCII device or database device.) The number of history files you specify here are in addition to the data file.

Note: If you do not want history files created, you need to enter 0 (zero) here, and set the [Device]CreateHistoryFiles parameter to 0; otherwise, 10 history files will be created as a default. You need to also verify that the data file is of a fixed size. (If the data accumulates, the file eventually fills the hard disk.)

If you specify -1 the data is appended to the end of one file.

If you are logging alarm, keyboard commands, or reports to the device, specify the number of files to be created, and the time of each file.

See Also

Using Device History Files

Time

The time of day to synchronize the beginning of the history file, in hh:mm:ss (hours:minutes:seconds). Enter a value of 32 characters or less. If you accepted the default number of history files above, and you specify a time and period, 10 history files will be created. If you do not specify a time, the file is synchronized at 0:00:00 (i.e. midnight).

If you omit both the time and the period, additional history files will still be created (with the default time and period). If you don't want history files to be created, you need to set the [Device]CreateHistoryFiles parameter to 0 (zero).

Period

The period of the history file, in hh:mm:ss (hours:minutes:seconds). Enter a value of 32 characters or less. Alternatively you can:

- Specify a weekly period by entering the day of the week on which to start the history file, for example Monday, Tuesday, Wednesday, etc.
- Specify a monthly period by entering the day of the month on which to start the history file, for example 1st, 2nd, 3rd, 4th, 5th, etc.
- Specify a yearly period by entering the day and month on which to start the history file, for example 1st January, 25th February, etc. The day and month needs to be separated by a space.

If you accepted the default number of history files above, and you specify a time and period, 10 history files will be created.

If you do not specify a period, the period defaults to Sunday (weekly).

If you omit both the time and the period, additional history files will still be created (with the default time and period). If you don't want history files to be created, you need to set the [Device]CreateHistoryFiles parameter to 0 (zero).

Cluster Name

Select a cluster from the list of clusters defined previously with the Servers, Clusters command in the Project Editor. See "Cluster Definitions ". If this is a single cluster system this field can be left blank.

Process

Select the type of server (or client) on which the process runs that sends data to the device. This field is used to prevent a history file being created while the device is active. If there is no history processing then this field can be left blank

Comment

Any useful comment. Enter a value of 48 characters or less.

Formatting Data in the Device

The device format specifies how to format the data in the device. The format is determined by the type of device, and the data that is sent to the device.

- Printer and ASCII devices format
- dBASE and SQL database devices format

See Also

Using a database device

Printer and ASCII devices format

The format specifies how each line of data is printed on the printer or written to the ASCII file, for example:

```
RFP3   Raw Feed pump 3    Overload    12:32:21
RFP9   Secondary Feed     Overtemp    13:02:45
```

When producing reports, the device format is ignored. The format defined for the report (i.e. the report format file) is used to write the report to the device.

To include CitectSCADA data you need to specify the field name and (optionally a width for each field to be printed or written to the file. The format has the following syntax:

```
{<field name>, [width[, justification]]}
```

You need to enclose each field in braces {}, for example:

Format	{Tag,8}{Name,32}

In this case, two fields are printed or written to the file - Tag, with 8 characters, and Name, with 32 characters. The width specifier is optional - if you do not specify a width, the width of the field is determined by the number of characters between the braces, for example:

Format	{Name }

In this case, Name is followed by four spaces - the field is printed or written to the file with 8 characters.

Creating lists and tables

To set the justification of the text in each field, use a justification specifier. You can use three justification characters, L (Left), R (Right), and N (None) - for example:

Format	{Tag,8,L} {Name,32,R}

The justification specifier is optional - if it is omitted, the field is left justified. If you use a justification specifier, you need to also use the width specifier.

To display field text in columns, use the tab character (^t) - for example:

Format	{Tag,8}^t{Name,32}^t{Desc,32} {Time,8,R}

Including fixed text

You can include fixed text by specifying the text exactly as it is to be printed or written to the file - for example:

Format	Name of Alarm:

Any spaces that you use in a text string are also included in the string.

See Also

Formatting Data in the Device
Using a database device

dBASE and SQL database devices format

The format specifies the structure (field names and field widths) of the database. The format has the following syntax:

```
{<field name>, <width>}
```

You need to use braces ({ }) to enclose each field, for example:

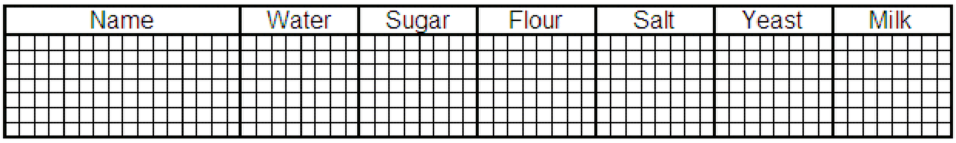
Format	{Tag,8}{Name,32}

In this case, the database is created with two database fields - Tag, with 8 characters, and Name, with 32 characters. The size of each database field is determined by the width you specify in the format. (Justification character are ignored.) Every database field is character (string) field types.

You can define your own fields (as well as the standard CitectSCADA fields) for the database device, for example:

Format	{Name,16}{Water,8}{Sugar,8}{Flour,8}
	{Salt,8}{Yeast,8}{Milk,8}

This database device has the following structure:



Don't leave any spaces between each field definition or at the end of the format string, or CitectSCADA creates extra fields for each space (in the device). Do not specify a field name longer than 10 characters, or CitectSCADA truncates the name to 10 characters. (The dBASE file format limits field names to a maximum of 10 characters.)

In this example, the database is created with seven database fields. To access the above dBASE device, use a Cicode function similar to the following:

```
hDev = DevOpen("Recipe");
DevFind(hDev, "Name", "Bread");
PLC_Water = DevGetField(hDev, "Water");
PLC_Sugar = DevGetField(hDev, "Sugar");
. . .
. . .
DevClose(hDev);
```

dBASE devices

If the database does not exist, it is created with the specified format. If you do not specify a format, and if the file name specifies an existing dBASE file, CitectSCADA uses the existing fields in the database.

SQL devices

You need to create the SQL database by an external application before it can be used.

If you edit a dBASE or SQL device record (in an existing project), the associated physical device is not edited. For example, if the device is a dBASE type device and you add an extra field in the device, the extra field is not added to existing database files (when you run CitectSCADA).

If you want to make changes to the file structure of an existing database and retain existing data, you need to manually edit the data using dBASE, Excel or some other database tool.

If you want to make changes to the file structure of an existing database and don’t want keep the existing data:

1. Close the device.
2. Manually delete the device file.
3. Make your modifications.
4. Open the device.
   When the device is opened no device file is available and a new data file is created with the necessary changes.

See Also

Formatting Data in the Device

Using a database device

Using a database device

Before you can use a database device, you need to open it. You can open several devices at the same time. The `DevOpen()` function returns an integer handle to identify each device, as in the following example:

```
INT hRecipe;
hRecipe = DevOpen("Recipe");
```

Writing dBASE records using a database device

To write data to a database device, first append a record to the end of the device, then add data to the fields of the record. For a dBASE database, the `DevAppend()` function appends the record, and the `DevSetField()` function writes data to a field. The following function writes a recipe record to a device:

```
FUNCTION
WriteRecipeData(INT hDevice, STRING sName, INT Water, INT Sugar,
       INT Flour, INT Salt, INT Yeast, INT Milk)
        DevAppend(hDevice);
        DevSetField(hDevice, "NAME", sName);
        DevSetField(hDevice, "WATER", IntToStr(Water));
        DevSetField(hDevice, "SUGAR", IntToStr(Sugar));
        DevSetField(hDevice, "FLOUR", IntToStr(Flour));
        DevSetField(hDevice, "SALT", IntToStr(Salt));
        DevSetField(hDevice, "YEAST", IntToStr(Yeast));
        DevSetField(hDevice, "MILK", IntToStr(Milk));
END
```

Writing SQL records using a database device

To use an SQL device in CitectSCADA, you cannot use every the Cicode Device function; you can only use the following functions:

`DevOpen()`	`DevClose()`	`DevGetField()`	`DevFind()`	`DevWrite()`
`DevNext()`	`DevSeek()`	`DevAppend()`	`DevWrite()`	`DevZap()`
`DevControl()`	`DevSetField()`			

To write CitectSCADA data to an SQL database, use the DevWrite() function, but you need to add a new record and write to fields in the new record. No data is written to the database if you do not write to all fields.

For example:

```
FUNCTION
WriteRecipeData(INT hDevice, STRING sName, INT Water, INT Sugar,
 INT Flour, INT Salt, INT Yeast, INT Milk)
        DevWrite(hDevice, sName);
        DevWrite(hDevice, Water);
        DevWrite(hDevice, Sugar);
        DevWrite(hDevice, Flour);
        DevWrite(hDevice, Salt);
        DevWrite(hDevice, Yeast);
        DevWrite(hDevice, Milk);
END
```

The following functions return error 267 (File mode is invalid) when the device type is SQL:

```
DevFlush()
DevPrev()
DevDelete()
```

The follow function returns error 263 (Cannot read file) when the device type is SQL:

```
DevRead()
```

The following functions return error -1 when the device type is SQL:

```
DevSize()
DevRecNo()
```

Locating and reading database records using a database device

To read data from a dBASE or SQL database device, use the `DevFind()` function to locate the record, and then the `DevGetField()` function to read each field:

```
FUNCTION
GetRecipe(STRING sName)
        INT hDev;
        hDev = DevOpen("Recipe");
        IF hDev >= 0 THEN
```

```
                  IF DevFind(hDev, sName, "NAME") = 0 THEN
                          PLC_Water = DevGetField(hDev, "WATER");
                          PLC_Sugar = DevGetField(hDev, "SUGAR");
                          PLC_Flour = DevGetField(hDev, "FLOUR");
                          PLC_Salt = DevGetField(hDev, "SALT");
                          PLC_Yeast = DevGetField(hDev, "YEAST");
                          PLC_Milk = DevGetField(hDev, "MILK");
                  ELSE
                          DspError("Cannot Find Recipe " + sName);
                  END
                  DevClose(hDev);
          ELSE
                  DspError("Cannot open recipe database");
          END
END
```

Deleting records using a database device

You can delete dBASE records with the `DevDelete()` function. The following Cicode function deletes records from a dBASE device:

```
FUNCTION DeleteRecords(INT hDev)
        WHILE NOT DevEOF(hDev) DO
                DevDelete(hDev);
                DevNext(hDev);
        END
END
```

To delete every record from a dBASE database, use the `DevZap()` function:

```
FUNCTION DeleteRecords(INT hDev)
        DevZap(hDev);
END
```

To delete records from an SQL database, use the Cicode SQL functions.

Closing a database device

When finished with a device, close it to free Cicode system resources by using the `DevClose()` function:

```
DevClose(hRecipe);
```

To define a group of devices:

1. Choose **System** | **Groups**. The Groups dialog box appears.
2. Complete the Groups form.

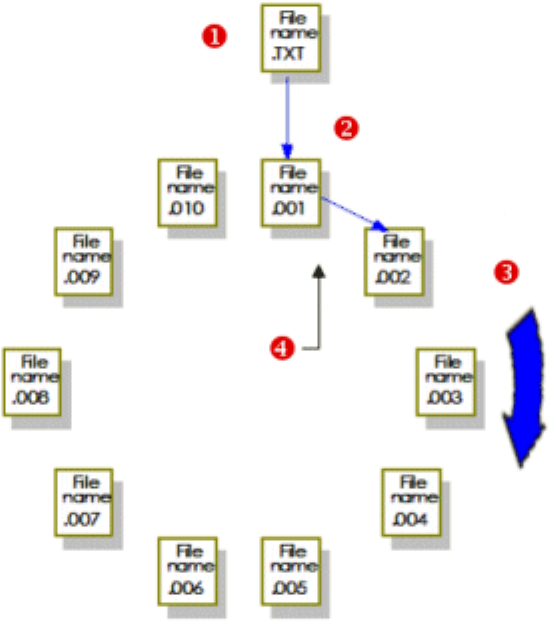
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

See Also

Using Device History Files

Using Device History Files

To makes the long-term storage of logged data easier to organize and more accessible, CitectSCADA uses a system of rotational history files to store historical data. To use this system, you need to specify how many device history files you want to keep. For example, if you want to keep 10 history files, they would be saved rotationally as illustrated below:



1. When CitectSCADA begins logging, data is written to a file called <filename>.txt or <filename>.dbf, depending on the type of device.
2. At midnight the following Sunday, the file <filename>.txt is renamed to <filename>.001 and a new <filename>.txt is created.
3. At midnight the following Sunday, the file <filename>.001 is renamed to <filename>.002, <filename>.txt is renamed to <filename>.001, and then a new <filename>.txt is created.
4. After week 10, the first file is overwritten (week 11 in the first cycle).

Note: The 10 history files are in addition to the default data file that is saved for each device.

By default, CitectSCADA uses 10 files (if history files are specified). You can change the default by specifying the number of files to use, for example:

No. Files	20
Comment	CitectSCADA uses twenty files for the data

The maximum number of files you can specify is 999.

You can also specify the period between files, i.e., when a new history file is used, for example:

Period	Comment
1:00:00	Use a new file each hour
6:00:00	Use a new file every six hours
72:00:00	Use a new file every three days
Monday	Use a new file each week beginning on Monday
15th	Use a new file every month beginning on the 15th of each month
25th June	Use a new file every year beginning on the 25th of June

Note: Marked improvement in system performance is observed when a period of one week or more is specified.

You can also specify the time of day to synchronize the beginning of the history file, for example:

Time	Comment
6:00:00	Synchronize the file at 6:00 am
12:00:00	Synchronize the file at 12:00 midday
18:30:00	Synchronize the file at 6:30 pm

The first file does not actually begin at this time: the first file begins when you start your runtime system. The time and period together determine when new history files are created, for example:

Time	Period
6:00:00	Monday

In the above example, CitectSCADA creates a new file each Monday at 6:00am. If you start your runtime system at 7:30am on Sunday, your first file only contains 22.5 hours of data. If you leave your system running, subsequent files start each Monday at 6:00am, and contain one full week of data.

Archiving data

To archive your data for long-term storage, back up your history files before they are overwritten. Use the Windows File Manager from the Main program group to check file creation dates (of the history files) and to back up files. Refer to your Windows documentation for a description of the File Manager.

See Also

Using Command Fields

Using Command Fields

You use the following fields (or combination) to format a command logging device:

Field Name	Description
{UserName,n}	The name of the user (User Name) who was logged on when the command was issued.
{FullName,n}	The full name of the user (Full Name) who was logged on when the command was issued.
{Time,n}	The time (in short format) when the command was issued (hh:mm).
{TimeLong,n}	The time (in long format) when the command was issued (hh:mm:ss).
{Date,n}	The date (in short format) when the command was issued (dd:mm:yy).
{DateLong,n}	The date (in long format) when the command was issued (day month year).
{DateExt,n}	The date (in extended format) when the command was issued (dd:mm:yyyy).
{Page,n}	The page that was displayed when the command was issued.
{MsgLog,n}	The message sent as the *Message Log* property (of the command record).

You can use the following fields (in the command field) for **Keyboard commands only**:	
{Arg1,n}	The first keyboard command argument (if any).
{Arg2,n}	The second keyboard command argument (if any).
. . .	
{Arg8,n}	The eighth keyboard command argument (if any).
{Native_MsgLog,n}	The native language version of the message sent as the *Message Log* property (of the command record).
Where n specifies the display field size.	

For example, you could have a device configured as follows:

Name	KeyLog
Format	{Date,9} {MsgLog,27} {Arg1,3} by {FullName,11}

Then a keyboard command (object, page, or system) could be created with the following configuration:

Log Device	KeyLog
Key Sequence	### ENTER
Log Message	Density setpoint changed to

Resulting in an output of the following kind: "01/01/99 Density setpoint changed to 123 by Timothy Lee".

About Print Management

The Windows printer management has been designed for page-based printers: laser printers and shared network printers. The printer driver does not print anything on the printer until the page is full; it then prints the page. This is the preferred printing method (when printers are shared on a network), because it minimizes the likelihood of conflict of data when more than one operator uses the print facility at the same time.

However, this method is inappropriate when logging alarms or keyboard commands. If you send alarm logging to this type of printer, CitectSCADA flushes the data to the printer when the current activity completes, or when the `[DEVICE]FlushTime` parameter has been exceeded (it defaults to 10 seconds). If, for example, you have one alarm occurring each minute, each alarm is printed on a new page (because the default flush time is less than the alarm frequency).

You can bypass the Windows print management by writing the output to a file. Set the device type to ASCII_DEV and specify the file name as `lpt1.dos`, `lpt2.dos` or `lpt3.dos` (depending on the port to which your printer is connected). The printer needs to be either on a local port, or a captured network printer. When you log to this device, the data is printed immediately on the printer with no extra form feeds.

For correct logging operation, reserve one printer to be your logging printer. This printer has to be a local printer, not on the network server. You can then send any other non-logging printouts, (for example, reports) to a shared network or local printer.

See Also

Using Devices

Using Equipment

The Equipment database is the central repository for information about equipment controlled by your CitectSCADA system.

To configure equipment:

1. Choose **System** | **Equipment** to display the equipment dialog box .
2. Enter the equipment properties.
3. Click **Add** to append a new record, or **Replace** to modify an existing record.

You use the Equipment Properties dialog box to configure your equipment. Alternatively you can use the DBF plug-in for Microsoft® Excel to edit and save records in the EQUIP.DBF file directly.

See Also

Equipment Properties

Equipment Database Functions

Equipment Properties

Equipment has the following properties:

Name

The name of the equipment. Enter a value of 254 characters or less.

Cluster Name

The name of the cluster that this equipment runs on. If the Cluster Name is not set, then this equipment will run on defined clusters.

Type

The specific type of equipment in the system. Enter a value of 254 characters or less.

I/O Device

The I/O Device used to communicate with this piece of equipment. Specify redundant devices with a comma between primary and standby, for example, "IODev1,IODev2". Enter a value of 254 characters or less.

Area

The area number or label to which this equipment belongs. Only users with access to this area (and any necessary privileges) will be able to perform operations on the equipment. For example, if you enter Area 1 here, operators need to have access to Area 1 (plus any necessary privileges) to perform operations on the equipment. Enter a value of 16 characters or less.

Location

A string describing the location of the equipment. Enter a value of 254 characters or less.

Comment

Any useful comment. Enter a value of 254 characters or less.

Page

The name of the page on which this equipment appears. Enter a value of 254 characters or less.

Help

The help context string. Enter a value of 254 characters or less.

The following fields are implemented with extended forms (press **F2**).

Extended forms fields

Custom1 .. Custom8

User-defined strings that can be used for filtering equipment when using the Cicode search functions (maximum 254 characters each).

Chapter: 31 Exchanging Data with Other Applications

You can transfer data between CitectSCADA and other software for storage, analysis, and post processing, or to control and tune your CitectSCADA system.

CitectSCADA uses the following methods to exchange data:

- dynamic data exchange (DDE), where CitectSCADA can act as a:
  - **DDE server** providing tag values to requesting clients
  - **DDE client** to request data from other applications.
- open database connectivity (ODBC), where CitectSCADA functions as an ODBC server, allowing other applications to read CitectSCADA variables directly.
- By using a common **external database**, where CitectSCADA and other applications use the same database to store and share information.

CitectSCADA also supports importing and linking variable tag data from external databases. See Linking, Importing, and Exporting Tags.

Using DDE (Dynamic Data Exchange)

Microsoft Windows DDE allows the continuous and automatic exchange of data between different Windows applications on the same machine without the need for any user intervention. For example, your company's Production group might use a spreadsheet application to graphically represent plant-floor data (product output). This could be dynamically updated with the latest live data using DDE to read values directly from CitectSCADA.

Windows DDE uses the DDE protocol to send messages between applications that share data.

Dynamic Data Exchange occurs between a DDE client application (which requests the data or service) and a DDE server application (which provides the data or service). The DDE Client starts the exchange by establishing a conversation with the DDE server, and requesting data or services. The DDE server responds to these requests by providing the data or services to the DDE Client. The DDE Client terminates the conversation when it no longer needs the DDE server's data or services.

Note: As the DDE protocol is not designed for high-speed data transfer, the use of

DDE is only appropriate when data communication speed is not critical.

- For information about DDE conversations, see DDE conversations and client syntax.
- To establish a DDE conversation between applications on the same computer, see Setting up DDE conversations.
- To establish DDE conversations between applications running on different computers over the same network, see Network DDE.
- To establish DDE Conversations with the CitectSCADA tag database directly, see Connecting to the CitectSCADA tag database using DDE.

Note: When reading or writing to CitectSCADA tags using DDE, you might unknowingly add to your CitectSCADA license point count. Once the dynamic point count is greater than the license point count, the software protection mechanism will terminate CitectSCADA runtime. Therefore, when accessing tags via DDE, it's important to remain aware of how many points you have used. For details, see license point count in the Installation and Configuration Guide.

DDE conversations and client syntax

Two applications participating in Dynamic Data Exchange are said to be engaged in a **DDEconversation**. The application that initiates the conversation is the **DDE Client**, and the application that responds to the DDE Client is the DDE server.

An application can have several DDE conversationsrunning at the same time. The application can be the DDE Client in some conversations (requesting data or services), and the DDE server (the data/service provider) in others. Each request or response in a DDE conversationspecifies the data or service to be sent or received.

Note: A DDE **conversation** is sometimes referred to as a **channel** or a **link**.

The syntax sent by the DDE Client when it tries to establish a DDE conversation with the DDE server, consists of three parts:

- The name of the application to retrieve the data from.
- The file or topic name which contains the data to be retrieved.
- The cell range, value, field, or data item that's being requested.

These are combined in the format:

```
<DDE server application name>|<DDE Topic name>!<DDE Data item
```

```
name>
```

where:

- **<DDE server applicationname>** identifies the DDE server application.
- **|** (pipe character) separates the DDE server application name from the DDE Topic Name with no spaces between them.
- **<DDE Topic name>**identifies the context of the data. For DDE Servers that operate on file-based documents, DDE topic names are typically file names. For other DDE Servers, they are other DDE application-specific strings.
- **!** (exclamation character) separates the DDE Topic Name from the DDE Data item name with no spaces.
- **<DDE Data item name>**is a string that identifies the data item that a DDE server can pass to a DDE Client during a DDE transaction. In some instances, the DDE Data item name is optional. Refer to the DDE application documentation for particulars.

> **Note:** In the DDE Client syntax structure example above, every placeholder shown inside arrow brackets (<placeholder>) has to be replaced with the actual name of the item that it describes. Do not include the arrow brackets and the placeholder words they contain in the statement, and are shown here only for your information.

As the DDE protocol was designed in an era before long file names, DDE only supports the use of short (8 character) file names. To overcome this limitation, enclose the three parts of the DDE syntax within single quotes respectively. For example:

```
Citect|Variable!'Process Variable 1'
```

This instructs DDE to treat the characters within the quotes as strings, thus permitting them to contain long file names, the space character (), the pipe character (**|**), the exclamation or bang character (**!**), or any other non alphanumeric character.

See Also

Setting up DDE conversations

Setting up DDE conversations

The DDE protocol itself does not support the launch of applications, so both the DDE Client application and the DDE server application needs to already be running before any DDE conversations can occur (unless the calling application is coded to detect and launch the DDE server application when necessary).

At the beginning of a DDE conversation, a DDE Client requests the services of a DDE server using DDE Client syntax (which contains the DDE server application name, topic or file name, and the data item name in the request). For DDE Client syntax details, see DDE conversations and client syntax.

To set up an application as a DDE Client, that is, to request data from a DDE server application, you need to use appropriate values in the DDE Client syntax as follows:

DDE server application name

The DDE server name is usually the DDE server application name, for example the DDE server name for CitectSCADA is "**Citect**", the DDE server name for Microsoft Excel is "**Excel**", the DDE server name for Microsoft Word is "**WinWord**", and the DDE server name for Microsoft Access is "**MSAccess**". Most DDE Servers respond to only one name.

DDE Topic name

The DDE Topic name for CitectSCADA is either "**Data**" (if you use the Cicode DDE functions) or "**Variable**" (if you use CitectSCADA as the DDE server and want to access the variable tag database directly). The DDE Topic name for Microsoft Excel is the name of the worksheet (which may also include the workbook name enclosed in square brackets). The DDE Topic name for Microsoft Word is the document name. The DDE Topic name for Microsoft Access is the Database name and Table name, Query name or an SQL string as detailed in the following note:

Note: The proper DDE Client syntax of the DDE Topic name section for accessing a Microsoft Access database is constructed like this: "`<DataBaseName>; TABLE <TableName>`".

The <**DataBaseName>** placeholder is for the name of the Access database file followed by a semicolon (**;**). You might have to include the file path; however this might not be the case (i.e. if it is known that Access will be running with the target file open). The .MDB suffix is optional (as .MDB is the default suffix for Access databases), unless the full path was included.

The **TABLE <TableName>** is the command string to instruct Access which table data you intend to converse with. DDE also supports the use of **QUERY <QueryName>** or **SQL <SQLString>** in place of **TABLE <TableName>**.

The use of the semi-colon (**;**) after the '<DataBaseName>' placeholder, and the use of UPPERCASE for the 'TABLE', 'QUERY', and 'SQL' commands in the DDE string syntax are necessary. The whole section needs to be enclosed in quotes (").

DDE Data item name

The DDE Data item name for CitectSCADA depends upon the DDE Topic name being used. When using 'Variable' as the DDE Topic name to access the variable tag database directly, the DDE Data item name is the CitectSCADA variable tag name. When using 'Data' as the DDE Topic name to access a value posted using the Cicode DDEPost() function, the DDE Data item name is the posted name.

The DDE Data item name for Microsoft Excel is the cell range in Row number Column number format (for example R1C1). The DDE Data item name for Microsoft Word is a bookmark name. The DDE Data item name for Microsoft Access is dependant upon which topic name was used. Refer to the Microsoft Access Help for details.

> **Note:** These CitectSCADA DDE help topics and examples were originally written for Windows 3.xx and subsequently updated for Office 95 on Windows 95. Microsoft has since introduced security measures with Office 2000 and later versions which, by default, block Office applications from being DDE Clients. For security details, see Using DDE with Microsoft Office applications.

CitectSCADA can perform as both a DDE server and as a DDE client as necessary. To set up CitectSCADA to use DDE, see Exchanging CitectSCADA data via DDE.

CitectSCADA comes with an Excel workbook file which contains macros to help you insert correct DDE Client syntax links from your CitectSCADA runtime project tag database directly into an Excel worksheet.

DDE function types

There are two classes of DDE functions in Cicode, the original **DDE** functions and the later **DDEh** functions.

DDE functions

The original Cicode DDE functions do not return a DDE Channel Number and were designed to insulate the user from the need to manage DDE Channels. The DDERead(), DDEPost(), DDEWrite(), and DDEExec() functions each perform a single exchange of data. Each of these functions starts a DDE conversation with the external application, sends or receives the data (or command), and ends the conversation - in one operation.

DDEh functions

The Cicode DDEhfunctions were introduced to afford more control over DDE communications, especially for Network DDE and for circumstances where it is necessary to explicitly terminate and re-initiate a DDE Channel (after deleting rows from a table for example).

The DDE handle (*DDEh...*)functions return a handle to the conversation - a DDE channel number.

Use the DDEh handle functions for Network DDE, and for Access DDE.

See Also

Exchanging CitectSCADA data via DDE

Exchanging data via DDE

CitectSCADA runtime can exchange data as a DDE server or a DDE Client.

CitectSCADA behaves as a DDE server when providing other applications with access to its data. When acting as a DDE server, CitectSCADA runtime can:

- Provide DDE access to the complete variable tag database **automatically** with no further setup necessary
- Provide access to selected variable values by posting select CitectSCADA data using DDE

CitectSCADA behaves as a DDE client when requesting other applications to provide access to their data. When acting as a DDE Client, CitectSCADA runtime can:

- Read data directly from another application
- Write data directly to another application

> **Note:** You can also execute commands in another application from CitectSCADA with the DDEExec() function. Similarly, you can run Cicode functions from another application by passing the functions through that application's DDE Execute command.

See Also

Connecting to the CitectSCADA tag database using DDE

Connecting to the tag database using DDE

CitectSCADA runtime behaves as a DDE server and automatically provides DDE access to the complete variable tag database with no further setup necessary.

To create DDE links to the CitectSCADA variable tags, use the DDE Client syntax. For syntax details, see DDE conversations and client syntax.

In the DDE Client call, the DDE Application name needs to be "Citect", the DDE Topic name needs to be "Variable", and the DDE Data item name needs to be the CitectSCADA tag name.

For instance, the PV1 tag value can be accessed from a cell in Excel containing the following formula:

```
=Citect|Variable!PV1
```

If the CitectSCADA variable tag name contains spaces or non-alphanumeric characters, the DDE data item section of the DDE Client call syntax needs to be enclosed within single quotes. For example:

```
=Citect|Variable!'Process Variable 1'
```

CitectSCADA runtime and the DDE Client application (for example Excel) need to both be running on the same computer. For information about DDE conversations, see DDE conversations and client syntax.

To establish a DDE conversation between applications on the same computer, see Setting up DDE conversations.

To establish DDE conversations between applications running on different computers over the same network, see Network DDE.

Posting select data using DDE

You might have a tag naming convention which is not DDE compatible, or inappropriate for use in a DDE call. CitectSCADA provides the ability to publish specific tags under different names in DDE. This involves using the DDEpost function.

To make selected CitectSCADA variable values or the results of calculations available to external DDE Client applications currently running on the same computer, use the Cicode DDEPost() function to have CitectSCADA runtime behave as a DDE server.

This conversation is one-way, which allows an external DDE Client application (like Excel, Word, etc.) to read the value from CitectSCADA using DDE. The Client application cannot change this value in CitectSCADA.

You can use this function to present data under a different name than its source. For instance, the following example presents the value of variable tag PV1 as "Process1".

The following Cicode example posts the value of variable PV1 using DDE:

```
DDEPost("Process1", PV1)
```

Once posted, this value can be accessed, for example, from a cell in Excel containing the following formula:

```
=Citect|Data!Process1
```

or from a field in Microsoft Word containing the following function:

{DDEAuto Citect Data Process1 }

Notes:

- The name of the posted value (for example Process1) not to exceed 8 characters or contain spaces if you want to link to it via a Microsoft Word DDE field. Microsoft Excel, however, accepts long data item names if enclosed within single quotes (for example 'Process Variable 1').
- You need to use Data as the DDE Topic name when accessing posted values. See Setting up DDE conversations.

This method of DDE connection requires that both CitectSCADA runtime and the DDE Client application (for example Excel or Word) are running on the same computer at the same time. Once the DDE connection has been made, the DDE Client would normally display the updated value of the CitectSCADA DDE posting as soon as it becomes available, usually within milliseconds of the post.

Unfortunately, this posting method of DDE linking is subject to breakage whenever CitectSCADA runtime is closed, even if CitectSCADA runtime is subsequently restarted. The DDE Client application might not detect the lack of connection, as updates to the data in the DDE Client (for example Excel) only occur after each post by the DDE server (CitectSCADA runtime). If no further posts occur, and in this scenario none will (as the DDE link is disconnected), the DDE Client application receives no update, and subsequently might display data which could be out of date. This disconnected state will remain until the DDE link in the DDE Client application is refreshed or the DDE Client application is restarted.

See Also

Writing values to a DDE application

Writing values to a DDE application

To write a CitectSCADA variable value directly to an external DDE server application currently running on the same computer as CitectSCADA, use the Cicode DDEWrite() function.

For example, writing data from CitectSCADA to a Microsoft Office Application (using CitectSCADA as the DDE Client and the Office application as the DDE server), could be done using the following Cicode DDEWrite() function examples:

```
! Write PV1 to Excel
! Assumes the existence of Sheet1
DDEWrite("Excel", "Sheet1", "R1C1", PV1);

! Write PV1 to Word
! Assumes the existence of TestDDE.doc already loaded in Word
! containing the bookmark named TagPV1
DDEWrite("Word", "TestDDE", "TagPV1", PV1);
! MS Access does not support direct DDE writes.
```

This DDE function is one-way, so to update the tag value in the remote DDE server application, this function will need to be called every time the value of this CitectSCADA variable changes.

Writing directly to a DDE server assumes a prior knowledge of the DDE server. In the above example, the spreadsheet or document needs to exist and Excel or Word (as appropriate) needs to be running for the DDE communication to be successful.

Notes:

- Instead of using the once only DDEWrite(), you could use the multiple use DDE handle function DDEhPoke().
- Verify that remote requests are enabled in the other application. For example, in Excel, you need to verify the **Ignore other applications** check box is cleared (the default setting). In Excel 2003, use **Tools** | **Options** | **General** to adjust this setting.
- The High security setting (if selected) on Microsoft Office 2000 and later versions does not appear to affect the use of remote DDE Client requests with those Office applications. See Using DDE with Microsoft Office applications.

See Also

Reading values from a DDE application

Reading values from a DDE application

To read a value into a CitectSCADA variable directly from an external DDE server application currently running on the same computer as CitectSCADA, use the Cicode `DDERead()` function. For example:

```
PV1 = DDERead("Excel", "[Book1]Sheet1", R1C1);
```

The data from Row 1, Column 1 on Sheet 1 of Book 1 in Excel is read and stored in the CitectSCADA variable "PV1". This DDE function is one-way, and will need to be called whenever the value needs to be updated in CitectSCADA.

Reading from a DDE server assumes a prior knowledge of the DDE server. In the above example, Excel needs to be running and the appropriately named spreadsheet needs to exist. The CitectSCADA variable PV1 needs to also have been previously declared.

Note: Instead of using the once only `DDERead()`, you could use the multiple use DDE handle function `DDEhRequest()`.

Verify that remote requests are enabled in the other application. For example, in Excel, you need to confirm that the **Ignore other applications** check box is cleared (the default setting). In Excel 2003, use **Tools** | **Options** | **General** to adjust this setting.

The High security setting (if selected) on Microsoft Office 2000 and later versions does not appear to affect the use of remote DDE Client requests with those Office applications. See Using DDE with Microsoft Office applications.

Using DDE with Microsoft Office applications

Microsoft has introduced security measures with Office 2000 and later versions which, by default, block Office applications from being DDE Clients. See Microsoft Office security.

Microsoft Office applications appear to support varying degrees of long file names with DDE. See Long file names in DDE.

To enable DDE remote requests in Microsoft Excel, you need to verify the **Ignore other applications** check box is cleared (the default setting). In Excel 2003, use **Tools** | **Options** | **General** to adjust this setting.

Long file names in DDE

According to MSDN Knowledge Base article Q109397, DDE does not support long file names, so DOS alias names needs to be used for directory and file names longer than eight characters (i.e. C:\mydocu~1\file.mdb).

Different Microsoft Office applications differ in their support for the use of long file names when used as DDE Clients. For instance, Microsoft Word does not appear to support the use of DDE data item names which exceed 8 characters, while Microsoft Excel however, accepts long data item names if enclosed within single quotes. See Posting select CitectSCADA data using DDE for an example.

Microsoft Office security

In the interests of data security, Microsoft Office 2000 and later versions have their security settings set to high by default. To view or change your security level in Excel or Word, choose **Tools | Macro**, click **Security**, and click the **Security Level** tab.

- If you have your security level set to **High** (the default setting), then communication with external DDE Servers will not be available unless they are digitally signed and trusted. What you see in Excel cells that use the DDE function is #N/A , and with no additional explanation as to why the DDE functions aren't working. The High security setting (if selected) does not appear to affect the use of remote DDE Client requests with those Office applications as DDE Servers.
- If you set your security level to **Medium**, you are asked if you want to run any DDE Servers that are not digitally signed and trusted and that are referenced by DDE functions.

- If you set your security level to **Low**, external DDE Servers are run regardless of whether they are digitally signed and trusted, or not.

Note: If you need to manipulate another application's objects from Microsoft Office, consider using OLE Automation.

Network DDE

Network DDE is a version of the DDE protocol for use across a network. For information about DDE, see Using DDE (Dynamic Data Exchange). For details about setting up CitectSCADA to use DDE, see Exchanging CitectSCADA data via DDE.

Just like the establishment of a DDE conversation between applications running on the same computer, a network DDE conversation uses the same DDE protocol to share data between applications running on separate computers connected to a common network.

Network DDE is a Windows Service used to initiate and maintain the network connections, security, and shared file space needed for using DDE over a network, and needs to be running on both computers at the same time. For startup details, see Starting network DDE services.

A network DDE trusted share needs to be manually created for the Network DDE server application on the Network DDE server application machine. This instructs the Network DDE Service (on the DDE server application machine), as to which application and topic to connect with. It is this share name which the Network DDE Client application can subsequently communicate with. For details, see Setting up network DDE shares.

The Network DDE Client starts the Network DDE conversation by connecting to the Network DDE Share on the Network DDE server computer. For details, see Using network DDE.

Starting network DDE services

For Network DDE to function, NetDDE.EXE needs to be installed and running on both machines before attempting to conduct a Network DDE conversation. NetDDE.exe is a Windows Service system file that is used to communicate the shared dynamic data exchange used by Network DDE. It has no graphical user interface (it runs as a background Windows service).

It is necessary to initiate the automatic activation of Network DDE Services, or manually run NetDDE.EXE on both machines before attempting connection.

To manually start Network DDE services:

- On the Windows Start menu, click **Start** | **Run**, type in "netdde" (without the quotes) and press the **Enter** key. Do so on both machines.

To automatically start the Network DDE Services on machine startup:

- With Windows 2000 and later, use the Windows Services Manager (select **Start** | **Control Panel** | **Administrative Tools** | **Services**) to set the **Network DDE** servicefrom **Manual** to **Automatic**. To do so, right-click the service and select **Properties** from the pop-up menu. On the **General** tab select **Automatic** from the drop-down list of the **Startup type** field. Click **OK**. Close evry window and restart the machine.

To verify that the NetDDE Services are running:

- The Windows Task Manager lists NetDDE.exe on the **Processes** tab when running. To view the Windows Task Manager, press **CTRL+ALT+DEL.**
- The Service Administrative Tools also lists the status of Network DDE and Network DDE DSDM. To view the Windows Services Manager, select **Start** | **Settings** | **Control Panel** | **Administrative Tools** | **Services**.

> **Note:** If you are using only the Microsoft Client Service for NetWare Networks, the NW IPX/SPX/NetBIOS compatible protocol needs to be enabled for NetDDE.exe to load.

To test that Network DDE is operational between two machines on the same network

Microsoft Windows ships with a network DDE application called Chat. It is installed in the `system32` folder.

1. On the Windows Start menu, click **Start** | **Run**, type in "winchat" (without the quotes) and press the **Enter** key. Do so on both machines.
2. On one machine, select the Chat menu **Conversation** | **Dial** or click the dial button. The **Select Computer** dialog will display.
3. Select the other computer from the list, and click **OK**. Chat will attempt to establish a network DDE conversation between the computers.

> **Note:** If Chat is not already running on the other computer, it times-out and states that the other computer didn't answer. If however, the other computer is already running Chat, it will keep dialling until answered.

4. On the other machine, (while it is being dialed), select the Chat menu **Conversation** | **Answer** or click the answer button. Type in a message and it will display in the Chat window on the other machine. The conversation will continue until either machine hangs up.

Once a Chat conversation is established, it proves that both machines are properly set-up and capable of handling network DDE conversations. You can view the share properties for Chat$ by using the DDEShares.exe application as described in Setting up network DDE shares.

You don't have to run Chat to use Network DDE with CitectSCADA. This Network DDE test only uses Chat as an example to confirm Network DDE functionality between two machines. Once you have established that Network DDE is functional, close the Chat windows, create the **Network DDE Trusted Share** for your Network DDE server application, and connect to the share using Network DDE. See Connecting to a network DDE shared application.

Setting up network DDE shares

To be able to create a DDE link over a network, the computer serving as the Network DDE server needs to be setup to provide a **NetworkDDE Share** to establish a network DDE Channel.

> **Note:** You only have to create a DDE Share if you intend to use DDE between two separate applications running on different machines, and you only have to create the DDE Share on the machine that contains the application which will be the DDE server.

The Windows DDESHARE.EXE utility enables users to manage DDE shares. The 32-bit version is shipped with each Microsoft operating system and is located in the Windows\System32 sub-directory.

To manually launch the DDE Share utility:

- On the Windows Start menu, click **Start** | **Run**, type in "ddeshare" (without the quotes) and press the **Enter** key.

When DDEShare.EXE is running, it displays the DDE Share utility window containing two icons which launch the DDE Shares dialog, and the DDE Trusted Shares dialog:

In the DDE Share utility, double-click the left icon (without the check mark) to display the DDE Shares dialog box:

The DDE Shares dialog is used to create, manage, and delete global DDE shares on your computer, and to view the DDE shares of any computer on the network. You can use this dialog to confirm the names of shares available on any machine on the same network. From the DDE Shares menu, select **Shares** | **Select Computer** and choose the computer name you're interested in from the list.

DDE Shares

There are three types of DDE shares: old style, new style, and static. CitectSCADA only supports the static type. The names of static shares follow the convention

```
<ShareName>$
```

so to set up a CitectSCADA server computer as a Network DDE share, use the name "Citect$" as the sharename on that computer. To expose the CitectSCADA runtime variable tag database for suitable DDE linking, use the word "Variable" as the DDE Share Topic name.

> **Note:** The trailing dollar sign ($) is necessary as part of the DDE share name syntax.

To create a DDE share:

1. In the DDE Shares dialog, click **Add a Share**. The **DDE Shares Properties** dialog appears. Verify that the fields are blank, or unchecked, except for the **Grant access to every item** radio button which needs to be checked.
2. Click **Permissions**. The DDE Share Name Permissions dialog appears.
3. **Read and Link** is the default permission setting. If you want to write data to the DDE Share application, change the permission to **Full Control**.
4. Click **OK**.
5. Click **OK** to save the Share, and return to the **DDE Shares** dialog.

See Also

Using DDE Trusted Shares

Using DDE Trusted Shares

When a network DDE Client user connects to a network DDE Share from a remote computer, Network DDE accepts the request only if both:

- The user who created the share has granted trusted status to the share.
- The user who created the share is currently logged on to the server computer.

To link to the CitectSCADA tag database, and permit write actions from an external application using Network DDE, the DDE Client computer needs to be granted appropriate Trusted status.

To create a trusted share:

1. On the DDE Shares dialog, highlight the new 'Citect$' share entry, and click **Trust Share to display the** Trusted Share Properties dialog box.
2. Check **Initiate to Application Enable** to allow new connections to the DDE share.
3. Click **OK**.

To view the trusted shares:

1. In the DDESHARE utility double-click the right icon (with the check mark) to display the DDE Trusted Shares dialog.
2. The DDE Trusted Shares dialog lists the DDE shares that are trusted in the current user's context. You can view and modify trusted share properties and remove DDE shares from the list of trusted shares.
3. Once setup is completed, close the DDE Share utility dialog box.

Using network DDE

Microsoft Network DDE Service needs to be running on both computers to communicate using Network DDE. For startup details, see Starting network DDE services.

Before a Network DDE Client can establish a DDE conversation with a Network DDE server application, the Network DDE server application computer needs to already have setup a **Network DDE Share**. For details, see Setting up network DDE shares.

> **Note:** You cannot connect using Network DDE to a shared application on the same machine. You can only connect using Network DDE to a shared application on another machine (which needs to also be on the same network).

To connect to a Network DDE shared application, you use an altered version of the DDE syntax, which replaces the "<ApplicationName>" with "<ComputerName>\NDDE$" and replaces the "<TopicName>" with the Network DDE server Share "<ShareName>", and continues to use the "<DataItemName>" as normal.

At first glance, there appears to be no way to specify the DDE Application or Topic names in the Network DDE syntax call, and indeed, that is the case. However, the DDE Application and Topic names are defined in the DDE server Share settings. So, when the Network DDE server machine receives the call (from the Network DDE Client) containing the Share name, it knows which application and topic to connect with. See Connecting to a network DDE shared application.

Connecting to a network DDE shared application

The network DDE Client specifies the remote DDE server share in the normal DDE Client syntax by replacing the DDE Application name and DDE Topic name with the DDE server computer name and DDE server share name in the call. For DDE client syntax details, see DDE conversations and client syntax.

With Network DDE Client syntax, the DDE Application name is replaced with the following string enclosed in single quotes:

```
'\\<ComputerName>\NDDE$'
```

where "<ComputerName>" is the name of the computer running the DDE server application, and "NDDE$" notifies Windows on the remote computer that the calling DDE Client wishes to establish a Network DDE channel. You cannot omit the NDDE$ string, or it won't work.

The DDE Topic name is replaced with the following string also enclosed in single quotes:

```
'<ShareName>'
```

where "<ShareName>" is the name of the DDE Trusted Share previously set-up on the DDE server computer. The DDE Share on the DDE server machine contains the details of which application and topic to create the Network DDE link with. Most often, DDE server share names end with a $ character.

> **Note:** You need to use a separate DDE share name on the remote computer for each combination of DDE application name and DDE topic name you want to share. You can not declare the topic as a wild card (*).

For example, to create a Network DDE link with the following criteria:

- CitectSCADA variable tag name: "PV1"
- CitectSCADA server computer name: "PlantSvr"
- Remote DDE Share name: "Citect$"

you would construct a Network DDE Client call containing:

```
'\\PlantSvr\NDDE$'|'Citect$'!PV1
```

In Excel, the following formula could be placed directly into a worksheet cell:

```
='\\PlantSvr\NDDE$'|'Citect$'!PV1
```

If prompted for a username and password, use one that has appropriate permissions on the DDE server computer.

> **Note:** You cannot omit the DDE syntax pipe character (|) or exclamation character (!), nor can you enclose those characters within quotes (').

CitectSCADA comes with an Excel workbook file which contains macros to help you insert correct DDE Client syntax links from your CitectSCADA runtime project tag database directly into an Excel worksheet.

Using the Citect Tags Excel macros

CitectSCADA provides the Citect Tags Excel macros, which permit you to display the value of CitectSCADA variables directly in an Excel worksheet cell (so that you do not need to use Cicode DDE functions). The macros are contained in a workbook named `ddeformu.xls` (in the `\CitectSCADA 7.10\bin` directory), which was updated to support Network DDE with product version 5.41 and later.

> **Note:** Microsoft Excel version 8 which shipped with Microsoft Office 97 provides macro virus protection to minimize the likelihood of potentially malicious macros from running. To enable macros, and be able to use the features provided with `ddeformu.xls`, in Excel 8, from the main menu, choose **Tools** | **Options** and on the General tab clear the **Macro Virus Protection** option.

Microsoft Excel version 9, which shipped with Microsoft Office 2000, provides security levels which silently disables macros by default. To enable macros, and be able to use the features provided with `ddeformu.xls`, in Excel 9 or later, from the main menu, select **Tools** | **Macro** | **Security** and select **Medium** or **Low**.

If your Excel security settings are enabled, when you attempt to open the `ddeformu.xls`, Excel informs you that the file contains macros. To enable the Citect Tags features, select **Enable Macros**.

When started, the Citect Tags macros expect that CitectSCADA runtime is operating on the same machine. If not, Excel displays a dialog requesting permission to start `citect.exe` (which may not succeed as `citect.exe` does not exist unless you're running Citect version 3 or earlier). If you further select **Yes**, it will search the system 'path' for the non-existent program and subsequently report that the application could not be found. If you select **No**, and if previous values were saved with the worksheet, those are the values that will display initially, and be replaced with '#REF!' when updated. In any case, no valid values will be displayed in the example worksheets until CitectSCADA runtime is started and Excel is subsequently refreshed or restarted.

The Citect Tags macros expect the `citect.ini` to exist in the `config` folder of the `Citect-SCADAv7.20` User and Data folder selected during installation. If the file is not found in the location, it will not search elsewhere and will instead display the 'ERROR Reading Citect.INI' dialog requesting the proper location. Enter the full path including the file name, and clear **Restore Defaults on Start Up** to prevent the same thing happening next time the macro is started. If you are using an alternative INI file, enter it instead.

Once running, the right-click menu in Excel contains four additional menu items, permitting you to perform two new workbook related commands, and two new Citect-SCADA-related commands to the cell beneath the mouse pointer location when you perform the right-click event. The new menu items provided with `ddeformu.xls` are:

- **Citect Settings** - Workbook command
- **Citect Get Tags** - Workbook command
- **Citect Select Tags** - CitectSCADA command
- **Citect Select Trends** - CitectSCADA command

Note: This feature is only compatible with Excel version 5.00 (or later).

Using External Databases

You can store and update runtime data from your plant floor in a database external to CitectSCADA. You can also use CitectSCADA to send information from the database (such as a recipe) to I/O Devices in your plant.

By using an external application to read and write the same database records, you can effectively interface to an external application through, for example, a relational database.

CitectSCADA supports two types of databases:

- dBASE databases
- SQL databases

dBASE databases

The dBASE file format has become an industry standard for data storage, and Citect-SCADA supports the standard dBASE format. You can use any database editor (that supports dBASE III files) to create a database, and to read and write database records.

To connect to a database, you need to define a CitectSCADA Device. Devices specify the format and location of the database, for example:

Name	Recipe
Format	{Name,16}{Water,8}{Sugar,8}{Flour,8} {Salt,8}{Yeast,8}{Milk,8}
Header	Name
File Name	[DATA]:RECIPE.DBF
Type	DBASE_DEV
Comment	Recipe Device (dBASE file)

SQL databases

SQL (Structured Query Language) also has become an industry standard. SQL is a command language that allows you to define, manipulate, and control data in SQL databases - and in other relational databases. Most database servers support SQL. SQL gives you direct access to database servers on other platforms, such as computers, mini-computers, and mainframe computers.

You can use the CitectSCADA Device functions to set up the format and locations of each of your SQL databases. Alternatively, use the SQL functions for direct control over SQL transactions.

You need to define a CitectSCADA Device to specify the format and location of the database, for example:

Name	Recipe

In the Format field, specify the field names in the SQL database, for example:

Format	{Name,16}{Water,8}{Sugar,8}{Flour,8} {Salt,8}{Yeast,8}{Milk,8}

The Header is the database connection string for ODBC connection, for example:

Header	DSN = ORACLEDATABASE

Enter the database table name in the File Name field:

File Name	RECIPE
Type	SQL_DEV
Comment	Recipe Device (SQL)

See Also

Using Structured Query Language

Using Structured Query Language

You can use Structured Query Language (SQL) functions for direct access to an SQL database, instead of accessing the database as a Device. Using direct database access can provide greater flexibility. The SQL functions provide access to SQL databases through any ODBC-compatible database driver, for example MS Access, FoxPro, Paradox, etc.

See Also

Connecting to an SQL database
Executing SQL commands
Using a transaction
Expressing dates and times in SQL

Limitations:

The SQL library used inCitectSCADA has a number of limitation and these are documented in our Knowledge Base (article Q4014). This library was not designed to handle large scale SQL data transactions. For large volumes of data it is recommended that CitectSCADA Reports (CitectHistorian) be used.

If you intend to use the embedded SQL library in CitectSCADA, consider the following:

1. Use "Simple" model for a SQL database to limit transaction information logging to transaction log file.
2. Use the fixed size of transaction log to restrict the log file growing.
3. Back up the database regularly to keep the transaction log size under control. When SQL Server finishes backing up a database or its transaction log, it automatically truncates the inactive portion of the transaction log. If the transaction log file is full, your database transactions will cease.
4. Keep working tables as small as possible. When CitectSCADA SQL adds or appends a new row to data table, it uses SELECT * first to get column information. If there are hundred and thousand of records in the table, this action will certainly hinder the performance and may cause lock ups as these SQL Cicode functions are block functions.
5. Use SQL Server trigger to remove records from the working tables to the permanent tables. In this way, sizes of the working tables that are directly interfaced with Citect are not growing unrestricted.

6. Do NOT run SQL Cicode functions on a critical CitectSCADA machine. As SQL Cicode functions are block functions and running in the main thread, #COM or trend miss samples are likely to occur when SQL functions are executed on IO or trend. Similarly, an alarm may miss a scan or a report may miss a trigger if the functions are running on an alarm or report server.

Connecting to an SQL database

Before you can use SQL commands, you need to connect to the SQL database system. The SQLConnect function provides this access. You need to call this function before any other SQL functions. It has the format:

```
SQLConnect(sConnect);
```

where `sConnect` is the connection string, for example:

```
INT hSQL;
hSQL = SQLConnect("DSN=DBASE_FILES;DB=C:\ODBC\EMP;LCK=NONE;CS=ANSI");
! Connect to a dBASE Compatible Database File.
INT hSQL;
hSQL = SQLConnect("DSN=EXCEL_FILE;DB=C:\ODBC\EMP;FS=10");
! Connect to an Excel File.

INT hSQL;
hSQL = SQLConnect("DSN=ORACLE_TABLES;SRVR=X:ACCTS;UID=SCOTT;PWD=TIGER");
! Connect to an Oracle Database.
```

Refer to the documentation that accompanied your SQL server for details about connecting to an SQL database.

See Also

Executing SQL commands

Executing SQL commands

SQL allows you to manipulate data in a non-procedural manner; you specify an operation in terms of what is to be done, not how to do it. SQL commands allow you to:

- Create tables in the database.
- Store information in tables.
- Select exactly the information you need from your database.
- Make changes to your data and to the structure of a table.
- Combine and calculate data.

The SQLExec() function executes any SQL command that your SQL server supports. For example, to create a database table, you would execute the SQL "CREATE TABLE " command:

```
SQLExec (hSQL, "CREATE TABLE recipe ('Name' CHAR(16), 'Water' CHAR(8),
 'Sugar' CHAR(8), 'Flour' CHAR(8), 'Salt' CHAR(8), 'Yeast' CHAR(8), 'Milk'
CHAR(8))");
```

To add records into the database table, use the "INSERT INTO" command. The command has the following syntax:

```
INSERT INTO <filename> [(<col_name>, . . .)] VALUES (<expr>, . . .)
```

This command adds the values for each field in the table, for example:

```
SQLExec(hSQL, "INSERT INTO recipe VALUES ('Bread', '10', '5', '7', '1', '1', '2')");
```

Column names are optional; however, if you omit column (field) names, the values are inserted into the fields in the same order as the values.

To read data from an SQL database, use the SQL "SELECT" command. You can use the "SELECT" command to read an entire set of records, or a row, from the table. You can then use the SQLGetField() function to read the data in each field, for example:

```
SQLExec(hSQL, "SELECT * FROM recipe WHERE NAME = 'Bread'");
If SQLNext(hSQL) = 0 Then
        PLC_Water = SQLGetField(hSQL, "WATER");
        PLC_Sugar = SQLGetField(hSQL, "SUGAR");
        PLC_Flour = SQLGetField(hSQL, "FLOUR");
        PLC_Salt = SQLGetField(hSQL, "SALT");
        PLC_Yeast = SQLGetField(hSQL, "YEAST");
        PLC_Milk = SQLGetField(hSQL, "MILK");
END
```

To delete database records, use the SQL "DELETE" command. The command has the following syntax:

```
DELETE FROM <filename> [WHERE <conditions>]
```

This command deletes values from the table, for example:

```
SQLExec(hSQL, "DELETE FROM recipe WHERE NAME = 'Bread'");
```

See Also

Using a transaction

Using a transaction

You can use a database transaction for more sophisticated database operations. A database transaction allows you to execute a series of SQL commands and then either commit the changes to the database, or 'roll back' (cancel) the changes, for example:

```
SQLBeginTran(hSQL); ! Begin the transaction
SQLExec(hSQL, "UPDATE recipe SET water = '12' WHERE NAME = 'Bread'");
SQLExec(hSQL, "UPDATE recipe SET milk = '1' WHERE NAME = 'Bread'");
IF . . . THEN
        SQLCommit(hSQL); ! Commit the transaction
ELSE
        SQLRollBack(hSQL);! Cancel the transaction
END
```

Check the ODBC-compatibility level of your database driver if you cannot use transactions.

See Also

Expressing dates and times in SQL

Expressing dates and times in SQL

The way in which SQL dates are expressed depends upon the particular database system. With dBASE, you normally specify a date in braces, for example {02/18/95}. For Oracle, use the format: to_date(`02/18/95', 'MM/DD/YY'). Other ODBC drivers might require another format - a common ODBC format is: `YYYY-MM-DD'.

Note: Date references in an external database have to be based on the Gregorian Calendar, or the database tables needs to be exported to text files before use in CitectSCADA. Dates in Microsoft Access database tables exported as text files are stored as Gregorian values.

Database independent date-time syntax

For database independence, you can use the following syntax for dates and times:

```
[<format>'YYYY-MM-DD HH:MM:SS.FFFFFF']
```

where:

- **<format>** (the first character after the opening square bracket) needs to be one of the following:

- d - date
- t - time
- dt - date and time.

Whether you are specifying a date, time, or date and time, you need to provide the full 26 character string, for example:

```
[d'1995-02-18 00::00.000000']
```

Refer to the documentation that accompanied your SQL server for further information about SQL commands.

Using ODBC drivers

CitectSCADA supports the open database connectivity (ODBC) standard. Many manufacturers of database packages now also supply an ODBC database driver for their software. As well as these there are independent parties manufacturing ODBC database drivers for various databases, such as Intersolv Q+E with their DataDirect ODBC Pack. Usually, any ODBC driver for your database will work.

See Also

Installing the ODBC driver
About the ODBC driver
Setting up ODBC
Getting the correct syntax with ODBC
Programming style with ODBC
Using CitectSCADA as an ODBC server

Installing the ODBC driver

You need to install and setup up your ODBC driver from the Windows Control Panel. To do this:

1. Open the Windows Control Panel.
2. Click the **ODBC** icon to start the ODBC setup utility (if you do not already have an ODBC icon in your control panel, you might need to install the icon. See the documentation provided with your ODBC driver).
3. Click **Add** to set up a DataSource.

> **Note:** If your ODBC driver is not included in the list of Installed ODBC Drivers, return to the ODBC setup utility and install your driver (select the **Drivers...**

button).

4. From the **Add Data Source** dialog box, select your ODBC driver. The Setup dialog is displayed.
5. Enter the Data Source Name of the driver that you used previously. For example, if you had used SQL with a dBASEIII database, then your connection string would have been "DRV=QEDBF". To avoid changing the connection strings throughout your project, use a Data Source Name of **QEDBF**.
6. Run your CitectSCADA project.

Some Cicode SQL functions will not work if your ODBC driver has limited functionality. This situation is not frequently encountered, and in most cases affects only the ability to use transactions with the SQLBeginTran(), SQLCommit(), and SQLRollBack() functions. If you are using the Intersolv Q+E ODBC drivers, it is unlikely you will experience any loss of functionality.

You might need to change the data source in the Control Panel each time you switch from using one ODBC-compatible driver to another, for example from a dBASE file to an Access database. Click the ODBC icon and select from the list of available data sources. (Refer to the documentation supplied with your driver for more information.)

Notes:

- For maximum compatibility with the Cicode SQL functions, the ODBC driver has to provide a minimum of functions. For example, if your driver does not support the ODBC function SQLTransact, you cannot use the Cicode functions `SQLBeginTran()`, `SQLCommit()`, and `SQLRollback()`.
- Any functions you might have created in early versions are backward-compatible. Q+E drivers are now ODBC-compliant, so you need to upgrade your old Q+E database driver to a Q+E ODBC database driver.

See Also

About the ODBC driver

About the ODBC driver

CitectSCADA connects directly to the Microsoft Access ODBC driver, which allows applications to access information stored in MDB (Microsoft Access Database) files without actually running Microsoft Access. (Microsoft Access uses the "Jet Engine" DLL to access information stored in MDB files.)

ODBC normally implies heavy use of SQL statements to manipulate data. SQL statements can become quite complex and verbose. To implement them in Cicode they often have to be separated into sub-strings so that the maximum string length for Cicode variables is not exceeded.

With Access, it is possible to call queries that have been defined in Access so that the SQL statements become quite simple and straightforward. The Access tables & queries can be used to implement **RELATIONSHIPS** and **JOINS**, to **SORT** & **SELECT** only those rows (records) and return only those columns (fields) of particular interest at the time.

Developing queries in Access also has an advantage that the resulting Recordsets can be viewed in Access to test that they contain the data that is expected. The queries can incorporate SQL Functions (such as BETWEEN & AND).

The Jet Engine can also call upon theVBA Expression Service. This means that many non ANSI functions can also be used (both in SQL statements and Access Query Definitions) provided there is no need to migrate to a non Access system at a later date. Refer to VBA Functions Reference in the Access or Excel help system (only those functions with (VBA) after them and are appropriate to an SQL environment, are likely to work in an SQL statement).

See Also

Setting up ODBC

Setting up ODBC

To use ODBC, the Access ODBC Driver needs to be installed. This can be obtained from Microsoft and is included with Microsoft Office. The installation programme (for example, for Microsoft Office) will copy the necessary drivers and the Jet Engine DLL into the appropriate Windows directories when the appropriate Data Access/ODBC options are selected.

With the Driver installed on the PC the ODBC Icon can be selected from the Control Panel and a Data Service Name set up for the desired MDB. This is used in the DSN= part of the connect string.

The Jet Engine DLL is quite large (1 MB) and the execution speed of the runtime (and your application) can be impacted if the Windows Virtual Memory Manager (VMM) swaps it out of memory. The next time an SQL is executed there will be short delay while the DLL is loaded back into memory. To force the VMM to keep the DLL in memory, design a simple dummy table with only one record and one field and set up a Cicode task that frequently (say every 10-15 seconds) calls a SELECT query based only on the dummy table. This has no significant effect on CPU load and keeps the DLL in memory.

See Also

Getting the correct syntax with ODBC

Getting the correct syntax with ODBC

The ODBC syntax for SQLs varies from the Access syntax in some ways. A good way to get the syntax correct and view the resulting Recordset is to use the query designer in **Microsoft Query** then copy the SQL text from it into Cicode. Because MS Query uses ODBC, any syntax that works in it will work when called via ODBC from Cicode. MS Query can also be used to confirm that the DSN is correct.

MS Query tends to create SQL text that is more complex than necessary. In particular it includes the path with the file name which is not necessary because the path is already defined in the DSN entry. It is considered bad practice to hard code file paths. MS Query also tends to prefix column (field) names with the table names to avoid any chance of ambiguity. Again this is not always necessary and it is desirable to keep the SQL text as brief as possible in your code.

The SQL statement text generated by the query designer can be pasted into Execute SQL window (under the File menu of MS Query), any surplus text removed and the SQL statement tested until the simplest syntax that works can be found. There is provision to save the SQL text if necessary. The final version of the SQL statement can be used with confidence in Cicode.

See Also

Programming style with ODBC

Programming style with ODBC

Most of the sample code in the following topics do not include error checking and reporting:

- Reading data from an access table with ODBC
- Writing data to an access table with ODBC
- Deleting rows from an Access table with ODBC
- Calling action queries with ODBC
- Parameter queries using ODBC

> **Note:** These examples exclude error checking to keep them as simple as possible. However, both common sense and accepted programming practices mandate that you include error checking in your ODBC code.

Give consideration to implementing most of the complexity of queries in Access Query Definitions where they are easier to design and the results are easily viewed. A WHERE clause can be used when calling the query to select only the desired rows at run time. Where tables have many columns (fields), the Access Query Definitions can be used to restrict any particular call to view only the fields of interest.

It is helpful to build the SQL test up into strings. Firstly the ODBC function calls become simpler. Secondly the strings can be passed to TraceMsg() to make debugging simpler.

Remember that the Jet Engine runs on the same PC as CitectSCADA and that complex queries returning large Recordsets can have an adverse impact on CPU and memory resources. However, excessive impact on PC resources can be avoided by careful table, query and relationship design.

If there is a need to execute the queries on a **Remote Computer**, the code can set up on a Reports Server or an Event Server. This is especially relevant if the code is to be triggered by an event in a PLC. If the code is to be triggered by a User at a Display Station, and the query is considered too CPU intensive, the Display Station can be used to set the PLC bit that calls to code or call the Report using the Cicode Report() function. Another possibility is to use the Cicode MsgRPC() function to call a Cicode function (with parameters, if necessary) on a remote computer. Each of these alternatives require CitectSCADA to be running on the remote computer.

See Also

Comparing DDE with ODBC

Comparing DDE with ODBC

Each has advantages and disadvantages. In general DDE is suitable for simple requirements but ODBC give serious thought if the limitations of DDE become too restrictive.

DDE Advantages

- No need to set up a Data Service Name (DSN); however, a DDEShareName is necessary for Network DDE.
- Can call Access Macros & Functions.

DDE Disadvantages

- Record sets with rows that exceed the maximum Cicode string length cannot be read directly.
- Rows (records) are returned to string variable with TAB characters between columns. You need to parse the string in Cicode to obtain the column (field) values.
- SQL over DDE cannot perform actions (such as INSERT, UPDATE or DELETE).
- DDE Client and server applications need to be be running at the same time.

ODBC Advantages

- MS Access does not have to be running. ODBC uses the JET Engine DLL on the same PC. This an advantage in many ways but can consume excessive PC resources if not managed properly.
- Large SQL statements can be split into sub-strings.
- SQLGetField makes easier to get data from fields (columns). There is no need to parse the data in Cicode.
- Can handle large numbers of fields (columns) in the Recordset.
- SQL can perform actions (such as INSERT, UPDATE or DELETE).

ODBC Disadvantages

- Requires that a Data Service Name (DSN) be set up.

The JET Engine DLL cannot be directly called on a remote PC, (Reports or MsgRPC() can be used however, to run SQL statements on a Remote Computer which needs to be be running CitectSCADA).

See Also

ODBC compatibility

ODBC compatibility

This section describes the necessary and optional ODBC functions that your database driver needs to support:

- Essential functions
- Optional functions

Essential functions

The CitectSCADA SQL devices and Cicode functions only work if the database driver supports the following ODBC functions:

Level 0	Level 1
SQLAllocConnect	SQLColumns
SQLAllocEnv	SQLDriverConnect
SQLAllocStmt	SQLGetData
SQLBindCol	SQLGetFunctions
SQLColAttributes	SQLGetInfo

SQLDescribeCol	SQLGetTypeInfo
SQLDosconnect	SQLParamData
SQLError	SQLPutData
SQLExecDirect	SQLSetConnectOption
SQLExecute	SQLSetStmtOption
SQLFetch	
SQLFreeStmt	
SQLGetCursorName	
SQLNumResultCols	
SQLPrepare	
SQLRowCount	
SQLSetParam	

Optional functions

CitectSCADA SQL devices and Cicode functions also use the ODBC functions listed in the table below. While these functions are not necessary for operation, if they are absent in a driver, the error code 307 (SQL database error) will be returned if the ODBC driver does not support the necessary ODBC functions. To get the message associated with the error, call the Cicode function SQLErrMsg.

Level 0

SQLTran-sact	If the driver does not support this function, the Cicode functions SQLBeginTran(), SQLCommit(), and SQLRollBack() are not supported.

Level 1

SQLSpecial Columns	CitectSCADA uses this function if it is available. There is no loss of functionality otherwise.
SQLTables	(no effect on CitectSCADA)

Level 2

SQLData Sources	The driver does not need to support this function. It is provided by ODBC.
SQLExtended Fetch	Without this function, CitectSCADA cannot take advantage of the native database's ability to fetch records at random.
SQLSetScroll Options	Without this function, CitectSCADA cannot take advantage of the native database's ability to fetch records at random.
SQLMore Results	(no effect on CitectSCADA)
SQLNativeSql	(no effect on CitectSCADA)
SQLProcedure Columns	(no effect on CitectSCADA)

See Also

Using CitectSCADA as an ODBC server

Using CitectSCADA as an ODBC server

The ODBC server support allows CitectSCADA to function as an SQL database server. This will allow third-party applications that support ODBC to access data directly from CitectSCADA. This means that users can have direct access to data in CitectSCADA without having to develop Cicode or reports to export the data.

Currently, the CitectSCADA ODBC server allows variable tags to be accessed. The table for the variable tags is named '**TAGS**' and the format is as follows.

NAME	Variable tag name	read only
VALUE	The current runtime value	read/write

CitectSCADA can only function as a database server at runtime. Using tags through ODBC at runtime can still add to your CitectSCADA License point count. Once the dynamic point count is greater than the license point count, the software protection mechanism will shutdown CitectSCADA runtime. Therefore, when accessing tags via the ODBC server, it's important to keep aware of how many points you have used. For details see License point count in the Installation and Configuration Guide.

Setting up the CitectSCADA ODBC server:

You need to have TCP/IP installed on your computer first.

1. Choose **Start** | **Settings** | **Control Panel.**
2. Double-click the ODBC icon.
3. Click **Add** on the **User DSN** tab.

> **Note:** If you select other tabs you will see that the CitectSCADA ODBC driver has been automatically installed.

4. Select the **Citect Driver** from the list and click **Finish**.
5. Enter "**Citect**" in the **Data Source** field. If you do not want to use this name, make sure the name you use is one word.
6. Enter the Computer Name in the **Host** field. The Computer Name is specified in the Network section of the Control Panel.
7. Click **OK**.

Accessing the CitectSCADA ODBC server using MS Query (V2.00):

All ODBC capable applications use different ways to construct queries for accessing CitectSCADA tags. The example instructions for using MS Query, given here, show a simple implementation. MS Excel and MS Access use the same method.

1. Verify that you have MS Query installed on your computer.
2. Set up the CitectSCADA ODBC server.
3. Run CitectSCADA.
4. Run MS Query.
5. From the **File** menu (in MS Query) select **New Query**.
6. Select the CitectSCADA Data Source Name (DSN) from the Available Data Sources list. Click the **Use** button.
7. Select the **Tags** table. Click the **Add** button and then the **Close** button.
8. You can now run a query to extract the Tag data from CitectSCADA. The simplest way to see this is by double-clicking **Names** and **Tags**.

Accessing the CitectSCADA ODBC server using MS Query (V8.00):

Unlike Version 2.00, User DSNs are not used by Version 8.00. Instead it uses File DSNs which by default are stored in `Program Files\Common Files\ODBC\Data Source` folder. File DSNs are not stored in the Windows registry, they are text files given the .DSN extension. When you connect to an existing data source, only the available File DSNs that are stored on that PC are displayed. MS Query V8.00 does not display User or System DSNs. The simplest solution is to create a File DSN that points to a User DSN.

To create a file DSN that points to a user DSN:

1. Use a text editor (Notepad for example) and create a file containing the following two lines:

```
[ODBC]
DSN=<MyUsrDSN>
```

   where <MyUserDSN> is the name of an existing user DSN that you have created via the ODBC icon in the Control Panel.

2. Click **Save As** on the **File** menu and type a name that includes a .DSN file extension. For example, "**Citect_File.dsn**" is a valid name. Include the quotation marks so that the .DSN file name extension is added correctly. Save it to the default File DSN directory listed above, then it will appear in the DSN list box.
3. Open the ODBC Manager from the Control Panel and verify that you can see your newly created File.DSN.
4. Open the ODBC Manager from the Control Panel and verify that you have created a User DSN called <`MyUsrDSN`>. For example:
   1. Select Citect Driver and click **Finish**.
   2. Type "**Citect**" in the **Data Source** field (i.e., <`MyUsrDSN`>).
   3. Enter *Computer Name* in the **Host** field.

When you run MS Query, you can now select your File DSN from the list.

See Also

Reading data from an access table with ODBC

Reading data from an access table with ODBC

You can use a SELECT query to read data from an Access table or to call an Access query.

A query is preferred over a table if there are many more columns in the table than are needed at the time, if the data needs to be sorted, or if you need to relate or join several tables. The Cicode necessary is as follows:

```
Function SQLTest
        INT hSQL, iResult;
        hSQL = SQLConnect("DSN=ODBCTest;UID=YourUID_C;PWD=YourPWD");
        IF hSQL <> -1 Then
                iResult = SQLExec(hSQL, "SELECT * FROM qryRecipes
               WHERE Recipe Between '3000' And '6000'");
                IF iResult = 0 Then
                        WHILE SQLNext(hSQL) = 0 DO
                        TraceMsg(">" + SQLGetField(hSQL, "Recipe")
                        + "<>" +SQLGetField(hSQL, "Flour")
                        + "<>" +SQLGetField(hSQL, "Water")
                        + "<>" +SQLGetField(hSQL, "Cocoa") + "<");
                        END
                        SQLDisconnect(hSQL);
                ELSE
                        Message("SQL Error", SQLErrMsg, 48);
                END
        ELSE
                Message("SQL Error", SQLErrMsg, 48);
        END
END
```

See Also

Appending data with ODBC

Appending data with ODBC

To append data to an Access table using ODBC, you can use an SQL INSERT statement.

```
Function SQLInsert
        INT hSQL, iResult;
        hSQL = SQLConnect("DSN=ODBCTest;UID=YourUID;PWD=YourPWD");
        IF hSQL <> -1 Then
                iResult = SQLExec(hSQL, "INSERT INTO tblRecipes
                (Recipe, Flour, Water, Cocoa) VALUES ('X1234', 2, 3, 4)" );
                SQLDisconnect(hSQL);
        END
END
```

To avoid having to deal with SQL statements, you can use the standard Cicode device functions to append records to an Access table. First configure an SQL device. If the table has many fields that do not need to be written to, define only those fields that are necessary in the device definition (this keeps the device definition simple and reduces the number of DevWrite instructions). DevOpen, DevWrite and DevClose can then be used to add records to the table.

CitectSCADA accepts successive DevWrites until they equal the number of fields in the device definition at which time it constructs an SQL INSERT statement. The DevWrites needs to contain data for fields in the same order as the device definition. Do a DevOpen followed immediately by successive DevWrites for as many records as are necessary then a DevClose to avoid data being out of context.

See Also
Editing data with ODBC

Editing data with ODBC

To edit data in an Access table there needs to be a unique (usually primary) key to identify the row (record) to be changed. This is to be able to provide a WHERE clause that will apply only to that row in an SQL UPDATE.

Toedit data, read the data in the normal way, keeping track of the unique key. Any changed values can later be written to the same row using an UPDATE query with a WHERE clause.

```
Function SQLUpdate
        INT hSQL, iResult;
        hSQL = SQLConnect("DSN=ODBCTest;UID=YourUID;PWD=YourPWD");
        IF hSQL <> -1 Then
                iResult = SQLExec(hSQL, "UPDATE tblRecipes SET Flour = 20,
                Water = 30,Cocoa = 40 WHERE Recipe = 'X1234'");
                SQLDisconnect(hSQL);
        END
END
```

Note: The ODBC/SQL environment does not let you edit the "Current Record" (there is no current record). Consequently you cannot use `DevAppend` or `DevSetField` to add or modify records.

See Also
Deleting rows from an Access table

Deleting rows from an Access table

The DELETE keyword is used with a WHERE clause to delete the necessary row or rows. If the WHERE clause is not based on a primary key, more than one record may be deleted.

```
Function SQLDelete
        INT hSQL, iResult;
        hSQL = SQLConnect("DSN=ODBCTest;UID=YourUID_C;PWD=YourPWD");
        IF hSQL <> -1 Then
                iResult = SQLExec(hSQL, "DELETE FROM tblRecipes WHERE
                Recipe = 'X1234'");
                SQLDisconnect(hSQL);
        END
END
```

See Also

Calling action queries with ODBC

Calling action queries with ODBC

Access ACTION queries cannot be called in a SELECT query such as:

```
"SELECT * FROM qdeDeleteRecipe"
```

To call an Access ACTION via ODBC, use the Call statement in SQLExec:

```
"{Call qdeDeleteRecipe}"
```

The statement needs to be enclosed in {curly} braces.

The Call statement can be used to Call SELECT queries, the resulting Recordset being accessible in the normal way.

See Also

Parameter queries

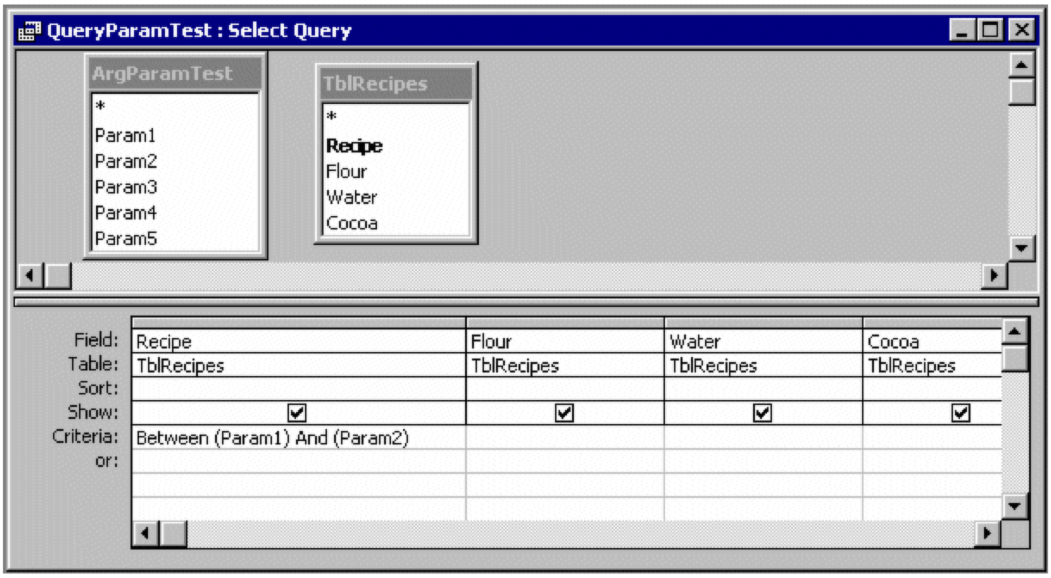
Parameter queries

Many ODBC Servers will accept PARAMETERS in a Call statement so that PARAMETERS can be defined in a Query Definition on the server and their values supplied by ODBC Clients at run time. Unfortunately, the Access Jet Engine uses Parameter Markers which are not supported in the standard Call statement. The method outlined here can be used as a work-around.

For each query that requires PARAMETERS, design a arguments table with the same name as the query but with a different prefix. For example, if the query is qryParamTest, the TABLE could be called argParamTest.

The TABLE could have, say, five fields called Param1, Param2, Param3, Param4, Param5.

Add this table to the Access Query Definition for qryParamTest. Then the field names can be used as PARAMETERS anywhere in the Query Definition.



Field:	Recipe	Flour	Water	Cocoa
Table:	TblRecipes	TblRecipes	TblRecipes	TblRecipes
Sort:				
Show:	☑	☑	☑	☑
Criteria:	Between (Param1) And (Param2)			
or:				

A simple Cicode function can be written to which the query name (without the prefix) and PARAMETERS are passed. The function inserts "arg" in front of the query name and executes a DELETE from the TABLE (to verify that it is empty) and then performs an INSERT to leave the table containing ONE RECORD with the desired PARAMETERS in the appropriate fields. The function then prefixes the query Name with "qry" and calls the query.

```
Function SQLCall(INT hSQL, STRING sQueryName, STRING sArg1 = " ", STRING sArg2 = "
",
       STRING sArg3 = " ", STRING sArg4 = " ", STRING sArg5 = " ")
        STRING sTable, sQuery;
        sTable = "arg" + sQueryName;
        sQuery = "qry" + sQueryName;
        SQLExec(hSQL, "DELETE FROM " + sTable);
        SQLExec(hSQL, "INSERT INTO " + sTable + " (Param1, Param2, Param3, Param4,
        Param5) VALUES ('" + sArg1 + "', '" + sArg2 + "', '" + sArg3 + "',
        '" + sArg4 + "', '" + sArg5 + "')");
        SQLExec(hSQL, "{Call " + sQuery + "}");
END
```

Calling the Function from Cicode is then as simple as:

```
SQLCall(hSQL, "ParamTest", "2000", "4000");
```

The default parameters for SQLCall needs to be SPACES if "Allow Zero Length" is "No" in the Access table Definitions for fields Param1, Param2 etc.

The function can be used to call many different PARAMETER queries.

An advantage of this work-around is that, even after CitectSCADA has been shut down, the query can be called from Access and, because the PARAMETERS are still stored in the arguments table, the resulting Recordset can be viewed in Access.

Another method is to design queries that perform any necessary joins, sorting and field selection the call them using a WHERE clause to select the desired rows (records).

See Also

Access and Cicode date/time conversions

Access and Cicode date/time conversions

Access and Cicode have different Date/Time variables data types. Date references in an external database has to be based on the Gregorian Calendar.

You can convert between date and time in three ways:

- **Convert to real numbers** - In both Cicode and Access you can equate real numbers to data/time variables, like this:

```
AccessTime = 25568.66667 + (CicodeTime/86400);
CitectTime = 86400*(AccessTime - 25568.66667);
```

- **Convert to strings** - The Data and/or Time are converted to and from text strings using the standard conversion functions available in each environment.
- **Use the #Date/Time# SQL syntax** - The Jet Engine will convert and Dates and Time strings enclosed in # markers. This date could be useful in a WHERE clause:

```
SELECT * FROM qryMyQuery WHERE 'Date' BETWEEN #3/20/96# AND #3/27/96#
```

The American Date format is used in this case, the Jet Engine DLL ignores the local Date and Time settings as set in Windows Control Panel.

Using Microsoft Excel to Edit .dbf Tables

CitectSCADA allows you to edit and save .dbf files (tables) used in CitectSCADA by opening them in Microsoft® Office Excel®.

Microsoft Office Excel® 2007 does not allow you to save files in .dbf format though you may open and edit them using the File > Open command. In order to overcome this limitation CitectSCADA includes an Add-In for Microsoft Excel called ProjectDBFAddIn. When this Add-In is loaded into Excel, it allows you to browse, open, edit and save CitectSCADA .dbf files in the correct format.

The ProjectDBFAddIn is an optional component included in the installation of CitectSCADA and is accessed from the installer. If you did not select installation of the Add-In during installation, you can install it at a later time by navigating to the ProjectDBFAddIn folder of your CitectSCADA installation and running the setup.exe in that folder.

> **Note:** The installation of the projectDBFAddIn will only be available if a version of Microsoft Excel 2003 or later is detected by the installer on the target computer.

Compatibility

Though this Add-In is specifically designed for use with Microsoft Excel 2007, it is compatible with Microsoft Excel 2003.Currently Microsoft Office 2010 is not supported.

Integration with Microsoft Excel

When you open Microsoft Excel, an additional toolbar, or an additional tab, called project DBF Add-in is available depending on the version of Excel that you are using.

The fields and buttons on this toolbar allow you to choose:

1. The path to the master.dbf.
2. A CitectSCADA project from the master.dbf .
3. A .dbf file (table) belonging to the selected project.
4. Save changes made to a .dbf file (table).
5. Save changes without re-indexing the table, or save and re-index.
6. Save a dbf file with a different name.
7. Open a dbf file in any location

However, you can also open .dbf files using the File > Open command.

Protected Tables

The installation of the Add-In incorporates a list of tables that by default are excluded from being opened using the Add-In interface. This list is in a file called ExcludedProjectTables.xml and can be found in the Add-In installation directory.

These tables are system tables that are maintained by CitectSCADA, and it is recommended that these files are not edited externally in Microsoft Excel. To minimize the likelihood of additional tables being edited with Microsoft Excel you can add their names to the ExcludedProjectTables file using a text editor. You can also view the default list of tables using a text editor, or an XML compliant browser.

Read Only Tables

Projects which are read only, or contain read only tables, are displayed in the list of tables when selected from the Add-In buttons. If you try to open a read-only table (.dbf file) an alert (sometimes referred to as a "warning") message is displayed advising that the table is read only. You can still open and edit a read only table but you will not be able to save it.

A table could be read-only for the following reasons:

- The project is read-only.
- The table is open in the CitectSCADA Project Explorer.
- The user trying to edit the file does not have read and/or write privileges.

If you attempt to save a read-only table an alert message is displayed advising that the table is read only.

See Also

Functionality

Functionality

The add-in is automatically available to Excel when the add-in is installed by the CitectSCADA installation. When you open Microsoft Excel you will see a new toolbar called Project DBF Add-in, containing the following fields and buttons:

1. Master.dbf location
2. SaveDBF table
3. SCADA project
4. SCADA table
5. Save DBF
6. Save As
7. Open DBF
8. Save Only/Save and re-index

To access the tables using the Add-In:

1. If the location of the Master.dbf file is not displayed in the master.dbf location field, it will display Enter path to master.dbf. Click on this field to display a dialog box where you can enter the location. This populates the SCADA Project field.
2. Click the drop down arrow in the SCADA Project Field to select the Project in which the .dbf file (table) resides that you want to open. This populates the SCADA Table field.

Note: Any tables listed in the ExcludedProjectTables.xml are excluded from the list of tables.

3. Click the drop down arrow in the SCADA Table field to select the table. The table will open in Microsoft Excel as a new worksheet named with the table name. If the table is a read only table an alert (sometimes referred to as a "warning") message is displayed advising you that it is read only.

Alternatively, if the dbf file that you wish to open is not the Master.dbf file, or is not located in the root dbf file location, click the Open DBF button and browse to the location of the dbf file that you want to open.

To edit the contents of the table in Microsoft Excel

A table in .dbf format is displayed in Microsoft Excel in the following equivalent format:

- A field is displayed as a column.
- A record is displayed as a row.
- A field within a record (data) is displayed as a cell.

You may edit any of the data in the table, shown as cells within a row. You may delete a record shown as a row, and add a record. However you cannot change the structure of the table by modifying the name of any Field, shown as a column, add a column or delete a column. If you do change the structure an alert message will be displayed when you attempt to save the table.

If you edit a cell in the table take care not to exceed the maximum field length set for that field in the table.

To save a table in Microsoft Excel

1. Select to Save and re-index the table, or Save Only, from the drop down control on the toolbar.

If you do not re-index an indexed table the table will be saved more quickly, but you will need to manually re-index the table when you have completed your edits. To re-index manually, select Pack from the File menu of the CitectSCADA Project Editor.

2. Click the SaveDBF table button.

When you save the table the Add-In performs the following checks prior to saving the table:

1. Verifies that the file type is .dbf.
2. Checks for any changes in table structure that may have been made. For restrictions on changes to the table structure, refer to "To edit the contents of the table in Microsoft Excel".

3. Updates the existing table according to the data in the Microsoft Excel worksheet.
4. Re-indexes the updated table if it is was an indexed table (if Save and re-index was selected).

If the edited table in the Microsoft Excel worksheets does not pass every check, an alert (sometimes referred to as a "warning") message is displayed and the existing table on disk will not be affected until you correct the condition that caused any check to not pass.

You can also save a dbf file with a different name by clicking the Save As button on the toolbar and entering a name for the file. This method of saving also includes the option of saving the dbf file with or without re-indexing the table.

Note: If you attempt to save a table in .dbf format in Microsoft Excel using the Microsoft Excel File > Save command an alert message is displayed advising you that the file cannot be saved in the current format and that use the SaveDBF button on the Microsoft Excel toolbar.

Chapter: 32 Genies and Super Genies

There are two types of Genies within CitectSCADA:

Genie - collection of related objects, which you add to your graphics pages when you configure your system. You can paste any number of Genies onto a graphics page (for example, multiple pumps on the same page).

Super Genie - dynamic pages (usually pop-ups), to which you pass information when the page displays at runtime. You can use Super Genies for pop-up type controllers (to control a process, or a single piece of plant floor equipment).

CitectSCADA includes libraries of Genies and Super Genies that you can use in your system. You can also define your own and add them to a specific library to be reused again within your project.

See Also

Genies
Configuring Super Genies

Genies

Usually each object on a graphics page is configured individually. With a Genie, you can combine several related objects into a group, and store the group in a Genie library (similar to a symbol library). The Genie can then be used as a single object (pasted, moved, resized, etc.), and the elements configured collectively. Many types of graphic objects, and their configuration data, can be stored with the Genie.

Genies work by substituting common information in an object. Genies can be groups of objects (start/stop controller) or even one object (a button), that you may want to save in the genie library so you can reuse it through out the project.

The advantage of using a Genie is that objects are defined once, and then after you place the Genie onto a page, you only need to configure the substituted properties.

Before using Genies you need to understand the following term:

Substitution –A substitution can be a number or name you use to define an object or group of object’s properties when creating a genie. Used as a placeholder the property is replaced at runtime with a real value.

See Also

Configuring Genies
Defining Substitutions for Genies

Using Genies

Defining Substitutions for Genies

To define a Genie, you use substitution strings for the properties of the objects that will be specific to each instance. You can use substitution strings for any text property in any object (in the group of objects). To specify a piece of text as a substitution string, enclose the string between percentage (%) characters.

For STRING variable type substitutions enclose the %% in quotation marks e.g. “%label%”, so that CitectSCADA does not read it as a tag.

> **Note:** You are not restricted to using only variable tags as substitution strings. Any expression can be substituted, such as constants or labels. Only fields that accept text can have Genie tag substitutions. You can also define substitutions to variables that aren't in the current project by using the IFDEF function.

See Also

Configuring Genies
Using Genie Substitutions in Templates

Configuring Genies

Configuring a new Genie is similar to creating a page with graphical objects, but with genies there is no background. Typically you create a new Genie using the Graphics Builder, add the objects, defining the Genie substitutions, and save the Genie in a Genie library.

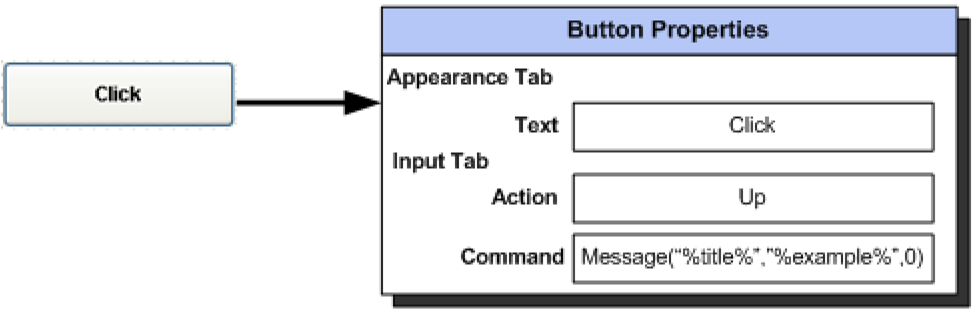
To create a new Genie:

1. From the **File** menu select **New**.
2. Click the **Genie** button.
3. Create your Genie object (defining your substitutions).

In the first example a button object will be created and saved as a genie:

1. Select the button object from the drawing toolbox
2. Draw a button and in the button properties dialog:

- Configure the appearance of the button
- Configure the input command
- Configure the Cicode function and its parameters.In the example the parameters have been substituted with a placeholder.



Button Properties

Appearance Tab

Text: Click

Input Tab

Action: Up

Command: Message("%title%","%example%",0)

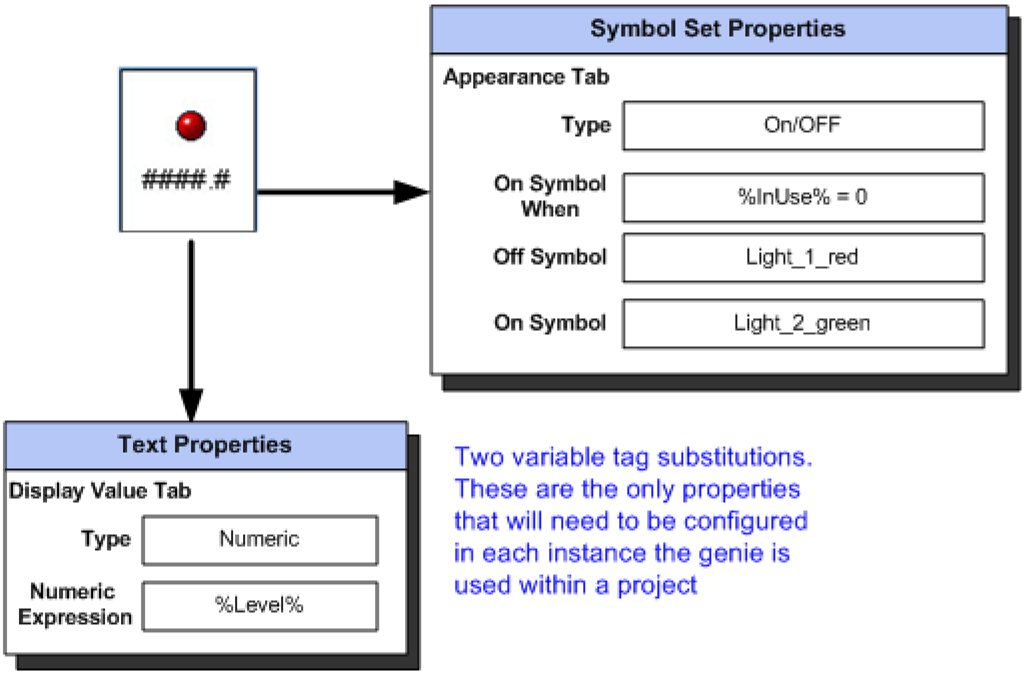
In this example both substitutions are STRING variable types, therefore the %% characters have been enclosed in quotation marks.

3. Click **OK** to close the button properties dialog.
4. Click the **Save** tool, or choose **File** | **Save**.
5. Select the Project and Library in which to store the Genie.
6. Enter a name for the Genie in Genie.
7. Click **OK**. (To create a new library for the Genie, click New. In the field provided name the new library and click OK. The name of the library is added to the library list.)

In the second example two objects are combined and saved as a genie.

A text object that shows the Level of Tank 1, and a symbol object that indicates if Tank 1 is in use.

1. Select the symbol set object from the drawing toolbox
2. Drag onto the blank genie page, the symbol set dialog will open
3. Configure the appearance of the symbol set (Type, On symbol when, OFF symbol, On symbol)
4. Select the text object and place onto the page under the symbol.
5. Configure the text object (Type, Expression)
6. Click **OK**
7. Click the **Save** tool, or choose **File** | **Save**.



Symbol Set Properties

Appearance Tab

Type: On/OFF

On Symbol When: %InUse% = 0

Off Symbol: Light_1_red

On Symbol: Light_2_green

Text Properties

Display Value Tab

Type: Numeric

Numeric Expression: %Level%

Two variable tag substitutions. These are the only properties that will need to be configured in each instance the genie is used within a project

See Defining Substitutions for Genies

See Also

Using Genies

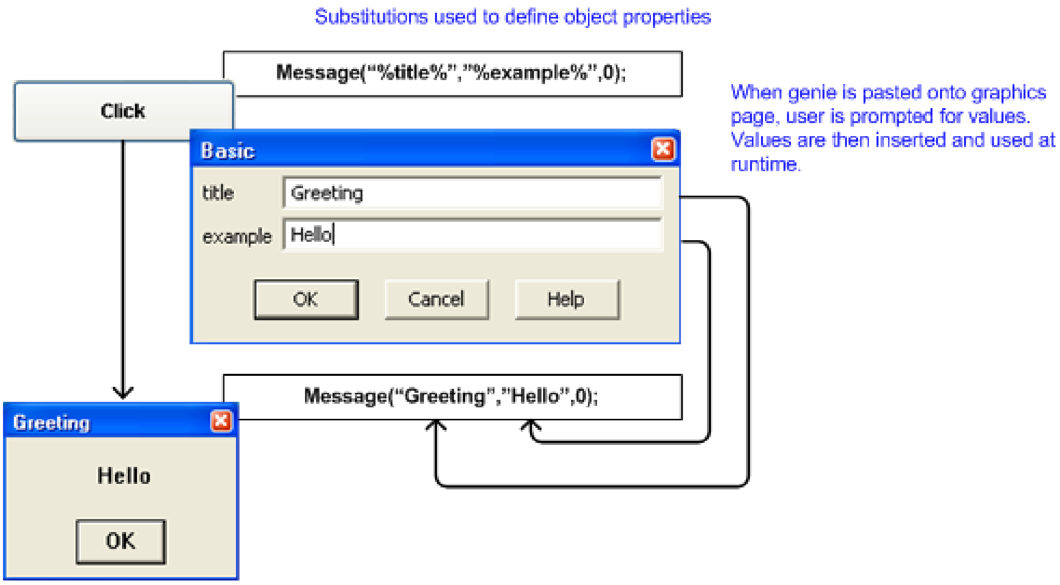
Maintaining Genies

Using Genies

Once you have created (defined and saved) your Genie, you can use it on any graphics page.

Using the first genie example (the button) from the create genie topic:

1. Click the Paste Genie tool in the toolbox or choose Edit | Paste Genie
2. In the Paste Genie dialog, select the library the genie belongs to.
3. Select the genie from the list and click OK (Or double-click the thumbnail of the genie)
4. The genie is pasted onto the graphics page. A dialog will open, prompting you to configure the properties of the genie. You will see that the field names match the ‘substitutions’ used when creating the genie. Click OK to close the dialog.


Substitutions used to define object properties

```
Message("%title%","%example%",0);
```

When genie is pasted onto graphics page, user is prompted for values. Values are then inserted and used at runtime.

Basic

title: Greeting

example: Hello

```
Message("Greeting","Hello",0);
```

At runtime the dialog values you entered into the prompt replace each substitution in the genie.

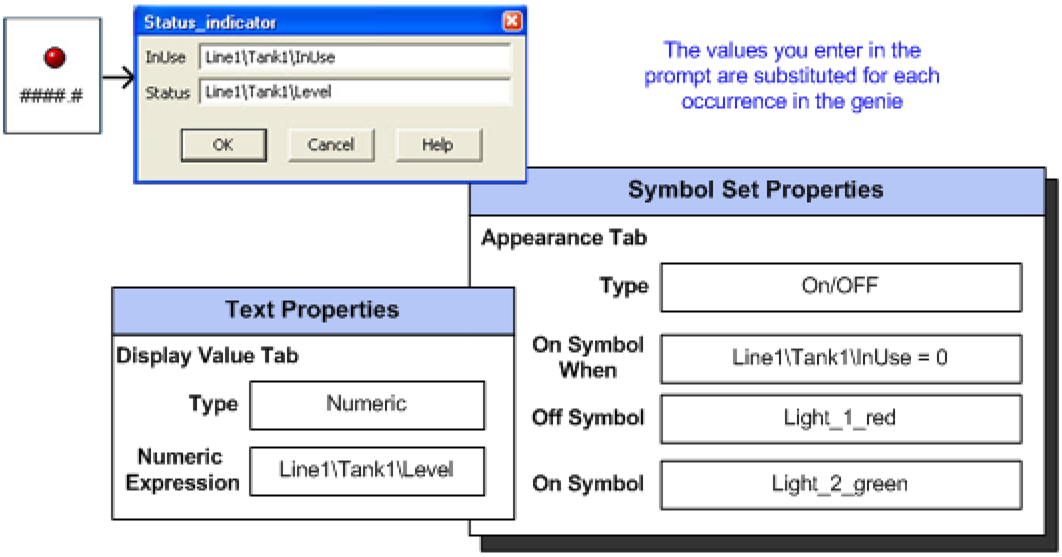
To reuse this genie in the project you would enter a different title and example text when prompted.

> **Note**: To display the properties of the individual objects in a Genie (instead of the Genie Properties), hold the Control (CTRL) key down and double-click the object. If, however, a link to the Genie has been retained, most of these properties will be read-only.

Using the second genie example:

For the on/off indicator you will need to configure the InUse and the Level substitution fields.

1. Click the paste genie tool.
2. Select the library (from the Library list) that contains the Genie.
3. Select the on/off indicator and click OK.
4. The genie is pasted onto the graphics page and you are prompted to configure the substituted values.



Status_indicator

InUse: Line1\Tank1\InUse

Status: Line1\Tank1\Level

The values you enter in the prompt are substituted for each occurrence in the genie

Symbol Set Properties

Appearance Tab

Type: On/OFF

On Symbol When: Line1\Tank1\InUse = 0

Off Symbol: Light_1_red

On Symbol: Light_2_green

Text Properties

Display Value Tab

Type: Numeric

Numeric Expression: Line1\Tank1\Level

The examples used above are simple uses of a Genie. You can define Genies that use many objects, with substitution strings for any text property (or properties) of an object.

> **Note**: If you use structured tags, you can use substitution strings within a tag name to construct more sophisticated Genies. See Using Structured Tag Names with Genies.

See Also

Maintaining a Genie

Maintaining a Genie

You can open an existing genie to edit it.

To open an existing Genie:

1. Click the **Open** tool or choose **File** | **Open**.
2. Select the **Genie** tab.
3. Select the **Project** and **Library** in which the Genie is stored.
4. Select the **Genie**.
5. Click **OK**.

To delete a Genie from the project, select the Genie name, and click **Delete**.

If you modify a Genie after you have used it in your project, occurrences of the Genie may be automatically updated throughout the project

If you modify a Genie when the project is running in the background, you need to perform an Update Pages to see the changes in the runtime project. If a runtime page containing the Genie is displayed when the change is made, it will not be updated until you exit then re-display it.

See Also

Configuring Genies
Using Genies
Using Genie Substitutions in Templates

Using Genie Substitutions in Templates

When you create a new custom template you can add objects to the template and configure the object or group of objects as a Genie.

Configure the Genie using substitution strings (%) for the relevant properties of each object. (If you have default values for any property, you can add the default values to the native objects.)

Save the new template.

When you subsequently create a new page based on the template, double-click the Genie and a single dialog will appear prompting you for the value or values of the substitution strings used in the template. This is how the templates for trending and SPC were created.

Using structured tags with Genies

When you define a Genie, you can add a prefix or suffix to a Genie property to generate the complete tag when the Genie is used. For example, if you define a Genie property as %tag%_PV, and then use DEV1 for the tag, the Genie will generate the complete tag DEV1_PV.

You can add extra information at the beginning (prefix), or on the end (suffix) of the Genie property, or use both a prefix and suffix in the same Genie property. For example, if you have defined a loop controller with three bar graphs (created using the fill property in a rectangle) to display the tags DEV1_PV, DEV1_SP and DEV1_OP, you can configure a Genie as follows:

Each rectangle has a separate Genie tag:

Level expression	%PV_Tag%
Level expression	%SP_Tag%
Level expression	%OP_Tag%

When you configure the Genie (with the Genie dialog), you have to enter three separate tags: DEV1_PV, DEV1_SP and DEV1_OP. However, if you use structured tags, you can configure the rectangles as follows:

Level expression	%Tag%_PV
Level expression	%Tag%_SP
Level expression	%Tag%_OP

In this case, you only have to enter one tag (DEV1) to generate six objects. The Genie automatically concatenates DEV1 with either _PV, _SP, or _OP, depending on where the tag is substituted. As well as a reduction in configuration time, this Genie is easier to maintain.

Note: The above example illustrates the power of Genies. The more complex and the greater number of objects in a Genie, the greater the advantage of using structured tags. You can also make complex Genies by using multiple variables for a Genie property. For example, `%Area%_TIC_%Occ%_PV` or any combination of prefix, suffix and number of Genie variables.

See Also

Using structured tags with Super Genies

Super Genies

Individual pages (popup controllers, loop tune pages, etc.) are often used to control and monitor devices. Super Genies are ideal when there are many devices of the same type within a project, because you can re-use them without re-configuring for each device. You can configure the common properties once, which is then passed to the Super Genie at runtime.

Before Configuring a Super Genie you need to understand the following terms:

Association- An association is a name or number that you can use when defining a Super Genie substitution. Used with ‘Ass’ Super Genie cicode functions to assign, and set the dynamically generated values of the Super Genie at runtime.

Substitution –Used as a placeholder, the Super Genie substitution is replaced at runtime with a real value. A Super Genie substitution is comprised of the data type (optional) and association that you use to define an object or group of object’s properties when creating a Super Genie.

See Also

[Configuring a Super Genie]
[Defining Substitutions for Super Genies]
[Defining Associations for Super Genies]

Defining Substitutions for Super Genies

You can only use Super Genie substitution in the properties of an object that accepts tags, commands and expressions. You can also use Super Genie substitutions in log messages for object touch and keyboard commands, tool tips, page keyboard commands, or as part of the comment for Trend objects, and Color Floods. However, you cannot use the Super Genie syntax in a report, alarm, trend, or background Cicode function.

To mark a tag as a substitution string, enclose the tag between question mark (**?**) characters, in the following format:

?<Data Type> <Association>?

where:

- **Data Type** is optional except in the case of the STRING variable type, which needs to be explicitly specified. When Data_Type is left blank, CitectSCADA will correctly typecast variable types other than STRING at runtime.

Note: Explicit type casting to BCD, BYTE, LONGBCD, UINT and ULONG types is not supported for the substituted tag. Use typeless substitutions for these types.

- **Association** - Can be a Number or a Name.
  - **Number** determines which variable tag (1 to 256) will be substituted when the Super Genie is displayed (using the Super Genie functions). If you use more than one substitution string in your Super Genie, make your numbers sequential. This will make the Super Genie functions easier to use.
  - **Name** determines which variable tag will be substituted when the Super Genie is displayed. For example, if a substitution is defined with the name SPEED, and explicitly specifies the type as REAL, the substitution would be referenced as ?REAL SPEED? on the graphics page. The page substitution may be referenced as many times as necessary on the Super Genie page.

If you do not specify a data type, it will default to TYPELESS. Typeless substitution allows you to pass tags of BYTE, BCD, DIGITAL, INT, UINT, LONG, LONGBCD, or REAL types, but not STRING. When you make a typeless substitution, CitectSCADA will automatically try to convert the substituted 'data' to the correct type at runtime.

Note: You might want to use typeless substitutions because they offer more flexibility, but be aware that errors can be harder to find.

See Also

Defining Associations for Super Genies

Configuring Super Genies

Defining Associations for Super Genies

You can define Associations for Super Genies two ways:

Using the Ass () Cicode functions. As part of the Cicode expression the association can be passed directly to the Super Genie substitution. The Ass () functions are processed just before runtime, to generate the real value of the Super Genie. However if the page association is not able to be performed there will be no default value or value on error available.

Using the Page Properties – Associations tab – you can reference or edit existing associations. In using this tab you can add those associations that are currently in use in a Super Genie substitution and define the appropriate default value and value on error fields. You can also add a description.

See Also

Defining Substitutions for Super Genies

Configuring a Super Genie

Configuring a Super Genie

You can configure and save a Super Genie either as a page or a library object.

See Also

Configuring a Super Genie as a Page

Configuring a Super Genie as a Library Object

Configuring a Super Genie as a Page

You can configure and save a Super Genie as a normal graphics page. This means that when you use the Super Genie you do not need to define a Genie to call it, and instead can use any graphics object.

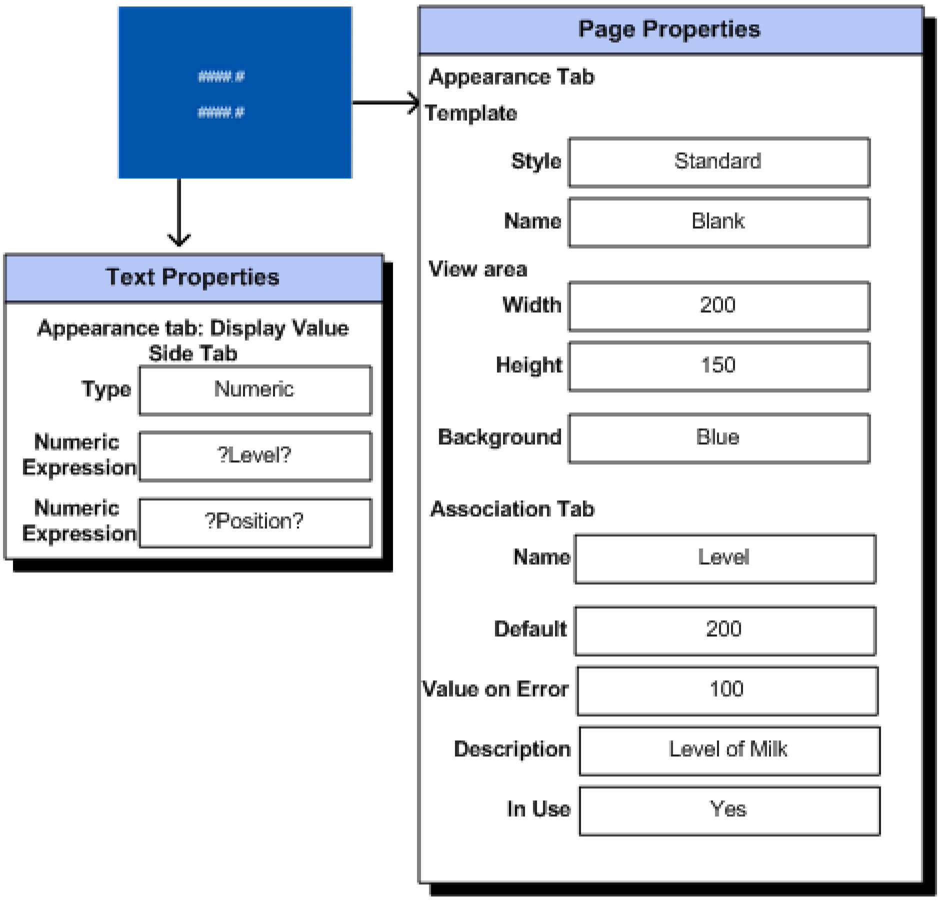
Note: If you configure a Super Genie in this way and name the page with an ! prefix to hide it, you need to select List System Pages from the Graphics Builder Options menu to show the page with the other pages, so you can then select it and edit the page.

To create a new Super Genie page:

1. From the toolbar select **New**, or go to **File** | **New**
2. Select **Page**
3. Choose a Page template and click **OK**
4. Define the page properties e.g. size of page
5. Place an object on the page e.g. numeric expression
6. Define the objects properties.
7. Select the type of object
8. In the expression field enter the association (placeholder) – don't forget to enclose the name or number in question mark characters. E.g. ?Level?

Note: When creating a Super Genie, if you reference an association that has not been defined on the Page Properties - Associations tab, it will be assumed to be an association type substitution with no Default, Value on error, or Description. You will need to define the Default, Value on error, and add a Description (if necessary) directly in the associations tab.

9. Click OK to apply the changes
10. Select **File** | **Save**
11. In the Save dialog, select the project the page will belong to (be aware that tab options are unavailable)
12. Name the Super Genie page and click **OK**



Text Properties

Appearance tab: Display Value Side Tab

- Type: Numeric
- Numeric Expression: ?Level?
- Numeric Expression: ?Position?

Page Properties

Appearance Tab

Template

- Style: Standard
- Name: Blank

View area

- Width: 200
- Height: 150
- Background: Blue

Association Tab

- Name: Level
- Default: 200
- Value on Error: 100
- Description: Level of Milk
- In Use: Yes

In the example above there are two numeric expressions on the Super Genie page. When configuring the Super Genie the association name Level was added directly to the page properties- associations tab. The other association name Position will be passed to it at runtime using an Ass() Cicode function.

See Also

Using a Super Genie Page

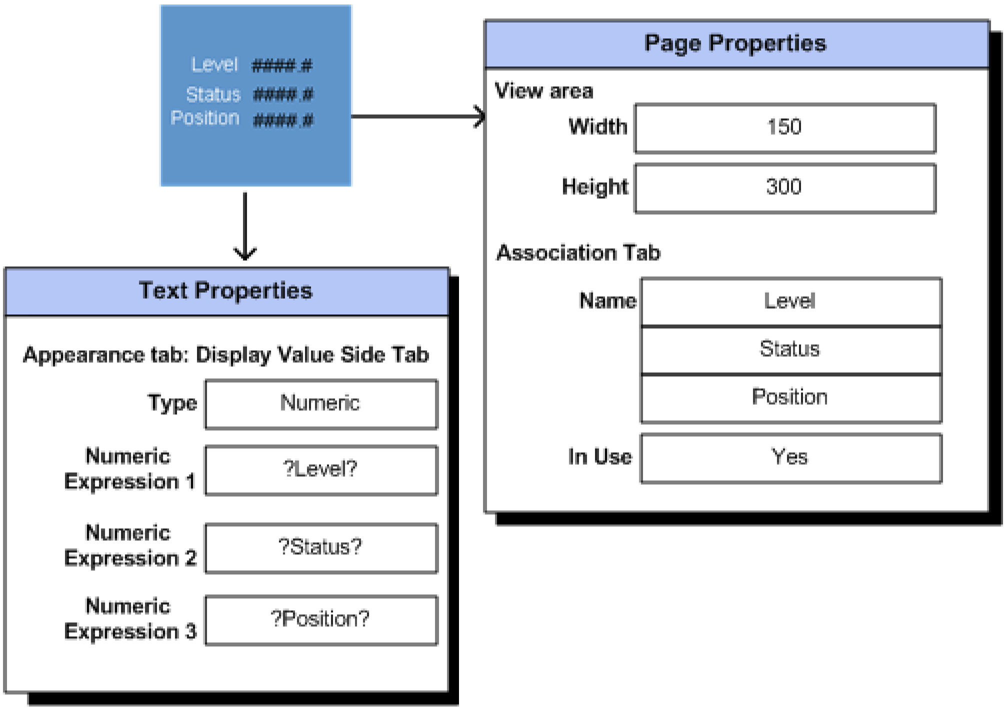
Configuring a Super Genie as a Library Object

Configuring a Super Genie as a Library Object

You can configure and save a Super Genie as a Super Genie library object. If you save your Super Genie as a library object it is not saved as a normal page and thus is not instantiated. A Super Genie library object is only instantiated when attached and then called from a genie. Once the library object is instantiated the Super Genie can be called from other objects.

To create a new Super Genie (using the Super Genie library option):

1. From the toolbar select **New**, or choose **File** | **New**.
2. Click the Super Genie option.
3. A blank Super Genie library object page will open.
4. In this example a pop up that displays numerous values will be created.



Page Properties

View area

Width	150
Height	300

Association Tab

Name	Level
	Status
	Position
In Use	Yes

Text Properties

Appearance tab: Display Value Side Tab

Type	Numeric
Numeric Expression 1	?Level?
Numeric Expression 2	?Status?
Numeric Expression 3	?Position?

5. From the tool bar select **Save** or go to **File** | **Save**.
6. The Save as dialog opens. The Super Genie tab is active, with no other options available. Enter a name for the Super Genie in the Super Genie field (!sg2_multi)
7. Select the Library and Project in which to store the Super Genie. Like Genie libraries, Super Genies libraries are global and can be used between projects.
8. Click **OK**.

> **Note**: The first eight characters of the Super Genie name needs to be unique for each Super Genie. Save it in a Super Genie library using an exclamation mark (!) prefix. This keeps the pages hidden in the configuration environment (they're visible only if attached to a Genie).

See Also

Using a Super Genie Library Object

Using a Super Genie Page

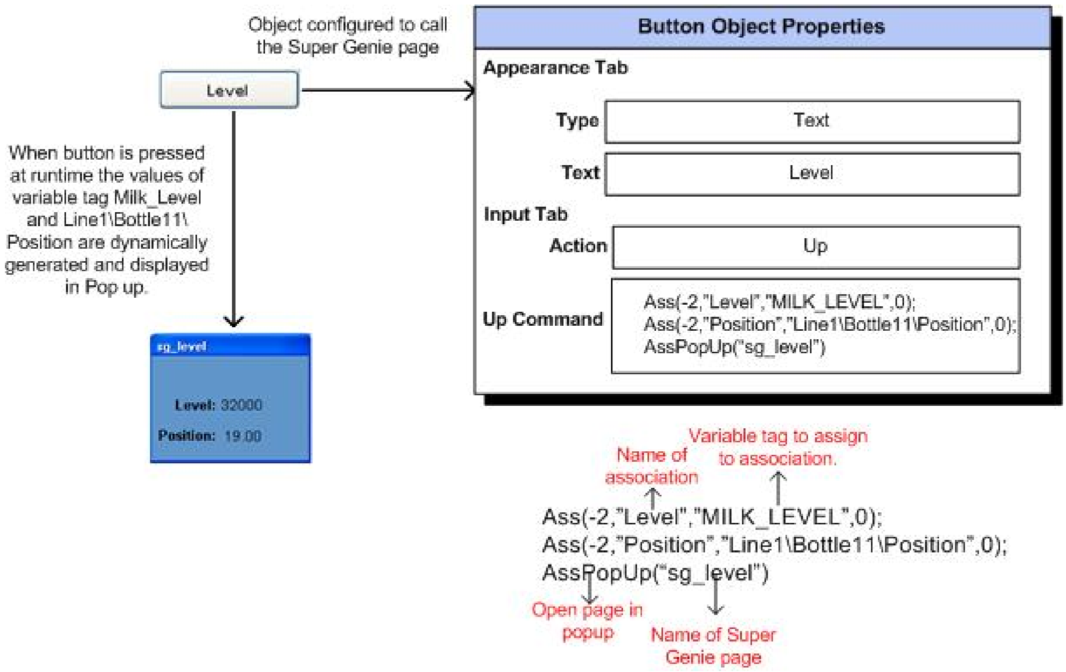
To use a Super Genie page you need to add an object to the graphics page. This object can be anything from a new menu item, a button, or a symbol that is configured to call the “Super Genie”. When configuring the object, you use one of the “Ass” Cicode functions that at runtime replace the substituted parameters with the associated values.

If you use the ‘Ass’ Cicode function you need to use it once for each value you want to pass to the Super Genie. If you use the ‘AssMetadata’ Cicode function matching metadata will be processed, with the values dynamically generated and set where applicable on the Super Genie page at runtime.

> **Note**: When embedding a Super Genie in another. The embedded Genie (for the embedded Super Genie) needs to use AssMetadata functions instead of Ass functions.

Using the basic Super Genie example created previously:

1. The button properties dialog will display
2. Configure the appearance of the text on the button
3. Configure the Input tab as outlined below. (In this example the Ass Cicode function is used to highlight how an association can be passed directly from the Cicode or defined in the page properties of the Super Genie.)



Object configured to call the Super Genie page

When button is pressed at runtime the values of variable tag Milk_Level and Line1\Bottle11\Position are dynamically generated and displayed in Pop up.

Button Object Properties

Appearance Tab

Type: Text

Text: Level

Input Tab

Action: Up

Up Command:

```
Ass(-2,"Level","MILK_LEVEL",0);
Ass(-2,"Position","Line1\Bottle11\Position",0);
AssPopUp("sg_level")
```

Name of association

Variable tag to assign to association.

```
Ass(-2,"Level","MILK_LEVEL",0);
Ass(-2,"Position","Line1\Bottle11\Position",0);
AssPopUp("sg_level")
```

Open page in popup

Name of Super Genie page

4. Save the graphics page

You can use Super Genie pages more than once. For example, another button could be added to the graphics page, with the Value defined in the Metadata tab a different variable tag. When the user clicks on that button at runtime the associated value for that variable tab will be displayed.

See Also

Using a Super Genie Library Object
Maintaining a Super Genie

Using a Super Genie Library Object

Using a Super Genie library object is initially not as simple as using a Super Genie page.

To use a Super Genie library object you need to:

- Create and save a Genie object
- 'Attach' the Super Genie library object to the Genie.
- Whenever you use that genie the Super Genie will be attached. You will have to 'Detach' the Super Genie if you do not want to use that genie with the Super Genie.

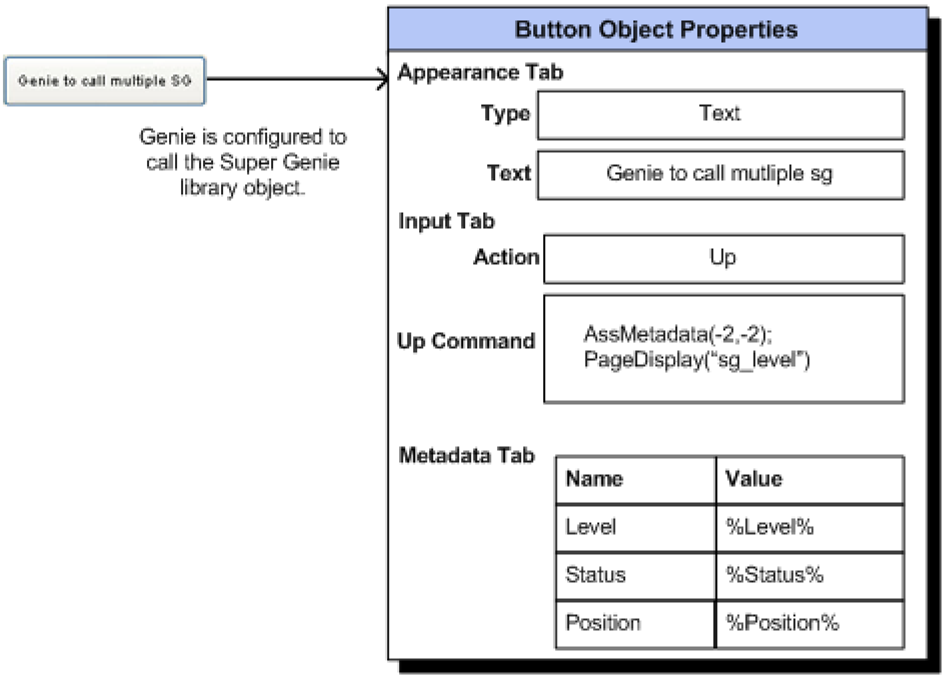
> **Note**: When the Super Genie is attached to a genie, the Super Genie library object is instantiated and becomes a new page in your project. This means any object can be configured to call the Super Genie page.

Create and Save a Genie object

In the following example a button will be created and saved as a genie. The Super Genie library object created in a previous example will then be attached to the genie.

1. Select **File | New Genie**
2. Draw the button that the user will click at runtime to display the popup. This button will call a Super Genie Cicode function, which performs the associations and displays the dynamically generated values
3. Configure the input tab. In the example below the AssMetadata Cicode function is used to call the Super Genie.
4. Configure the Metadata tab – define the name | value pairs. Match the name of the metadata with the name of the association, so the dynamically generated value is assigned to the correct association at runtime.

> **Note**: For metadata values with no corresponding tag configured, place the value within single quotes otherwise a hardware alarm will be raised. For example if a value MyTitle has been configured to substitute a Super Genie window title (?TItle?), then configure the value as 'MyTitle'


Genie to call multiple SG
Genie is configured to call the Super Genie library object.

Button Object Properties

Appearance Tab

Type: Text

Text: Genie to call mutliple sg

Input Tab

Action: Up

Up Command:

```
AssMetadata(-2,-2);
PageDisplay("sg_level")
```

Metadata Tab

Name	Value
Level	%Level%
Status	%Status%
Position	%Position%

5. Save your genie to a genie library.

Attach a Super Genie to the Genie

Attaching a Super Genie to a Genie:

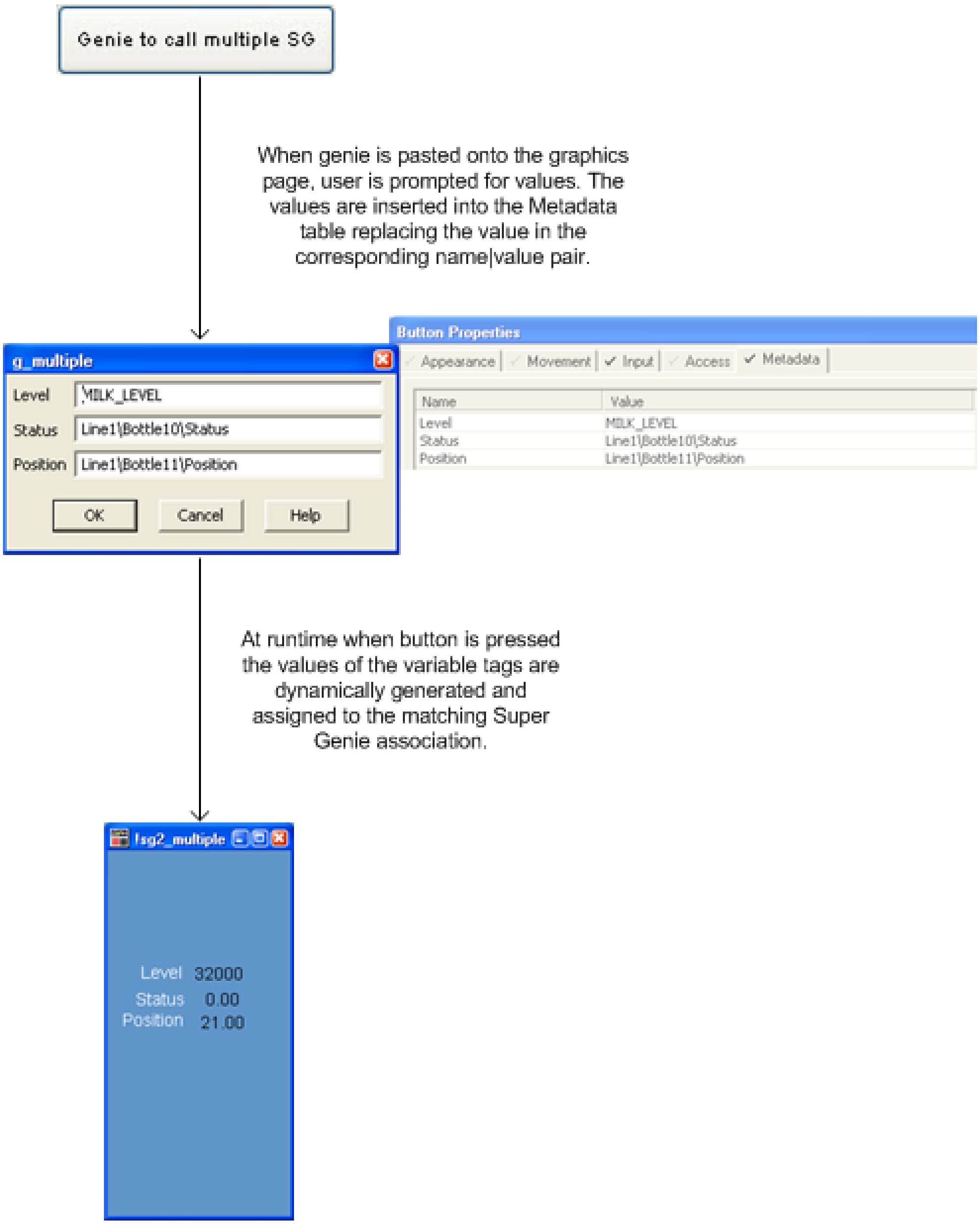
1. With the Genie open, select **Edit | Attach Super Genie**.
2. Click **Add**, and the Super Genie Dialog box will open.

3. Select the library the Super Genie belongs to.
4. Select the Super Genie to attach e.g. !PopUp_SG and click OK.
5. The Super Genie is added to the Attach Super Genie list.
6. Click **OK** to save the changes, or click **Cancel**.

Note: A Super Genie can be attached to more than one Genie.

Using a Super Genie with a Genie:

1. From the graphics page select Edit | Paste Genie, browse for the Genie you just created, and select it.
2. On Pasting the Genie, a dialog will open prompting you to fill in the fields and enter the variable tags.



When genie is pasted onto the graphics page, user is prompted for values. The values are inserted into the Metadata table replacing the value in the corresponding name|value pair.

Level: MILK_LEVEL
Status: Line1\Bottle10\Status
Position: Line1\Bottle11\Position

Name	Value
Level	MILK_LEVEL
Status	Line1\Bottle10\Status
Position	Line1\Bottle11\Position

At runtime when button is pressed the values of the variable tags are dynamically generated and assigned to the matching Super Genie association.

Level 32000
Status 0.00
Position 21.00

Note: To avoid compilation errors variable tags used in a Super Genie needs to be defined in the variable tags database. Alarm tags can also be used (allowing you use alarm tag properties).

Using tags through Super Genies at runtime increases your dynamic license point count. Super Genies called after you have reached your point limit will return #COM. For more information see license point count in the Installation and Configuration Guide.

See Also

Maintaining a Super Genie

Using a Super Genie Page

Maintaining a Super Genie Page

You can open an existing Super Genie page to edit it.When you modify a Super Genie when the project is running in the background, you will be prompted to 'Update Pages' to see the changes in the runtime project. If a runtime page containing the Super Genie is displayed when the change is made, it will not be updated until you exit then re-display it.

For those pages created from a Super Genie library object, if the parameter [CtDraw.RSC]-AllowEditSuperGeniePage is set to 0 a message will be displayed that will help prevent you from editing the Super Genie page directly. Instead you will need to edit the Super Genie Library object that created that page.See Maintaining a Super Genie Library Object. If the parameter [CtDraw.RSC]AllowEditSuperGeniePage is set to 1 a message will be displayed that will ask for confirmation before allowing you to edit the page directly.

> **Note**:Changes made to a page (created from a Super Genie library object) will be over-ridden when you ' Update Pages'.

To open an existing Super Genie Page:

1. Click the **Open** tool or choose **File** | **Open**.
2. Select the **Pages** tab.
3. Select the **Project**the page belongs to.
4. Select the **Super Genie**.
5. Click **OK**.

To delete a Super Genie page from the project:

1. Click the **Open** tool or choose **File** | **Open**.
2. Select the **Pages** tab.
3. Select the **Project**the page belongs to.
4. Select the **Super Genie** and click **Delete**.
5. A message will display. Click Yes to delete the page from the project and its records. Click No to delete the page from the project only. Click Cancel to cancel the operation.

> **Note**: If you delete a Super Genie page that is in use,the object configured to call the page at runtime will become inoperable.

See Also
Using Constants and Arrays with Super Genies

Maintaining a Super Genie Library Object

You can open an existing Super Genie library object to edit it.If you modify a Super Genie library object when the project is running in the background, you will be prompted to 'Update Pages' to see the changes in the runtime project. If a runtime page containing the Super Genie is displayed when the change is made, it will not be updated until you exit then re-display it.

To open an existing Super Genie library object:

1. Click the **Open** tool or choose **File | Open**.
2. Select the **Super Genie** tab.
3. Select the **Project**the **Library** in which the Super Genie is stored.
4. Select the **Super Genie**.
5. Click **OK**.

> **Note**: If you delete a Super Genie page that is in use,the object configured to call the page at runtime will become inoperable.

To delete a Super Genie library from the project:

1. Click the **Open** tool or choose **File | Open**.
2. Select the **Super Genie** tab.
3. Select the **Project**and the library the Super Genie belongs to.
4. Select the **Super Genie** and click **Delete**.
5. A message will display. Click OK to delete the library object.

Deleting a Super Genie library object that has been attached to a Genie will not cause that Genie to become inoperable at runtime.

See Also
Using Constants and Arrays with Super Genies

Using Constants and Arrays with Super Genies

You can use Constants and Arrays with Super Genies.

See Also
Constants
Arrays

Constants

The ability to pass constants into Super Genies is restricted in that, the constant association can only be where you can enter a normal Cicode tag - keyboard command, symbol address field etc. The following types of constants are supported: STRING, INTEGER, DIGITAL, REAL, and LONG.

To pass a constant you need to format the argument in the Ass function to include a single quote on either side. For example, to pass the constant data **1.2345** into a Super Genie, you would call the Ass function like this:

```
Ass(hWin, sArg, "'1.2345'");
```

To pass a variable tag, you don't need the single quotes. For example, to pass variable tag **TAG1** into a Super Genie, you would call the `Ass` function as follows;

```
Ass(hWin, sArg, "TAG1");
```

See Also

Arrays

Arrays

Super Genies can accept array elements or entire arrays as substitution. Passing an element of an array is straightforward, and is done by reference to the element, as shown here:

```
AssPopUp("MyPopUp", "DigArray[42]");
```

To pass an entire array to a Super Genie, only the array name is used. For example:

```
AssPopUp("MyPopUp", "DigArray");
```

To pass an entire array to a Super Genie, configure it to accept the array instead of a single value. Use the following syntax for the Super Genie substitution string:

```
?<Data Type><Substitution String>? [<element>]
```

Only arrays of data type DIGITAL, INT, REAL, and LONG are supported.

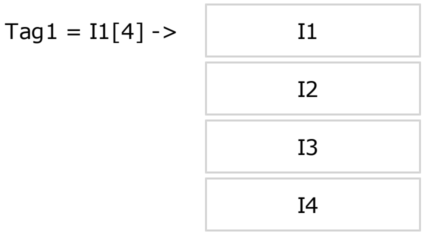
See Also

Using array offsets

Using array offsets

Using array offsets with a tag that is an array

If you have a tag *Tag1* defined as an array of 4 elements. For example, when using the GENERIC protocol the tag has an address I1[4]:



Tag1 represents the registers I1, I2, I3 and I4. When Tag1 is used with the Ass() Cicode function it behaves as follows:

- If you associate a value using Ass(hWin, "X", "Tag1", 0) then the substitution string ?X?[0] gives the value in I1.
- If you associate a value using Ass(hWin, "X", "Tag1", 0) then the substitution string ?X?[2] gives the value in I3.
- If you associate a value using Ass(hWin, "X", "Tag1[0]", 0) then the substitution string ?X?[2] gives the value in I3.
- If you associate a value using Ass(hWin, "X", "Tag1[1]", 0) then the substitution string ?X?[2] gives the value in I4. This is because the two offsets are added together to determine the final offset within the array variable.
- If you associate a value using Ass(hWin, "X", "Tag1[2]", 0) then the substitution string ?X?[2] gives the error #ERR. This is because the sum of the two array offsets gives a position (Tag1[4]) that is outside the bounds of the array.

Using array offsets with a tag that is not an array

If you have a tag *Tag2* defined as a single value. For example, when using the GENERIC protocol the tag has an address I1:

When Tag2 is used with the Ass() Cicode function it behaves as follows:

- If you associate a value using Ass(hWin, "X", "Tag2", 0) then the substitution string ?X?[0] gives the value in I1. This is because a non-array tag is equivalent to an array with one element.
- If you associate a value using Ass(hWin, "X", "Tag2[0]", 0) then the substitution string ?X?[0] gives the value in I1. This is because a non-array tag is equivalent to an array with one element.

- If you associate a value using Ass(hWin, "X", "Tag2", 0) then the substitution string ?X?[2] gives the error #ERR. This is because a non-zero offset cannot be applied to a non-array tag.
- If you associate a value using Ass(hWin, "X", "Tag2[0]", 0) then the substitution string ?X?[2] gives the error #ERR. This is because a non-zero offset cannot be applied to a non-array tag.
- If you associate a value using Ass(hWin, "X", "Tag2[1]", 0) then the substitution string ?X?[2] gives the error #ERR. This is because a non-zero offset cannot be applied to a non-array tag.
- If you associate a value using Ass(hWin, "X", "Tag2[2]", 0) then the substitution string ?X?[2] gives the error #ERR. This is because a non-zero offset cannot be applied to a non-array tag.

Nesting Super Genies

CitectSCADA allows you to nest Super Genies. Nesting refers to where one Super Genie is embedded in another. For this to work, the embedded Genie (for the embedded Super Genie) need to use `AssChain` functions instead of `Ass` functions.

See Also

Super Genie areas

Super Genie areas

When you display a Super Genie, the area of the Super Genie is inherited from its parent. For example, if the parent page is in area 1, when you display a Super Genie it will also be area 1. This allows you to call the same Super Genie from different pages in different areas.

The inherited area may be avoided by defining the Super Genie to have a specific area. Then, every instance of the Super Genie will have the same area, no matter which area its parent is from. Super Genies will only inherit areas if their area is blank.

See Also

Super Genie Library Objects and Associations

When you define a Super Genie, you are actually creating a Super Genie template, similar to a page template. When a Genie calls the Super Genie library object, this template is used to create a new Super Genie page. At this point, any Associations saved with the template are copied across to the Super Genie page. However, if subsequent changes are made to the Associations of the template, the Associations of the Super Genie page are not updated.

After the Super Genie template associations have been modified the Super Genie page created from the template needs to also be updated. If it is not updated then your page may display out-of-date e associations.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

To update the Super Genie page associations with changes made to the template:

- Find and delete the Super Genie page (remember it may be prefixed with an exclamation mark (!))
- Re-attach the Super Genie page to the Genie. On attaching it this will create a new Super Genie page that has the updated associations.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Using structured tags with Super Genies

Super Genies do not support direct concatenation of the Super Genie tag with other information (as do Genies). For example, ?INT 1?_PV is not valid and will generate a compiler error. However, you can concatenate the tag using a Cicode expression. You need to use a unique Super Genie variable for each real tag, and concatenate the tag with the Ass Cicode function. For example, if you have defined a loop controller with three bar graphs (created using the fill property in a rectangle) to display the tags DEV1_PV, DEV1_SP and DEV1_OP, you can configure a Super Genie as follows:

Each rectangle has a separate vGenie tag:

Level expression	?INT 1?
Level expression	?INT 2?
Level expression	?INT 3?

If you do not use structured tags, you can call the Ass function for the above Super Genie as follows:

```
AssPage("PageName", "DEV1_PV", "DEV1_SP", "DEV1_OP");
```

To concatenate information for the Super Genie, you could also write your own Cicode function, as follows:

```
FUNCTION
AssMine(STRING sPage, STRING sTag)
        AssPage(sPage, sTag + "_PV", sTag + "_SP", sTag + "_OP");
END
```

With this function, you can call your AssMine() function (for example, from a command button), and pass a single tag (DEV1), as follows:

```
AssMine("PageName", "DEV1");
```

Writing your own Cicode function to call a Super Genie provides extra flexibility; however, you can also use a Super Genie (for example, from a button command) to call the Ass function, as follows:

Execute command	AssPage("%Page%", "%tag%_PV", "%tag%_SP", "%tag%_OP");

When you use the above Super Genie, you only enter the page name and tag once.

You need to pass the tag name (by enclosing it in quotation marks) to the Super Genie functions. You cannot pass the tag values. For example, if you pass **%tag%_SP** (no quotes), the value of the variable and not the tag name is passed to the Super Genie, and the association will not succeed, and a runtime error may result.

See Also

Using structured tags with Genies

Hiding Graphics Objects

You can configure a graphics object so that if the variable tag specified for the object is not defined in the tag database at compile time, the object does not display on a graphics page.

The expression entered in the **Hidden When**field of an object's property is used to determine if the object will display. The expression evaluates to either TRUE or FALSE and the object is hidden when the expression is TRUE.

You define the variable tag and conditions under which the object is hidden by entering an IFDEF statement into the **Hidden When** field when you configure the object. The IFDEF statement is evaluated by the compiler and the value of the resulting expression or variable tag will determine whether or not the object is hidden.

This can significantly reduce the number of necessary genies, as the configuration engineer does not need to generate several smaller genies to cater to operations driven by a slightly different range of tags.

See IFDEF macro for more information on using the IFDEF macro for hiding graphics objects. For more information on the IFDEF macro in general, see IFDEF.

IFDEF macro

The IFDEF statement consists of three arguments:

```
IFDEF (<"Tag name">, <Hidden When value if tag defined>, <Hidden When value if tag
undefined>)
```

The first includes a variable tag name. If the variable tag is defined in the tag database at project compilation, the IFDEF statement is replaced in the **Hidden When** field by the second argument. If the variable tag is undefined, the **Hidden When** field will contain the third argument.

Example 1

```
IFDEF("Bit_1", 0, 1)
```

In the above example, if Bit_1 is defined in the tag database, the value in the **Hidden When** field will be 0. If Bit_1 is undefined, the value will be 1. Since the object is hidden when the value is TRUE, the object will be hidden when BIT_1 is undefined (i.e. when the **Hidden When** field contains 1).

Example 2

```
IFDEF("Bit_2",,"1")
```

If the second argument is omitted, as in Example 2, the variable tag specified in the first argument is used. If Bit_2 is defined, therefore, the **Hidden When** field will contain Bit_2. The value of the variable tag Bit_2 is then used to determine if the object is hidden. A non-zero value will equate to TRUE, causing the object to be hidden.

If Bit_2 is undefined, the Hidden When expression evaluates to 1 (TRUE) and the object is hidden.

To enter an IFDEF statement in the Hidden When Field:

1. Double-click the graphics object for which you want to edit the field.
2. Select the **Appearance** tab.
3. Click the **Hidden When** field and enter the IFDEF statement.
4. Click **OK**.

Chapter: 33 Working with Multi-Language Projects

CitectSCADA's language switching facility allows you to use one language to configure a project, and another for runtime text items such as alarm descriptions, button text, keyboard/alarm logs, graphic text, Cicode strings, and so on. You can also dynamically change languages during runtime.

Notes:

- There is no automatic translation, you need to arrange translation of native language strings into the desired local languages and insert these strings into the localized language database.
- After translation review the foreign language interface and verify that translated strings fit in their graphics elements correctly.

For example, if your native language is English, you could enter an English alarm description when configuring the project, but specify to display it in the French or German (or any other language) equivalent at runtime. You can specify the language you want before running the project, or change it dynamically at runtime (using the `SetLanguage()` function) without affecting any of the project's normal operations.

CitectSCADA distinguishes between the *native* language (that is, the language of the developer), and the *local* language (the language of the end user). Language changes are achieved by using a language database, which has a field for native text, and a field for the translated local text. When the project is run, native text is replaced with the equivalent local text.

Alarm and keyboard logs can be processed in both the native and the local language, enabling native and local users to read historical logs. The data can be logged to the same device, or to separate devices.

See Also

Changing Languages

Changing Languages

This section describes how to change the language used by CitectSCADA.

See Also

Marking text for language change
Language databases
Multiple languages

Multiple projects
Changing languages at runtime
Logging data in different languages
ASCII and ANSI character sets
OEM character sets

Marking text for language change

During project development, you need to mark any text you want change to another language at runtime with a language change indicator, like this:

```
@( Native Text [,Width [,Justify]])
```

where Native Text is the identifying text to be displayed when configuring. This text will be replaced by the local equivalent at runtime. Be aware that the brackets are necessary as they specify the extent of the native language text; `Width` and `Justify` are optional (indicated by the square brackets).

For example, if English is the native language, you could enter the following alarm description:

Alarm Desc	@(Motor Inoperative)

This indicator serves two purposes: It flags the text as native, and tells CitectSCADA to change the text from native to local at runtime.

By default, the text that you enter here can be in any combination of upper- and lowercase. In other words, *Motor Inoperative* will be considered the same string as *motor inoperative* or *MOTOR Inoperative*, and they will have the same local language translation. Case-sensitivity can be introduced by setting the `[Language]CaseSensitive` parameter to 1.

`Width` can be assigned any value from 0 to 254. If the local text is longer than specified, it is truncated and left-justified. If a width is not specified, the field is the length of the local text and the text left-justified.

`Justify` specifies the text justification and can only be used with `Width`. `Justify` can be one of the following values:

- **l** or **L** - Left
- **r** or **R** - Right
- **c** or **C** - Center
- **n** or **N** - None

For example, to limit the local text in the previous case to 20 characters with right justification:

Alarm Desc	@(Motor Inoperative, 20, R)

Characters that are normally part of the formatting - **@ , ()** - can also be used within the native text. To do this, place a caret (^) character before them. For example, to include a comma without introducing a formatting error:

Alarm Desc	@(Motor Inoperative^, thermal overload, 20, R)

> **Note:** The caret (^) character does appear at runtime or in the language database.

See Also

Language databases

Language databases

When the project is compiled, CitectSCADA creates a language database (dBASE III format), consisting of two fields: NATIVE and LOCAL. Text marked with a language change indicator is automatically entered in the **NATIVE** field. You can then open the database and enter the translated text in the **LOCAL** field.

For example:

NATIVE	**LOCAL**
Line Disconnected Alarm at Line Speed {LineSpeed1}	<Translation of *Line Disconnected Alarm at Line Speed {LineSpeed1}*>
Main Menu page	<Translation of *Main Menu page*>
Conveyor Belt Trip	<Translation of *Conveyor Belt Trip*>

When the project is run, the translation of *Line Disconnected Alarm at Line Speed {LineSpeed1}* will display in place of the English text, the translation of *Main Menu page* will display in place of the English text etc.

For the language change to occur automatically when the project is run, you need to specify the language database to use *before* running the project by using the `[Language]LocalLanguage` parameter. Otherwise, you can change the language yourself at runtime using the `SetLanguage()` function.

If you do not enter a local equivalent of the native text string, the native text is displayed by default. You can specify to display "#MESS", instead of the native text by setting the `[Language]DisplayError` parameter to 1 (one); the default is 0 (zero).

For single-byte languages (such as French), the database can be edited using Microsoft Excel; for double-byte languages (such as Chinese), use Visual FoxPro.

By default, the language database created by the compile is called **English.dbf** (this can be changed using the `[Language]LocalLanguage` parameter). It is saved to the project directory. Once the database is created, it us updated when you compile. Text marked since the last compile is appended to the database; the rest of the database is unchanged.

See Also

Multiple languages

Multiple languages

> **Note:** To use characters for Baltic, Central European, Cyrillic, Greek, Turkish, and Asian languages, or right-to-left languages (Arabic, Hebrew, Farsi, and Urdu) the operating system needs to have the corresponding language version of Windows, or have installed system support for that language.

Each local language need to have its own language database, so that it can be displayed in place of a specified native language at runtime. Also, it needs to be set as the local language using the `[Language]LocalLanguage` parameter. With this parameter set *before* you compile, CitectSCADA automatically creates/updates the relevant language database.

For example, to display text in French at runtime, set the `[Language]LocalLanguage` parameter to French, flag necessary native text in the project with **@()**, and compile. After compiling, look in the project directory for **French.dbf**, open it, enter the necessary French translations in the **Local** field, and save the database. When the project is run, marked native text will be replaced by the appropriate French text.

Because you can have any number of databases, you can use as many different languages as you like.

When you compile, text marked with a language change indicator is entered in the **Native** field of whatever database is set as the local language using the `[Language]LocalLanguage` parameter. Therefore, know what database is set *before* you compile.

Also, if you have several language databases with the same native language, remember that newly marked text is only appended to the *current* local language database (as specified by the `[Language]LocalLanguage` parameter). To add this text to other databases with the same native language, change the `[Language]LocalLanguage` parameter, update pages, and recompile for each database. Remember that for each database, only relevant changes made since the last compile are added.

See Also

Multiple projects

Multiple projects

A language database can contain entries which are not actually included in a project. This means that a single language database can be developed that is applicable to many projects.

See Also

Changing languages at runtime

Changing languages at runtime

The language of runtime display items such as alarm descriptions, button text, keyboard/alarm logs, graphic text, Cicode strings and so on can be changed dynamically at runtime using the `SetLanguage()` function. Normal operations of the project continue unaffected.

> **Note:** Forms do not automatically update when the language is changed using the `SetLanguage()` function. They need to be closed and reopened for the change to take place.

Local translations that are missing from the specified language database are replaced by the native equivalent. You can specify to display "#MESS", instead of the native text, by setting the `[Language]DisplayError` parameter to 1 (one); the default is 0 (zero).

See Also

Logging data in different languages

Logging data in different languages

Alarm and keyboard logs can be processed in both the native and the local language. This means that both native and local users can read the historical logs. The logs can employ the same device, or separate devices.

Logs in the local language are produced using the standard field names. For example, if `{NAME} {DESC} {COMMENT}` is entered in the format field of an alarm category, the alarm name, description and comment of alarms in that category will be logged in the local language.

All fields which support the automatic language change facility can also be logged in the native language. To do so, just precede the field name with **NATIVE**. For example, to log the name, description and comment of a category of alarms, enter `{NATIVE_NAME} {NATIVE_DESCRIPTION} {NATIVE_COMMENT}` in the format field for that category.

To log both native and local to the same device, just enter the standard fields, and the native fields together in the format field. To log them to different devices, use a Group of two devices, and enter the local fields as the format for one, and the native fields as the format for the other.

See Also

ASCII and ANSI character sets

ASCII and ANSI character sets

Each screen character is defined by a code (number). Operating systems and applications need to know these codes to attach meaning to individual characters. A character set provides a code for every character. For your operating system/application to interpret a character correctly, you need to use the correct character set.

Note: Character sets are distinct from fonts. A font defines the visual/appearance properties of a character, not its meaning.

ASCII (American Standard Code for Information Interchange) is a widely adopted 7-bit code specifying the basic alphanumeric character set of the English language. For example, the character capital "A" has the ASCII value of 65, the character lowercase "a" a value of 97.

The ASCII character set contains 96 characters and is commonly used as a standard for protocols and files.

Much of CitectSCADA uses ANSI (American National Standards Institute) character sets. ANSI character sets are language-based, with each different language version of Windows (French, Korean and so on) requiring a specific ANSI character set. Codes 32 to 127 contain standard ASCII characters.

Windows uses Unicode, but still supports ANSI character sets. Several of CitectSCADA's utilities have been created for Unicode, for instance Process Analyst. Unicode accommodates known character sets by having one 16-bit (worldwide) character encoding standard.

See Also

OEM character sets

OEM character sets

OEM character sets are those which are used by MS-DOS or Console applications (they are operating system dependent). Most OEM character sets do not match the ANSI character sets. For example, line drawing characters commonly used in MS-DOS character sets were replaced with language characters in ANSI character sets.

As explained below, when building multiple language projects give adequate consideration to the role that ANSI and OEM character sets play, in the way the language strings are stored and interpreted.

Language configuration information is stored in dBASE files (a database standard defined primarily for MS-DOS applications) where string information is customarily stored as OEM characters. When using a Windows application (such as Excel) to edit dBASE files, the characters on screen are in the ANSI character set. When you save this information to the dBASE file, Excel converts it to an OEM equivalent. For this conversion to work correctly the OEM character set needs to be compatible with the ANSI character set used in Excel. For example, if you have prepared strings for a project in Russian (using Excel), the OEM character set needs to support the Russian (Cyrillic) character set. The OEM character set used by Windows is primarily determined by your system setup and cannot easily be changed. This presents a challenge for multi-language projects.

For example, consider a project intended to support Russian, French, and English. Excel is used to prepare the language dBASE files. When saving information from Excel, it is translated from the respective ANSI character sets to the OEM equivalent. To display this information, CitectSCADA will need to convert it from OEM back to ANSI. However, Russian requires a Cyrillic OEM character set and French and English requires a Latin OEM character set. Because Windows can only use one OEM character set at a time (which cannot be changed dynamically), only one of the three project languages can be correctly supported during any given session.

The only way to support multiple languages with differing character sets within one CitectSCADA project, is to verify that the language information you store in dBASE files is stored in the ANSI (not the OEM) format. Further, the [CtEdit]ANSItoOEM parameter needs to be set to 0 (zero) to prevent a conversion from occurring automatically. The challenge for the developer in preparing the project is in saving this information in the first place, because most applications store the language information in OEM format.

Note: A multi-language project is included in the samples directory on your installation CD. This project allows you to enter information into the language dBASE files in ANSI format.

Chapter: 34 Working with Multiple Monitors

Multiple monitors are now supported in the core product instead of within a set of templates. CitectSCADA runtime will display a new window on each of the multiple monitors at start-up. Once the windows are displayed, the user can use the standard page navigation functions to change pages on the individual window. The built-in page navigation functions work under the multi-monitor environment. Several built-in Cicode functions are made available for the user to query information for each of the monitors installed on the machine for further customization.

Multiple monitors are configured by CitectSCADA at startup as follows:

- Reads [MultiMonitors] Monitors to determine which monitors will be shown with CitectSCADA windows.
- Enumerates through the specified monitors and show a new CitectSCADA window at the origin of each of the monitors. The page to be displayed on each monitor will be determined by the parameter [MultiMonitors] StartupPage<n>.
- Once a CitectSCADA window is displayed on each of the specified monitors, you can use the standard page navigation functions to display a page on each of the windows independently.

If this conflicts with your own implementation of multiple monitors support, for example if you are using CSV templates, you can disable the core support by setting parameter [MultiMonitors] DisableAutoStart to 1.

See Also

Configuring Startup Pages for Multiple Monitors
MultiMonitors Parameters

Configuring Startup Pages for Multiple Monitors

You can display several graphics pages simultaneously across multiple computer screens. If your hardware can run multiple monitors off a single computer, you can display a different page on up to eight screens at a time.

To successfully run your project across multiple monitors, adjust the following parameters:

- `[MultiMonitors]Monitors`. Adjust this parameter to indicate how many monitors your project will be running on.

- **`[MultiMonitors]StartupPage`***n*. This parameter determines which page will appear on each monitor at startup. You have to duplicate this parameter for each monitor. For example, `StartupPage1` determines the page that appears on the first monitor at startup, `StartupPage2` determines what appears on the second monitor, and so on. If you do not specify a startup page for a particular monitor, the page specified in parameter [Page]Startup will be displayed.

Many types of hardware support multiple-monitor display. CitectSCADA has been tested on a PC with multiple outputs from a single video card, but not on other architectures. The ability to display project pages on multiple screens might be affected by hardware architectures that have not been tested.

See Also

MultiMonitors Parameters

Chapter: 35 Using OPC Server DA2.0

CitectSCADA OPC Server allows you to access data available in the CitectSCADA runtime environment through any OPC Client application (v1.0 or v2.0).

This section contains:

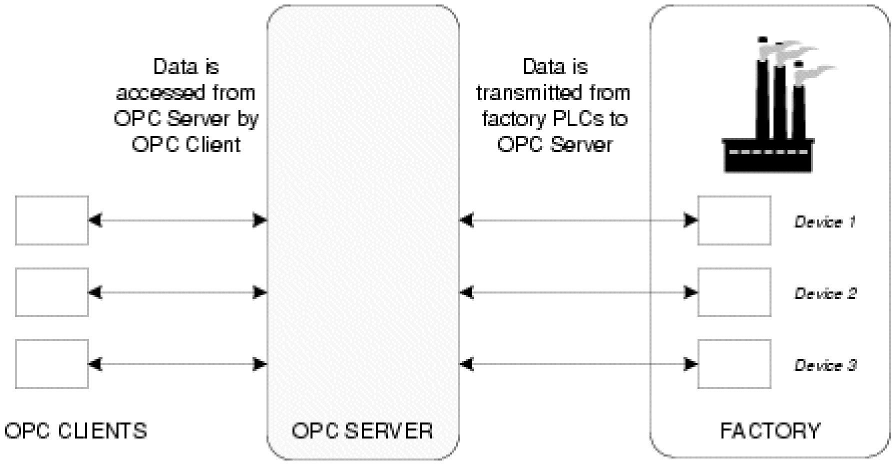
- OPC Overview
- CitectSCADA OPC Server
- CitectSCADA OPC Server Installation
- Configuring Remote Access to the CitectSCADA OPC Server
- Troubleshooting

OPC Overview

OPC (OLE for Process Control) is a set of open standards communication specifications intended for the industrial automation industry. OPC provides a common way for applications to access data from sources such as PLCs and databases.

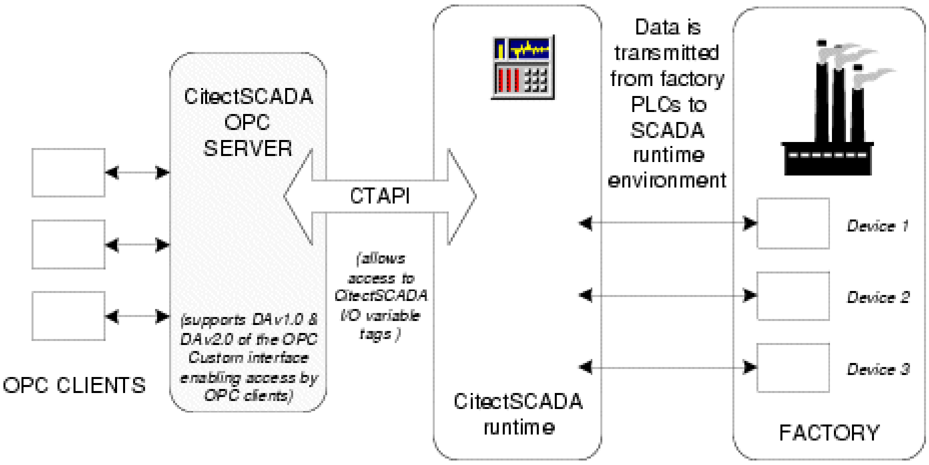
The specifications are maintained by the OPC Foundation and are reviewed constantly to verify that they meet the needs of industry. The OPC Foundation's goal is to promote interoperability (open connectivity) through the use of open standards. CitectSCADA runtime implements vDA2.05 of the OPC specifications. For details on the OPC specifications, navigate to www.opcfoundation.org.

When an OPC Server is used to collect data from physical equipment in a plant, any OPC Client can access data directly from the OPC Server, as shown below:


Data is accessed from OPC Server by OPC Client
Data is transmitted from factory PLCs to OPC Server
Device 1
Device 2
Device 3
OPC CLIENTS
OPC SERVER
FACTORY

CitectSCADA OPC Server

When CitectSCADA is used to monitor and control a plant, data from PLCs is collected and displayed in the CitectSCADA runtime environment. OPC Clients can access device and tag information from CitectSCADA through the OPC interface to the OPC Server, which in turn interacts with the CtAPI interface to the CitectSCADA Runtime, as shown here:


CitectSCADA OPC SERVER
(supports DAv1.0 & DAv2.0 of the OPC Custom interface enabling access by OPC clients)
OPC CLIENTS
CTAPI
(allows access to CitectSCADA I/O variable tags)
CitectSCADA runtime
Data is transmitted from factory PLCs to SCADA runtime environment
Device 1
Device 2
Device 3
FACTORY

The CitectSCADA OPC Server can be invoked from an OPC Client on the same machine or on a different machine. When connecting to an OPC Server running on the same machine as the client, use the Citect.OPC connection option. When connecting to an OPC Server running on a remote machine, use the Citect.OPCRemote option.

The OPC Server starts automatically when you start the CitectSCADA runtime environment and does not require configuring.

CitectSCADA OPC Server Installation

During CitectSCADA installation:

- The CitectSCADA OPC Server is installed.
- The OPC Core components are installed.
- The registry settings for the workstation are configured to support the OPC Server. Set out in the tables below are the registry settings for:
    - The In-Process OPC Server - Citect.OPC
    - The Local/Remote OPC Server - Citect.OPCRemote

In-Process OPC Server

In-Proc OPC Server		
Application ID (*AppID*)	Citect.OPC	
Class ID (*CLSID*)	{BA198B61-32E3-11d1-A1B5-00805F35623C}	
Binary file	CtOpc32.dll	
Interfaces	*DA 1.0 Custom Interface:*	
	OPCServer:	IOPCServer IOPCBrowseServerAddressSpace (optional)
	OPCGroup:	IOPCItemMgt IOPCGroupStateMgt IOPCSynchIO IOPCASynchIO IDataObject
	DA 2.0 Custom Interface:	
	OPCServer:	IConnectionPointContainer IOPCItemProperties
	OPCGroup:	IOPCAsyncIO2 IConnectionPointContainer
System Requirements	2000, XP and Windows Server 2003	
Server Organization	Flat (OPC_NS_FLAT)	
Implemented Categories	*OPC DA Servers v1.0*	{63D5F430-CFE4-11D1-B2C8-0060083BA1FB}
	OPC DA Servers v2.0	{63D5F432-CFE4-11D1-B2C8-0060083BA1FB}

Local/Remote OPC Server

Local / Remote OPC Server		
Application ID (*AppID*)	Citect.OPCRemote	
Class ID (*CLSID*)	{ BA198B62-32E3-11d1-A1B5-00805F35623C}	
Binary file	CtOpc32.exe	
Interfaces	*DA 1.0 Custom Interface:*	
	OPCServer:	IOPCServer IOPCBrowseServerAddressSpace (optional)
	OPCGroup:	IOPCItemMgt IOPCGroupStateMgt IOPCSynchIO IOPCASynchIO IDataObject
	DA 2.0 Custom Interface:	
	OPCServer:	IOPCCommon IConnectionPointContainer IOPCItemProperties
	OPCGroup:	IOPCAsyncIO2 IConnectionPointContainer
System Requirements	2000, XP and Windows Server 2003	
Server Organization	Flat (OPC_NS_FLAT)	

Note: In the OPC architecture, each variety of server is allocated a unique identifier, known as a Class ID. OPC Clients use these 128-bit numbers, frequently displayed in the form 6B29FC40-CA47-1067-B31D-00DD010662DA, to access the particular vendor's OPC Server. To make it easier for users, these numbers are often referred to by a more manageable string identifier called a *ProgID (Program Identifier)*. These identifiers usually take the form **Vendor.Application** with an optional version number appended; for example, the CitectSCADA OPC Server ProgID is **Citect.OPC**.

Once successfully installed, the CitectSCADA OPC Server is ready for use. The OPC interface that OPC Clients access starts automatically when you start the CitectSCADA runtime environment. Further configuration is only necessary where the CitectSCADA OPC Server will be accessed by a remote OPC Client. See Configuring Remote Access to the CitectSCADA OPC Server for more information.

Configuring Remote Access to the OPC Server

You can use any OPC Client to access CitectSCADA OPC Server. While each OPC Client has its own interface and therefore operates differently, the same steps needs to be performed in order to access data exposed through the OPC Server:

- Configure the OPC Server
- Configure the OPC Client
- Create a data group
- Add data items to the group

For details on how to operate your OPC Client, refer to the product's documentation.

Note: No configuration is needed if the client and server are on the same machine.

Configure the OPC Server

OPC operates using the Microsoft Distributed COM (DCOM) architecture. This means that if an OPC Client is connecting to a remote OPC Server, you need to configure DCOM security settings on both the client and server machines.

Note: You need to assign your users sufficient Citect and DCOM security privileges or the OPC Client will not be able to connect to the OPC Server.

The process of configuring DCOM security settings differs depending on which platform you're using:

- Windows Vista and Windows XP SP2
- Windows 2000

- Windows Server 2003

Windows Vista and Windows XP SP2

This section describes how to configure the OPC Server on Windows Vista and Windows XP Service Pack 2.

> **Note:** After configuring your OPC Server, you need to restart the server so that your settings can take effect.

To configure the OPC Server:

1. In Windows Control Panel, double-click **Security Center**, and then click **Windows Firewall**.
2. Select the **Exceptions** tab, and then click **Add Program**.
3. Locate the OPC Servers and clients you want to add as exceptions. You need to add the **Microsoft Management Console** (used by the DCOM configuration utility, below) and the OPC utility OPCEnum.exe in the `Windows/System32` folder.
   You might have to **Browse** for other executables installed on the computer to add them as exceptions. Please be aware that only `.exe` files are added to the exceptions list. For in-process OPC Servers and clients (DLLs and OCXs) you need to add the `.exe` applications that call them to the list instead.
4. Click **OK** to save your exceptions.
   You need to now add TCP port 135 to initiate DCOM communications and allow incoming echo requests.
5. From the Windows Firewall Exceptions tab, click **Add Port**.
6. In the **Name** text box type **DCOM**.
7. In the **Port number** text box type **135** and select the **TCP** option.
8. Click **OK** to save your changes.
   You have now defined your exceptions, so you can restart your firewall.
   You are now ready to configure DCOM for your launch, activation, and access permissions.
9. Choose **Start | Run**. Type **dcomcnfg** and click **OK**.
   The **Component Services** dialog will appear.
10. In the tree pane, locate and select **Computers** under **Component Services**.
11. Right-click **My Computer** in the pane on the right, and choose **Properties** from the context menu.
12. Select the **Default Properties** tab. Make sure the settings are adjusted as follows:

Property	Setting

Enable Distributed COM on this computer	selected
Enable COM Internet Services on this computer	not selected
Default Distributed COM Communication Properties	
Default Authentication Level	Connect
Default Impersonation Level	Identify
Provide additional security for reference tracking	not selected

Click **Apply** to save your changes.

> **Note:** For some OPC servers, setting the Default Authentication Level to "Connect" does not work. If this is the case, try setting this property to "None".

13. You need to then edit the access permissions. Select the **COM Security** tab, and click **Edit Limits** in the **Access Permissions** section.
14. Use the **Access Permission** dialog to select **Anonymous Logon,** and then check **Allow** for the **Remote Access** option.
    This is needed for `OPCEnum.exe` to function correctly, and for any OPC Servers and clients that set their DCOM authentication level to None in order to allow anonymous connections. If you don't use `OPCEnum.exe`, you might not need to enable remote access for anonymous users.
15. Click **Apply** to save your changes.
    You need to then edit the launch and activation permissions.
16. Select **Edit Limits** in the **Launch and Activation Permissions** section of the Default COM Security tab.
    In the **Group or user names** list, select **Everyone** then select every check box in the **Allow** column.
17. Click **OK** to save your changes.
    Finally, you're ready to edit the default permissions for **Access** and **Launch** for each user (or group) that participates in OPC communication (for example, "OPC Users").
18. In the **Access Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.
19. In the **Launch and Activation Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.

You have now configured your OPC Server.

Note: You need to restart this machine so that your settings can take effect.

You are now ready to configure your OPC Client.

Windows 2000

This section describes how to configure the OPC Server on Windows 2000.

To configure the OPC Server:

1. Choose **Start | Run**. Type **dcomcnfg** and click **OK**.
   The **Component Services** dialog will appear.
2. In the tree pane, locate and select **Computers** under **Component Services**.
3. Right-click **My Computer** in the pane on the right, and choose **Properties** from the context menu.
4. Select the **Default Properties** tab. Make sure the settings are adjusted as follows:

Property	Setting
Enable Distributed COM on this computer	selected
Enable COM Internet Services on this computer	not selected
Default Distributed COM Communication Properties	
Default Authentication Level	Connect
Default Impersonation Level	Identify
Provide additional security for reference tracking	not selected

   Click **Apply** to save your changes.
   You need to edit the **Access Permissions** and **Launch and Activation Permissions** settings.
5. Select the **Default Security** tab, and then click **Edit Default** in the **DefaultAccess Permissions** section.
6. In the **Registry Value Permissions** dialog, select the user you want to edit permissions for and then choose **Allow Access** from the **Type of Access** menu.
   This setting is needed for `OPCEnum.exe` to function, and for some OPC Servers and clients that set their DCOM authentication level to None in order to allow anonymous connections. If you don't use `OPCEnum.exe` you might not need to enable remote access for anonymous users.
   Click **Apply** to save your changes.

Now you need to edit the default launch permissions.

7. Click **Edit Default** in the **Default Launch Permissions** section. Make sure the type of access for necessary users (including System and Interactive) is set to **Allow Launch**. Click **OK** to save your changes.
   You have now configured your OPC Server.

> **Note:** You need to restart this machine so that your settings can take effect.

You are now ready to configure your OPC Client.

Windows Server 2003

This section describes how to configure the OPC Server on Windows Server 2003.

To configure the OPC Server:

1. Choose **Start | Run**. Type **dcomcnfg** and click **OK**.
   The **Component Services** dialog will appear.
2. In the tree pane, locate and select **Computers** under **Component Services**.
3. Right-click **My Computer** in the pane on the right, and choose **Properties** from the context menu.
4. Select the **COM Security** tab, and click **Edit Limits** in the **Access Permissions** section.
5. Use the **Access Permission** dialog to select **Anonymous Logon,** and then check **Allow** for the **Remote Access** option.
   This is needed for `OPCEnum.exe` to function correctly, and for any OPC Servers and clients that set their DCOM authentication level to None in order to allow anonymous connections. If you don't use `OPCEnum.exe`, you might not need to enable remote access for anonymous users.
6. Click **Apply** to save your changes.
   You need to then edit the launch and activation permissions.
7. Select **Edit Limits** in the **Launch and Activation Permissions** section of the Default COM Security tab.
   In the **Group or user names** list, select **Everyone** then select every check box in the **Allow** column.
8. Click **OK** to save your changes.
   Finally, you're ready to edit the default permissions for **Access** and **Launch** for each user (or group) that participates in OPC communication (for example, "OPC Users").
9. In the **Access Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.

10. In the **Launch and Activation Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.
    You have now configured your OPC Server.

> **Note:** Now you need to restart this machine so that your settings can take effect.

You are now ready to configure your OPC Client.

Configure the OPC Client

You need to configure the OPC Client before it can communicate with the OPC Server.

The process of configuring DCOM security settings differs depending on which platform you're using:

- Windows Vista and Windows XP SP2
- Windows 2000
- Windows Server 2003

Windows Vista and Windows XP SP2

This section describes how to configure the OPC Client on Windows Vista and Windows XP Service Pack 2.

> **Note:** After configuring your OPC Client, you need to restart the client so that your settings can take effect.

To configure the OPC Client:

1. In Windows Control Panel, double-click **Security Center**, and then click **Windows Firewall**.
2. Select the **Exceptions** tab, and then click **Add Program**.
3. Locate the OPC Servers and clients you want to add as exceptions. You need to add the **Microsoft Management Console** (used by the DCOM configuration utility, below) and the OPC utility OPCEnum.exe in the `Windows/System32` folder.
   You might have to **Browse** for other executables installed on the computer to add them as exceptions. Please be aware that only `.exe` files are added to the exceptions list. For in-process OPC Servers and clients (DLLs and OCXs) you need to add the `.exe` applications that call them to the list instead.
4. Click **OK** to save your exceptions.
   You need to now add TCP port 135 to initiate DCOM communications and allow incoming echo requests.

5. From the Windows Firewall Exceptions tab, click **Add Port**.
6. In the **Name** text box type **DCOM**.
7. In the **Port number** text box type **135** and select the **TCP** option.
8. Click **OK** to save your changes.
   You have now defined your exceptions, so you can restart your firewall.
   You are now ready to configure DCOM for your launch, activation, and access permissions.
9. Choose **Start | Run**. Type **dcomcnfg** and click **OK**.
   The **Component Services** dialog will appear.
10. In the tree pane, locate and select **Computers** under **Component Services**.
11. Right-click **My Computer** in the pane on the right, and choose **Properties** from the context menu.
12. Select the **Default Properties** tab. Make sure the settings are adjusted as follows:

Property	**Setting**
Enable Distributed COM on this computer	selected
Enable COM Internet Services on this computer	not selected
Default Distributed COM Communication Properties	
Default Authentication Level	Connect
Default Impersonation Level	Identify
Provide additional security for reference tracking	not selected

   Click **Apply** to save your changes.

13. You need to then edit the access permissions. Select the **COM Security** tab, and click **Edit Limits** in the **Access Permissions** section.
14. Use the **Access Permission** dialog to select **Anonymous Logon,** and then check **Allow** for the **Remote Access** option.
    This is needed for `OPCEnum.exe` to function correctly, and for any OPC Servers and clients that set their DCOM authentication level to None in order to allow anonymous connections. If you don't use `OPCEnum.exe`, you might not need to enable remote access for anonymous users.
15. Click **Apply** to save your changes.
    You need to then edit the launch and activation permissions.
16. Select **Edit Limits** in the **Launch and Activation Permissions** section of the Default COM Security tab.

In the **Group or user names** list, select **Everyone** then select every check box in the **Allow** column.

17. Click **OK** to save your changes.
    Finally, you're ready to edit the default permissions for **Access** and **Launch** for each user (or group) that participates in OPC communication (for example, "OPC Users").
18. In the **Access Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.
19. In the **Launch and Activation Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.
    Some older OPC Clients cannot locate CitectSCADA OPC Server on a client machine. If this is the case, copy the `ctopc.reg` file located in the `bin` directory to the client machine. Double-click the file to add OPC Server-specific data to the registry.
    You have now configured your OPC Client.

Note: You need to restart this machine so that your settings can take effect.

Windows 2000

This section describes how to configure the OPC Client on Windows 2000.

To configure the OPC Client:

1. Choose **Start | Run**. Type **dcomcnfg** and click **OK**.
   The **Component Services** dialog will appear.
2. In the tree pane, locate and select **Computers** under **Component Services**.
3. Right-click **My Computer** in the pane on the right, and choose **Properties** from the context menu.
4. Select the **Default Properties** tab. Make sure the settings are adjusted as follows:

Property	Setting
Enable Distributed COM on this computer	selected
Enable COM Internet Services on this computer	not selected
Default Distributed COM Communication Properties	
Default Authentication Level	Connect
Default Impersonation Level	Identify

Provide additional security for reference tracking	not selected

Click **Apply** to save your changes.

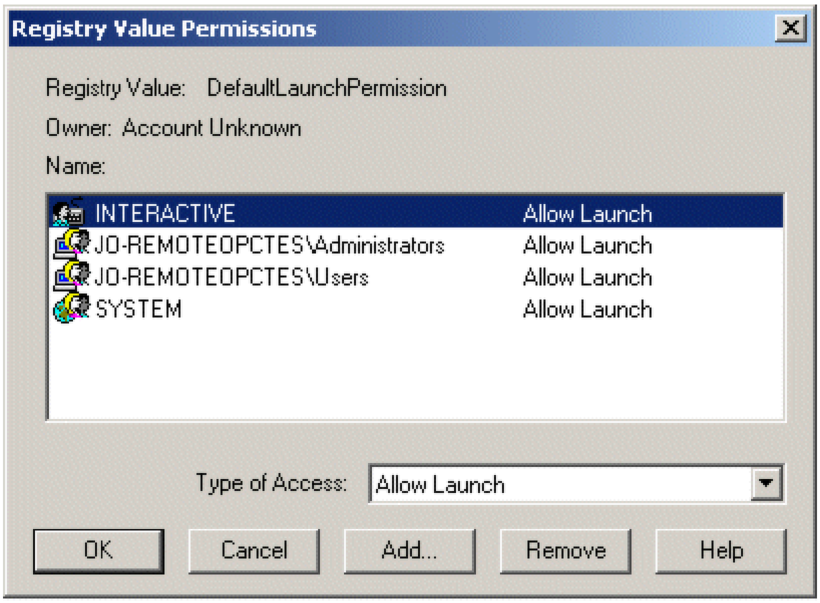
You need to edit the **Access Permissions** and **Launch and Activation Permissions** settings.

5. Select the **Default Security** tab, and then click **Edit Default** in the **DefaultAccess Permissions** section.
6. In the **Registry Value Permissions** dialog, select the user you want to edit permissions for and then choose **Allow Access** from the **Type of Access** menu.

   This setting is needed for `OPCEnum.exe` to function, and for some OPC Servers and clients that set their DCOM authentication level to None in order to allow anonymous connections. If you don't use `OPCEnum.exe` you might not need to enable remote access for anonymous users.

   Click **Apply** to save your changes.

   Now you need to edit the default launch permissions.
7. Click **Edit Default** in the **Default Launch Permissions** section. Make sure the type of access for necessary users (including System and Interactive) is set to **Allow Launch**.



Registry Value: DefaultLaunchPermission

Owner: Account Unknown

Name:

INTERACTIVE	Allow Launch
JO-REMOTEOPCTES\Administrators	Allow Launch
JO-REMOTEOPCTES\Users	Allow Launch
SYSTEM	Allow Launch

Type of Access: Allow Launch

Click **OK** to save your changes.

You have now configured your OPC Client.

Some older OPC Clients cannot locate CitectSCADA OPC Server on a client machine. If this is the case, copy the `ctopc.reg` file located in the `bin` directory to the client machine. Double-click the file to add OPC Server-specific data to the registry.

Note: You need to restart this machine so that your settings can take effect.

Windows Server 2003

This section describes how to configure the OPC Client on Windows Server 2003.

To configure the OPC Client:

1. Choose **Start | Run**. Type **dcomcnfg** and click **OK**.
   The **Component Services** dialog will appear.
2. In the tree pane, locate and select **Computers** under **Component Services**.
3. Right-click **My Computer** in the pane on the right, and choose **Properties** from the context menu.
4. Select the **COM Security** tab, and click **Edit Limits** in the **Access Permissions** section.
5. Use the **Access Permission** dialog to select **Anonymous Logon,** and then check **Allow** for the **Remote Access** option.
   This is needed for `OPCEnum.exe` to function correctly, and for any OPC Servers and clients that set their DCOM authentication level to None in order to allow anonymous connections. If you don't use `OPCEnum.exe`, you might not need to enable remote access for anonymous users.
6. Click **Apply** to save your changes.
   You need to then edit the launch and activation permissions.
7. Select **Edit Limits** in the **Launch and Activation Permissions** section of the Default COM Security tab.
   In the **Group or user names** list, select **Everyone** then select every the check boxe in the **Allow** column.
8. Click **OK** to save your changes.
   Finally, you're ready to edit the default permissions for **Access** and **Launch** for each user (or group) that participates in OPC communication (for example, "OPC Users").
9. In the **Access Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.
10. In the **Launch and Activation Permissions** section of the COM Security tab, click **Edit Default**. Select the necessary users/groups and allow **Local Access** and **Remote Access**. Click **OK** to save your changes.
    You have now configured your OPC Client.
    Some older OPC Clients cannot locate CitectSCADA OPC Server on a client machine. If this is the case, copy the `ctopc.reg` file located in the `bin` directory to the client machine. Double-click the file to add OPC Server-specific data to the registry.

> **Note:** Now you need to restart this machine so that your settings can take effect.

Create a data group

Before you can access data from an OPC Server, you need to define the data group to which it belongs. A data group is a way to organize data retrieved from the OPC Server into logical selections of related data.

Note: OPC Clients usually allow properties to be set on a per group basis, such as the rate at which the data is collected. therefore consider arranging and naming your groups according to these requirements.

As this procedure is OPC Client-specific, refer to the documentation accompanying your OPC Client.

Add data items to the group

Within each group you can assign one or more data items. Each variable tag defined in CitectSCADA runtime is available for selection as a data item.

As this procedure is OPC Client-specific, refer to the documentation accompanying your OPC Client.

Troubleshooting

If you experience connectivity issues with your OPC Server, OPC Client, or both, refer to the CitectSCADA Knowledge Base.

Chapter: 36 Compiling and Running a Project

This section describes how to compile and run a CitectSCADA project, and the functionality available to support these processes.

See Also

Compiling a Project
Running the System
Running Your System Over the Internet
Restarting the System Online
Software Protection

Compiling a Project

The CitectSCADA compiler brings together elements of your project - the configuration databases, graphics and Cicode files - to create a runtime system.

Compilation checks the project for errors and optimizes your system for fast and efficient operation. The time necessary to compile a project depends on its size and on the speed of your computer. Typically, compiling only takes several minutes.

When the CitectSCADA compiler runs, it normally opens files in exclusive mode. In this mode only CitectSCADA has access to the files (while the compiler is running). This improves the performance of the compiler, but can also result in an incomplete compilation if two people try to compile different projects at the same time, and these projects have one or more Include projects in common. The [General] ShareFiles parameter tells the compiler to open files in shared mode. This option allows shared network users to run the compiler at the same time, but it can increase the time necessary for the compilation.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

Restart the client process if the hardware alarm "Cicode library timestamp differs" is raised after a page is opened.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Note: A hardware alarm of "Cicode library timestamp differs" will be raised if the Cicode library used by a page has a different timestamp from the one in memory. The timestamps will be different if the project has been fully recompiled, the project has been incrementally recompiled after the page has been modified, or if the project has been incrementally recompiled after any Cicode has been modified.

To compile a project:

1. Select the Project Editor.
2. Click **Compile**, or choose **File** | **Compile**.

The results will be displayed in a dialog when compilation is complete.

The Citect Compiler result dialog reports the number of errors and alerts separately to avoid confusion. A CitectSCADA project can compile successfully even though the compile may generate alerts; however, compile errors prevent a project from compiling successfully.

If there are compile errors, you need to first fix the errors, and then recompile.

CitectSCADA automatically compiles the project (if uncompiled) when you try to run it.

Post Compilation

After a project has compiled successfully you can execute an optional command, script or batch file. This offers useful functionality if you have tasks that could be automated after a successful compile. This provides an expansion point for you to add your own script or command to perform additional tasks. Examples of this would be:

- Launch an application or script to merge the language files that your project is using into a single file in the root project to aid in localization.
- For the web client deployment, launch an application or script that would package up your files into the misc.zip and active.zip files.

You can also launch an optional command, script or batch file to execute after an unsuccessful compile. Generally this would be used to create a log file to show warnings or errors generated during a compile. To use either of these functions, add the command line containing the command or script to either the [CtEdit]CompileSuccessfulCommand or [CtEdit]CompileUnsuccessfulCommand parameter in the Citect.ini file.

See Also

Incremental compilation
Debugging the compilation
Compile Error Properties
Compile Error Messages
Setting the Project Editor options
Improved Client Side Online Changes

Incremental compilation

You can compile the project incrementally. With incremental compilation, CitectSCADA only compiles the database records that were added (or changed) since the last compilation. The remainder of the project is not re-compiled.

> **Note:** Some database records are dependent on other database records. If you change a dependent record, CitectSCADA compiles the entire database.

Before you run a system on a live plant, perform a complete compilation (switch off Incremental Compile) as the compilation will be more rigorous. Similarly, when you restore a project from floppy disk, you need to perform a complete compilation the first time (switch off Incremental Compile).

To switch to Incremental Compile:

1. Select the Project Editor.
2. Choose **Tools** | **Options**.
3. Select the **Incremental Compile** check box, and then click **OK**.

See Also

Debugging the compilation

Debugging the compilation

If the compiler detects any errors during compilation, the errors are written to an error file. The compiler will notify you of any errors as it compiles, and you can opt to cancel the compilation at any stage. If there are multiple or severe errors, the compiler might automatically cancel. Once the compiler is finished, you can locate each compile error and display information on it.

The compiler does not verify the operation of your project. Just because your project compiles does not mean it will work correctly at runtime. For example, the compiler checks that the tags you use are defined correctly, and that your Cicode has acceptable syntax. But, it does not check your tags for incorrect scaling, or that your Cicode has no potential divide by zero errors.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

- Do not attempt to run your system until you have resolved errors reported during project compilation.
- Always perform a complete compilation before promoting your project to the test or live environments.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

To view compilation errors:

- Select the Project Editor and choose **File | Compile Errors**.

To get more information on an error:

1. Click **Help** at the bottom of the Compile Errors dialog box.
2. Read the Help topic associated with the error.

To locate the error (in the project):

- Click **Go To** at the bottom of the Compile Errors dialog box.

See Also

Compile Error Properties

Compilation options

The compiler has a number of options available to help simplify the process of resolving compilation issues. Typically, these options modify the output of the compile error log, making it more practical to assess specific error types and entries.

The options listed here can be adjusted via the Project Editor Options dialog, which is accessible from the Project Editor's **Tools** menu.

- **Log "tag not defined" warnings during compile**
  If you select this option, the compiler will generate a "tag not defined" warning in the error log for any tags detected that are not defined in the variable database.
  As CitectSCADA v7.20 now allows you to include undefined tags on your graphic pages, this warning may be redundant and impractical. By unchecking this option, the warnings are still included in the displayed warning count, but they are not added to the error log.
- **Log deprecated warnings during compile**
  If you select this option, the compiler will generate a warning to identify any deprecated elements it detects in a project, i.e. any functions, parameters, or Kernel commands that are no longer supported.

By unchecking this option, the warnings are still included in the displayed warning count, but they are not added to the error log.

- **Warn about unused tags during full compile**
  This option enables the generation of warning entries for unused tags that are not used directly in a CitectSCADA project. The warning entries are included in the compile errors form when a full compile is run. By default this option is not selected.

See Also

Setting the Project Editor options

Compile Error Properties

You might encounter errors when you compile your CitectSCADA project. Compile errors can be viewed in Project Editor using the Compile Errors dialog.

Compiler errors have the following properties:

Property	Description
Type	The type of error. Three types can occur during compilation. These are: **ERROR** - The compilation process continues, however the project will not compile successfully until you have corrected the error. **FATAL** - The severity of this error is such that it halts the compilation process. The project cannot be compiled until you correct the error. **WARNING** - The error was not serious enough to stop the project being compiled successfully, however investigate and correct the error.
Record	The number of the database record where the error has occurred.
Name	The name of the graphics page, library, or report format file where the error has been detected.
Field	The database field where the error has been detected.
Table	The database table where the error has been detected.
Error	A brief description of the error.
Context	The location in the database field, report format file, or Cicode library where the error has been detected. The context of the error is enclosed in braces {. . . }.

See Also

Compile Error Messages

Compile Error Messages

You might see the alert messages described below during project compilation.

Error Message	Description
Address on bad boundary	When reading a long or real from the memory of an I/O Device, addresses need to be on odd or even boundaries. Addresses cannot be mixed. You can disable checking with the [General]CheckAddressBoundary parameter.
Analog address not supported	An INT or other analog tag is specified where a DIGITAL tag is expected. Check that the tag name is correct, or that a DIGITAL data type is specified for the tag.
Array size exceeded	A tag is indexed but that tag is not declared as an array, or no index has been specified when a tag is declared as an array, or the wrong number of dimensions are specified for an array, or more than four dimensions are specified for an array.
Bad analog format	The format is incorrectly specified for an analog variable. Check the Format field in the Variable Tags form.
Bad factor specification	A Cicode expression that contains an invalid expression has been used. Check the syntax of the expression.
Bad floating point value	A floating-point number cannot be found where one is expected, or the floating-point value is out of range.
Bad I/O Device variable	The variable tag address is not a valid format for the I/O Device protocol you are using: check the address format. (See the **Data Types** topic in the Help for each supported I/O Device for a list of appropriate address formats for that device).
Bad integer value	An integer value cannot be found where one is expected, or the integer value is out of range.
Bad point limit	The incorrect point limit is specified in the Citect.ini file. The point limit needs to correspond to your CitectSCADA license.
Bad raw data type	An invalid raw data type or a mismatch of data types is specified, for example, an attempt was made to convert an integer into a string. This may be due to a variable address implying a data type that is not compatible with the data type specified. **Example 1**

Error Message	Description
	Protocol: S7NT or PSDIRECT Tag Data Type: REAL Address: DB101,54.3[32] In this case, the address implies that the variable is an array of 32 DIGITALs (via bit access) but the tag data type is REAL. **Example 2** Protocol: ABMLXEIP Tag Data Type: INT Address: T4:0/0 In this case, the address implies that the variable is a DIGITAL (via bit access) but the tag data type is INT.
Bad string conversion parameter	An invalid format parameter is specified in a string conversion. Check the format specification of the variable in the Variable Tags form.
Cannot compile every function	Error message(s) were detected while compiling the function library.
Cannot open file	The file cannot be opened. The file does not exist, or it has become corrupt, or your system is out of file handles.
Cannot read from file	The file cannot be read. The end of file was found, or it has become corrupt.
Cannot return value from void function	A RETURN statement cannot be used in a function that does not return a value. Remove the RETURN statement or declare a return data type for the function.
Cannot use an array inside function	You cannot declare an array within a function. Arrays can only be declared as library variables, i.e. at the beginning of the library file.

Error Message	Description
Cannot use RETURN outside of functions	A RETURN statement can only be used within a function.
Cannot write to file	The file cannot accept a write operation. The file has become corrupt or the disk is full.
Cicode data limit reached	An array in a Cicode module cannot exceed 60 KB. Reduce the size of the array.
Close bracket expected	The Cicode statement has a different number of open and close brackets. Another close bracket ')' or ']' is expected in the statement.
Close comment delimiter expected	A comment opened with /* needs to be closed with the */ delimiter. Add the */ delimiter or use a single line comment that starts with an exclamation mark (!) and has no end delimiter.
Close quotation mark expected	The Cicode statement has a different number of open and close quotation marks. Another close quotation mark (") is expected in the statement.
Database is empty	The database does not contain any records.
Database not found	The main project database cannot be found.
Database table full	The database is full. If the error persists, contact Technical Support for this product.
Disk full	The disk is full. Remove unwanted files from the disk, or replace the existing disk with a larger disk.
DO expected	A DO statement needs to be used in a WHILE statement.
END expected	An END statement needs to be used at the end of a conditional statement or function definition.

Error Message	Description
Error reading file	An include file specified in a database field cannot be found or cannot be opened. Check that the file name is correct, and that the file has been specified correctly, i.e. <@FILENAME>.
Expression too big	An expression is too large for the compiler. Reduce the length of the expression by splitting the expression into two or more smaller expressions.
File already opened in SINGLE mode	The file has been opened by another user. Set the [General] ShareFiles parameter to 1 in the Citect.ini file to open files in shared mode.
File does not exist	The file cannot be found. Check that the file name is correct.
File is read only	An attempt was made to write to a read-only file. Check that the file name is correct or change the attributes of the file.
File locked	A file is in use by another network user.
File not indexed	The database needs to be indexed, but the index file associated with the database cannot be found. Pack the database.
File size error	A Cicode functions file, or Report format file, or an include file is too big. The maximum file size is 1 MB.
FUNCTION expected	A function needs to be declared with the FUNCTION keyword.
GLOBAL function is not allowed, use PUBLIC	You have declared a Global function within your Cicode. Global is not a valid function type. Instead, the type Public needs to be used.
Group not found	A group name was expected. Check that the group name is correct, or that the group has been specified correctly in the Groups form.
Include project not found	An included project (specified in the Included Projects database) does not exist. Check the name of the included project.

Error Message	Description
Incompatible types	There is a mismatch of data types in a statement. For example, a string is specified where a number is expected. This error can be generated when a variable tag is placed on a graphics page and the name of the tag is identical to the name of a Cicode function in the project or an included project.
Incorrect number of arguments for function	Too many or too few arguments have been passed to a Cicode function.
Index key has changed	The database has a corrupted index. Pack the database.
Invalid BOOLEAN value	A non-integer value was found where a TRUE or FALSE value is expected. For example, the controlling expression in an IF, WHILE, or FOR statement needs to be an integer.
Invalid font name	The font does not exist in the project. Check that the font name is correct, or specify the font with the Fonts form.
Invalid group definition	A group does not exist in the project. Check that the group name is correct, or specify the group with the Groups form.
Invalid time format	The time is incorrectly specified in the Time, Period or Sample Period field of a Reports, Events, Trend Tags, SPC Trend Tags, or Devices form. Time formats needs to be in the format HH:MM:SS and needs to be in the range of 0:00:00 - 23:59:59. Only the hour is necessary, for example a value 16 means 16:00 (4:00 PM). Also 24:00:00 is accepted for historical purposes, and maps directly to 0:00:00. Period formats needs to be either a valid date or a time in the format HH:MM:SS with the minutes and seconds in the range of 0 - 59. Only the seconds are necessary, for example a value of 22 means 22 seconds. Sample Period formats need to either be a milliseconds value (for example 0.200 for 200 milliseconds) or a time in the format HH:MM:SS with the minutes and seconds in the range of 0 - 59. Only the seconds are necessary, for example a value of 22 means 22 seconds.
Label argument error	The syntax of the argument is incorrect, or the incorrect number of arguments has been specified, or the number of characters in an argument is incorrect.

Error Message	Description
Label is defined twice	Label names needs to be unique. Check the Labels form for duplicated names.
Label too big	The label is too big. The size of a label cannot exceed 8 kb.
Maximum report size exceeded	The report file size needs to be less than 63 kb. Reduce the size of the report or configure two reports.
MODULE function is not allowed, use PRIVATE	You have declared a Module function within your Cicode. Module is not a valid function type. Instead, the type Private needs to be used.
Must return value from function	If a Cicode function is declared to return a value, it needs to have a RETURN statement.
No I/O Devices defined	No I/O Devices are defined in the project.
No user defined	A CitectSCADA system is read only until someone logs in. Therefore, before compilation can occur, a user and role needs to be defined to allow someone to log in. No User has been defined in the configuration for this project. At least one user needs to be defined.
Not database format	The database has become corrupt or the file format is unknown. Pack the database.
Open bracket expected	You need to use parentheses () in Cicode functions, even when they have no parameters, for example MyFunction(). This error can be generated when a variable tag is placed on a graphics page and the name of the tag is identical to the name of a Cicode function in the project or an included project.
Operand expected	A Cicode operator needs to be followed by an operand.
Out of file handles	CitectSCADA uses a file handle to open each file. When you try to open too many files or databases simultaneously, CitectSCADA can need more file handles than are available.

Error Message	Description
	You are most likely to run out of file handles if you have many included projects. When CitectSCADA compiles your project, it will open several files in each include project at the same time, so each extra project you include will increase the usage of file handles. If you get this alert message when you have added another include project, you have run out of file handles. To verify this, remove one of the included projects to see if CitectSCADA can then compile your project. With Windows running on a network, the setup of the number of file handles is located in various places. To increase the number of file handles in DOS, the setup is in the CONFIG.SYS file. If you are using Novell Netware you need to also increase the file handles in the NET.CFG or SHELL.CFG file. You need to also increase the number used by CitectSCADA with the [CtEdit]DbFiles parameter. Adjust the following settings the associated files: CONFIG.SYS FILES=120 NET.CFG or SHELL.CFG file handles=120
Out of memory	CitectSCADA has run out of memory. Increase the amount of memory in the computer or use smaller databases.
Page Name cannot start with underscore	A Page Name needs to start with an alphanumeric character (A - Z, a - z, or 0 - 9).
Point limit reached	The maximum number of points that can be referenced has been reached. The maximum point limit is determined by your CitectSCADA license. Contact Technical Support for this product.
PRIVATE variable is not allowed, use MODULE	You have declared a Private variable within your Cicode. Private is not a valid variable type. Instead, the type Module needs to be used.
Protocol expected	The protocol field in the I/O Devices database is blank. You need to select a protocol for the I/O Device.
PUBLIC variable is not allowed, use GLOBAL	You have declared a Public variable within your Cicode. Public is not a valid variable type. Instead, the type Global needs to be used.

Error Message	Description
Reached the end of table	The end of the database has been reached or the database has become corrupt. Pack the database. If the error persists, contact Technical Support for this product.
Read remap is not supported for this variable	A mapped variable cannot be written when Remap Write is disabled, and cannot be read when Remap Read is disabled. Check the Remapping form.
Semicolon expected	Cicode statements needs to be separated with semi-colons (;).
Software error	An internal CitectSCADA software error has been detected. Contact Technical Support for this product.
Specification file invalid	A system file has become corrupt, or been deleted. Re-install CitectSCADA on your system. If the error persists, contact Technical Support for this product.
Statement expected	CitectSCADA is expecting a statement. Check the Cicode for syntax errors.
String expected	Only strings can be used in database fields.
String too big	The string size has been exceeded. The maximum size of the string not to exceed 255 characters.
Super Genie must be on a page	Super Genie syntax (?) can only be used on pages. You cannot use a Super Genie in a report or Cicode function library. Use the TagRead() and TagWrite() functions instead.
Symbol search failed	A database record does not exist. Check that the record name is correct.
Syntax error	A malformed Cicode expression has been specified. Check the structure of the expression.
Tag already defined	Tag names need to be unique. Check the Variable Tags form for duplicated names.

Error Message	Description
Tag expected	A tag name was not found where one was expected, or an expression has been passed to a function that expects a tag. Check that the tag name is correct, or specify the tag with the Variables Tag form.
Tag not found	The tag does not exist. Check that the tag name is correct, or specify the tag (with the Variables Tag form). If the tag does exist in the variables database, the index to the database might be incorrect. This can occur if you have edited the variables database using Excel or some other database editor. To re-index the database choose **File** | **Pack** (in the Project Editor).
THEN expected	A THEN statement needs to be used in an IF statement.
Too many arguments	Too many arguments are specified in a Cicode function. The maximum number of arguments allowed is 32.
Too many Cicode functions	More than 4500 user functions have been defined. To increase the number of functions allowed (up to 10000), use the CtEdit parameter MaxCicodeFunctions This error is often due to Cicode functions being defined in a number of included projects. Extending this parameter might affect system performance. Only set it when advised by Citect Customer Service.
Too many fields in database	Too many fields have been specified in the database. This error will only occur if the `Citect.frm` file has been changed or become corrupt.contact Technical.Support for this product.
Too many files open	The maximum number of .DBF files that can be open simultaneously has been exceeded. Increase the limit by changing the [CtEdit] DbFiles parameter.
Too many Include projects	More than 240 Include projects have been defined.
Too many records in database	Too many records have been specified in the database. This error should only occur if the `Citect.frm` file has been changed or become corrupt. Contact Technical Support for this product.
Trailing characters in Cicode	There are extra trailing characters in a Cicode statement, following the semi-colon.
Trailing characters in Name	The database record name contains invalid characters. Remove any invalid characters from the record name.

Error Message	Description
Unexpected beginning of file	The beginning of the database has been reached or the database has become corrupt. Pack the database. If the error persists, contact Technical Support for this product.
Unexpected end of file	The end of the database has been reached or the database has become corrupt. Pack the database. If the error persists, contact Technical Support for this product.
Unknown bin error	An output file could not be opened during compilation. Pack the database. If the error persists, contact Technical Support for this product..
Unknown DBA error	An internal CitectSCADA software error was detected. Contact Technical Support of this product.
Unknown field	A field is being referenced that does not exist. The database has been modified or has become corrupt. Pack the databases. If the error persists, contact Technical Support of this product.
Unknown file	The include file cannot be found. Check the name of the include file, or that the included file is in the correct directory.
Unknown I/O Device	The I/O Device (unit) does not exist in the project. Check that the I/O Device name is correct.
Unknown protocol	The protocol does not exist.

Distributing the Project

After compiling the project you will need to distribute the runtime files to your live system. A project can be distributed to a computer in various ways:

- Using the archive feature - See Archiving projects
- Using a manual file transfer - pushing the runtime files directly onto your live system at the operating system level
- Using automatic file transfer - pulling runtime files from the configuration machine to the live server using [CtEdit]Run and [CtEdit]Copy
- Using a shared folder on a network drive

> **Note:** Shared folders are not recommended for servers or control computers.

Automatic file transfer

This is where runtime files are pulled down to the runtime system using the [CtEdit]Run and [CtEdit]Copy parameters.

- [CtEdit]Run points to the location of the runtime files. To avoid running a system over the network and potentially slowing down the system, it is recommended that this is a local folder under the [user] directory.
- [CtEdit]Copy points to a copy of the runtime files. This can be a remote shared location (for example the [user] folder shared on another computer) or another local folder. When CitectSCADA detects that the files in the RUN directory have a different timestamp to those in the COPY directory, the files are copied across to the RUN directory.

By updating the runtime files in the COPY directory, they will be automatically copied to the RUN directory whenever they are changed.

Shared Folders

This is where the runtime files are stored in a network shared location, with a backup server specified in case the main files are not available.

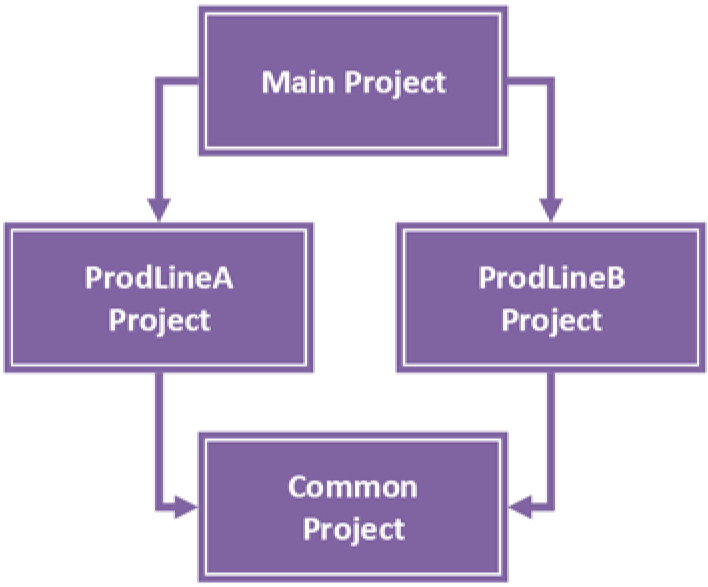
- [CtEdit]Run points to the main remote shared location of the runtime files.
- [CtEdit]Backup points to the alternate remote shared location.

> **Note:** Shared folders are not recommended for servers or control computers.

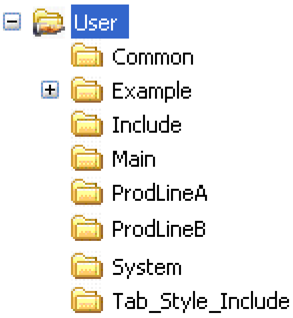
Example of Using Run/Copy to Distribute Project Runtime Files

File Server (PC1)

Consider the example project structure below. The overall system comprises four parts: a main project which features two separate production line projects, each of which draws on some common elements from a common project.



Run/Copy can be used to update project files from the File Server (PC1) to the File Client (PC2). Run/Copy does not support nested project folders, therefore project folders need to be on the same level as shown below.



The [User] folder (or a parent of the [User] folder) needs to be shared on the network. The shared folder needs to be accessible for machines to download the project runtime files. In this example, [User] is shared.

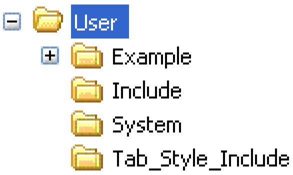
File Client (PC2)

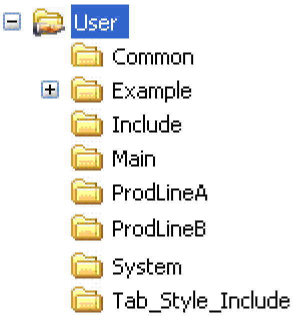
File Client (PC2) will use Run/Copy to get the latest updated projects from the File Server.

The settings for Run/Copy on the client are:

[ctEdit]Copy = PC1\User\Main
[ctEdit]Run = D:\Program Files\CitectSCADA V7.20\User\Main

The [Run] folder needs to be a child of the [User] folder on the target machine. The following pictures shows the project folders before and after the first project start-up with Run/Copy.





It can be seen that the project folders are automatically created by the Run/Copy process if they don't exist. In this example, the basic runtime files are copied to the [Run] folder. Any other files will be copied over on demand, such as page *.ctg and *.rdb files. After a successful project start-up, it is also possible for the project to start up when the [Copy] path is not available (if, for example, the File Server is uncontactable). In this case, only pages that have been previously viewed (that have been copied over previously on demand) are available.

If the target machine has the project development environment installed, do not launch CitectSCADA Explorer after Run/Copy are set up. Doing so will override the [Run] path setting and you may end up running the wrong project.

See Also

File server redundancy

Running the System

This section of the help describes how to run a CitectSCADA project, the adjustments that can be made to the runtime system, and the processes you can use to update and test a system while it is operational.

> **Note:** Before you attempt to run a project, check that the host computer has been appropriately configured using the Computer Setup Wizard. See Configuring Your System for more information.

See Also

Startup and runtime configuration
Running servers independently
Server redirection

Startup and runtime configuration

You can specify a Cicode function to execute automatically when CitectSCADA starts up. This Cicode gets executed as soon as the Cicode system comes online. use the Computer Setup Wizard to specify the name of the startup function.

You can also run a report on startup. CitectSCADA searches for a default report called "Startup" when it starts up. If you have configured a report called "Startup", it is run automatically. You can change the name of the startup report (or disable it altogether) by using the Computer Setup Wizard.

You can customize many elements of CitectSCADA's runtime and startup behavior. Only The Computer Setup Wizard is necessary, but you can also use Parameters for more control.

To start your runtime system:

- Click **Run**, or choose **File** | **Run**.

To compile and run CitectSCADA online:

- Click **Run**, or choose **File** | **Run**.

> **Note:**CitectSCADA automatically compiles the project (if uncompiled) when you try to run it.

If your operating system is Windows Vista, CitectSCADA runs a compatibility check at runtime to see if drivers used by the running project are compatible with your operating system.

- If an I/O Device uses a driver marked as compatible with Windows Vista, the project proceeds as normal.
- If an I/O Device uses a driver marked as incompatible with Windows Vista, a dialog box is displayed asking you to remove the I/O Device from the project configuration or run the I/O Server on a compatible operating system.
- If an I/O Device uses a driver that hasn't been confirmed as either compatible or incompatible, a dialog box is displayed asking if you want to proceed with running the project.

Running servers independently

You can run individual processes in a multi-process system, allowing a particular server or client to be launched independently. This can be useful for hardware testing and system analysis.

The Runtime Manager can be launched during the configuration of a project via the **Tools** menu in **Citect Explorer**, **Graphics Builder** and **Project Editor**. Launching the Runtime Manager this way presents it in an idle state, allowing you to start a particular process by itself.

Once a process is started (by right-clicking on it and selecting **Start**), the Runtime Manager's monitoring pane displays the startup log for the selected processes, and launches Runtime.

Note: Only one instance of Runtime Manager can run on a machine at any time.

For more information on using Runtime Manager, click on its **Help** button.

See Also

Server Redirection

Server Redirection Using Address Forwarding

CitectSCADA allows you to temporarily transfer the communications of a server to another computer to assist with hardware maintenance and system analysis.

By including an address forwarding section within a Citect.ini file, you can override a server's project-configured address, redirecting network traffic to a different address and port.

For example, if you would like to test a new server before adding it to a project configuration, you could make the necessary address forwarding adjustments within the citect.ini file of a single client to direct it to the new hardware. The server could be tested within the context of a system without having to recompile the project.

You need to make the necessary adjustments within the citect.ini file for every computer that is likely to be impacted by address forwarding.

For example, if you have a computer configured to run two I/O Servers, and you would like to temporarily redirect one of them, then both server machines and every connecting client need to have the necessary address forwarding adjustments made to their citect.ini files, so that the servers are aware of each other.

Address forwarding is implemented using the following syntax within a citect.ini file:

```
[AddressForwarding]
<ClusterName>.<ServerName>=<ipaddress>:<port>
```

You can also redirect the Peer Port connection for an I/O Server using the following syntax:

```
[AddressForwarding]
<ClusterName>.<ServerName>_PeerPort=<ipaddress>:<port>
```

For Alarm Servers that have alarm properties enabled, you can redirect the alarm properties connector using the following:

[AddressForwarding]
<ClusterName>.<ServerName>_AlarmProps=<ipaddress>:<port>

Address forwarding is only interpreted and used during startup of Citect Runtime. It is recommended (but not necessary) that the Computer Setup Wizard is run before running up a project to confirm your changes.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

- Do not use address forwarding as the primary mechanism for specifying client/server communications. It bypasses the normal configuration checks enforced by the compiler.
- carefully track any adjustments you make to the citect.ini file, as they needs to be manually corrected once redirection is no longer necessary.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Note: The CitectSCADA Web Client uses address forwarding to manage communication across corporate firewalls. Any manual adjustments may also impact the ability to connect a Web Client to a deployment on a Web Server.

Using an Alternative INI File

When CitectSCADA starts up, it reads default values from `citect.ini`. From v7.10, CitectSCADA expects the Citect.ini file to exist in the `config` folder in the User and Data directory selected during installation. If the file is not found in this location, it will not search elsewhere and will instead display an error.

If you need to store your INI file elsewhere, specify the path to it on the command line when starting citect32.exe and ctexplor.exe. If you are running multiple projects, you can specify an alternative `ini` file for each project. The new `ini` file name needs to be passed to the CitectSCADA system applications; i.e., to `CtExlplor.exe` and `Citect32.exe` as a command line argument.

To specify an alternative INI file for CitectSCADA :

1. Click the icon that you use to start Citect Explorer or Runtime.
2. Right-click to display the shortcut menu and select **Properties**.
3. Click the **Shortcut** tab.
4. Add the name of the INI file to the command line property of the appropriate icon using the **`-i`** option. For example, to start CitectSCADA v 7.20 using the initialization file `my.ini`, enter the following line:

```
C:\Program files\Citect\CitectSCADA 7.20\bin\citect32.exe -ic:\-
citect\user\myproj\MY.INI
```

> **Note:** The Computer Setup Wizard uses the same `.ini` as specified for Citect Explorer. You can specify different `.ini` files for both the Runtime and Explorer programs. However, if you initiate Runtime from the Explorer, it uses the `ini` that is specified for the Explorer.

System tuning

CitectSCADA is designed for optimal performance, so it is not necessary for most users to tune their system. However, special circumstances might require that you adjust your system for optimal performance. The Kernel allows you to locate areas that need tuning, and the tuning itself is usually done through parameters. For example, you can improve performance of the client by using the `[Page]ScanTime` and `[Alarm]ScanTime` parameters.

Cache tuning

Tune the cache large enough so that unnecessary reads are not generated, and small enough that old data is not returned while keeping the communication channel busy. If the cache is too large, the communication channel might become idle for a while and so waste its bandwidth. Also if the cache is too large, a CitectSCADA client might start to short cycle on reads request, which will generate unnecessary network or internal traffic load.

Read short cycling occurs when a client requests data from the I/O Server, and the data is returned from the cache, so it is returned quickly. The client will process the data (display it on screen) then ask for the same data again. If the I/O Server again returns the same data from the cache, the client will process the same data again which is redundant and a waste of CPU and the network (to transmit the request and response). When short cycling starts to occur, the CPU and network loading will rise while the PLC communication traffic will start to fall.

To tune the cache you need to balance the cache time between unnecessary reads and short cycling. The method described below assumes you know how to use the Citect-SCADA debugging Kernel.

1. Turn off unit caching, use the CACHE command in the Kernel so you don't have to re-compile your project.
2. Run one CitectSCADA client only on the network, use the Client in the I/O Server for the test.
3. Display a typical page to generate normal PLC loading for your system.
4. In the Kernel use the STATS command to reset the CitectSCADA statistics.
5. In the Kernel display the page 'PAGE TABLE STATS'. This page shows the cycle and execution time of various CitectSCADA tasks, some of which consume PLC data. The tasks called 'Citect *n*' where *n* is a number are the tasks which get data from the PLC and display on screen. Look at the Avg Cycle time, this is the third column from the left. Assume that the Avg cycle time is 1200 ms. T his will mean that the current page is gathering PLC data and displaying its data on the screen in 1200 ms.
6. Always set the cache time below this average cycle time to minimize short cycling. On average set it to less than half this time, that is 600 ms.
7. Set the cache time to half the cycle time (600 ms). You might not see any improvement in performance with a single client, as caching will only improve performance with multi clients. You might see improvements if you are also running trends, alarms or reports which are requesting the same data.
8. Add another CitectSCADA client that is displaying the same data. Reset the STATS and check the Average cycle time. Each new client will not increase the cycle time, it will drop slightly. Also look at PAGE GENERAL, to see that each new client services its reads from the cache; i.e., the % cache reads increases.
9. If the average cycle time drops to less than half the original time then short cycling is occurring and you need to decrease the cache time until this stops.
   Tuning the cache is a trial and error process - as you change it, the read cycle time will also change. The cache time will also depend on what the current PLC traffic is. The current traffic is dynamic as CitectSCADA will only read what is needed depending on the current page, trend, alarm and reports running. Monitor the average cycle

time under lower loading conditions and set the cache as low as necessary to stop or help prevent short cycling.

Client Side Online Changes

You can now reload a page when the Cicode library has changed.When you attempt to open a page the system locates the necessary runtime database file on the local disk (or a server location in the Run/copy setup). If the file timestamp is different to the one in the memory then it will try to reload the new page file. On page load the system compares the version of the Cicode library with which the page was created to the one in memory. If the Cicode library mismatches then a hardware alarm of "Cicode library timestamp differs" is raised. The page is displayed and any Cicode function that is executed will be from the memory version of the Cicode library and not the latest compiled version.

Note: It is recommended that the ServerGetProperty cicode function be used with the *LibRDBMemTime* and *LibRDBDiskTime* properties to check if there is a change to the Cicode library before attempting a reload. Following a reload please check the corresponding server's syslog.dat file for any reload messages. The cicode changes will not be reloaded.

A restart of the client is needed to update the Cicode library version to the latest compiled version.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Restart the client process if the hardware alarm "Cicode library timestamp differs" is raised after a page is opened.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Note: A hardware alarm of "Cicode library timestamp differs" will be raised if the Cicode library used by a page has a different timestamp from the one in memory. The timestamps will be different if the project has been fully recompiled, the project has been incrementally recompiled after the page has been modified, or if the project has been incrementally recompiled after any Cicode has been modified.

See Also

Client Side Online Changes

Compiling the Project

Linking Projects

PageDisplay

PageGoTo

Server Side Online Changes

Previously in CitectSCADA v7.0 the display client supported online changes for graphic pages and tags, but the servers needed to be restarted if any changes were made to the configuration. This meant that simple changes such as removing an alarm or trend record necessary the large change of restarting the server. With CitectSCADAv7.20, records can be modified during runtime using the project editor, recompiled and the resulting database files updated. You can initiate a configuration reload for each server using the Runtime Manager or using Cicode functions.

Note: It is recommended that the ServerGetProperty cicode function be used with the *LibRDBMemTime* and *LibRDBDiskTime* properties to check if there is a change to the Cicode library before attempting a reload. Following a reload please check the corresponding server's syslog.dat file for any reload messages. The cicode changes will not be reloaded, therefore a restart may be more appropriate.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Restart the server process if a "Cicode library timestamp differs" error is detected. The library mismatch is indicated on the server in either the hardware alarm or the server's syslog.dat file.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Note: A message in the Syslog.dat file and hardware alarm of "Cicode library timestamp differs" (error code 454) will be raised if the Cicode library used by one or more server runtime databases is different from the one in memory. The timestamps will be different if the project has been fully recompiled (with or without Cicode modification), or if the project has been incrementally recompiled after any Cicode has been modified.

The following fields are reloaded on their respective servers:

Alarms

- Digital Alarms
- Time Stamped Alarms
- Analog Alarms
- Advanced Alarms
- Multi-Digital Alarms
- Time Stamped Digital Alarms
- Time Stamped Analog Alarms
- Argyle Analog Alarms
- Alarm Categories - The Alarm Category is not reloaded on the client side.

The following Alarm Category fields are used by the server and will be reloaded:

- ON Action
- OFF Action
- ACK Action
- Summary Device
- Log Device, ON, OFF and ACK log devices

Trends

The following data will be reloaded on a Trend Server when adding, removing and modifying the records of:

- Trend Tags
- SPC Tags

Modifying a record's archive properties:

- Sample Period
- Type (from Periodic to non Periodic or vice versa)
- Filename
- Storage Method
- No. Files
- Time
- Period

will put the record in an error state and no acquisition will be done for corresponding records. A hardware alarm, a syslog entry and a reload error message will be raised. The history files will not be deleted. For periodic to periodic Type changes, reload will not generate an error.

If you want to have a valid archive for the modified record, you need to delete the old history files (after archiving manually) and then trigger the reload again after forcibly compiling the Trend configuration. This is necessary because the trend system doesn't reload the trend rdb file if the compile time of in-memory file is same to the on disk file. Therefore changes to an existing trend record's archive properties is a two staged reload.

Due to reload the display clients will need to request the latest configuration data for trend records. The Process Analyst will periodically refresh configuration data from the trend server. The Trend server tag browsing is optimized to handle the more frequent requests.

For legacy trends including SPC, configuration data will be refreshed from the trend server when a page update is triggered.

Reports

Changes to devices will not be reloaded because the devices may be in use by other Cicode tasks. The following fields will be reloaded on a Report Server:

- Reports
- Accumulators

See Reference for more information.

See Also

ServerReload

Effects of Server Reload on Servers

The following sections detail the effects on the history of reloaded Alarm, Trend and Report records. This is dependent on which fields have changed.

A 'Yes' in the New Alarm column indicates changes to that field will make a new alarm and delete the old one and therefore the history is not kept.

A 'Yes' in the Invalid History column means changes to that field will make the history of that record invalid.

Alarm Servers

Common Alarm fields

Field Name	New Alarm	Invalid History
Alarm Tag	Yes	No
Cluster Name	Yes	No
Alarm Description	No	No
Alarm Category	No	No

Field Name	New Alarm	Invalid History
Help	No	No
Delay	No	No
Comment	No	No
Privilege	No	No
Area	No	No
Custom 1..8	No	No
Paging	No	No
Paging Group	No	No

Analog and Time Stamped Analog Alarm fields

Field Name	NewAlarm	Invalid History
Variable Tag	No	Yes
High High	No	Yes
High	No	Yes
Low Low	No	Yes
Low	No	Yes
SetPoint	No	Yes
Deviation	No	Yes
High High Delay	No	No
High Delay	No	No
Low Low Delay	No	No
Low Delay	No	No
DEviation Delay	No	No
Rate	No	Yes
Deadband	No	Yes
Format	No	No

Digital and Time Stamped Digital Alarm fields

Field Name	New Alarm	Invalid History
Variable Tag A	No	Yes
Variable Tag B	No	Yes

Time Stamped Alarm fields

Field Name	New Alarm	Invalid History
Variable Tag	No	Yes
Timer	No	Yes

Advanced Alarm fields

Field Name	New Alarm	Invalid History
Expression	No	Yes

Multi-Digital Alarm fields

Field Name	New Alarm	Invalid History
Variable Tag A	No	Yes
Variable Tag B	No	Yes
Variable Tag C	No	Yes
Realarm	No	Yes
On Function	No	No
Off Function	No	No
State	No	Yes
State Description	No	No

Argyle Analog Alarm fields

Field Name	New Alarm	Invalid History
Variable Tag	No	Yes
SetPoint	No	Yes
High High	No	Yes
High	No	Yes
Low	No	Yes
Low Low	No	Yes
Deviation	No	Yes
Rate	No	Yes
Deadband	No	Yes
Format	No	No
Trigger	No	No
Latch	No	No
Control Timeout	No	No

Trend Servers

Changes to the following properties of trends make existing trend files invalid and result in the creation of a new trend file (renaming the old one for backup):

- Sample Period
- Type
- Storage Method
- No of Files
- Time (File)
- Period (File)

Changing the scale of a tag will make the trend history invalid.

Effect of reload on a modified Trend:

Field Name	New Trend	Invalid History
Trend Tag Name	Yes	No
Cluster Name	Yes	No
Expression	No	Yes
Trigger	No	Yes
Comment	No	No
Privilege	No	No
Area	No	No
Eng Units	No	No
Format	No	No

Report Servers

Effect of reload on a modified Report:

Field Name	New Reports	Invalid History
Report Name	Yes	No
Cluster Name	Yes	No
Time	No	Yes
Period	No	Yes
Trigger	No	No
Report File Format	No	No
Output Device	No	No

Field Name	New Reports	Invalid History
Privilege	No	No
Area	No	No

Effect of reload on a modified Accumulator:

Field Name	New Accumulators	Invalid History
Name	Yes	No
Cluster Name	Yes	No
Trigger	No	Yes
Run Time	No	Yes
No. of Starts	No	No
Totaliser Inc	No	No
Totaliser	No	No
Comment	No	No
Privilege	No	No
Area	No	No

Restarting the System Online

With the online restart facility, you can change your project configuration and examine the results in the runtime system without having to shut down CitectSCADA. You can update your system while it is running.

Note: If you've configured CitectSCADA to use multiprocessor support, it is recommended that you use the Runtime Manager instead of the restart facility. The runtime manager allows you to restart different processes individually, whereas the restart facility will restart every CitectSCADA processe on the machine.

The time taken for the system changeover depends on the size of the project and the extent of the changes to the project:

- If you only change graphics pages, CitectSCADA does a *partial restart* (changing only the pages in the runtime system). Changeover is instantaneous.
- If you change any databases (for example, add a new alarm tag, trend tag, or Cicode function), CitectSCADA does a *full restart* to run the updated project.

See Also

Restarting a networked system online

Restarting a networked system online

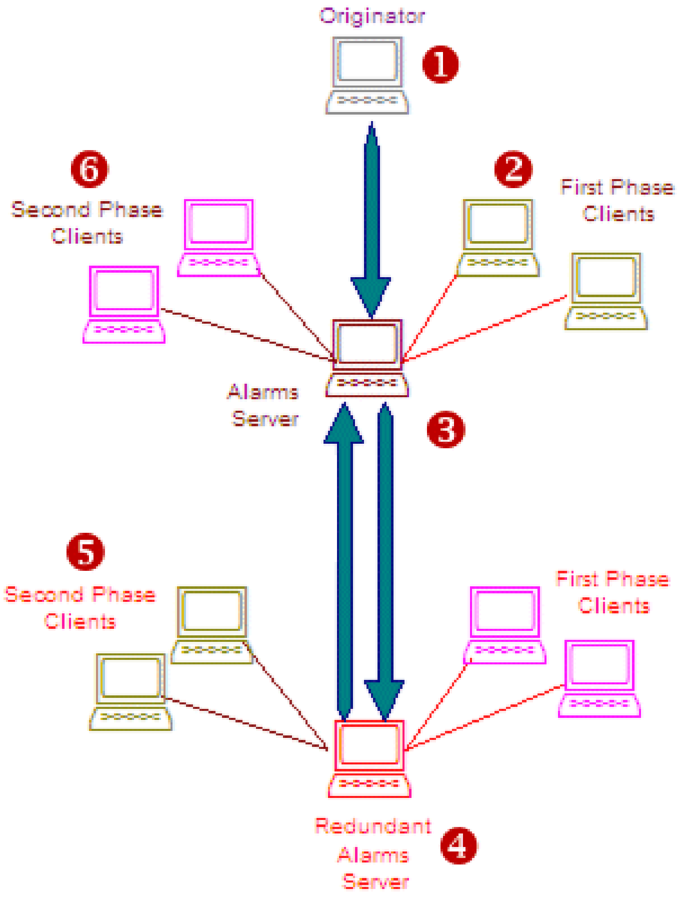
If you are using CitectSCADA on a network, and would like to restart CitectSCADA without undue impact on the control of the plant, use a structured restart procedure. You can use any CitectSCADA computer on the network to initiate the online restart.

To phase shutdowns:

1. Set the citect.ini parameter [Shutdown]Phase on each client to either 1 or 2 to identify the phase of the client.
2. Issue a Shutdown() command from the server.
   When the Shutdown() command is issued, clients with phase set to "1" are shutdown first. When the first phase 1 client has reconnected to the server, clients with phase set to "2" are shutdown.

CitectSCADA automatically manages the online restart in the following sequence:

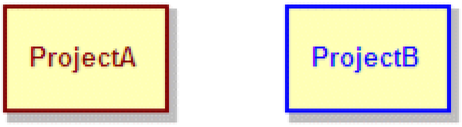
1. The Originator issues the Shutdown("Everybody") command.
2. The Alarm Server that the originator is connected to shuts down its first phase clients.
3. The Alarm Server that the originator is connected to advises the other Alarm Server then shuts itself down.
4. The second Alarm Server shuts down its first phase clients and waits for the other Alarm Server to restart (running the new project).
5. When the first Alarm Server is back on line, the second Alarm Server shuts down its second phase clients and then shuts itself down.
6. The first Alarm Server restarts its first phase clients and shuts down and restarts its second phase clients.



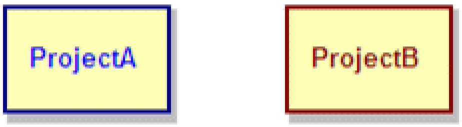
Note: If you’ve configured CitectSCADA to use multiprocessor support, it is recommended that you use the Runtime Manager instead of the restart facility. The runtime manager allows you to restart different processes individually, whereas the restart facility will restart every CitectSCADA processe on the machine.

Using multiple projects

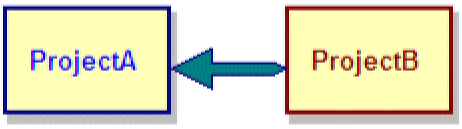
The most effective method of using the online restart facility is to use two projects. The first project becomes the current runtime system while the second project is in the development stage. You can manage both projects as follows:



Project A is currently the runtime system while Project B is under development.



When Project B is complete , you can use the online restart facility to change the runtime system to Project B.



You can then copy Project B to Project A for further development.

> **Note:** If you've configured CitectSCADA to use multiprocessor support, it is recommended that you use the Runtime Manager instead of the restart facility. The runtime manager allows you to restart different processes individually, whereas the restart facility will restart every CitectSCADA processe on the machine.

Initiating the online restart

To initiate the online restart, the originator (any CitectSCADA computer on the network) issues a shutdown command with the Shutdown function, for example:

```
Shutdown("Everybody", "MyProject", 2);
```

Where possible, balance clients across both phases of the shutdown. The `[Shutdown]Phase` parameter defines the phase to which each CitectSCADA computer responds.

You can exclude selected computers (for example I/O Servers) from the online restart procedure with the `[Shutdown]NetworkIgnore` parameter.

For security, you can prevent selected computers from initiating the online restart procedure with the `[Shutdown]NetworkStart` parameter.

> **Note:** If you've configured CitectSCADA to use multiprocessor support, it is recommended that you use the Runtime Manager instead of the restart facility. The runtime manager allows you to restart different processes individually, whereas the restart facility will restart every CitectSCADA processe on the machine.

Using a Callback function

You can use a callback function (with the `OnEvent` function) to perform housekeeping tasks before the system shuts down. You would normally call `OnEvent()` in the main startup function (defined with the `[Code]Startup` parameter). Each time a `Shutdown()` call is made, the callback function is run.

```
/* A user shutdown procedure. */
INT
FUNCTION
MyStartupFunction()
        ...
        OnEvent(25, MyShutdown);
        ...
END

INT
FUNCTION
MyShutdown()
        STRING sPath;
        // Perform housekeeping tasks
        ...
        sPath = ProjectCurrentGet();
        If sPath = "ProjectA" Then
                ProjectSet("ProjectB");
        Else
                ProjectSet("ProjectA");
        END
        Shutdown("Everybody", sPath, 2);
END
```

Running Your System Over the Internet

If you have a computer with Internet access, you can use it to run your project over the Internet from a remote location. Your computer would then be called an Internet Display Client. This is basically a runtime-only version of CitectSCADA; you can run your project from that computer, just as you would from any normal client. However, an Internet Display Client cannot be a server, and it cannot be used to make configuration changes - you can only run your project.

The Internet Display Client is not supported on Windows Vista. It will still operate correctly on earlier Operating Systems, however there are no plans to provide support for IDC on Vista and above in the future.

Note: CitectSCADA also allows you to run your projects in a standard Web browser

> across a network of LAN-connected computers. See the CitectSCADA Web Client.

See Also

CitectSCADA Internet Display Client
CitectSCADA Internet server
Startup and runtime configuration
Server - client file updates

The Internet Display Client

The Internet Display Client is a runtime-only version of CitectSCADA. An Internet Display Client cannot be a server, and it cannot be used to make configuration changes - you can only run your project.

A computer can have a normal CitectSCADA installation as well as the Internet Display Client. Before you can run a project over the Internet, you need to switch the **[Internet]Client** parameter on.

You can also run multiple instances of the Internet Display Client at the same time. This allows you to work with more than one project, in runtime-only mode, on the remote computer.

Notes:

- The Internet Display Client is not supported on Windows Vista. It will still operate correctly on earlier Operating Systems, however there are no plans to provide support for IDC on Vista and above in the future.
- If an ActiveX object has an associated data source, you need to verify the data source can be located by the computer hosting the Internet Display Client. See the topic Managing associated data sources.
- If you want to use the Process Analyst via the IDC, you need to copy the Process Analyst view (.pav) files to the IDC in order to do so.

See Also

CitectSCADA Internet server

The Internet server

Any I/O Server can be a CitectSCADA Internet Server: you just need to use the Computer Setup Wizard. (A special protection key is necessary for the Internet Server. Please contact Technical Support for this product for protection key details.)

Note: The Internet Display Client is not supported on Windows Vista. It will still operate correctly on earlier Operating Systems, however there are no plans to provide support for IDC on Vista and above in the future.

See Also

Startup and runtime configuration

Startup and runtime configuration

You can customize many elements of CitectSCADA's runtime and startup behavior via the Computer Setup Wizard.

You can also configure an IDC installation to automatically connect to a CitectSCADA Internet Server at startup. To do this, you need to adjust the parameters [Internet]IPAddress, [Internet]Password and [Internet]ShowSetupDlg within the citect.ini file of the IDC computer.

The `citect.ini` file is stored in the `bin` directory of the Internet Display Client installation.

Note: The Internet Display Client is not supported on Windows Vista. It will still operate correctly on earlier Operating Systems, however there are no plans to provide support for IDC on Vista and above in the future.

See Also

Server - client file updates

Server - client file updates

When you log on to the CitectSCADA Internet Server, the files needed to run the project are downloaded to your computer. Because these files needs to be up to date, CitectSCADA periodically compares the files on the Internet Server with the downloaded files on the Internet Display Client. (This period is defined using the **[Internet]UpdateTime** parameter.) If a file has been changed since the last update, it is copied to the Internet Display Client.

Note: The Internet Display Client is not supported on Windows Vista. It will still operate correctly on earlier Operating Systems, however there are no plans to provide support for IDC on Vista and above in the future.

To set up your CitectSCADA Internet Server:

1. Run the Computer Setup Wizard and select **Custom Setup**.
2. Select **Server and Control Client** from **Network Computer** section.
3. Once you reach the Internet Server screen, select **Internet Server**.
4. Enter the TCP/IP address of your Internet server computer (for example 10.5.6.7 or plant.yourdomain.com). This information is downloaded and stored in the Internet Display Client's citect.ini file when a connection is made.
5. To determine the TCP/IP address of the Internet server computer:
   - For Windows NT4 or 2000, go to the Command Prompt, type **IPCONFIG**, and press **Enter**.
   - For Windows 95, select **Start | Run**, type **WINIPCFG**, and press **Enter**.
6. After completing the Computer Setup Wizard, define the passwords necessary by users of your Internet Display Clients using the `[Internet]Manager` and/or `[Internet]Display` parameters.
7. If the Runtime project on the Internet Server has links to any included projects, the Internet Display Client can only access the included project files if they are stored on the same directory level as the Runtime project. For example, if the current project is located at:

`C:\Citect\User\`<*current project*>

then any included projects needs to be located on the same level:

`C:\Citect\User\`<*included project*>

8. Any files you want to make accessible to an anonymous FTP user have to be placed in the Internet Server's`\Internet` directory, located at `C:\Citect\User\Internet` by default. This is where CitectSCADA stores the `idc.exe` file to allow remote installation of the Internet Display Client.

The`Internet` directory on the Internet Server is only accessible to anonymous FTP users if it shares the same directory level as the current Runtime project. For example, if you use CitectSCADA's default settings, the current project folder will be located at:

`C:\Citect\User\`<*current project*>

andthe `Internet` directory will be located on the same level at:

`C:\Citect\User\Internet`

If the runtime project on the Internet server is stored elsewhere, an appropriately located Internet directory needs to be created.

To install the Internet Display Client:

1. On the remote computer, start your Internet browser.
2. Type in the FTP address of your CitectSCADA Internet Server (for example, **`ftp://sanctus.citect.com.au/idc.exe`**).

3. Save the file (`IDC.exe`) to a temporary folder.
4. Go to this temporary folder and double-click **`IDC.exe`**.
5. Follow the prompts to complete the installation.

To run your project over the Internet:

1. Verify that you have installed the Internet Display Client and set up your Internet server.
2. At the Internet Display Client, double-click the **Citect Runtime** icon in the CitectSCADA IDC program group.
3. In the **Citect Internet Client Setup** dialog, type the TCP/IP address of your CitectSCADA Internet Server (for example 10.5.6.7 or plant.yourdomain.com). You can also optionally enter a port number separated by a colon ":". If you do not enter a port number the port number specified in the [Internet]Port parameter in the Citect.ini file will be used, the default is 21.
4. Type your password
5. Click **OK**. The relevant data will be downloaded to your computer and your project will run (if the CitectSCADA Internet Server is running).

To connect to a different CitectSCADA Internet Server once your project is running on the Internet Display Client:

1. Click **Client Setup** from the runtime **Control** menu.
   In the **Citect Internet Client Setup** dialog box, complete the following properties:

Option	Description
Address	Enter the TCP/IP address of the CitectSCADA Internet Server (for example 10.5.6.7 or plant.yourdomain.com). You can also optionally enter a port number separated by a colon ":". If you do not enter a port number the port number specified in the [Internet]Port parameter in the Citect.ini file will be used, the default is 21. The address of the last Internet Server used will be automatically entered here. The addresses of every Internet Server previously used are retained in the menu (with the most recently used at the top).
Password	Enter the password supplied to you by the CitectSCADA Internet Server administrator. Your password will be encrypted before it is sent across the Internet. If you enter an incorrect password, the connection attempt will not succeed.

3. Click **OK**. The relevant data is downloaded to your computer and your project will run.

Software Protection

CitectSCADA uses a hardware key to help prevent license infringement. The hardware key is a physical key that plugs into either the parallel port or USB port of your computer. The hardware key contains details of your user license, such as type and I/O point limit.

See Also

CiUSAFE dialog properties
Demo mode

CiUSAFE dialog properties

The CiUSAFE dialog box has the following properties:

Serial Number

The serial number of the computer's hardware key. It will only appear if the key was delivered after September 11 2000, or has been updated since this time. If this is not the case, you can read the number from the label on the hardware key. You need to enter the serial number at the Schneider Electric (Australia) Pty. Ltd. web site to update the key.

KeyID

Each time you launch CiUSAFE, a Key ID will display in the **KEYID** field. You might need to provide the Key ID plus the serial number when updating the hardware key. This depends on the status of the key in the CitectSCADA license database, and you are prompted if the Key ID is necessary. Click **Save KeyID** to save the Key ID and serial number to a text file, which you can refer to when visiting the Schneider Electric (Australia) Pty. Ltd. web site.

Authorization Code

To update the hardware key, enter the 106-character authorization code. You are asked for this code once you have entered the Key ID and serial number, and your license and Customer Service agreement have been verified. Click **Update** to update your hardware key.

Return Code

The Return Code indicates the result of the key update:

0	The key was updated successfully.
1,3	Either the KeyID or the Authorization code you entered is invalid.

2	Either the KeyID or the Authorization code you entered has been corrupted.
4,16	Either the KeyID or the Authorization code you entered is invalid.
9	No hardware key could be found.

To close the program, click **Exit**.

Demo mode

You can run CitectSCADA without the hardware key in demonstration (Demo) mode. Demonstration mode lets you use every CitectSCADA feature normally, but with restricted runtime and I/O.

In demonstration mode you can work in multi-process (with the networking model selected as Stand alone) or single process mode.

The following demonstration modes are available:

- 15 minutes with a maximum of 50,000 real I/O.
- 10 hours with no static points and a maximum of 1 dynamic real I/O. This is useful for demonstrations using memory and disk I/O. CitectSCADA starts in this mode if no static points are configured.
- If you want to demonstrate DDE, CTAPI, or ODBC writes to CitectSCADA in this mode, you can only write 1 point. To write to more than 1 point, you need to force CitectSCADA to start in 15 minute-50,000 I/O demo mode by creating at least one static I/O point.
  For this to work, you need to configure a real variable tag, with an accompanying PLC or I/O Device. The tag needs to be used by a page or in Cicode. If you do not have a real I/O Device connected, CitectSCADA gives a hardware alert message, which you can disable using the `IODeviceControl` function.
- 8 hours with a maximum of 42,000 real I/O. This is only available through special CitectSCADA Integration Partners (CIP) keys.

Monitoring and Debugging the Runtime System

Debugging CitectSCADA runtime system involves two processes:

- gathering information about Runtime
- analyzing logged data to identify problems

The information gathered from your system will include:

- hardware alarm
- log files

This information can be analyzed directly to identify problems, or you can use the CitectSCADA Kernel to perform advanced debugging.

The Kernel can perform low-level diagnostic and debugging operations, and runtime analysis of your CitectSCADA system. Use it to display low-level data structures, runtime databases, statistics, debug traces, network traffic, I/O Device traffic and so on.

You can also call built-in Cicode function or user-written Cicode functions.

Note: The process involved in debugging device communications is described in the 'Communicating With I/O Devices' section of the help. See Debugging I/O Devices and Protocols.

See Also
Gathering Runtime Information
Using the Kernel

Gathering Runtime Information

There are two types of Runtime information you can use to identify and resolve operational problems:

- hardware alarms
- log files

Hardware alarms are typically displayed on a dedicated alarm page to alert operators to current problems.

Log files are a record of time-stamped system data that can be analyzed to determine the cause of a problem.

See Also
Hardware Alarms
Log Files

Hardware alarms

When an error is detected that affects CitectSCADA's operation, a hardware alarm is generated. Hardware alarms are usually displayed on a dedicated Hardware Alarm page, available as a standard template.

The hardware alarm page indicates what is happening in your CitectSCADA system. If loss of communication occurs, if Cicode can't execute, if a graphics page is not updating correctly, or if a server becomes inoperative, this page shows you. Hardware alarms consist of a unique description and error code.

The hardware alarms do not have detailed information, but serve to point you in the right direction. For example, if you have a Conflicting Animation alarm, CitectSCADA will not tell you the cause. You need to observe which page causes the hardware alarm, and locate the animations yourself.

Note: Do not allow your system to have any recurring hardware alarms.

There are two hardware alarm fields that are not always shown on the hardware alarms pages. ERRPAGE will display the name of the page that was displayed when the error was detected. This is useful for finding errors caused by improperly programmed animations. ERRDESC provides information that is specific to the type of the alarm. For example, if the alarm is an I/O Device error, ERRDESC shows the name of the device.

See Also

Log files

Log Files

CitectSCADA supports the following log files.

Log file	Description
syslog.dat	The syslog.dat file is the primary log file for CitectSCADA. It contains useful system information, from low-level driver traffic and Kernel messages, to user defined messages. Trace options (except some CTAPI traces) are sent to this file. CitectSCADA locks syslog.dat while running. However, you can still view it by using the 'SysLog' command in the Kernel.
tracelog.dat	The tracelog.dat file contains managed code logging, mainly in relation to data subscriptions and updates. Field traces and requests to native drivers go to the syslog.dat or a specific driver log file.
debug.log	This file contains information about a crash or other serious internal issues. If a crash occurs, it will identify the version and path of each DLL being used at the time. It can be used to confirm you have the right version of files.
kernel.dat	Kernel.dat contains a copy of the kernel screens. It has the "dumpkernel(0x8000)" mode added to it on a crash, and is also available via Cicode calls to "dumpkernel".
ipc.log	This log is used for CTAPI communication traffic.
<driver>.dat	Driver logs relate to the operation of a particular driver and are named accordingly. For example, the OPC driver is logged in 'OPC.dat'.

Log files may have additional suffixes included in their name., for example, an archived log file will include a timestamp.

If a system uses separate processes, a log file is appended with the component name. An example of this could be:

```
syslog.IOServer.Cluster1.dat
```

> **Note:** If CitectSCADA suffers an unexpected shut down, the Crash Handler will create a compressed file containing a number of log and data files that may be useful in determining the cause. See The Crash Handler.

Time-stamping

With the release of CitectSCADA v7.20, log file entries use the following timestamp format:

```
yyyy-mm-dd<SPACE>HH:mm:ss.fff<TAB>TZD
```

where:

yyyy = year (e.g. 2008, 2009)

mm = month (e.g. 01, 05, 12)

dd = date (e.g. 01, 02, 31)

HH = hour of the day (24 hour format)

mm = minute (e.g. 00, 02, 59)

ss = seconds (e.g. 00, 02, 20, 59)

fff = milliseconds (e.g. 000, 123, 999)

TZD = local time offset from the UTC (e.g. +10:00, -09.00)

For example:

2009-06-03 11:19:33.249 +01:00

File locations

The CitectSCADA log files are located in the following directory in versions of Windows prior to Vista.

```
C:\Documents and Settings\All Users\Application Data\Citect\CitectSCADA x.xx\Logs
```

In Windows Vista, the files are located in the Program Data directory:

```
C:\ProgramData\Citect\CitectSCADA x.xx\Logs
```

In both cases, driver log files will appear in the Logs directory in a folder named after the particular driver.

Performing Calculations

If you wish to perform calculations on the time and dates recorded in the log files, this can be accomplished by opening the files in Microsoft Excel. Open the file in Excel by selecting settings in the Text Import wizard such that the date and time fields are NOT split into separate cells. Once opened in Excel, select the column containing the date and time and format the cells in the column with a custom format of yyyy-mm-dd hh:mm:ss.000. This will allow Excel to correctly parse the date/time, and store the value in days, as the number of days (and fractions of a day) since 1900-01-01 00:00:00.000 (this date/time has a value of 1).

Calculations between date/time values can be simply performed. However, it needs to be remembered that, since Excel treats times as days, if the difference in milliseconds between two date/times is necessary, the result of the difference calculation needs to be multiplied by (24*60*60*1000).

Parsing the UTC offset

Using the UTC offset in Excel is a little more complicated, as Excel will not accept negative times. Therefore format the UTC offset cell/s as a text string, and write a formula to convert it into a fraction of a day. For example, if the UTC offset is held in cell B4, the formula is:

=LEFT(B4,1)&TIME(MID(B4,2,2),RIGHT(B4,2),0)

This will convert the UTC offset into a correctly signed numerical value expressed as a fraction of one day.

For example:

- "+09:00" is converted to "+0.375"
- "-04:30" is converted to "-0.1875"

This value can then be directly added to the UTC time, to yield local time.

See Also

Configuring Logging

Configuring Logging

You can make adjustments to the way CitectSCADA logs data using Citect.ini parameters. This includes the ability to filter logs according to priority, category or severity.

The available parameters are listed within the Logging Settings page of the Computer Setup Editor. From here, you can learn how each parameter will impact system logging, and make adjustments directly into the local Citect.ini file.

The Logging Settings page is accessible from the home page of the Computer Setup Editor. For more information, see Using the Computer Setup Editor.

Some of the available logging parameters can be updated while your CitectSCADA system is running. For more information, see Adjusting logging during runtime.

> **Note:** Logging can cause a strain on system resources. When configuring logging, be aware of the potential impact on normal operation. For example, a large number of traces can affect CPU performance, while log file archiving may use up disk space.

Archiving syslog.dat

The syslog.dat file is restricted in size (to 2000 kb by default). When it reaches its size limit, CitectSCADA renames it `syslog.bak`, and starts a new `syslog.dat`.

You can make this size restriction larger or smaller by using the `[Debug]SysLogSize` parameter. For example, the following lines in the `Citect.ini` will set the `syslog.dat` size to 30 Mb:

```
[DEBUG]
SysLogSize=30000
```

If you want to archive more than one system log file, you can set `[Debug]SysLogArchive` to 1. This adds a timestamp to the filename.

See Also

Log files

Adjusting Logging During Runtime

You can modify and query logging settings during runtime using the following Cicode functions:

- SetLogging() - allows you to make changes to the current logging configuration without the need to restart a system that is already running. An optional parameter can be used to persist the settings to the INI file.
- GetLogging() - allows you to view current logging settings.

The parameters you can modify and query include the following:

- [CtApi]Debug
- [Debug]DriverTrace
- [Debug]SysLogSize
- [Debug]EnableLogging
- [Debug]Priority
- [Debug]CategoryFilterMode
- [Debug]CategoryFilter
- [Debug]SeverityFilterMode

- [Debug]SeverityFilter
- [Debug]LogShutdown
- [Debug]DebugAllTrans
- [IOServer]RedundancyDebug
- [General]Verbose
- [General]VerboseToSysLog
- [Trend]DebugClient

There is also a subset of the logging related INI parameters that can be modified online within the Citect.ini file, as the system will read their values periodically or on demand. The parameters you can modify in this way include the following:

- TranDebug <name>
- [PubSub]LogLevel
- [PubSub]LogDevice
- [General]ShowDriverError
- [Debug]SysLogArchive
- [Dial]DebugLevel
- [Trend]TrendDebug

The parameters that support this functionality are identified on the Logging Settings page of the Computer Setup Editor.

See Also

Log files
The Crash Handler

The Crash Handler

The Crash Handler can be used to help determine the cause of an unexpected program shut down.

If CitectSCADA suffers an unexpected shut down, the Crash Handler will create a compressed file containing a number of log and data files that may be useful in determining the cause. These files include:

- syslog.dat
- Citect.ini
- tracelog.dat
- kernel.dat
- debug.log
- user.dmp (see below)

In a multi-process system, these files may use extended names, for example:

```
syslog.IOServer.Cluster1.dat
```

You can configure CitectSCADA to save the Crash Handler zip file to the local Logs directory. If an unexpected shut down occurs, the path to the saved zip file will be recorded in the syslog.dat. The compressed file can be also emailed to Schneider Electric (Australia) Pty. Ltd. for analysis, or to a specified email address.

These features are not enabled by default; to enabled and configure the Crash Handler you need to set the [CrashHandler] parameters.

> **Note:** Schneider Electric (Australia) Pty. Ltd. may use the information included in an unexpected shutdown email to help resolve problems in future releases, but it cannot reply or follow up a problem with users directly. To discuss these, contact Technical Support.

Configuring user.dmp

User.dmp is a configurable file that logs exception details. When an unexpected shut down occurs, it captures objects and their variables in memory in a process referred to as a "mini-dump". This dump file can be analyzed by Technical Support.

The information included in this process is adjustable via the parameter [Debug]MiniDumpType. It supports a "light" mini-dump with local stack information for unmanaged code, through to a "heavy" dump (the default setting) including accessible memory for a process and thread information. You can also switch the mini-dump off.

To reduce the amount of disk space used by a heavy dump, it is compressed after it is created (along with other log files) into a file called "Citect32Exception_[timestamp].zip".

To help manage the amount of disk space used by this process, you can adjust the number of Citect32Exception_[timestamp].zip files that are stored using [Debug]MaxMiniDumps. You can also raise a hardware alarm to warn when disk space on the drive where these files are stored falls below a specified minimum amount using [Debug]LogsDriveMinimumFreeSpace.

See Also

Log Files

Using the Kernel

You use the CitectSCADA kernel to perform low-level diagnostic and debugging operations, and for runtime analysis of your CitectSCADA system. Use it to display low-level data structures, runtime databases, statistics, debug traces, network traffic, I/O Device traffic and so on. You can also call built-in Cicode function or user-written Cicode functions.

The Kernel includes the following areas:

- **General** - presents statistics and information on the overall performance of CitectSCADA. For example, this page shows memory usage, summaries of protocol and I/O Device statistics, as well as CPU usage. Access this by using the Page General command.
- **Table** - displays information about CitectSCADA runtime data structures. This area is extensive and is initially difficult to navigate. However, Page Table ***Stats*** is insightful. Access this by using the Page Table command.
- **Driver** - displays statistics and information about the individual protocols running on the I/O Server. Each individual port has its own page of information. Access this by using the Page Driver command.
- **Unit** - similar to the driver information, this shows specific statistics and information about each I/O Device. Access this by using the Page Unit command.

> **Note:** restrict access to the Kernel: anyone using the Kernel has total control of CitectSCADA (and subsequently your plant and equipment).

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Do not use the kernel for normal CitectSCADA operation. The kernel is only for diagnostics and debugging purposes.
- Configure your security so that only approved personnel can view or use the kernel.
- Do not view or use the kernel unless you are an expert user of CitectSCADA and Cicode, or are under the direct guidance of Technical Support for this product.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Access to Cicode and Cache commands

So that there is no unauthorized use of Cicode and Cache commands in the Kernel, only Kernel users have access to these commands. The Kernel user needs to be defined in the User database (with the username 'kernel' and a non-blank password) for the CitectSCADA project in which they want to access the commands. The Kernel user does not need to have any areas or privileges defined.

During runtime if the Kernel user attempts to access Cicode or Cache Kernel commands, the Kernel will ask for the Kernel user password. If the Kernel user is not defined in the User database for that project (or if the user provides the incorrect password), access to the Cicode (or Cache) commands is denied.

See Also

Displaying the kernel window
Inside the kernel

Using Kernel Commands
Kernel commands

Displaying the kernel window

You can display the kernel window in several ways. CitectSCADA can open the window automatically at startup, provide a command option on the Control menu, or you can define a runtime command to display the Kernel window when necessary.

- Displaying the kernel from the control menu
- Displaying the kernel at startup
- Defining a runtime command
- Closing the kernel window

Displaying the kernel from the control menu

To add a **Kernel** option to the Control menu of the runtime system, use the Computer Setup Wizard. Run the wizard, select **Custom** mode, and select the **Kernel on menu** option on the Security Setup | Control Menu page**.**

You can then display the Kernel window by selecting the Kernel option from the Control Menu (top-left corner) at runtime. If you do not have a title bar displayed, you can access the Control Menu by pressing ALT-SPACE (verify that the **Alt-Space enabled** option is selected on the Security Setup - Keyboard page).

> **Note:** Clear these options (the default) after commissioning so that there is no accidental or unauthorized use of the Kernel.

Displaying the kernel at startup

To display the Kernel window automatically when CitectSCADA starts up, set the `[Debug]Kernel` parameter to 1**.** The Kernel window is opened at startup and closed at shutdown. The display is off (0) by default.

> **Note:** Reset this parameter after commissioning so that there is no accidental or unauthorized use of the Kernel.

Defining a runtime command

To display the Kernel window, define a runtime command that calls the `DspKernel()` function, passing 1 in the iMode argument:

Command	DspKernel(1);
Comment	Displays (opens) the Kernel window

To close the Kernel window, call the `DspKernel()` function again, passing **0** in the **iMode** argument:

Command	DspKernel(0);
Comment	Closes the Kernel window

Note: put the highest privilege level on the DspKernel command to prevent your operators from opening the Kernel window.

Closing the kernel window

You can close the Kernel window by choosing **Close** from the Control menu of the main Kernel window.

See Also

Inside the kernel

Inside the kernel

When displayed, the CitectSCADA Kernel consists of an application (client) window called Main and one or more child windows. At startup, the Kernel window displays information about your CitectSCADA startup processes. The Kernel window also displays runtime system messages, providing a continuous operational history of your CitectSCADA system.

The Kernel window contains a command line interface (similar to the Command prompt) where you can type in Kernel commands to perform a Kernel operation or to display other child windows.

Note: Access to the Cache and Cicode Kernel commands is password-protected for security. For details, see Access to Cicode and Cache commands.

By default, CitectSCADA runs in one instance on a single processor. With this configuration, a single Kernel window reports on every processe.

If CitectSCADA runtime is configured to run in multiple instances, each instance has a separate Kernel window that displays instance-specific system messages; for example, trend-related system messages appear in the Trend window in the Kernel. You can switch between Kernel windows by using the Kernel menu on the Kernel window menu bar.

For a default configuration, the Kernel window displays the following messages:

- **Initializing Sub Systems** - The primary parts of CitectSCADA are getting started.
- **Initializing Font System** - Creating fonts that have been defined within CitectSCADA. These are fonts used for displaying items such as alarms, and pre-V5.0 dynamic text.
- **Initializing Client System**.
- **Starting IO Server** - Only visible if the CitectSCADA computer is an I/O Server. If the computer is an I/O Server and this message does not display, the computer is improperly set up: run the Computer Setup Wizard to check your configuration. **IO Server Started** - The server has started and is functioning correctly.
    - **Initializing I/O Server** - Starting to check what is necessary for the I/O Server to work, and initializing any cards that are necessary.
    - On a Client, these messages will be replaced with **Calling '<I/O Server>' Connected**.
- **Initializing Cicode System** - Cicode has been loaded into memory, and is prepared to run.
- **Initializing com System** - Checking that ports and hardware are responding and functioning correctly.
- **Initializing Request System** - The system that handles requests from the Client part of CitectSCADA to the Server parts of CitectSCADA.
- **Initializing Trend Client System** - The Trend Client is slightly different than the normal client, so it needs separate initialization.
- **Starting Trends Server** - You will only see these messages if the CitectSCADA computer is a Trends Server. If the computer is a Trends Server, and these messages do not display, most likely the computer is improperly set up. run the Computer Setup Wizard to check your configuration.
    - **Trend Startup** - CitectSCADA is checking for the trend files, and making new ones if they can't be found.
    - **Initializing Trend Acq System** - Every trend you define has its own sample rate. Here CitectSCADA is setting up the system so it can poll the data at the correct rate for each trend pen.

On a Client, these messages will be replaced with **Calling '<Trends Server>' Connected**.

- **Initializing Alarm System**.

- **Loading Alarm Databases -** You will only see these messages if the CitectSCADA computer is an alarms server. This is loading alarm data into memory. If the computer is an alarms server, and these messages do not display, most likely the computer is improperly set up. run the Computer Setup Wizard to check your configuration.
- **Open Alarm Save File**
  Loading Alarm Save File
  Alarm Save File Loaded - CitectSCADA gets the alarm save file (that was created in the specified directory), and examines it in order to see the status of existing alarms. If you have a redundant alarms server, then this alarms server will interrogate the other instead of using the alarm save file - since the information on the other server will be newer than any file.
  Starting Alarm Processing- The server is now processing (and serving) alarm data. On a Client, these messages will be replaced with **Calling '<Alarm Server>' Connected**.
- **Initializing Report System**.
- **Starting Reports Server** - You will only see this message if the CitectSCADA computer is a reports server. If the computer is a reports server, and this message does not display, most likely the computer is improperly set up. run the Computer Setup Wizard to check your configuration. On a Client, this message will be replaced with **Calling '<Reports Server>' Connected**.
- **Initializing Page System** - CitectSCADA will now display the Startup Page. At this time CitectSCADA will cover up the Kernel if it is displayed.
- **Initializing Functions** - Executing any Cicode functions that have been defined as running at start up.

The next line of information is the start up time and CitectSCADA version number.

- **Channel PORT# is Online**
  Channel PORT# is Online
  Channel PORT# is Online - You will only see these messages if the CitectSCADA computer is an I/O Server. These are messages telling you that any ports you have defined in the I/O Server have come online. If you get a messages saying that the port is not online, or could not be opened, check the configuration of your project. *PORT#* is the Port Name specified in the Ports form.
- **Unit 'UNIT#' Port PORT# is Online**
  Unit 'UNIT#' Port PORT# is Online
  Unit 'UNIT#' Port PORT# is Online - Only visible if the CitectSCADA computer is an I/O Server. This indicates that the I/O Device with the Unit Number of ***UNIT#*** (as defined in the I/O Devices form), is connected to port ***PORT#*** .
- **Communication System Online** - CitectSCADA has completed startup operations and is now fully operational (running).

What to look for

All systems in CitectSCADA start smoothly. When commissioning a system, check the Kernel. If any element does not initialize properly or reports errors at startup, your CitectSCADA system is not working correctly and investigate.

Common causes of startup errors are:

- Incorrect computer setup (usually solved by the Computer Setup Wizard).
- Networking errors or bad hardware.
- Communication errors (usually just a configuration issue).

Use the Main window to check that your I/O Devices come online correctly when starting. First the ports need to initialize, then the I/O Device itself will come online. When a device does not initialize as expected, CitectSCADA displays a message such as "PLC not responding", "I/O Device Offline" or similar.

Some I/O Devices might take two attempts to come online. If so, CitectSCADA waits (usually 30 seconds) and tries again. If the I/O Device does not come online after the second attempt, check your configuration (at both ends) and cabling.

Note: The Kernel continues to report changes in the status of I/O Devices to the Main window. This information might also be reported as alarms to the Hardware Alarms page.

See Also

Using Kernel Commands

Using Kernel Commands

Commands are issued at a command line interface (similar to the DOS prompt), usually from the Main Kernel window. Some commands display their results in the Main Kernel window; others open a child window for information display (or for further commands). You can open a maximum of five windows at once.

You can use several keyboard keys to scan and reuse commands from the command history, to speed up the issuing of Kernel commands. (The command history is a list of commands that you have previously issued). These keyboard keys are listed below:

Key	Description
Up arrow	Scans backward through the command history. (Commands are displayed in the command line.)

Down arrow	Scans forward through the command history. (Commands are displayed in the command line.)
F3	Puts the last command you issued in the command line.
Left arrow	Moves the cursor back one character at a time (in the command line).
Right arrow	Moves the cursor forward one character at a time (in the command line).
Delete	Deletes the character to the right of the cursor (in the command line).
Back-space	Deletes the character to the left of the cursor (in the command line).
Insert	Switches from over-strike mode to insert character mode (in the command line).

Note: When typing a command in the Main window, if a message appears in the middle of your command, you can still execute the command normally. Use theShell command to open a new command window.

See Also

Kernel commands

Kernel commands

The table below describes the kernel commands.

Command	Description
Cache	Changes the cache timeout for each I/O Device.
Cicode	Opens a child window that you can use to call Cicode functions.
Cls	Clears text from the Main or Cicode windows.
Debug	Enables the debugging of raw data transfer between CitectSCADA and a driver.
DriverTrace	Lists the driver control blocks (DCBs) on the I/O Server awaiting delivery to a driver.

Exit	Closes a Cicode or Shell window.
Help	Displays a list of some of the commands available in the Kernel.
INI	Displays the local `citect.ini` file
Kernel Alarm	Deprecated from Version 7.0
Kernel Trend	Deprecated from Version 7.0
Kernel Report	Deprecated from Version 7.0
Kernel IOServer	Deprecated from Version 7.0
Kernel Client	Deprecated from Version 7.0
Log	Enables or disables the logging of I/O Device reads and writes.
NetBIOS	Obsolete in v7.20.
Page General	Displays general statistics information.
Page Driver	Displays information about each driver in the CitectSCADA system.
Page Memory	Displays the memory debug heap.
Page Netstat	Obsolete from Version 7.0
Page Table	Displays information about CitectSCADA's internal data structures.
Page RDB	Displays information about CitectSCADA's Runtime Databases.
Page Unit	Displays information about each I/O Device in the CitectSCADA system.
Pause	Pause debug output.
Probe	Obsolete from Version 7.0

Shell	Opens a new command (shell) window.
Stats	Resets system statistics.
SysLog	Displays the `syslog.dat` file.

Cache

Changes the cache timeout for each I/O Device with which CitectSCADA is communicating.

Syntax

Cache *<I/O Device name> <Timeout>*

Where:

<I/O Device name>

Any valid I/O Device defined in the project (using the I/O Devices form), or * for I/O Devices.

<Timeout>

Timeout in milliseconds, or 0 (zero) to disable timeout.

This command allows you to tune the cache timeout while the I/O Server is communicating with the I/O Devices. If you set the timeout to 0 (zero), the cache is disabled. If you specify a cache timeout for an I/O Device that has the cache disabled, the cache is enabled.

Any changes made to an I/O Device only apply while the I/O Server is running. If you restart the I/O Server, the cache timeout reverts to the value configured in the project. Once you have determined the optimum cache timeout, make the change persistent by setting the value in the I/O Devices form (for the particular I/O Device).

See Also

Kernel commands
Displaying the kernel window

Cicode

Opens a child window that you can use to call Cicode functions, on either a local or remote computer. Any built-in or user-written function can be called from this window.

Syntax

Cicode *[<Name>]*

Where:

<Name>

Optionally the name of a Citect Server (for example, Alarms, Reports, Trends) or a client computer name.

If you enter the Cicode command with no Name argument, a local Cicode window is created.Cicode commands are executed on the local computer.

In a multi-cluster systems when connecting to a server, the following syntax needs to be specified:

```
<clustername>.<server>
```

For example, to connect to the Alarm Server on Cluster1, use:

```
Cicode Cluster1.Alarm
```

If you enter a server or computer name as the Name argument, you can create a Cicode window to the remote server or computer. Cicode commands entered in a remote Cicode window are executed on the remote computer. For example, to create a Cicode window where commands execute on the Alarm Server, use:

```
Cicode Alarm
```

If you issue the command from a server, you can create a window to the "MyComputer" computer:

```
Cicode MyComputer
```

If the remote computer can be found, its name is displayed in the title of the Cicode window, otherwise a local window is created.

> **Note:** You can only specify a computer name if you are issuing the command on a server. This function only supports Client to Server or Server to Client connection.

Each Cicode command is executed with its own Cicode task, so you can start tasks that take a long time to complete. The Cicode prompt returns immediately after the Cicode task has started and the task continues to run in the background. If the function is completed immediately, the return result of the function is displayed. If the function continues to run, the result is not displayed and cannot be returned - the message "Task still running" and the task handle is returned instead.

> **Note:** Remember that there is no Privilege check on any command issued from this window, so you have full access to the system.

The Cicode prompt 0:> shows the current window number with which any object is associated. To change the current window, use the WinGoto() function (or any other Cicode function that affects the current window).

The Cicode window does not recognize any variable names, so when you call a Cicode function, you can only pass constants (for example numbers or strings). When you call a function that expects a string, pass a string constant, for example Prompt("Hello from the Kernel"). If the string is only a single word, you do not have to use delimiters, for example Prompt(Hello). The Cicode window tries to convert whatever you enter (as arguments) into the correct data type. If it cannot convert the arguments, it passes either 0 (zero) or an empty string to the function.

> **Note:** Some Cicode functions are implemented as label macros by the compiler. These macros allow backward compatibility when the number of arguments to a function has been changed. Because the Cicode window does not expand macros, you cannot call these functions directly. You need to use the macro expansion. If the function you are trying to use cannot be found, try again by adding an underscore (_) to the front of the function name, for example _DevClose(1).
> You can also shut down the Citect system from this window by using the Shutdown() function.

See Also

Kernel commands

Displaying the kernel window

Cls

Clears text from the Main or Cicode windows, and moves the cursor to the top left-hand corner.

Syntax

Cls

Use this command to clear the current window when it has become cluttered (from displaying debug data or too many commands).

See Also

Kernel commands

Displaying the kernel window

Debug

Enables the debugging of raw data that is transferred between Citect and a selected driver. Protocol traffic is displayed in the Kernel window and logged to the SysLog.DAT file.

Syntax

Debug*<Port> <Mode> [<Display>]*

Where:

<Port>

A port name configured in the project (using the Ports form)

<Mode>

The mode:

ALL - Trace low-level communication traffic to the Kernel window.

READ - Trace the low-level communication traffic of read commands to the Kernel window.

WRITE - Trace the low-level communication traffic of write commands to the Kernel window.

ERROR - Trace any low-level communication traffic that contains a protocol error. This mode only generates traces when an error is detected, so you can leave this option on for long periods of time (to find difficult problems).

OFF - Stop the debug trace of any type of command.

Note: The TO and FROM modes are now obsolete.

<Display>

Optionally the display scenario:

MONO - Send the debug to a monochrome monitor (if one is attached) as well as the Kernel.

MONOONLY - Send the debug to the monochrome monitor only.

To place a port in debug mode, enter DEBUG followed by the port name and the mode you require. If you do not know the name of the port, enter DEBUG (without any arguments), and Citect displays a list of available ports.

Only the I/O Server communicates with the I/O Devices, so this command is generally used only on the I/O Server.

When you enable a debug trace mode, Citect displays protocol traffic in the Kernel window and logs it to the SysLog.DAT file. This tends to reduce Citect's performance (as there may be a lot of data), and therefore not enable debug trace on an I/O Server that is important to your current operation. Use this command only during commissioning or on a non-vital section. Excessive use of this command may cause the I/O Device to go offline.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not enable or use the Debug Trace mode in an operational environment. This mode is only for use during commissioning.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

When the debug trace is sent to a Monochrome monitor, it is displayed directly from the interrupt routine of the driver and therefore at a much faster rate. This trace (if MONO-ONLY is used) causes less CPU overload of Citect while the trace is active, and provides instantaneous output. This method is useful if you are developing your own driver and your driver crashes before the debug trace is displayed in the Kernel.

You can use the Shell command to create an extra command window while the trace is active. This allows you to enter more commands (in the new window).

You can use the Pause command to stop the debug output (to view the data).

See Also

Kernel commands
Displaying the kernel window

DriverTrace

Lists the commands and information issued to a driver by CitectSCADA. It can be useful in debugging a driver, for example, it can help determine why a driver is unable to fully initialize.

Note: Running syslog traces can draw heavily on a CPU usage. monitor the impact on CPU performance when implementing a large number of traces.

Syntax

DriverTrace
[OFF|CMDS|VER[BOSE]|ERR[OR]|PORT=<*name*>|MASK=<*xxxxxxxx*>|DUMP]

Where:

blank

Returns the current DriverTrace settings. This can be useful to determine the current mask or port settings before implementing a new trace. Just key DriverTrace into the Kernel and hit the return key to view the current settings.

OFF

Turns the DriverTrace off.

CMDS

Turns the DriverTrace on, but does not display the contents of the data buffer.

Note: Specific driver commands need to be enabled with the Kernel command DriverTrace[Mask], or the DriverTraceMask INI parameter.

VER[BOSE]

Turns the DriverTrace on, and displays the contents of the data buffer.

Note: Specific driver commands need to be enabled with the Kernel command DriverTrace[Mask], or the DriverTraceMask INI parameter.

ERR[OR]

Turns the DriverTrace on, but only traces DCBs with errors (DCBs are the internal data structures used for communication between the I/O server and a driver).

PORT

Allows traces to be limited to a particular driver port, by defining a port name. The default setting, PORT=*, will trace all ports.

MASK

A 4-byte hexadecimal number that represents a bit mask used to either include or exclude driver commands from the DriverTrace. The driver commands and their values are as follows:

Command	Bit Position
CTDRV_INIT	00000001
CTDRV_OPEN	00000002
CTDRV_INIT_CHANNEL	00000004
CTDRV_INIT_UNIT	00000008
CTDRV_READ	00000010
CTDRV_WRITE	00000020
CTDRV_CONVERT	00000040
CTDRV_CANCEL	00000080
CTDRV_CPU	00000100
CTDRV_DATABASE	00000200

CTDRV_STOP_UNIT	00000400
CTDRV_STOP_CHANNEL	00000800
CTDRV_CLOSE	00001000
CTDRV_FORMAT	00002000
CTDRV_STATS	00004000
CTDRV_DEBUG	00008000
CTDRV_INFO	00010000
CTDRV_STATUS_UNIT	00020000
CTDRV_INIT_CARD	00040000
CTDRV_UPDATE_INFO	00080000
CTDRV_UI_READ	00100000
CTDRV_UI_WRITE	00200000
CTDRV_EXIT	00400000
CTDRV_UNITACTIVATES	01000000
CTDRV_SUBSCRIPTIONS	02000000
CTDRV_EVENTUPDATES	04000000
CTDRV_STATUS_DISCONNECT	40000000

For example, the value you would use to include only the CTDRV_OPEN, CTDRV_INIT_UNIT and CTDRV_READ commands would be: 0000001A.

Most users will want to exclude the CPU function call, as this happens often. Do this by setting a mask of 7ffffeff.

The default <mask> is 7FFFFFFF.

DUMP

Lists the driver control blocks (DCBs) on the I/O Server awaiting delivery to a driver and actioning from the driver.

Examples

```
-> 07f96fdc Cmd: 03 CTDRV_INIT_UNIT   ,MOCKOPC, Port: PORT1, Unit: IODev1
(UR0,N1)
| 07f96fdc UnitType: 0 (0x0), UnitAddr: 0 (0x0), BitWidth: 1, UnitCount: 16
<-07f96fdc Cmd: 03 CTDRV_INIT_UNIT    ErrDriver 23 (0x17) and took 0ms
-> 07ff1584 Cmd: 04 CTDRV_READ        ,DISKDRV, Port: DISKDRV, Unit: IODevDisk
(UR1,N2)
| 07ff1584 UnitType: 1 (0x1), UnitAddr: 1 (0x1), BitWidth: 16, UnitCount: 5,
RawType: INTEGER
<+07ff1584 Cmd: 04 CTDRV_READ         ErrDriver 0 (0x0) and took 0ms
<+07ff1584 UnitType: 1 (0x1), UnitAddr: 1 (0x1), BitWidth: 16, UnitCount: 5,
RawType: INTEGER
<+07ff1584 000:      1      0      0      0      5
~> 00000000 Cmd: 25 CTDRV_SUBSCRIBE   ,MOCKOPC, Port: PORT1, Unit: IODev1
(UR0,N1)
>-00000000 SUBSCRIBE Tag=Tag4 UpdateRate=500ms
~< 000003e9 Cmd: 26 CTDRV_EVENTUPDATES,MOCKOPC, Port: PORT1, Unit: IODev1
(UR0,N1)
<-000003e9 DS Event UPDATE_MODE=ALL Tag=Tag4 RawValue=0x00
Timestamp=<time_not_set> RawQual=0x0000 DSError=0 (0x0)
```

Explanation of arrows used above

-> Flags information going down to the driver

|

<- Flags information back from the driver straight away

<+ Flags information back from the driver asynchronously (i.e. after some

small time delay)

~> Flags subscription based calls into the driver

>-

~< Flags subscription based updated from the driver

<-

(URx,Ny) addition to UNIT and Data driver traces

UR stands for Unit Record number and is 0 based, so a value of 3 would be the 4th device in the IODevice form. N stands for the (Network) Number in the IODevice form. The use of this is to distinguish between redundant units which may use the same unit name. Thus the trace will positively confirm which unit is in use.

See Also

Kernel commands

Displaying the kernel window

Exit

Closes the Cicode or Shell windows.

Syntax

Exit

> **Note:** You cannot use this command to close the Main window. See Closing the kernel window.

To close every other window, select Close from the window's control-menu box, or press Esc, or double click the control-menu box.

See Also

Kernel commands
Displaying the kernel window

Help

Displays a list of some of the commands that are available in the Kernel.

Syntax

Help

See Also

Kernel commands
Displaying the kernel window

INI

Displays local citect.ini file. (If you are using a Citect Internet Display Client to run a project over the internet, the .ini file in the bin directory will display). You can use the Page Up, Page Down, and arrow keys to move through the file, but you cannot edit or save it.

Syntax

INI

> **Note:** This window is the same that is used to display the syslog.dat file. You can display either the citect.ini or syslog.dat in this window, but you cannot display both at the same time.

See Also

Kernel commands
Displaying the kernel window

Log

Enables or disables the logging of I/O DEVICE reads and writes to the syslog.dat file.

Syntax

Log*<I/O Device> <Mode>*

Where:

<I/O Device>

The name of the I/O Device to log

<Mode>

Either: READ, WRITE, or OFF

See Also

Kernel commands

Displaying the kernel window

Page General

Displays general statistics information on the overall performance of Citect.

Syntax

Page General

General Statistics

Server Name	The server name of this computer or, if it is a client, the name of the primary server this client talks to.
Node Name	The computer name of this computer. You can set this through the Computer Setup Wizard. This is only used if you have Citect in a networking configuration.
Time	The (current) time of day and the date.
CPU Index	An indication of the performance of the Computer. This number provides a rough indication of the performance of the computer. On a Compaq 486/25M it will be 25.
Running	The total time Citect has been running
Stat Reset	The time since the Cicode command to reset the statistics was run.

Memory Total	The total amount of free memory (including virtual memory).
Memory Physical	The total amount of free physical memory. The physical RAM that is free on the computer (not including virtual memory).
Memory Resources	The % of free Windows system resources.
Scheduler Cycles	The number of times that the Citect scheduler is looking for a task to execute. A good indication of how fast the computer can respond to an event. This number depends on the speed of the computer and the CPU resources that Citect is using. This value is completely different depending on whether you are running 32-bit or 16-bit Citect. This is because Citect can use the 32-bit operating system to perform process scheduling, an operation that Citect needs to perform itself in the 16-bit environment. 16-bit: The busier Citect is, the lower the number. Typical values of 10,000 to 50,000; a highly loaded system may drop to below 1000. 32-bit: The busier Citect is, the higher the number. Typical values of 100 to 1000; a highly loaded system may rise above 5000.
CPU Usage	The percentage of the total available computer processing power that is being used on this computer. As the percentage increases to a high level, the performance of Citect may level off or become sluggish. If the CPU usage is consistently very high (greater than 70%), there could be a problem with your system or you may be overloading the CPU. For best performance, run your system between 0% and 40%.
Tasks Per Sec	The number of tasks per second that the Citect scheduler is executing, to show how busy Citect is. This number will be between 30 and 200 tasks per second. If the computer is an I/O Server, the task number is higher (because each protocol uses many tasks).
Lost Errors	Citect keeps track of internal errors in its local error buffers. The Lost Errors counter is incremented when an error is detected and there is no buffer in which to put the alert message. This number should normally be 0 (zero). If this field is incrementing, you may have a problem with your system.

I/O Server Statistics

The following statistics are only incremented if the computer is configured as an I/O Server:

Read Requests	The first number is the total number of read requests sent to the I/O Server from client Citect computers, including the local Citect client. This number continues to increment from 0 (to billions) depending on how fast clients are requesting data and how long the server has been running. The second number is more useful - it records the read requests per second. This value also depends on how fast the clients are requesting data from the I/O Server, and should be between 5 and 400 requests per second
Physical Reads	The first number is the total number of physical reads made to the I/O Devices. Because the I/O Server can optimize the number of read requests made to the I/O Devices, this number is usually smaller than the number of read requests. This number is similar to the read requests and increments forever. The second number is the number of physical reads per second (that the I/O Server is performing). The rate depends on the clients and how fast the server can communicate to the I/O Devices (typically 5 to 200).
Blocked Reads	The total number of times a request was made for the same I/O Device address while the I/O Server was already reading that address. The server blocks the two requests together as an optimization.
Digital Reads	The total number of digital points that the I/O Server has read from I/O Devices.
Digital Reads per Sec	The number of digital reads per second that the I/O Server is processing. This provides a general indication of performance. It is dependent on the protocol.
Write Requests	The first number is the total number of write requests sent to the I/O Server from client Citect computers, including the local Citect client. The second number is the write requests per second (that the I/O Server is performing). The number of requests depends on the rate at which clients are sending write requests to the I/O Server (typically 0 to 20). If the number of requests continually exceeds 10, then this may be causing performance to decline (usually caused by Cicode performing too many writes to the I/O Devices).
Physical Writes	The first number is the total number of physical writes made to the I/O Devices. Because the I/O Server can optimize the number of write requests made to the I/O Devices, this number is usually smaller than the number of write requests. The second number is the number of physical writes per second (that the I/O Server is performing). The number of writes depends on the rate at which clients are sending write requests to the I/O Server (typically 0 to 20). If the number of requests continually exceeds 10, then this may be causing performance to decline (usually

	caused by Cicode performing too many writes to the I/O Devices).
Blocked Writes	The total number of times a request was made for the same I/O Device address while the I/O Server was already writing that address. The server blocks the two requests together as an optimization.
Register Reads	The total number of register points that the I/O Server has read from I/O Devices.
Register Reads per Sec	The number of register reads per second that the I/O Server is processing. This provides a general indication of performance. It is dependent on the protocol.
Cache Reads	The number of read requests serviced from the read cache.
Cache Reads(%)	The percentage of reads serviced from the cache. If the cache read % is large, (for example greater than 40%), the I/O Device cache timeout could be set too high. Try reducing the I/O Device cache time to bring the % cache below 40%. The % of read cache depends on the configuration of your Citect system and the number of clients connected. If you have many clients looking at the same I/O Device data, the cache % may be very high, however this does not usually impact performance. For example, if you have 10 Citect clients viewing the same page, a cache read of 90% would be acceptable.
Cache Flush	The number of times a cache buffer was flushed because a write request was directed to the same location. When you write to the I/O Device and that address is in the read cache, Citect flushes the data from the cache. The next time the data is read, it is reloaded from the I/O Device to reflect the new value.
Cache RD Ahead	The number of read-ahead updates made to the cache buffer. When read ahead caching is enabled, the I/O Server will try to read any I/O Device data which is coming close to cache 'timeout' time. The I/O Server will only read this data if the communication channel to the I/O Device is idle - to give higher priority to other read requests.
Cache Buffers	The number of active cache buffers allocated. Each block of data that is read from the I/O Device requires a cache buffer when stored in the cache. The number of cache buffers active at once depends on the dynamic operation of the Citect computers and their project configurations (typically 0 to 100).
Cache Short	The number of times the cache needed to allocate additional buffers but no buffers were available. When this happens the server cannot cache the data. This does not generate errors but it does lower performance. If this field increments too quickly, increase the available memory.

Response Times	The time taken by the I/O Server to process read and write requests (i.e. the time from when a request arrives at the server to when it is sent back to the client). This time depends on the physical response time of the I/O Device and how long the request had to wait in a queue for other requests to be completed. This time increases as the server's loading increases, and is a good indication of the total system response. The average, minimum, and maximum times are displayed. Typical values depend on the I/O Device protocol, however Citect should have an average response of 500 to 2000 ms. Slower protocols or a heavily loaded system may make the response time higher, but be able to get response times of less than two seconds even for very large systems.
Points	The maximum number of I/O points that can exist in your Citect system. The combined Static and Dynamic count cannot exceed this number. The point limit is part of the Citect software protection, and is programmed into your hardware key.
Max Full	The maximum number of fully functional Citect computers (Full Licenses) that can exist in your Citect system. This number is part of the Citect software protection, and is programmed into your hardware key.
Current Full	The current number of fully functional Citect computers (Full Licenses) that are running in your Citect system.
Peak Full	The peak number of fully functional Citect computers (Full Licenses) that your Citect system has experienced since it was re-started.
Static	The number of static I/O points.
Max Mngr	The maximum number of View-only Clients that can exist in your Citect system. This number is part of the Citect software protection, and is programmed into your hardware key.
Current Mngr	The current number of View-only Clients that are running in your Citect system.
Peak Mngr	The peak number of View-only Clients that your Citect system has experienced since it was re-started.
Dynamic	The number of dynamic I/O points.
Max Dsp	The maximum number of Display-only Clients that can exist in your Citect system. This number is part of the Citect software protection, and is programmed into your hardware key.

Current Dsp	The current number of Display-only Clients that are running in your Citect system.
Peak Dsp	The peak number of Display-only Clients that your Citect system has experienced since it was re-started.

See Also

Kernel commands

Displaying the kernel window

Page Driver

Displays information about each driver in the Citect system. This window is only displayed if the Citect computer is configured as an I/O Server with physical I/O Devices attached.

Syntax

Page Driver

Use Page Up and Page Down keys to scan the driver list.

Driver Statistics

Port Name	The name of the physical port the driver is using for communication.
Protocol	The name of the protocol.
Title	The protocol title and version string. Protocol drivers for Citect Versions 3.xx, 4.xx, and 5.xx, are Version 2.0 type drivers.
Read Requests	The first number is the total number of read requests sent to the driver from every client Citect computer, including the local Citect client. The second number is the read requests per second (that the I/O Server is performing).
Physical Reads	The first number is the total number of physical reads made to the I/O Devices. Because the I/O Server can optimize the number of read requests made to the I/O Devices, this number is usually smaller than the number of read requests. The second number is the number of physical reads per second (that the I/O Server is performing).
Blocked Reads	The total number of times a request was made for the same I/O Device

	address while the driver was already reading or about to read that address. The driver blocks the two requests together as an optimization.
Digital Reads	The total number of digital points that the I/O Server has read from every I/O Device.
Digital Reads per Sec	The number of digital reads per second that the I/O Server is processing. This provides a general indication of performance. It is dependent on the protocol.
Write Requests	The first number is the total number of write requests sent to the driver from every client Citect computer, including the local Citect client. The second number is the write requests per second (that the I/O Server is performing).
Physical Writes	The first number is the total number of physical writes made to the I/O Devices. Because the I/O Server can optimize the number of write requests made to the I/O Devices, this number is usually smaller than the number of write requests. The second number is the number of physical writes per second (that the I/O Server is performing).
Blocked Writes	The total number of times a request was made for the same I/O Device address while the driver was already writing that address. The driver blocks the two requests together as an optimization.
Register Reads	The total number of register points that the I/O Server has read from every I/O Device.
Register Reads per Sec	The number of register reads per second that the I/O Server is processing. This provides a general indication of performance. It is dependent on the protocol.
Cache Reads	The number of read requests serviced from the read cache.
Cache Reads(%)	The percentage of reads serviced from the cache. If the cache read % is large (for example greater than 40%), the I/O Device cache timeout could be set too high. Try reducing the I/O Device cache time to bring the % cache below 40%. The % of read cache depends on the configuration of your Citect system and the number of clients connected. If you have many clients looking at the same I/O Device data, the cache % may be very high, however this does not cause a problem. For example, if you have 10 Citect clients viewing the same page, a cache read of 90% would be acceptable.
Error Count	The total number of errors encountered by the driver.

Short buffers	The number of times the server needed a buffer for the driver but none were available. This can reduce communication performance and result in loss of requests. Increase available memory if this field increments too quickly (that is, more than 10 per minute).
Driver Errors	The number of low-level driver errors encountered before retries were performed. This field may continue to increment even if no errors are reported because the driver retries and it may complete the command on the second attempt. If this field increments quickly (i.e. 10 or more per minute) without other errors being reported, you may have a low-level error that affects performance.
Out of Buffers	The number of times the driver requires a buffer but none were available. The driver needs to discard the data. If this field increments too quickly, increase the number of pending commands the driver can process. This field only increments with client drivers. I/O Device drivers do not require pending buffers.
Time Outs	The number of timeouts encountered by the driver during operations. This field may continue to increment with no visual errors as the driver retries the operation. If this field increments excessively, there may be a communication problem.
Retries	The number of retries executed by the driver. If this field is incrementing, there may be a communication problem.
Maximum Pending Commands	This is the number of requests that are kept in a buffer waiting to be serviced by the protocol. There is a pre defined default value for this for every protocol. If a protocol is capable of handling multiple requests or commands at a time then this number may be high. If the protocol is capable of only handling 1 command at a time then this will probably be 1 or 2.
Blocking Size (Bytes)	This is the range that the I/O Driver will use when blocking read or write requests together at runtime. The default value for this is typically set after quite a lot of experimentation. The value is calculated as the optimum range of data that the I/O Device can respond to, to get the fastest response times from your I/O Device. For example, if the Blocking Size is 100 and you have a graphics page that has Address1 and Address 99 on it, Citect will read both of these addresses in one request. If you have Address 1 and Address 101 on the page Citect will issue 2 separate read requests. The block size may not be the maximum packet size for the protocol, since a particular type of I/O Device may respond faster to smaller requests of data than larger requests. There is one important consideration when using this method; many I/O Devices need to have their Memory tables created by the user. If the user does not define every memory address in a range, then Citect may try and read a block of memory from the I/O Device that does not exist (giving a hardware alarm). This is because Citect will ask for the whole range of addresses between the starting address and the ending address. So, if in our previous

	example the I/O Device did not have address 76, it would report back to Citect that it could not read address 76. The I/O Driver does not know that it doesn't need this address and will retry the command, and in some cases will eventually put the I/O Device offline. Confirm that you always have defined the memory addresses that Citect will need to read.
Timeout Period (ms)	This is the period of time that the I/O Driver will wait before re-requesting data, if no answer comes from the I/O Device.
Maximum Retries	The number of times the I/O Driver will attempt to get data from the I/O Device. Combining this with Timeout gives you the total period before Citect will put an I/O Device offline. If the Timeout=2000 and Retry=2 then Citect will wait 2 seconds for a response, then retry, wait 2 seconds, retry, wait 2 seconds, Offline. Total time between losing communications and deciding it is offline is now 6 seconds. You can modify these parameters, but if you set them too low you will generate unneeded retries and possibly get I/O Device Offline messages.
Poll Time (ms)	This is the time in milliseconds that Citect will check the port for data or write data to the port. If this is 0 then the protocol is operating in Interrupt mode.
Transmit Delay (ms)	This is the time that Citect will hold a packet of data between receiving a response from the last request and sending the new request. This is usually 0, however some protocols can become saturated and start to misbehave. In these cases the default value has been calculated while the protocol was being tested, and modifying this value to something smaller will cause problems. However, making it bigger will only have a very slight impact on the overall response times in your system, but may make the communications more stable.
Watchtime (seconds)	This is the period of time that Citect will wait after deciding an I/O Device is offline before trying to re-establish communications. This is typically 30 seconds. It can be made smaller but not be made smaller than the period that Timeout and Retry will be - otherwise you will not be able to re-establish communications with an I/O Device.
Response Times	The time taken by the driver to process read and write requests (i.e. the time taken to process a single read or write operation to the I/O Device). This time depends only on the physical response time of the I/O Device, because no queue waiting time is included. This field reflects any tuning of the communication channel (for example increasing the baud rate will reduce the response time). The average, minimum and maximum times are displayed.

Channel Usage	This field displays the percentage of the total capacity of the hardware channel that is currently being used. The I/O Server tries to keep the utilization as high as possible, however if the client Citect computers are requesting data slower than the channel can supply, the total will be below 100%. It is possible for the channel usage to rise above 100% as some I/O Device drivers can process more than one command at the same time (having two or more commands using the channel at the same time).
Bytes Per Second	The number of bytes transferred (each second) by the driver. This number provides a simple performance indication that is useful when tuning the driver.
Special Variables	By enabling verbose mode (press V) or by pressing the down arrow, twenty special variables are displayed. The meaning of these variables is driver-specific.

See Also

Kernel commands
Displaying the kernel window

Page Memory

Displays memory debugging information. This window is designed for use by Citect specialists, and requires a high degree of expertise to use.

Syntax

Page Memory

See Also

Kernel commands
Displaying the kernel window

Page Queue

Queues are the main data structure used in CitectSCADA, and this allows the various queues active on the system to be displayed. To view the different queues, press the page up and page down keys.

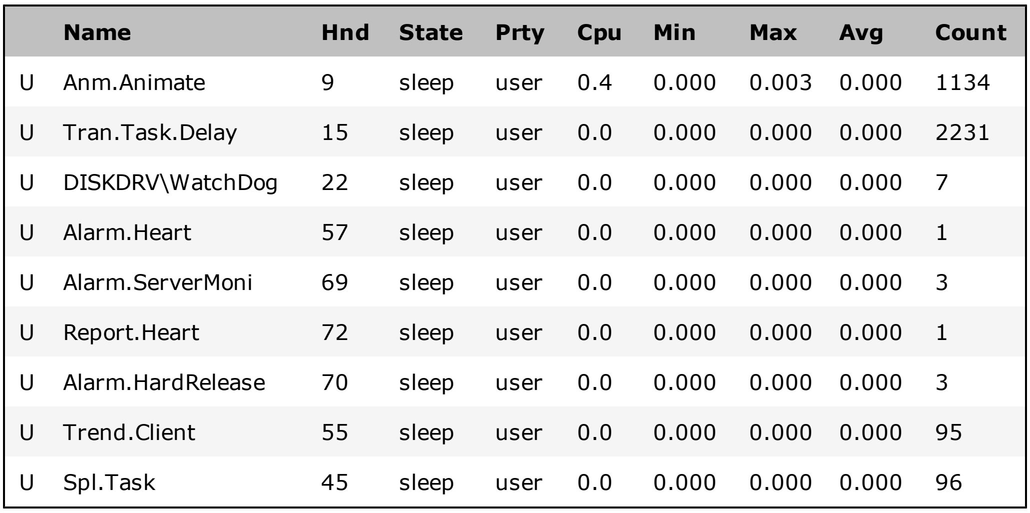
On each page, the handle of the queue being viewed, its name, and its length is displayed. The number of queues and entries within these depends entirely on the custom configuration of Citect32 and the individual project.

Queue formats can vary but several common formats exist.

Example

Queue Sleep

Handle 5 Length 49



	Name	Hnd	State	Prty	Cpu	Min	Max	Avg	Count
U	Anm.Animate	9	sleep	user	0.4	0.000	0.003	0.000	1134
U	Tran.Task.Delay	15	sleep	user	0.0	0.000	0.000	0.000	2231
U	DISKDRV\WatchDog	22	sleep	user	0.0	0.000	0.000	0.000	7
U	Alarm.Heart	57	sleep	user	0.0	0.000	0.000	0.000	1
U	Alarm.ServerMoni	69	sleep	user	0.0	0.000	0.000	0.000	3
U	Report.Heart	72	sleep	user	0.0	0.000	0.000	0.000	1
U	Alarm.HardRelease	70	sleep	user	0.0	0.000	0.000	0.000	3
U	Trend.Client	55	sleep	user	0.0	0.000	0.000	0.000	95
U	Spl.Task	45	sleep	user	0.0	0.000	0.000	0.000	96

This is the most common queue type. Column meanings are as follows :

Mode (Name hidden): U for user-space task, and S for system tasks.

Name: The name of the task.

Hnd : The internal handle for the task.

State: The state of the task, it can be one of: Free, Curr, Ready, Sleep, Wait, Susp and Dodgy.

Prty: The priority of the item in the queue. It can be one of; Low, User and High.

CPU: Shows the percentage of CPU time used by this task.

Min: Minimum response time for the task (from statistics).

Max: Maximum response time for the task (from statistics).

Avg: Average response time.

Count: The number of times this task has been run.

See Also

Kernel commands
Displaying the kernel window

Page RDB

Displays information about each of Citect's runtime databases (RDBs).

Syntax

Page RDB

Use the Page Up and Page Down keys to move between the tables.

The runtime databases contain your compiled project configuration information. There are two types of RDB; resident and non-resident.

Resident databases are loaded at Citect startup and remain in memory. Examples of resident databases are Alarms, Trends, Reports, and Functions. The names of resident databases start with the underscore character (_).

Non-resident databases are those that are associated with pages. These databases are loaded (and unloaded) as necessary - when the page is displayed.

Each database is divided into 10 (or so) tables; <name>, TEXT, REQUEST, PIECE, CODE, SCALE, RUN, WRITE, FUNC, SYMB, and <name>.

See Also

Kernel commands
Displaying the kernel window

Page Table

Contains information about Citect's runtime data structures. This area is very extensive and is initially a bit difficult to navigate. Currently there are about 50 tables of information - most of which are only relevant in specific circumstances and are otherwise not useful.

Syntax

Page Table [<*Name*>]

Where:

<Name>

Optionally the name of the table to display.

Use Page Up and Page Down keys to navigate through the table list. Use Up Arrow and Down Arrow keys to scroll the table data.

The MASTER_TABLE (Table 1) lists every table. The following tables are particularly useful:

Page Table Stats

This very useful table contains the cycle and execution times of every task that is running in Citect. The Execution time is the time taken for the entire task to run. The Cycle time is the time between when a task starts and when it starts again. The CPU is the percentage of total available CPU that the task is using (fast tasks often have 00 CPU).

The Citect 0 entry is the display task (graphics page updates) for the main window. That is, the total time taken for the Client to request data from the I/O Server, the I/O Server to get the data and send it back to the Client, and the Client to update the display.

Note: Citect 0 corresponds to the display task for the main window. Citect 1 for the first child window, Citect 2 for the third, and so on.

The CodeX entries correspond to Cicode tasks, where X is the handle of the task. You can find out which task corresponds to which handle by viewing Cicode table.

Note: There will be a Trend Acq entry for every different trend sampling period you have defined in your project.

Page Table Cicode

This table contains a list of Cicode tasks currently running. It contains the task name, handle and running state, as well as some statistics. CPU_Time is the total time that the task has run for - it is incremented each time the task runs. The CPU is the percentage of total available CPU that the task is using (fast tasks often have 00 CPU).

Page Table Users

This table displays a list of users currently logged into the system either locally or remotely.The Page table users contains the following columns:

Field	Description
Name	The user name for a particular user
Last Login	Time at which user last logged in
Last Logout	Time at which user last logged out
Num connected	Number of references, both internal and external.
Default	Indicates whether user is the default user for the Citect process. Only one user can be a default user.
Logged In	Indicates if user is currently logged in.
Local User	Indicates if user is the local user of the current process. Only one user can be the current processes local user.

Page Table Tran

This table shows a list of channels of communication between CitectSCADA components. A tran exists between exactly two separate components. A client tran initiates a connection, a server tran waits for a connection. Client and server in this context bears no relation to the type of component that owns the tran.

There are two modes for viewing the tran table: Standard and Verbose. The Standard mode shows information for every tran in a tabular format. It can be scrolled using the up/down arrow keys, the top row indicates the currently selected tran. Extended information for the selected tran can be viewed by toggling to Verbose mode by pressing the "v" key. Once in Verbose mode, the tran to display can be changed using the Page Up/Page Down keys.

In standard mode the tran table contains the following columns:

Field	Description
Name	The name and type of communication channel truncated to 20 characters: • Client Format: <cluster name><service name> Examples: Cluster1Alarm, Cluster1Report, Cluster1Trend • Server Format: <server name><cluster name> Examples: AlarmServer1Cluster1, ReportServer1Cluster1, TrendServer1Cluster1 • Dedicated (Client and Server) Format: @@<cluster name><server name> Examples: @@Cluster1.AlarmServer1, @@Cluster1.ReportServer1, @@Cluster1.TrendServer1 • Platform (Client and Server) Format: <server name><cluster name> Examples: AlarmServer1Cluster1, IOServerCluster1
Node	Either the node to which the tran is connected, or the status of the connection: • **<call>** The client tran is attempting to connect. • **<listen>** The server tran is waiting to be connected. • **<disabled>** The tran is currently disabled. • *node name* The tran has connected and is online. The value is the name of the node/computer to which the tran is connected. This could be the current computer or a different computer depending on the TCP/ IP configuration of the project.

Field	Description
Type	This is either: • **Client** - indicates a client tran. This tran actively attempts to connect to a server tran. • **Server** - indicates a server tran. This tran passively waits for a connection attempt from a client tran. • **SerRnd** - indicates a server-to-server redundant tran. This is the connection between the primary and standby servers when a project is configured for server redundancy.

Field	Description
Mode	This is either: • **Local** This tran connects two components which exist in the same process. For instance, when CitectSCADA is run in single process mode, every server component runs in the same process as the client. In this case, the connections between these components would be marked as Local. • **OutPro** (Out of Process) This tran either has an established connection or is attempting to establish a connection between two components via TCP/IP. The components exist in different processes. They may exist on the same computer or on different computers. • **OutPrD** (Out of Process Dedicated) A dedicated connection exists between the client and each server process on a single machine when CitectSCADA is run in multi-process mode. These trans use named pipe connections to verify that a communication path exists between every CitectSCADA component running on one computer regardless of the project's TCP/IP configuration. Dedicated connections do not exist between different computers. • **Platfo** (Platform Tran) This tran uses the Platform networking module as the transport layer between components. For v7.0, it is only used to connect to the I/O Server. **Note**: There are client-side Alarm Server platform trans which will show up in the Tran Table. The server-side Alarm Server platform trans were not necessary or implemented for v7.0 so these client-side trans will always remain in a state of Connecting. They can safely be ignored. • **Remote** This server tran is waiting for a connection attempt via TCP/IP. Upon connection, this mode changes to OutPro.
Hnd	This value represents the handle number in the tran table of the record.

Field	Description
Cnt	This value indicates the number of times the tran has established a connection. Specifically it counts the number of times the tran receives the MSG_OPEN message. This value has no meaning for local trans for which it remains 0. If the number is high, it can indicate that your network is dropping and then re-establishing connections. However, it could also mean that the server has been running for a long time and many clients have started and stopped, thereby closing and opening sessions.
Send	This value displays the number of messages that have been sent by the tran.
Rec	This value displays the number of messages that have been received by the tran.
Wait	This value represents the number of times the tran had to wait to get a buffer in order to send a message.
Stack	This value indicates the protocol number. Historically, this incorporated the NetBIOS LanA numbers with the TCP/IP protocol stacks. However, NetBIOS support was removed for v7.0 while TCP/IP support was enhanced to include redundant Network Interface Cards (NIC). The Stack value displays an index (1-based) which indicates on which redundant IP address the server tran is listening or connected. Its value is only used for out-of-process TCP/IP server trans; it has no meaning for every other tran.
Service	This column displays the type of the service used by the tran (regardless of the tran mode). The valid services are Alarm, IO, Report and Trend.
State	This is the current state of the tran. Valid states are Online, Offline, Connecting (client trans only), Disconnecting, Listening (server trans only) and Disabled.
Login	The user login name for remote Tran connection.

Page Table CSAtoPSI.Subs

Displays a list of client tag subscriptions.

See Also

Kernel commands

Displaying the kernel window

Page Unit

Displays information about each I/O Device in the Citect system. This information is displayed if the Citect computer is configured as an I/O Server or simply as a client. If the computer is a client, then every I/O Device for every I/O Server is displayed. If the computer is an I/O Server, then only the I/O Devices for that I/O Server are displayed. You can display I/O Devices from other I/O Servers by using the Verbose mode (press V to enable Verbose mode). If the computer is a client, then not every one of the I/O Device Information is updated (because only the I/O Server has this information).

Syntax

Page Unit

If the Citect computer is a client, the status and error codes are only local to the computer and do not reflect the true status of the I/O Device on the I/O Server. Be aware that every configured I/O Device is displayed in this window, not just the I/O Devices for the particular I/O Server. (Any remote I/O Devices do not reflect the true status of the I/O Device).

Use the Page Up and Page Down keys to scan the I/O Device list.

I/O Device Information

I/O Device	The name of the I/O Device defined in the project (with the I/O Devices form).
I/O Server	The name of the I/O Server that is servicing this I/O Device.
Comment	A description of the I/O Device defined in the project (with the I/O Devices form).
I/O Device No	The I/O Device number defined in the project (with the I/O Devices form).
PLC Number	The physical I/O Device address defined in the project (with the I/O Devices form).
Port Name	The communication port to which the I/O Device is connected.
Protocol	The protocol used for communication with the I/O Device.
Server Status and Client Status	The status of the I/O Device. The Server Status is only valid if the computer is an I/O Server and it is servicing this I/O Device. The Client Status field is valid for Clients only, and indicates the status of the I/O Device that is attached to the I/O Server. The I/O Device status can be one of the following: RUNNING - Indicates that the communication link with the I/O Device is good. STANDBY - Indicates that the communication link with the I/O Device is good, but communication with that I/O Device is currently being performed by another port. This port is in standby mode. STARTING - Indicates that the server is currently establishing a com-

	munication link (with the I/O Device). STOPPING - Indicates that the server is currently relinquishing control of the communication link (with the I/O Device). OFFLINE - Indicates that the server cannot establish a communication channel with the I/O Device. If a standby port or server is available, Citect tries to communicate to the I/O Device using that port. REMOTE - Indicates that the status of the I/O Device is OK, but it is not currently connected.
Primary	Indicates if the I/O Device is in primary mode; Yes = Primary, No = Standby. If the I/O Device is in primary mode, the server starts a communication channel with the I/O Device as soon as the server is activated. If an I/O Device is in standby mode, the I/O Device remains inactive when the server starts (until a primary I/O Device becomes inoperative).
Client Using	The name of the I/O Server that this client is using. This allows you to identify the primary and standby I/O Servers.
Generic Error	The last generic error code returned by the driver. Because most protocol drivers have their own special errors, they cannot be recognized by the I/O Server. The drivers convert their special errors into generic errors that can be identified by the server.
Error Handle	The error handle that is assigned by the I/O Server to each error. This handle is not used by Citect (at this time).
Driver Error	The driver-specific error code. Each driver has its own special error codes. Refer to the driver specific errors (for the particular protocol) for an explanation of each of the error codes.
Error Message	The alert message associated with the generic error code.
Error Count	The total number of errors from the I/O Device.
Restarts	The number of times the server has tried to establish a connection with the I/O Device. This number is normally 1, because the server establishes a connection at startup. If this field displays a number greater than 1, there is a problem with the communication channel.
Response Times	The time taken by the driver to process read and write requests (i.e. the time taken to process a single read or write operation to the I/O Device). This time depends only on the physical response time of the I/O Device, because no queue waiting time is included. This field reflects any tuning of the communication channel (for example doubling the baud rate will half the response time). The average, minimum, and maximum times are displayed.

	Note: One I/O Device with a slow response can slow down your entire system. For example, if you have an I/O Device with a response of 2000 ms, any pages in your system that use data from that device, will have a minimum update time of 2000 ms.
Cached	This field indicates if the I/O Device data is cached.
Cache Timeout	If the I/O Device is cached, this field displays the cache timeout value. Data is held in the cache for this timeout period before being discarded and re-read from the I/O Device. Only read data is cached.
Blocking Constant	The current blocking constant value for this I/O Device, as specified in the protocol.
Dial-up Connection	The status and history of the dial-up connection. SUCCESS - The number of successful dial-up attempts. FAIL - The number of unsuccessful dial-up attempts. TOTAL - The total number of dial-up attempts. NEXT - The time of the next scheduled dial-up attempt.

See Also

Kernel commands

Displaying the kernel window

Pause

Pauses debug output in the Kernel window.

Syntax

Pause

See Also

Kernel commands

Displaying the kernel window

Shell

Opens a new command (shell) window.

Syntax

Shell

You can use shell windows in a similar manner to a Main window. Shell windows are useful for displaying debugging information, or entering commands when the Main window is displaying debug trace data. You can close the shell windows by selecting Close from the window's system icon or with the Exit command.

See Also

Kernel commands
Displaying the kernel window

Stats

Resets system statistics (used in the page general, page drivers, page table (stats), and page I/O Device windows) to 0 (zero).

Syntax

Stats

This command allows you to reset the statistics after Citect has been running for a long time, and therefore provides an indication of the statistics now (instead of an average over the total time that Citect has been running).

> **Note:** Some I/O Server statistics are automatically reset every few minutes.

See Also

Kernel commands
Displaying the kernel window

SysLog

Displays local SysLog.DAT file. The SysLog.DAT is read from disk before being displayed, but is not updated once it is displayed. You need to close and re-open the window to force it to update. You can use the Page Up, Page Down, and arrow keys to move through the file, but you cannot edit or save it.

Syntax

SysLog [Delete]

Use the optional Delete keyword to clear (purge) the contents of the SysLog.DAT file.

> **Note:** This window is the same that is used to display the Citect.INI file. You can display either the Citect.INI or Syslog.DAT in this window, but you cannot display both at the same time.

See Also

Kernel commands

Displaying the kernel window

Using the Web Client

This section contains information on the Web Client and describes the following:

Chapter: 37 The Web Client

The CitectSCADA Web Client allows you to view a live CitectSCADA project within a Web browser. It provides easy access to CitectSCADA Runtime for LAN-connected users requiring read/write access to current production information.

For example, a senior manager could monitor a facility and access current production information from any computer on the LAN without the need for extensive downloads or software installation.

Note: On 64 bit Windows operating systems (XP 64 bit and Vista 64 bit), there are two Internet Explorer options. One for 32 bit mode and one for 64 bit mode. The Web Client needs to be used in Internet Explorer 32 bit mode.

If you start the Control Client and then start the Web Client on the same machine running Windows Vista operating system, an alert message will be displayed. This will occur for projects that use ActiveX executable components, for example the Example project. In order to use these projects you need to add the Web Server address as a trusted site in Internet Explorer.

See Also

System architecture

System architecture

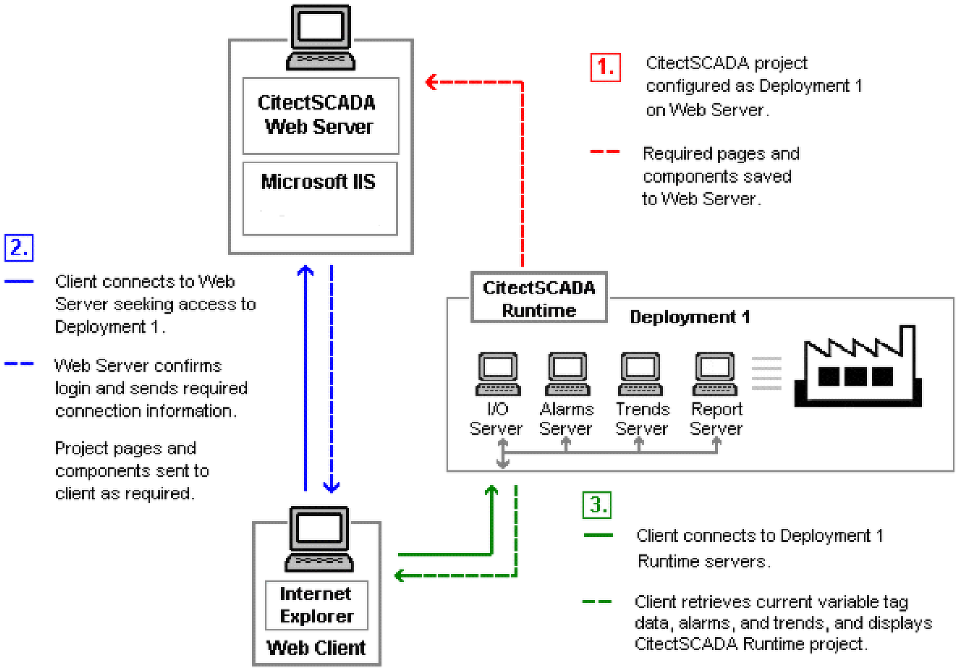
To display a live CitectSCADA project in an Internet browser, you need to combine the content of the project pages and the current data these pages present using standard, Web-based communication protocols. To understand the communication architecture for the CitectSCADA Web Client, it's easiest to consider the role each of the following components play in achieving this outcome:

- **CitectSCADA Web Server** - Performs the server-side functionality of the system. It operates by accepting requests from the client, and providing a response to the client when the clients details are authenticated. It then directs a client to the graphical and functional content of a CitectSCADA project and the location of the runtime servers. This information is stored on the Web Server when a CitectSCADA project is configured as a "deployment". A CitectSCADA Web Server can contain multiple deployments.

- **CitectSCADA Runtime Servers (including the I/O Server, Alarm Server, Trends Server and Reports Server)** - Monitor the physical production facility and contain the live variable tag data, alarms and trends that the Web Client will display.
- **Web Client** - provides the platform to merge a deployed project's pages and content with the raw data drawn from the runtime servers. Again, standard Web technologies are necessary, so the client uses Microsoft Internet Explorer.

The following diagram shows how these components interact.

CitectSCADA Web Client communications architecture.



1.

CitectSCADA project configured as Deployment 1 on Web Server.

Required pages and components saved to Web Server.

2.

Client connects to Web Server seeking access to Deployment 1.

Web Server confirms login and sends required connection information.

Project pages and components sent to client as required.

3.

Client connects to Deployment 1 Runtime servers.

Client retrieves current variable tag data, alarms, and trends, and displays CitectSCADA Runtime project.

Once the Web Client has connected to the Runtime servers, steps 2 and 3 become an ongoing process, with the necessary content being called upon as the user navigates the project pages.

The citect.ini file settings used by a Web Client are taken from the citect.ini file on the Web Server at the time of connection.

This diagram has the system components set up on different computers purely for the sake of explaining the communications model. In reality, the flexibility of the architecture allows these components to be distributed in any necessary arrangement; they can even share a common location.

Getting Started

The CitectSCADA Web Client Help is designed to guide you through the steps necessary to successfully set up a Web Client system.

For detailed information on installing and configuring the web server, refer to the The CitectSCADA Installation and Configuration Guide which is available in a PDF format on the Installation CD, or in the Documentation folder of your CitectSCADA installation.

To facilitate your installation, first familiarize yourself with the System architecture, and then work your way through the following steps, as they will logically guide you through the correct set up procedure.

> **Note:** In order to implement a web client solution you need to first install Microsoft .NET Framework 2.0 on the web client machine.

1. **Preparing a** CitectSCADA **project for deployment**
   This section explains the adjustments that need to be made to a CitectSCADA project prior to deployment on the Web Server.
   - Preparing a Project for Deployment
2. **Configuring a deployment**
   This section describes how to deploy a project on the Web Server, by identifying its source location and associated servers.
   - Configuring a deployment
3. **Multi-language support**
   If you are using IIS as your Web Server platform, there are several language options you can implement on the Web Server interface.
   - Implementing Multiple Language Support

If you have followed the procedures outlined above and your deployed project does not seem to be performing as expected, use the Frequently Asked Questions section to help resolve any issues you might be having.

Preparing a Project for Deployment

Before deploying a project on a Web Server, you will need to makes adjustments in the CitectSCADA configuration environment to get it ready for Web-based delivery. In preparing to make these adjustments, consider the following:

- Functionality limitations of the Web Client platform
- Preparing a project's user files for delivery

- Running the Web Deployment Preparation tool

> **Note:** To use the Web Client, your CitectSCADA system needs to be configured to communicate using TCP/IP. This means the network addresses in your project needs to be defined using standard IP addressing.

Functionality limitations of the Web Client platform

Due to the architecture necessary to support Web-based execution of CitectSCADA projects, the Web Client cannot offer the full functionality of a standard CitectSCADA system.

consider the following list of unsupported features and Cicode functions to assess if this will be detrimental to the performance of your project. Some adjustments might be necessary.

Feature limitations

Cicode Function Limitations

Feature limitations

The following features are not supported:

- Cicode Debugger
- Remote shutdown
- Fuzzy Logic
- Kernel windows
- Keyboard shortcuts that clash with Internet Explorer's keyboard shortcuts
- Web Client is unable to act as a CitectSCADA Server
- Pages based on the default Menu Page template will only show buttons for pages previously visited
- The **Page Select** button on the default Normal template only lists pages previously visited
- The CSV_Include project's Update Page List menu item will not work

> **Note:** If your project is based on the CSV_Include template, you need to create a customized menu to access pages from the menu bar.

See Also

Functionality limitations of the Web Client platform

Cicode Function Limitations

Cicode Function Limitations

Several Cicode functions are unavailable with the Web Client, or limited in their capabilities:

Cluster Functions	Cluster functionality not supported
DebugBreak	Cicode debugger not supported
DelayShutdown	Programmatic Shutdown not supported
FTP Functions	Not every FTP function is supported
Fuzzy Logic Functions	Removed from control due to size
KerCmd	Kernel windows not supported
ProjectRestartGet	Programmatic Shutdown not supported
ProjectRestartSet	"
ProjectSet	"
Shutdown	"
ShutdownForm	"
SwitchConfig	Configuration environment not available
TraceMsg	Kernel windows not supported
UserCreate	Changes to user profiles need to be made on the machine where the project is compiled and auto-deployed.
UserCreateForm	"
UserDelete	"

UserEditForm	"
UserPassword	"
User-PasswordForm	"
GetWinTitle	Windows other than the main window only
WinFree	"
WinMode	"
WinMove	"
WinPos	"
WinSize	"
WinTitle	"
WndShow	"
WndViewer	Invokes multimedia applications. Feature not supported

See Also

Preparing a project's user files for delivery

Functionality limitations of the Web Client platform

Feature limitations

Preparing a project's user files for delivery

If the content of your CitectSCADA project incorporates user-created files, such as DBF files, HTML files, or CSV files, you need to manually place these into a special zip file called **Misc.zip** for delivery to the Web Server. Similarly, if a project contains ActiveX objects, these also needs to be included in a zip file called **ActiveX.zip**.

To prepare any user-created files for deployment:

1. Identify the user-created files that are associated with the project you want to deploy. These files could include CSV or DBF files associated with tables presented on project pages, or HTML content.

2. Use a compression tool to zip these files up into a single file called **`Misc.zip`**.
3. Place `Misc.zip` in the main folder for the project. For example, in the case of the Example project, this would be:
   `[User]\Example`

   > **Note:** If your project has included projects that use ActiveX objects, verify that these are also zipped up in an `Activex.zip` file in the included project's directory.

The files are now ready for deployment on the Web Server.

To prepare any included ActiveX objects for deployment:

1. Identify the ActiveX objects associated with the project you want to deploy.
2. Use a compression tool to zip these files up into a single file called **`ActiveX.zip`**.
3. Place `ActiveX.zip` in the main folder for the project. For example, in the case of the CSV_Include project, this would be:
   `[User]\CSV_Include`

> **Note:** If an ActiveX object has an associated data source, verify that the data source can be located by the computer hosting the Web Client. See the topic Managing associated data sources under the section on ActiveX objects in the CitectSCADA User Guide Help.

See Also

Running the Web Deployment Preparation tool

Running the Web Deployment Preparation tool

The final step in preparing a project for deployment involves running it through the Web Deployment Preparation tool. This takes a freshly compiled project and creates the necessary files and directories for Web-based delivery.

To run a project through the Web Deployment Preparation tool:

1. Verify that the project you want to deploy has its associated user files and ActiveX objects zipped up for delivery (see Preparing a project's user files for delivery).
2. Locate the project you want to deploy in Citect Explorer and do a fresh compile.
3. Go to the Citect Explorer **Tools** menu and select **Web Deployment Preparation** (or click the following icon on the Explorer toolbar):
4. A progress indicator appears. The size of the project significantly affects how long this process takes; a large project with many files can take over ten minutes to

process, depending on your hardware. (You can abort the deployment preparation if you want.)

5. When complete, a dialog appears stating the preparation was successful. Click **OK**. The project is now ready for deployment on the Web Server. If you change a project, you need to do a fresh compile and run the Web Deployment Preparation tool again.

> **Note:** You can run the Web Deployment Preparation tool automatically when you compile a project. To do this, go to the Citect Project Editor **Tools** menu and select **Options**. Select the **Prepare for Web Deployment** option and click **OK**. Be aware that this increases the time taken for each compile, particularly for large projects.

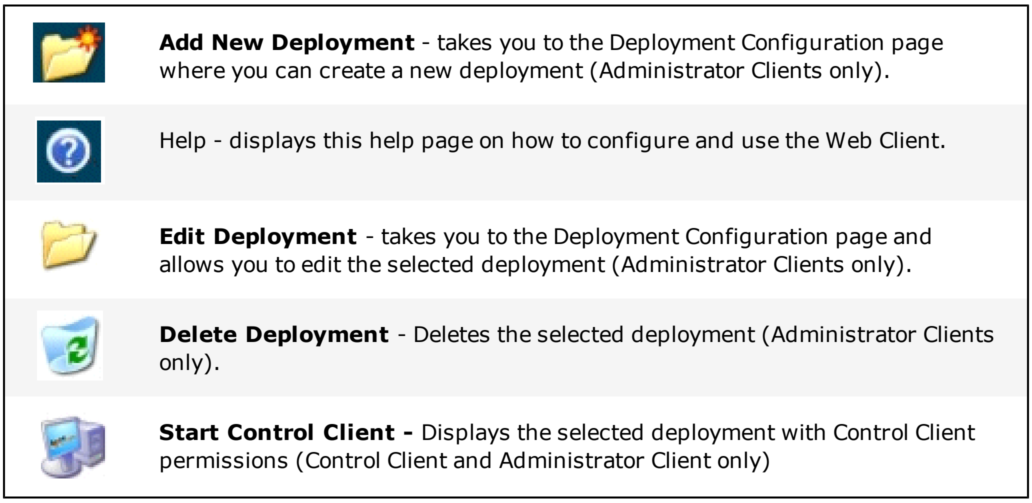
Configuring a deployment

A deployment represents the implementation of a CitectSCADA project on the Web Server. It incorporates the files and components necessary to display a project, and keeps a record of the location of the servers where CitectSCADA Runtime data is generated.

The deployments configured on a Web Server are listed on the Web Client home page, which is the page that appears when you initially log in. The configuration details for a deployment can be displayed by clicking the small plus (+) icon to the left of the deployment name.

The type of action you can implement for a deployment depends on the permissions granted by your log in. For example, if you log in as a View-only Client, you can only view a deployment. If you are an administrator, you can edit deployments and create new ones.

The following list describes the functionality associated with each of the icons presented on the home page.:



	Add New Deployment - takes you to the Deployment Configuration page where you can create a new deployment (Administrator Clients only).
?	Help - displays this help page on how to configure and use the Web Client.
	Edit Deployment - takes you to the Deployment Configuration page and allows you to edit the selected deployment (Administrator Clients only).
	Delete Deployment - Deletes the selected deployment (Administrator Clients only).
	Start Control Client - Displays the selected deployment with Control Client permissions (Control Client and Administrator Client only)

Start View-only Client - Displays the selected deployment with View-only Client permissions

Additionally, the **System Messages** panel provides notification of events that impact the current status of the Web Server.

> **Note:** The Citect.ini file settings used by a Web Client are taken from the Citect.ini file on the Web Server at the time of connection. This includes which clusters a Web Client has access to. To switch clusters in a project viewed using a Web Client, use the functions ClusterActivate and ClusterDeactivate.

See Also

Preparing a Project for Deployment
Creating a new deployment
Deploying a project from within CitectSCADA
Displaying a deployment
Editing an existing deployment
Updating a deployment to reflect project changes
Deleting a deployment

Creating a new deployment

To configure a deployment of a CitectSCADA project on a Web Server, you need to log in to the Web Client with Administrator permissions. This will provide you with access to the full functionality of the home page.

To add a new deployment

1. Click the **Add New Deployment** icon.

   This will display the Deployment Configuration page.

2. Type a name in the **Deployment** text box, and include a **Description** if necessary. A deployment name cannot contain any of the following characters: `\ * ? | . , / " ' : ; < > # &`

   > **Note:** If you've upgraded your version of the Web Client, you can still view your legacy deployments that you created using version 6.1 or earlier of the Web Client. For details, see Web Client Upgrade Considerations.

3. Identify the source of the CitectSCADA project's content in the **Project Path** field.

If the project is located locally on the Web Server, you can use a normal path address. The path needs to point directly to the project within the CitectSCADA`User` directory. For example, the location of the MyExample project would be:

```
[User]\MyExample
```

> **Note:** If you are remotely administering the Web Server and use a local path address, verify that the path represents the location of the project on the Web Server computer, not the computer you are currently using.

If the project is not located on the Web Server, you need to use a UNC address that identifies the host network computer and the directory it can be found in. For example,

```
\\ComputerName\<path to application data>\User\MyExample
```

> **Note:** You need to share the directory a project resides in to allow the Web Server access to it. Ideally, create a share from the directory (called WebShare, for example) and then use the following project path:

`\\`*ComputerName*`\WebShare`

Remember that if you are trying to access the project directory from a remote computer, a "local" administrator log in will not provide you with appropriate access on a different computer. use a network user profile that will be recognized by other computers on the same domain.

4. Determine if any of the I/O, Alarms, Reports or Trends Servers associated with the project are protected by a firewall. If they are, you need to confirm with the firewall administrator if the CitectSCADA ports have been opened to allow direct access, or if the firewall is using address forwarding.
   If forwarding is being used, you will need to identify each server by typing the name in the **Server** field, using the following format:

```
<ClusterName>.<ServerName>
```

If you have alarm properties enabled on an Alarm Server, you will need to configure an alarm properties connector as a separate server to let the Web Server know which port it is running on. Type the Alarm Server name in the **Server** field, using the following format:

```
<ClusterName>.<AlarmServerName>_AlarmProps
```

For example:

```
ClusterOne.AlarmServerOne_AlarmProps
```

Type in the **Address** and **Port** for each server, as supplied by the firewall administrator.

> **Note:** The Web Client will automatically add any servers that are redirected in this way to the [AddressForwarding] section of the local Citect.ini file. See Using address forwarding for more information.

You can add additional servers to the list by selecting the **Add New Server** icon.

5. Use the **Client Control** text box to specify the use of a particular version of the Web Client component when the deployment is displayed.
   The menu lists the different versions of the Web Client control currently installed on the Web Server. Typically, choose the version of the control that matches the version of CitectSCADA your project was compiled on.
6. Click **Apply Changes**.

   This is important, as you'll lose your changes if you jump straight back to the home page.
   All the project files are retrieved from the path indicated, and copied to the Web Server ready for access by the Web Clients.
   Once complete, information about the size of the project appears in the File Paths banner above the Project Path field. The number to the left indicates how many files are included in the project; the number to the right indicates the total size of the project.

The deployment is saved. When you return to the Web Client home page, by clicking the home icon, your new deployment is listed.

See Also

Deploying a project from within CitectSCADA
Displaying a deployment

Deploying a project from within CitectSCADA

The Web Client architecture lets you deploy a project from within the CitectSCADA configuration environment, avoiding the need to use the Web Client interface to setup a system.

This process requires you to adjust two parameters in the Citect.ini file:

- `[WebServer]WebClientCab`
- `[WebServer]DeployRoot`

These parameters identify the client component used with the project and the location of the deployment root directory. When the project is compiled and prepared for deployment, it is placed directly on the Web Server.

Notes:

- If you've upgraded your Web Client to version 7, you can still view your legacy deployments. For details, see Web Client Upgrade Issues.
- When implementing this option, pay attention to your `Citect.ini` file configuration, as any errors with these parameters are difficult to diagnose. To avoid input errors, use the Web Deployment Tool on the Citect Explorer tool bar with the Web Server's Web Deployment GUI.
- If the project name contains non-ASCII characters, deploying from within CitectSCADA might be unsuccessful and raise errors. Under these circumstances, use the Web Server interface to create the deployment.

To deploy a project from within CitectSCADA:

1. Confirm that your CitectSCADA system is configured to use TCP/IP. If you run the Computer Setup Wizard, the **Networking** page will identify which communications protocol is being used.
2. Adjust the [WebServer]DeployRoot parameter within the `Citect.ini` file. This parameter represents the directory where the deployment will be located on the WebServer. If you have set up an IIS-based Web Server, the default location will be the Deploy directory within the installed directories. For example:

```
[webserver]
DeployRoot="C:Program Files\Citect\CitectSCADA 7.10\WebServer\deploy"
```

> **Note:** When setting the `[WebServer]DeployRoot` ini parameter, the path needs to contain "deploy" as the last subfolder name, otherwise the deployment will be unsuccessful. Use a mapped drive instead of a UNC address if deploying to a network destination from a Windows 2000 system. Do not map a drive directly to the deployment location, as the path needs to finish with a "deploy" subfolder.

3. Adjust the **[WebServer]WebClientCab** parameter within the `Citect.ini` file. This parameter represents the directory path and client component to use when a deployment is run, in relation to the installed Client directory. For example:

```
[webserver]
WebClientCab=700/CitectSCADAWebClient_7_0_176.cab
```

   Note the use of a forward slash in the defined path.

4. Compile your project and then prepare it for deployment. Go to the Citect Explorer **Tools** menu and select **Web Deployment Preparation** or select the following icon on the Explorer toolbar.

   Your project will now appear as a deployment within the Web Client home page next time you log in.

> **Note:** You can run the Web Deployment Preparation process automatically when you compile a project. To do this, go to the Citect Project Editor **Tools** menu and choose **Options**. Select the **Prepare for Web Deployment** option and click **OK**. Be aware, however, that this might increase the time necessary for a project to compile.

5. Determine if any of the I/O, Alarms, Reports or Trends Servers associated with the project are protected by a firewall. If they are, you need to confirm with the firewall administrator if the CitectSCADA ports have been opened to allow direct access, or if the firewall is using port forwarding.
   If port forwarding is being used, you will need to log in to the Web Client as an Administrator, select the project, and then the **Edit Deployment** button:

   📂

   This will take you to the deployment configuration page.
6. Identify each server that port forwarding is being used for by typing the name in the **Server** field, using the following format:

```
<ClusterName>.<ServerName>
```

   If you have alarm properties enabled on an Alarm Server, you will need to configure an alarm properties connector as a separate server to let the Web Server know which port it is running on. Type the Alarm Server name in the **Server** field, using the following format:

```
<ClusterName>.<AlarmServerName>_AlarmProps
```

   For example:

```
ClusterOne.AlarmServerOne_AlarmProps
```

7. Type in the **Address** and **Port** for each server, as supplied by the firewall administrator.

> **Note:** The Web Client will automatically add any servers that are redirected in this way to the [AddressForwarding] section of the local Citect.ini file. See Using address forwarding for more information.

   You can add additional servers to the list by selecting the **Add New Server** icon.

See Also

Displaying a deployment

Displaying a deployment

When you display a deployment, it downloads the necessary Web Client component file from the Web Server, enabling you to run the associated CitectSCADA project in your Web browser.

> **Note:** The Citect.ini file settings used by a Web Client are taken from the Citect.ini file on the Web Server at the time of connection. This includes which clusters a Web Client has access to. To switch clusters in a project viewed using a Web Client, use the functions ClusterActivate and ClusterDeactivate.

To display a deployment:

1. Locate the deployment you want to display in the list of available deployments.
2. Click the relevant icon (**Start Control Client** or **Start View-only Client**) to display the deployment.

The display options available to you depend on your login permissions. If you select the View-only Client icon (the one with the gold lock), you can only read the current values for the CitectSCADA project.

Once the necessary project files and components have been downloaded, the CitectSCADA project appears. You can now navigate the project pages as necessary.

> **Note:** An alert message might appear if the current user on the client machine does not have Windows administrator rights when a new or updated component file (.cab file) is downloaded. Verify that the current Windows user has administrator rights if a new deployment is run or an updated .cab file needs to be downloaded.

See Also

Editing an existing deployment

Editing an existing deployment

If necessary, you can edit the settings for a deployment. For example, you can change the name of the deployment or specify a new address for a runtime server.

To edit a deployment's settings, you need to be logged in as an Administrator Client.

To edit an existing deployment

1. Select the deployment you want to edit in the list of available deployments.
2. Click the **Edit Deployment** icon.

   This takes you to the Deployment Configuration page.
   Change the fields as necessary. For field descriptions, see Creating a new deployment.

   > **Note:** If you give a deployment a new **Name**, it is duplicated instead being updated and overwritten. This allows you to easily copy an existing deployment; however, to avoid confusion, delete the original deployment with the old name if it's no longer necessary.

3. Click **Apply Changes**. (This is important, as you'll lose your changes if you jump straight back to the home page.)

The Web Server retrieves a fresh set of pages and components for the CitectSCADA project, which will include any recent changes.

See Also

Updating a deployment to reflect project changes

Updating a deployment to reflect project changes

If you change a source CitectSCADA project, you need to update its associated deployment to publish these changes on the Web Server.

Updating a deployment uploads the latest project pages and components to the Web Server for distribution. This is important as discrepancies might occur between the project pages and the data being pulled from the runtime servers if the content is not up to date.

To update a deployment:

1. Verify that the project you want to update has been compiled and processed within the CitectSCADA by the Web Deployment Preparation tool. See Running the Web Deployment Preparation tool.
2. Select the deployment you want to update.
3. Click the **Edit Deployment** icon.

This takes you to the Deployment Configuration page.

4. Click **Apply Changes**.

The Web Server retrieves a fresh set of pages and components for the CitectSCADA project, which will include any recent changes.

See Also

Editing an existing deployment

Deleting a deployment

Deleting a deployment

To delete a deployment from a Web Server, you need to log in as an Administrator Client.

To delete a deployment from the Web Server:

1. Select the deployment you want to delete from the list of available deployments.
2. Click **Delete Deployment**.

   A dialog asks you to confirm that you want to delete the deployment. Click **OK**.

See Also

Configuring a deployment

Implementing Multiple Language Support

The Web Client deployment configuration interface can be displayed using languages other than English. The following languages are supported by default:

- French
- German
- Spanish
- Chinese
- Japanese
- Korean

You can also implement other languages by translating the resource message file that defines the text displayed. In the case of the languages listed above, this file has already been translated with a version for each language stored in the installed `locales` folder.

See Also

How default languages are implemented
Using a language different to the current system locale setting
Implementing a non-default language

How default languages are implemented

When you connect a client computer to the Web Server, the script on the web page automatically detects the language code currently defined as the default for the browser. This code is drawn from the system locale setting defined in **Control Panel|Regional Options** on the client machine.

Once the browser's language code has been determined, the script attempts to match it with those available on the Web Server. If a match is made, the associated language is automatically used for the Web Client deployment configuration interface. If a match cannot be made, it defaults to English.

For example, if your Windows **Locale** setting is Chinese (PRC), the language code set for your browser would be "**zh-cn**". This is compared to the current list of language codes on the Web Server, which by default is the following:

Language	Windows Language Code
English	en
French	fr
German	de
Spanish	es
Simplified Chinese	zh
Japanese	ja
Korean	ko

Having found no match in "zh-cn", the script tries to load Simplified Chinese language, "zh", as a match. The interface will automatically display in Chinese.

See Also

Using a language different to the current system locale setting

Using a language different to the current system locale setting

You can display the content of the Web Client's deployment configuration pages using a language that's different to the current system locale setting for the computer. To do this, use a URL query string in the address field of your browser.

To switch to a language other than the default:

1. Decide which language you want to use and determine its associated language code. (See How default languages are implemented for a list of the codes for the default languages supported by the Web Server).
   For example, if you want to use Chinese, the code necessary would be **zh**.

   > **Note:** If the language you want to use is not one of the supported languages, you need to create and translate your own message file. See Implementing a non-default language.

2. Use a URL query to indicate the language you want to use for the Web Client deployment pages. For example, if the address field on your browser currently reads:
   `http://localhost/CitectSCADA`
   add a "/`?lang=`" query to the end of the address. For example, Chinese would be:
   `http://localhost/CitectSCADA/?lang=zh`

   > **Note:** If you use a code that represents a regional variation of one of the default languages and that specific code cannot be matched, the Web Server can only implement the available default version of the language. For example, using the language code for Chinese (PRC), "zh-cn", results in the Simplified Chinese being used, "zh".

Your Web browser now displays the Web Client's deployment configuration pages using the appropriate language.

See Also

Implementing a non-default language

Implementing a non-default language

If you need to use a language on the Web Client's deployment configuration interface other than one of the default languages supported by the Web server, you can implement your own translation of the messages file that defines the text that appears.

To display a language other than those supported by default:

1. Using a text editor that supports the language you want to edit, open one of the existing message files located in the Web Server's `locales` directory; the default path is:
   `C:\Program Files\Citect\CitectSCADA 7.10\Web Server\locales`
   Open a file that includes the language that will be easiest to translate. The language code at the start of each file name can be used to identify the language each file represents; for example, the English language file is called **enmsg.xml**.
2. Save the file back to the `locales` directory, using the appropriate language code in the name.
   To name the file correctly, check the list of Windows Language Codes for the appropriate code. This will allow your translated resource file (XXmsg.xml) to be automatically loaded when the Web Client home page is launched, provided it matches the current system locale setting.
   For example, to implement Hebrew on the Web Client's configuration pages, you would name your file **hemsg.xml**. To use the Taiwanese variation of Chinese, you would call the file **zh-twmsg.xml**.
3. Now change the file content. Firstly, set the correct encoding format.
   The encoding format is defined in the top line of the file, which appears as follows:
   `<?xml version="1.0" encoding="iso-8859-1" ?>`
   If the language uses English characters, the format you would use is ANSI, which is defined as "iso-8859-1" (see example above).
   If the language uses non-English characters, you would use Unicode, which is defined as "UTF-8" (see example below).
   `<?xml version="1.0" encoding="UTF-8" ?>`
4. Now translate the text that appears on the Web Client interface.
   The content that needs to be translated is divided across two sections within the file: "labels" and "messages". The labels section includes the content used to describe and identify the elements of the interface; the messages section includes the notifications that appear in the system messages panel.
   To translate these sections, alter the text between the enclosing XML tags. Do not alter the tags themselves. The XML tags define where each label is used.

   > **Note:** Make sure you maintain any "%" characters, as these are used to insert system information.

   For example, the English file:

```
<!-- Labels -->
<span id="TITLE">CitectSCADA Web Client Deployment</span>
<span id="SYSMSG">System Messages</span>
<span id="DEP">Deployment</span>
<span id="DESC">Description</span>
<span id="ACTION">Action</span>
<!-- Messages -->
<sysmsg id="DELOK">% deleted.</sysmsg>
<sysmsg id="DELCAN">% will NOT be deleted.</sysmsg>
<sysmsg id="DEPNULL">You can't % an empty deployment.</sysmsg>
```

...

would appear as follows in Spanish:

```
<!-- Labels -->
<span id="TITLE">Despliegue del Cliente Web CitectSCADA</span>
<span id="SYSMSG">Mensajes del Sistema</span>
<span id="DEP">Despliegue</span>
<span id="DESC">Descripción</span>
<span id="ACTION">Acción</span>
<!-- Messages -->
<sysmsg id="DELOK">% eliminado.</sysmsg>
<sysmsg id="DELCAN">% NO será eliminado.</sysmsg>
<sysmsg id="DEPNULL">No puede % un despliegue vacío.</sysmsg>
```

Once you have translated the file and saved it with the appropriate name to the `locales` folder, your Web Server will be able to support the language.

> **Note:** When you save your file, make sure the text editor you used saves the file in the appropriate format depending on the language coding used, i.e. ANSI or Unicode (UTF-8) . See step 3 above.

Web Client Upgrade Considerations

If you have upgraded your Web Client from an earlier version to version 7.0, read the following sections about installation and how to use the upgraded Web Client tool with existing deployments.

Installation

If you intend to use the Web client on Windows 2000 or Windows 2003 Server, you need to first install the latest Windows Installer module on your machine. This is available via the Windows Update feature in Windows 2000 or Windows 2003 Server.

Upgrading to the new version of the Web Client adds a new folder named `700` to the `client` folder of the WebServer folder.

If you're planning on installing Web Client version 7.0 but want to view legacy (that is, pre-version 7.0) deployments, before installing the new version, back up your old deployments to a durable storage medium and place them in a secure location. Then, after installing the new version, copy your old deployment(s) back to the `deploy` folder (see above), and the legacy .cab file(s) to the corresponding folder in the `client` folder; this will make your old deployments available for use.

Creating new deployments

When creating a new deployment, be aware that the .cab file you use (**Client Control**) for the deployment needs to correspond to the correct version of the project you want to access or the deployment will not succeed. For example, to create a deployment based on a version 6.0 CitectSCADA project, from the **Client Control** menu choose **600/***filename*.cab; to create a deployment based on a version 7.0 CitectSCADA project, from the **Client Control** menu choose **700/***filename*.cab.

Deploying a project from within CitectSCADA

If you are deploying a project from within CitectSCADA, edit the `[webserver]` section in your `citect.ini` file to specify the correct cab file for the version of the Web Client you're using. For example, for a version 7.0 deployment, specify a .cab file located in the `700` folder; for a 6.0 version, specify the `600` folder.

Frequently Asked Questions

This section answers frequently asked questions concerning the Web Client. One section is dedicated to considerations pertaining to Windows 2003 Server, and the other to general considerations:

- Windows 2003 Server-related considerations
- General considerations

Windows 2003 Server-related considerations

This section describes considerations relating to the Windows 2003 server product.

Q. My Web Client Deployment Page displays incorrectly on Windows 2003 Server: Show Server Details is missing and the icons for Start client, Delete Deployment, and Edit Deployment are also missing. How do I fix this?

A: There are two possible answers:

- When IIS 6.0 is installed, it defaults to a "locked" mode, meaning it can serve up only static content. ASP, ASP.NET, and FrontPage Server Extensions are disabled and needs to be explicitly and separately enabled. The CitectSCADA Web Server needs ASP enabled on the IIS.

To enable ASP for IIS6 on Windows 2003 Server:

1. Choose **Start|Control Panel**, then double-click **Add or Remove Programs**.
2. In the Add or Remove Programs dialog box, click **Add/Remove Windows Components**.
3. In the Windows Components Wizard dialog box, select **ApplicationServer**, and then click **Details**.

4. In the Application Server dialog box, select **Internet Information Services(IIS)** and click **Details**.
5. In the Internet Information Services (IIS) dialog box, select **World Wide Web Service** and click **Details**.
6. In the World Wide Web Service dialog box, select the **Active Server Page** option.
7. On Windows 2003 Server, the default setting is to have the web locations except localhost as an untrusted site. Consequently you need to modify your browser's security settings.

To update your Trusted Sites settings for Windows 2003 Server:

1. Choose **Tools | Internet Options**.
2. Click the **Security** tab and then **Trusted Sites | Sites.**
3. In **Add this Web site to the zone** field, add the web server's IP address as follows:

```
http://<ip address>
```

Note: Whenever Web Client pages do not load, display, or perform correctly, verify your security settings even if you are not running Windows 2003 Server.

Q: When I try to start the Web Client, I get the alert message "Starting Citect Web Client Failed: Can not initialize Citect system", and then the Web Client browser window stops responding to my inputs. How do I correct this?

A: First check that you haven't accidentally deleted the `#DisplayClient` folder from the installed Web Server directory, as this will cause this error. By default, this directory is located at:

```
C:\Program Files\Citect\CitectSCADA 7.10\WebServer\
deploy\#displayclient
```

If this is not the case, this is due to an incorrect MIME setting: the initialization files are not being recognized in Windows 2003 as registered file extensions. To correct this, add the correct MIME type extension by doing the following:

1. Run the IIS manager and go to **Web Sites | Default Web Site | Citect | deploy**.
2. Choose **Properties** from the folder's right-click menu.
3. Go to **HTTP Headers | MIME types**.
4. In the Extension field, enter "**.***".
5. In the MIME type field, enter "**all**".
6. Restart your Web server and client.

General considerations

This section describes general considerations relating to the Web Client product.

Q. When I try to run a deployment in Internet Explorer, I get the following error: "Problems with this page might prevent it from being displayed properly...". What is the cause?

A. When you first try to run a deployment, Citect will attempt to download the client component (the .cab file) associated with that deployment if it is not already present on the local machine. If this download of the client component is triggered, and the currently signed in user does not have Windows local administrative rights on the client, then this alert message occurs.

The solution is to verify that the person who runs a deployment for the first time is a Windows local administrator on the client machine. Once the components have been downloaded, any user will be able to access and run the deployment. The alert message will not reoccur unless the .cab file is updated.

Q. I deployed a project from within CitectSCADA using the appropriate Citect.ini [WebServer] parameters, but the project does not appear in the list of deployments on the Web Server. A dialog informed me that the deployment was successful. What has happened?

A. This situation can occur if you make an error with the syntax for the `[WebServer]DeployRoot` parameter. If, for example, you use a curly bracket instead of a square bracket, (for example, "[WebServer}DeployRoot"), the compiler cannot read the parameter and deployment files are sent to the CitectSCADA project directory instead:

`[User]\<Project Name>`

The deployment is flagged as successful, but it cannot be located by the Web Server.

If you have deployed a project but it does not appear in the Web Server's list, check the location above for a subfolder called "Web Deploy". If such a folder exists, correct the syntax used in your `Citect.ini` file.

Q. I deleted a user from the list of users configured for access to the Web Server, but they can still log in. How do I deny them access?

A. Sometimes a user can connect to the Web Server even after their user account has been deleted. This can occur when the operating system does not immediately report the user deletion to the web server. The period between the deletion of a user and the restriction of access can be about a half an hour.

The solution is to deny access to the user before deleting them. That way, they cannot gain access. See the topic "Deleting a user account" in the Web Client section of the CitectSCADA Installation and Configuration Guide.

Q. When I try to run the Web Client component for the first time, I get a "System Settings Change" message instructing me to restart my computer. What do I do?

A. This message appears on computers that contain old versions of some system files necessary by the Web Client Control. If these files are used by another application during installation, this System Settings Change message appears. Click **OK** to restart your machine to allow the newer versions of the necessary files to be installed during system reboot. The alert message will not reoccur unless another application reloads the old files.

Q. One of the ActiveX Object's included in my project cannot locate its associated data source. Where is it?

A. If an ActiveX object has an associated data source, you need to verify the data source can be located by the computer hosting the Web Client. See the topic Managing associated data sources for details.

Q. Why does a pop-up saying "Client control (CitectSCADAWebClient_7_0_xxx.cab) is not in the option list!" when I try to edit my deployment from the Web Client Deployment Configuration Page?

There are two possible reasons for this dialog:

1. You might have set the [Web Server] parameter WebClientControl incorrectly. The Web page might not recognize the name or version of the .cab file. also check that you've used a forward slash in front of the cab file name, as this is necessary for the Web Server to locate the correct file.
2. A user might have deleted the necessary .cab file from `C:Program Files\Citect\CitectSCADA 7.10\WebServer\client\700`(or the specified location). Therefore, the Web page cannot find it.

Q. The Process Analyst interface normally displays in a foreign language as I translated the language resource DLL, but it displays in English on the Web Client platform. How do I correct this?

A. A Process Analyst control running inside a CitectSCADA Web Client supports runtime language switching, but you need to configure which languages the Web Client will download to the client machine.

To configure the languages to download:

1. Create a zip file in the `C:\Program Files\Citect\CitectSCADA 7.10\WebServer\client\700` folder called bin.zip.
2. Add to the zip file the language resource DLL files that you want the client to download and use. (You can find these files in your `\Program Files\Common Files\Citect` folder.)

> **Note:** The bin.zip file and its contents are not version-checked. This means you

need to manually remove the `bin.zip` from the Web Client machines if your server contains a more recent `bin.zip` file. To do this:

1. Find the installation directory of the Analyst.dll file on your Web Client machines and look for a file called **bin.zip** in this directory.
2. Delete this file.
3. Reconnect to the Web server to download the latest bin.zip file.

Q. I have keyboard shortcuts configured in my CitectSCADA project, but they do not work properly when the project is deployed in the Web Client. What's wrong?

A. Keyboard shortcuts configured for Internet Explorer (IE) take precedence over keyboard shortcuts configured within your CitectSCADA projects. For example, the Example project has F11 assigned to call up Help on a selected animation point on a graphics page. If the project is run as a Web Client deployment, F11 will toggle the view to full screen, as is the case normally with IE.

This is a limitation of using Internet Explorer to host CitectSCADA projects. The easiest solution is to return to the CitectSCADA configuration environment and assign your shortcuts so that no clashes occur. See the Internet Explorer Help for details of preconfigured keyboard shortcuts.

Q. I can't print from the Web Client. Why not?

A. You *can* print from the Web Client, but not by using your browser's **File | Print** command. Instead, in your CitectSCADA project, create a Print control that uses the Cicode `WinPrint()` function to print the page you want.

Q. The new page that I added to my CitectSCADA project does not appear in the Page Select list or the default menu page in the Web Client. How can I correct this?

A: If the page you added to your CitectSCADA project does not appear in Web Client, you can manually type in the page name in the Page Select list to view this new page. In this version of the Web Client, the new page is not added to the default menu page.

Q. How does the Web Client deal with ActiveX controls (for example, CiMeterX.ocx) and user files (Recipes.dbf, for example) that are necessary by a user project?

A. If your user project requires files such as these, you need to create special zip files to contain them. Create an `ActiveX.zip` file to contain the ActiveX files necessary by your project, and a `Misc.zip` file to contain other files that your project needs; for example, `recipes.dbf`, `Chinese.dbf`, `Japan.dbf`, and so on. Add these files under the main project path (for example, `[User]\Example`).

Note: You can have subfolders within the zip files, but your project needs to be configured to use the same relative path structure.

During compilation, any zip files that contain supporting files necessary by a CitectSCADA project are copied to the Webdeploy subfolder. During startup, the Web Client will check the timestamp of any zip files to determine if the zip files have been updated; if the files have been updated, the zip files will be downloaded.

Q. My project was created using CitectFacilities and incorporates the Citect TimeScheduler. The TimeScheduler is not working when I run the project on the Web Client. What's wrong?

A. If you want to run the Time Scheduler on a CitectSCADA Web Client, you need to verify that the user profile you log in with has appropriate network access to the configuration tool, and the location of the configuration files. The user needs to be able to execute the configuration tool, and write to the configuration files.

Q. The Web Client Deployment Page displays incorrectly on Windows 2000 Advanced Server. 'Show Server Details' is missing, and the icons for Start client, Delete Deployment and Edit Deployment are also missing. What is wrong?

A. This appears to be caused by Windows Automatic Update installing several components at the same time after a fresh install of the operating system. Even though Internet Explorer might have been upgraded to the latest version (for example, 6.0.2800.1106) it might still behave as a version 5 browser; for example, it offers limited support for "iframes". If you call up **About Internet Explorer** from the Help menu, and a Version 5-style dialog appears with a version 6 release number, then your computer is affected in this way.

A complete uninstall/reinstall of Internet Explorer will correct the problem.

Q. When I launch the web server, the login form prompts me for a user name and password. When I enter the valid local administrator user name and password the HTTP 500 Internal server error message pops up saying that the page cannot be displayed. What is wrong?

A. The problem is caused by the fact that the computer was in an invalid state in the way that it lost its trust relationship with the domain controller. Removing the computer and re-adding it to the domain will correct the problem. To do so perform the following steps:

1. Open the Computer properties dialog. (WinKey + Pause/Break).
2. On the Computer Name tab click Change Enable workgroup and enter any string, such as "Test".
3. Click Ok and enter the credentials, Apply (no need to restart at this point) Click Change again, Enable domain and enter "Citect.com".
4. Click Ok, enter the credentials, click OK to reboot.

Q. When I start the control client from a web client the Internet Explorer information bar displays telling me that it has blocked the running of scripts or ActiveX controls.

A. One of the reasons for this issue is Internet Explorer security preventing the download of any ActiveX controls. To resolve this issue, add the web site to the list of trusted sites under Tools -> Internet Options -> Security

Chapter: 38 Windows Language Codes

code	Windows locale setting	code	Windows locale setting
af	Afrikaans	hu	Hungarian
sq	Albanian	is	Icelandic
ar-sa	Arabic (Saudi Arabia)	id	Indonesian
ar-iq	Arabic (Iraq)	it	Italian (Standard)
ar-eg	Arabic (Egypt)	it-ch	Italian (Switzerland)
ar-ly	Arabic (Libya)	ja	Japanese
ar-dz	Arabic (Algeria)	ko	Korean
ar-ma	Arabic (Morocco)	ko	Korean (Johab)
ar-tn	Arabic (Tunisia)	lv	Latvian
ar-om	Arabic (Oman)	lt	Lithuanian
ar-ye	Arabic (Yemen)	mk	FYRO Macedonian
ar-sy	Arabic (Syria)	ms	Malaysian
ar-jo	Arabic (Jordan)	mt	Maltese
ar-lb	Arabic (Lebanon)	no	Norwegian (Bokmal)
ar-kw	Arabic (Kuwait)	no	Norwegian (Nynorsk)
ar-ae	Arabic (U.A.E.)	pl	Polish
ar-bh	Arabic (Bahrain)	pt-br	Portuguese (Brazil)

code	Windows locale setting	code	Windows locale setting
ar-qa	Arabic (Qatar)	pt	Portuguese (Portugal)
eu	Basque	rm	Rhaeto-Romanic
bg	Bulgarian	ro	Romanian
be	Belarusian	ro-mo	Romanian (Moldavia)
ca	Catalan	ru	Russian
zh-tw	Chinese (Taiwan)	sz	Sami (Lappish)
zh-cn	Chinese (PRC)	sr	Serbian (Cyrillic)
zh-hk	Chinese (Hong Kong SAR)	sr	Serbian (Latin)
zh-sg	Chinese (Singapore)	sk	Slovak
hr	Croatian	sl	Slovenian
cs	Czech	sb	Sorbian
da	Danish	es	Spanish (Traditional)
nl	Dutch (Standard)	es-mx	Spanish (Mexico)
nl-be	Dutch (Belgium)	es-gt	Spanish (Guatemala)
en	English	es-cr	Spanish (Costa Rica)
en-us	English (United States)	es-pa	Spanish (Panama)
en-gb	English (United Kingdom)	es-do	Spanish (Dominican Republic)
en-au	English (Australian)	es-ve	Spanish (Venezuela)

code	Windows locale setting	code	Windows locale setting
en-ca	English (Canada)	es-co	Spanish (Colombia)
en-nz	English (New Zealand)	es-pe	Spanish (Peru)
en-ie	English (Ireland)	es-ar	Spanish (Argentina)
en-za	English (South Africa)	es-ec	Spanish (Ecuador)
en-jm	English (Jamaica)	es-cl	Spanish (Chile)
en	English (Caribbean)	es-uy	Spanish (Uruguay)
en-bz	English (Belize)	es-bo	Spanish (Bolivia)
en-tt	English (Trinidad)	es-sv	Spanish (El Salvador)
et	Estonian	es-hn	Spanish (Honduras)
fo	Faeroese	es-ni	Spanish (Nicaragua)
fa	Farsi	es-pr	Spanish (Puerto Rico)
fi	Finnish	sx	Sutu
fr	French (Standard)	sv	Swedish
fr-be	French (Belgium)	sv-fi	Swedish (Finland)
fr-ca	French (Canada)	th	Thai
fr-ch	French (Switzerland)	ts	Tsonga
fr-lu	French (Luxembourg)	tn	Tswana
gd	Gaelic (Scotland)	tr	Turkish
gd-ie	Gaelic (Ireland)	uk	Ukrainian

code	Windows locale setting	code	Windows locale setting
de	German (Standard)	ur	Urdu
de-ch	German (Switzerland)	vg	Valley Girl
de-at	German (Austria)	ve	Venda
de-lu	German (Luxembourg)	vi	Vietnamese
de-li	German (Liechtenstein)	xh	Xhosa
el	Greek	ji	Yiddish
he	Hebrew	zu	Zulu
hi	Hindi		

Using the Tab Style Page Templates

This section contains information on the Tab Style Page Templates and describes the following:

Introducing Tab Style Page Templates
Using Pages and Templates
Creating a New Project

"Tab Style Page Templates Reference" in the CitectSCADA Technical Reference

Chapter: 39 Introducing Tab Style Page Templates

The Tab Style Template Project is a preconfigured project that includes a set of templates styled for the Windows 7 environment. It will help you reduce the time necessary to configure a new project.

When a new CitectSCADA project is created, the Tab Style Template (Tab_Style_Include) project is automatically incorporated as an included project. This means the project's templates are available for implementation when creating your graphics pages in Graphics Builder.

As well as including a standard graphics page template for creating plant mimics, the project also includes predefined process analyst and alarm templates, a set of file templates for displaying text, Rich Text Format and HTML files. They have identical tab style navigation and alarm menus for consistent "look and feel" across an entire project. The project also supports multimonitor displays, allowing you to simultaneously display several graphics pages across several computer screens.

Note: Do not modify the Tab Style Template project for use as a runtime project. It will not compile successfully, and be set aside for use as a template for new projects. CitectSCADA upgrades install a new version of the Tab Style Template project, which will overwrite changes you make to the project when this happens.

Where to Find Information

Use the following links for details about the preconfigured content in the Tab Style Template Project:

- Predefined templates. See Using Templates.
- Common toolbars. See Common Toolbars.

It also describes the processes used to create a project based on Tab Style templates:

- Creating a new project. See Creating a New Project.
- Creating pages. See Creating Pages.
- Creating custom menus. See Creating Custom Menus.

Reference material is also included, describing the citect.ini parameters and Cicode functions available to customize a project.

Chapter: 40 Using Pages and Templates

When you create a new project based on the Tab Style templates, the following templates are available for you to use as you create your graphics pages. These templates share Common Navigation Functionality, such as a customizable tabbed menu, a navigation toolbar and an alarm toolbar. This section describes the templates and their buttons, menus and tools.

The Tab Style template project has the following templates:

Template Type	Description
Normal	Standard graphics page. See Normal Page Template.
Blank	For users who want to create their pages from scratch. See Blank Page Template.
Alarm	Includes Alarm, Disabled, Hardware and Summary for displaying different types of alarm list. See Alarm Page Templates.
Process Analyst	Includes SinglePA, DoublePA, PopPA. See Process Analyst Page Templates.
SPC	Includes SpcPareto, SpcCpk, SpcXRSChart, EventSPCXRS, MeanMeanChart, RangeChart and StandardChart. See SPC Page Templates.
File	Includes File, File_RTF and File_HTML for displaying a plain text file, rich text format file and HTML file / URL respectively. See File Page Templates.

See Also

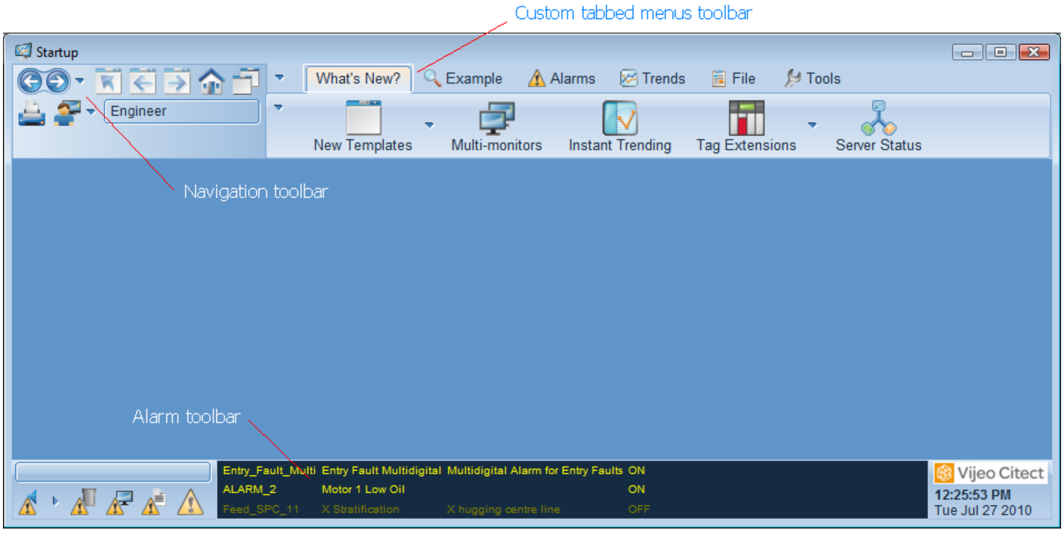
Creating a New Project
Creating Pages

Common Navigation Functionality

The tab style page templates include common toolbars for easy navigation and feature access, as well as a consistent appearance. This helps ensure pages that are created based on these templates will look and operate in the same style.

The following three toolbars remain on screen during operation:

- Custom tabbed menus toolbar: Provides tabbed menus capable of navigation to a specific page or calling Cicode expression. The content of the menus (and its associated buttons and sub-menus) is read from the menu configuration database configured via Project Editor.
- Navigation toolbar: Provides navigation buttons and common page functions such as direct access to home page, printing of current page and user login.
- Alarm toolbar: Provides access to alarm pages and displays the last three active alarms.



Entry_Fault_Multi	Entry Fault Multidigital	Multidigital Alarm for Entry Faults	ON
ALARM_2	Motor 1 Low Oil		ON
Feed_SPC_11	X Stratification	X hugging centre line	OFF

12:25:53 PM
Tue Jul 27 2010

See Also

Custom Tabbed menus Toolbar
Navigation Toolbar
Alarm Toolbar

Custom Tabbed Menus Toolbar

The tab style templates provides a tabbed toolbar to present the custom menus defined in the menu configuration database. The tabbed menu toolbar consists of following components:

Tab buttons

Tab buttons are equivalent of selecting a menu option in a traditional menu system. Each tab contains a list of menu items (represented as icons) relating to that tab.

Menu icons (buttons)

Each tab contains a row of menu icons, each of which either performs a function such as calling up a page or running a Cicode expression.

Sub-menu drop down lists

Optionally, a drop-down list may be provided next to the menu icon for further options. You can also view it as a sub-menu to the top-level tab. In this case, the menu icon serves as a collection point for related sub-menu items.

Hidden menu items

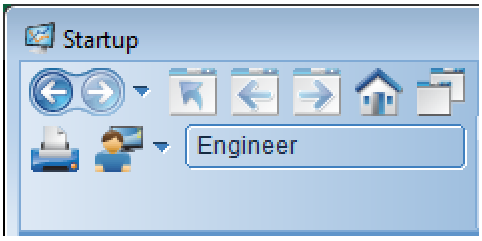
Where there are too many menu items to fit comfortably within the menu bar (either for the tabs or the menu icons), a pair of drop-down list buttons are supplied at either end of the menu bar which provide access to the menu items. The horizontal arrow buttons on the menu bar displays only those items not visible on either side of the screen. The vertical arrow button on the left side of the menu bar displays menu items (either tabs or the menu icons for the currently selected tab)

See Also

Creating Custom Menus

Navigation Toolbar

The Navigation toolbar includes clickable icons to quickly access the common pages and functions from every page in the project.



The following items are available from the toolbar:

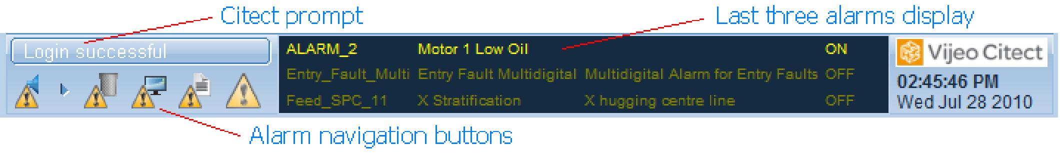
Icon	Name	Description
←	Goes back one page in the page history	Goes back to the page displayed before the current page.
→ ▾	Goes forward one page in the	Goes forward to the page that was displayed prior to the back button being pressed. The arrow to the right of the button allows you to select from menu of recently visited pages.

Icon	Name	Description
	page history	
↖	Displays the parent page	Changes the display to the "parent page" of the current page. You can assign a parent page to a graphics page by setting an environment variable in Graphics Builder. To do this, open the page you want to assign a parent to. Go to the properties dialog for the page (**File | Properties**), and click the **Environment** tab. Add a new variable called "ParentPage" with a value of the page name of the parent page.
←	Displays the previous page	Moves backwards through pages in a browse sequence if a sequence is configured. To configure a browse sequence, go to the Page Properties dialog (**File | Properties**) for each page in the sequence and set the **Previous** field on the **General** tab accordingly.
→	Displays the next page	Moves forwards through pages in a browse sequence if a sequence is configured. To configure a browse sequence, go to the Page Properties dialog (**File | Properties**) for each page in the sequence and set the **Next** field on the **General** tab accordingly.
🏠	Displays the home page	Displays the "home" page. A home page act as a default page for the project. To set a home page, use the [Page]HomePage parameter.
⧉	Displays the Page Select dialog	Displays a dialog box with a list of graphics pages defined in the project. The user can select a page name for display.
🖨	Prints the current page	Prints the current graphic page to the selected printer using the built-in WinPrint() Cicode function. You can specify your own printing function for the page by setting an environment variable in Graphics Builder. To do this, open the page you want to assign a printer function to. Go to the properties dialog for the page (File | Properties), and click the Environment tab. Add a new variable called "PrintPage" with a value of Cicode command in the syntax of "?" followed by a valid Cicode function name and comma separated arguments, for example, ?MyPrintFunction "arg1","arg2".

Icon	Name	Description
	Login / Logout	Displays the standard CitectSCADA login prompt. The menu to the right offers extra options. The default options are Login, Logout, Change Password, Edit User (restricted by login) and Create User (restricted by login). Logged in user is indicated to the right of the button. You can override the default options by defining your own login menu in the menu configuration database via Project Editor. The Page field and Level 1 field of these menu items needs to be set to "Template" and "Login" respectively.

Alarm Toolbar

The alarms toolbar provides access to current alarm information and navigation buttons for single-click access to alarm pages. It also includes standard CitectSCADA prompt and current date and time.



```
ALARM_2           Motor 1 Low Oil                                               ON
Entry_Fault_Multi Entry Fault Multidigital  Multidigital Alarm for Entry Faults OFF
Feed_SPC_11       X Stratification          X hugging centre line               OFF
```

02:45:46 PM
Wed Jul 28 2010

The alarms toolbar contains the following features:

Icon	Name	Description
n/a	**Last Alarms**	The center panel of the toolbar displays the last alarms triggered within the system, providing the operator with an immediate visual cue to alarm occurrences. You can modify the format of this list and control which alarms appear. For example, you can set the list to only display alarms of a particular type, category or priority. For details, see the Alarm parameters LastAlarmFmt, LastAlarmCategories, LastAlarmPriorities, and LastAlarmType.
	Active Alarms	Changes the display to the active alarms page. This button is animated and will blink when there is an unacknowledged alarm in the system. The page displayed by this button is determined by the parameter [Page]AlarmPage.
	Alarms Summary	Changes the display to the alarm summary page, which provides a historical log of alarm occurrences. The page displayed by this button is determined by the parameter [Page]SummaryPage.

⚠	**Hardware Alarms**	Changes the display to the hardware alarms page. This is an animated button which will blink when an unacknowledged hardware alarm occurs. The page displayed by this button is determined by the parameter [Page]HardwarePage.
⚠	**Disabled Alarms**	Changes the display to the disabled alarms page. The page displayed by this button is determined by the parameter [Page]DisabledPage.
⚠	**Alarm Silence**	Silences the audible alarm buzzer that sounds when there are pending unacknowledged alarms. See Implementing Audible Alarms.

Normal Page Template

A template designed for creating user-necessary content and plant mimics. This page contains the custom tabbed menus, the standard navigation and alarm toolbars featured on tab style templates. If you are creating a project based on the tab style templates, use this template for your standard graphics pages.

The Normal template creates a page in display format to be shown with a title bar. Several sizes of templates are available including XGA (1024 x 768), SXGA (1280 x 1024), HD1080 (1920 x 1080) and WUXGA (1920 x 1200).

Note: Starting in version 7.20, you can customize the behavior of the standard window titlebar to lock down its functions to achieve the same result that used to be only possible by hiding it in the previous versions. This is achieved by overriding the titlebar events or setting the startup mode of the window.

See Also

Creating Tab Style Pages
Securing the Window Titlebar

Blank Page Template

The blank template is included for users who want to create their pages from scratch. There are no differences between the tab style blank template and the template in the include project.

See Also

Creating Pages

Alarm Page Templates

Templates are included for the following types of alarm displays:

- **Active Alarm Page (Alarm)**: Used to create a page displaying active alarms that are unacknowledged or acknowledged and still in alarm state. To create a default active alarm page for your project, please create a new page based on this template and name it "Alarm". You can change the default page name by setting parameter [Page]AlarmPage.
- **Hardware Alarm Page (Hardware)**: Used to create a page displaying details of any system errors that are unacknowledged or acknowledged and still in alarm state, for example, if communications are lost, if Cicode can't execute, if a graphics page is not updating correctly, or if a server becomes inoperative, this page will alert you. To create a default hardware alarm page for your project, please create a new page based on this template and name it "Hardware". You can change the default page name by setting parameter [Page]HardwarePage.
- **Disabled Alarm Page (Disabled)**: Used to create a page displaying alarms that are presently disabled in the system. The alarms will appear on this page when they have been disabled from the normal alarm system due to a maintenance shutdown, nuisance tripping etc. To create a default disabled alarm page for your project, please create a new page based on this template and name it "Disabled". You can change the default page name by setting parameter [Page]DisabledPage.
- **Alarm Summary Page (Summary)**: Used to create a page displaying a historical log of alarms that have occurred. This page can be used for troubleshooting purposes. To create a default alarm summary page for your project, please create a new page based on this template and name it "Summary". You can change the default page name by setting parameter [Page]SummaryPage.

The new alarm templates (except the hardware alarm template) display alarm lists using a "no draw" mode. The display of alarm fields is handled by a new set of template specific Cicode functions and genies to control each of the alarm fields' position and size. The alarm list is installed with a callback event function to redraw the alarm list upon data change. The "no draw" mode and data change callback event are supported by a modified form of the existing Cicode function AlarmDsp().

Common Functionality

The alarm page templates offered as part of the tab style page templates include common graphical user interface (GUI) features for easy operation and consistent appearance.

These include:

- Task Panels - Provides controls to carry out common tasks that apply to the whole of the alarm list.
- Interactive Alarm List - Provides controls to manipulate the columns displayed on the alarm list and select alarms.

See Also

Task Panels
Interactive Alarm List

Task Panels

The alarm page templates share panels to the left of the alarms list that support the following functionality:

Page Tasks

Allows the user to navigate through the list of alarms a page at a time. The Print / Export hyperlink (not available in the hardware alarm page) allows the user to send the alarms on the list to a printer or a file. The output will be presented in a HTML table, and reflects the columns, filter and sort settings that are currently used on the display. There are options to output available alarms (starting from page 1) or specific pages of alarms staring from the current page off the alarm list. The blue box indicates the current page number of the alarm list.

Filter Tasks

Allows the user to filter the current list based on date time, alarm name, plant area, alarm category and other criteria. The Reset filter hyperlink clears the effect of the filter criteria specified through the filter form. The blue box indicates the current filter setting. It may display one of the following states:

- No Filter - no filter is applied
- Advanced Filter - filter criteria specified in the filter form is applied
- Custom Filter - filter criteria is applied via Cicode function AlarmSetInfo() by the user

Control Tasks

Offers the option to:

- Acknowledge alarms on the current page (only available on Active Alarms and Hardware Alarms page)
- Disable alarms on the current page (only available on Active Alarms page)
- Enable alarms on the current page (only available on Disabled Alarms page)
- Silence alarm sound (not available on Hardware Alarms page)

This panel is not featured on the Alarm Summary page.

View Tasks

Offers the option to:

- Add new column to the alarm list.
- Automatically fit the width of columns to show the most content.
- Reload the column settings according to the alarm format set in parameter [Format]<FormatName>
- Save the current column settings to parameter [Format]<FormatName>, so that the same settings are used when this page is displayed in the future.

This panel is not featured on the Hardware Alarm page.

Interactive Alarm List

The alarm page templates (except the Hardware Alarms template) feature an interactive alarm list that supports the following functionality:

Adjustable Columns

The user can adjust the columns displayed on the alarm list. Options include:

- Reposition a column by drag-and-drop its column heading to the new position.
- Delete a column by drag-and-drop its column heading (vertically) beyond the column heading row.
- Insert or delete a column via a pop-up menu by right-clicking a column heading
- Adjust the size of a column by dragging the column separator that appears at the right edge of a column heading.

Selection of Alarms

The user can select single or range of alarms on the alarm list so that the same operation can be applied to multiple selected alarms in one go. Alarm selection works as follows:

- To select a single alarm: Left-click an alarm that is not already selected to select it.
- To select a range of alarms: Select a single alarm to mark the start of the selection. Then hold the SHIFT key and left-click another alarm to extend the selection. You can adjust the range of the selection by SHIFT left-clicking another row of alarm.
- To deselect alarm(s): Left-click any alarm within a selection range to deselect the whole range.

Context Menu

Right-click on any alarm row to bring up the context menu for alarm operations. If you right-click on an alarm that is not already selected, this will select it. Subsequent commands (selected from the context menu) will then only operate on this single alarm. On the other hand, if you right-click on any alarm within a selection range, this will not change the selection. Subsequent commands selected will operate on each alarm within the selection range. Several operation options are available in the context menu:

- Information: displays the detailed information of a single alarm that has the cursor focus (indicated by surrounded by a white box).
- Help: displays the Help page (which is set in the Help field of the alarm record in Project Editor) of a single alarm that has the cursor focus.
- Acknowledge: acknowledges selected alarms. This option is restricted by the privilege set by the parameter [Privilege]AckAlarms. This is not available on the Alarm Summary page.
- Disable: disables selected alarms. This option is restricted by the privilege set by the parameter [Privilege]DisableAlarms. It is not available on the Disabled Alarms page.
- Enable: enables selected alarms. This option is restricted by the privilege set by the parameter [Privilege]DisableAlarms. It is not available on the Active Alarms page.

Note: The Hardware Alarms template only allows a single alarm to be selected. Its context menu has only the Acknowledge option which is also restricted by the privilege set by parameter [Privilege]AckAlarms.

Process Analyst Page Templates

The Process Analyst template is the default template for you to create trend pages. The Tab Style project includes templates for the three following types of trend display:

- **SinglePA** - A template that displays a single Process Analyst chart on the page. To create a default trend page for your project, please create a new page based on this template and name it "ProcessAnalyst". You can change the default page name by setting parameter [Page]ProcessAnalystPage
- **DoublePA** - A template that displays two Process Analyst charts on the page.
- **PopPA** - A template that displays a single Process Analyst chart on the page. The size of the page is reduced to make it suitable for pop-up pages. To create a default trend pop-up page for your project, please create a new page based on this template and name it "!ProcessAnalystPopup". You can change the default page name by setting parameter [Page]ProcessAnalystPopupPage

The functionality of traditional trends, double trends, and popup trends are largely incorporated into the Process Analyst on the new templates.

There is a key difference between the new PA templates and the existing trend templates, which is pen assignment. The trend templates can get pens assigned to the page at both design time and runtime, but every PA template needs the pens assigned at runtime only. This can be done by calling the Process Analyst Cicode functions and pass the PAV file or individual trend pens to those functions.

There are two new items on the PA templates:

- The Show/Hide Trend Statistics button shows and hides trend statistics columns in the Process Analyst.
- The Std Deviation column works out the sample standard deviation value of the samples within the display area.

See Also

Process Analyst for Operators

Statistical Process Control Trend Page Templates

The Tab Style project includes the following trend templates for different types of statistical process control:

- **SPCPareto**: Displays a Pareto chart for multiple variable tags
- **SPCCpk**: Displays an SPC mean chart and a Capability chart
- **SPCXRSChart**: Displays SPC mean, range and standard deviation charts
- **EventSPCXRS**: Displays SPC mean, range and standard deviation charts for Event SPC trend tag
- **MeanMeanChart**: Displays SPC mean charts for two SPC trend tags
- **RangeChart**: Displays SPC mean and range charts for an SPC trend tag
- **StandardChart**: Displays SPC mean and standard deviation charts for an SPC trend tag

File Page Templates

The tab style project includes the following templates for displaying different types of file:

- File: Displays a plain text file on the page. This template replicates the the functionality of the original file template found in the standard Include project. The new template is compatible with any existing PageFile() or DspFile() Cicode functions. It can display up to 24 lines of text from a file at a time. Buttons are provided for the user to browse to different rows and columns of the file.
- File_RTF: Displays a rich text format (RTF) file on the page using the CiTextBox ActiveX control. It can also display plain text file.
- File_HTML: Displays a hypertext markup language (HTML) file or an universal resource locator (URL) link on the page. It needs Microsoft Internet Explorer to be installed on the machine.

Several common features are presented on the templates:

The new file templates replicate the functionality of the original file template in the Include project. The new templates are compatible with any existing PageFile() or DspFile() Cicode functions. They can display up to 24 lines of text from a file at a time. Buttons are provided for the user to browse to different rows and columns of the file.

Several new features are added to the templates:

- When you create a page based on any of these templates in Graphics Builder, you can predefine the file for display at runtime. Clicking anywhere on the page will show a genie style dialog to allow you to enter a file path to assign a file to the page. A Citect path substitution string can be used in the file path. When the page is recalled at runtime, for example via PageDisplay(), the specified file will be automatically displayed.
- The "Open File" button displays an Open File dialog to select a different file to display.
- The "Refresh" button reopens the already opened file to get the latest contents.
- The full path name of the currently opened file is displayed.

Chapter: 41 Creating a New Project

Creating a project based on the Tab Style templates project is simple; by default, it is incorporated as an included project in new projects. This means whenever you start a new project, the tab style templates are ready to use as necessary.

There are two ways to create a project based on the tab style templates:

- Use the pre-defined starter project
- Create a project from scratch

Using the pre-defined starter project

To base a project on an existing starter project:

1. Choose **New Project** from the **File** menu, or click the **New Project** button.
2. Type a name for your project and choose a location for the files.
3. Click the **Create project based on starter project** checkbox.
4. Choose the project on which you want to base your new project.
5. Click **OK**.

Four pre-defined starter projects with page resolutions of XGA, SXGA, WUXGA and HD1080 (Full HD) are provided as part of the product installation. Each of the starter projects will be based on project "Tab_Style_1" and contain:

- A cluster named "Cluster1".
- A role named "Administrators" which is linked to the "BUILTIN\Administrators" Windows group and have global privilege of 8. This allow a user to log in CitectSCADA at runtime using a Windows user account who belongs to the Administrators group of the local machine.
- Pages of Alarm, Summary, Disabled, Hardware, ProcessAnalyst and !ProcessAnalystPopup based on the relevant templates found in the Tab_Style_Include project.

The newly created project will be immediately compilable, and will contain a basic level of built-in functions such as viewing alarms and trends.

Create a project from scratch

To make it easier to configure a project based on the tab style templates from scratch, follow these steps:

1. Create a privileged user; see Creating a Privileged User.
2. Run the Computer Setup Wizard; see Running the Computer Setup Wizard.
3. Set up instant trending; see Using Instant Trending.
4. Set up multiple monitor display; see Displaying a Project on Multiple Monitors.
5. Implement audible alarms; see Implementing Audible Alarms.

Don't modify the tab style templates project for use as a runtime project; set it aside for use as a template for new projects.

If creating a project based on the tab style templates, it is not recommended to include pages based on templates that use a different style, including the earlier Include project. Doing so might affect functionality.

See Also

Creating Pages

Creating a Privileged User

Some tab style templates project content is restricted via a user login. Without a valid login, some project functionality will be disabled. For example, the Close window (the cross) button on title bar will not work if you log in as a user with restricted privileges.

By default, the following elements within the tab style templates project are restricted by global privileges.

Element	Global Privilege	Associated Parameter
Editing users	8	[Privilege]EditUser
Project shutdown	8	[Privilege]Shutdown
Acknowledge alarms	1	[Privilege]AckAlarms
Disable alarms	8	[Privilege]DisableAlarms
Silence alarms	0	[Privilege]SilenceAlarms

When configuring a tab style templates project,check that your users have appropriate access to the available functionality. Verify that your users can acknowledge alarms if necessary.

To adjust the global privileges for the elements previously listed for a more complex security architecture, adjust the `[Privilege]` parameters in the `Citect.ini` file. Click the relevant link in the previous table above for details.

See Also

Running the Computer Setup Wizard

Privilege Parameters

Running the Computer Setup Wizard

As with any CitectSCADA project, you need to fill inthe Computer Setup Wizard on any machine where a tab style templates project will be run. See Running the Computer Setup Wizard for more information.

How you set up a computer using the Wizard is mostly up to you; however, due to some necessary settings, you need to run the Computer Setup Wizard as a custom setup. The Wizard page you need to pay attention to are:

- Security Setup - Control Menu Page

Security Setup - Control Menu page

All tab templates are designed to be displayed with the title bar. You need to select the "Show Title Bar" option when you select to display the page in fullscreen mode.

Window title bar controls can be secured from unauthorized access. See Securing the Windows Title bar.

See Also

Creating a Privileged User

Using Instant Trending

Implementing Audible Alarms

Using Instant Trending

Instant trending is natively supported by the Process Analyst. The steps to implement instant trending are as follows:

1. **Create a Process Analyst page.** Create a page that has a Process Analyst object on it. You may create these pages based on the provided tab style templates. For a full-screen page, create the page based on the SinglePA template. For a pop-up page, create the page based on the PopPA template.
2. **Set the sample cache duration.** Samples collected for Instant Trends are kept in the cache up to a default number of samples and time period. You can adjust these settings using the parameters [Trend]InstantTrendSamples and [Trend]InstantTrendIdleTime.

3. **Call the Process Analyst Display.** At runtime, call your process analyst page either as a fullscreen page or pop-up page using the Cicode function PageDisplay(pagename) or PagePopup(pagename) respectively.
4. **Add Variable Tags / Local Variables for Trending.** Use the Add Pens dialog provided with the Process Analyst to add pens to the process analyst chart as normal. In the dialog, you can now select any variable tags or local variables for trending.

See Also

Process Analyst Page Templates

Displaying a Project on Multiple Monitors

For projects based on the tab style templates, multiple monitors will be supported through the core product. You can display a CitectSCADA window on each of the monitors on the machine by adjusting the following parameters:

- [MultiMonitors]Monitors - Adjust this parameter to indicate how many monitors your project will be running on.
- [MultiMonitors]StartupPage<n> - This parameter determines which page will appear on each monitor at startup. You have to duplicate this parameter for each monitor. For example, StartupPage1 determines the page that appears on the first monitor at startup, StartupPage2 determines what appears on the second monitor, and so on. If you do not specify a startup page for a particular monitor, the page specified in parameter [Page]Startup will be displayed.

See Also

Working with Multiple Monitors

Implementing Audible Alarms

The tab style templates project offers support for audible alarms. You can configure a project so that a selected wav file is sounded whenever an alarm of a particular priority is triggered. You can even assign different sounds to different alarm priorities, allowing the urgency of an alarm to be determined from its sound.

To implement audible alarms in your project:

1. Verify that the alarms you want to trigger an audible alert are assigned to a category and given a particular priority.
2. Adjust the parameter `[Alarm]Soundn`. This parameter determines the wav file used when an alarm sounds, based on the priority of the alarm (n). For example, `[Alarm]Sound1` identifies the .wav file that will be sounded whenever a priority 1 alarm is triggered.

3. If necessary, adjust the parameter `[Alarm]SoundnInterval`. This parameter determines how long the selected wav file sounds, based on the priority of the alarm (n). For example, `[Alarm]Sound1Interval` sets the length of time a priority 1 alarm is sounded for.
4. Verify that your users have appropriate privileges associated with their login to silence alarms, otherwise they will not be able to silence an alarm. See Creating a Privileged User and the parameter [Privilege]SilenceAlarms.

See Also

Alarm Page Templates

Creating Pages

If you created your project by selecting the **Create project based on starter project** checkbox in the **New Project** dialog, pages of Alarm, Summary, Disabled, Hardware, ProcessAnalyst and !ProcessAnalystPopup will already be created in your new project. See Creating a new project.

If you created your project from scratch, there are no predefined pages in your project. The navigation controls on the tab template expect several default pages that being exist in your project for basic functionality such as alarm and trend displays. You are recommended to add the following pages to your project based on the provided templates of your chosen size:

Template	Description
SinglePA or DoublePA	Process Analyst page for displaying trends.
!PopPA	Process Analyst pop-up page.
Alarm	Active Alarms page, called from the Alarms toolbar.
Hardware	Hardware Alarms page, called from the Alarms toolbar.
Disabled	Disabled Alarms page, called from the Alarms toolbar.
Summary	Alarms Summary page, called from the Alarms toolbar.

The links between the pages listed and the controls that call them are defined by the [Page] parameter settings within the Citect.ini file. The relevant parameters are [Page]ProcessAnalystPage, [Page]ProcessAnalystPopupPage, [Page]AlarmPage and [Page]HardwarePage, etc. If you want a button to call a page with a different name, adjust these parameter settings. For details see Page Parameters .

Creating new pages

The normal template in the tab style template library are designed with minimal content to enable the presentation of customer-necessary information and plant mimics.

Pages are created based on this template within Graphics Builder by choosing the necessary template from the Use Template dialog (**File | New Page**). You can access the tab style templates by selecting **tab_style_1** from the **Style** field.

Note: If you try to launch a tab style template from within Citect Explorer, you need to drill down to `[Project Name]/Graphics/Templates/tab_style_1/`*`[Resolution]`* to locate it. Four different resolutions are provided including XGA, SXGA, HD1080 and WUXGA.

Creating Custom Menus

The content of the tabbed menus on pages based on the tab style templates can be configured via the Menu Configuration dialog, which is launched from Citect Project Editor at configuration time. This is different to how menus are defined in the CSV_Include project. If you are migrating from CSV_Include project to the tab style templates, you can use the Migration Tool to migrate the CSV_Include menu definition to the Menu Configuration database.

The tabbed menus read the menu configuration data in a custom fashion which are explained in Defining page menus.

You can choose to use the default menus offered by the tab style templates which can be turn on or off based on parameter [Page]AddDefaultMenu. Items in the default menus are assigned with a sorting order of 100. When you configure to show the default menus and your own defined menus at the same time, the two will be merged and ordered accordingly. The default menus contain the following items:

Tabs	Buttons
Pages	Automatically populate the non-system pages configured in the project and included project(s)
Alarms	Active Alarms

Tabs	Buttons
Alarm Summary	
Disabled Alarms	
Hardware Alarms	
Trends	Process Analyst Calls Cicode function: PageProcessAnalyst("", "")
	Process Analyst Popup Calls Cicode function: ProcessAnalystPopup("","")

See Also

Defining page menus

Defining template popup menus

Defining page menus

Page menus in the tab style templates are defined using the Menu Configuration dialog. For tab templates the fields have the following properties:

Order

The relative display position of the tabs / buttons or menu items among themselves. For the tabs and buttons, it determines the display order from left to right. For the drop-down menu items, it determines the display order from the top to the bottom. If this field is left blank, it will take the default value of 0. The relevant entries will be displayed at the start as a result. If more than 1 entry of the same order exists, they are displayed in the same order as they are defined in the database.

Level 1

This represents the tabs on the tab bar.

Level 2

This represents the buttons displayed under a particular tab.

Level 3

This represents the menu items of the dropdown menu next to a particular button.

Level 4

This represents the sub-menu items of a particular dropdown menu item.

Menu Command

The Cicode expression to be executed when the menu item is selected. String of 256 characters.

Symbol

The symbol to associate with the menu entry in the format of <library name>.<symbol name>. The template expects the symbol to be a certain size when it is configured against different menu types: For tabs, symbols are 16 x 16 pixels. For buttons, symbols are 32 x 32 pixels. Some common symbols can be found in the symbol libraries of icons_16x16 and icons_32x32. The field is not applicable for menu and sub-menu items.

Comment

A comment about the menu entry. String of 128 characters.

Page

The page on which this entry will exist. If blank, the entry exists every page. The page name of "Template" is a keyword used in the templates for defining custom pop-up menus.

Hidden when

The tab menu bar on the template does not support visibility. When the expression defined in this field is true, the relevant menu entry is disabled instead.

Disabled when

Cicode expression to determine when the menu entry is disabled on the page. String of 256 characters.

Disabled style

This field is not used by the tab menu bar on the template.

Width

The width (measured in pixels) of the item appear on the page. If this is not specified, the item is automatically sized to show the most content. This field only applies to the tabs and buttons.

The objects used in the template are subjected to minimum widths: Tab (Level 1)= 63 pixels, and Button(Level 2) = 61 pixels.

Checked

Whether the menu item shows a check mark next to it. This field only applies to the menu items and sub-menu items.

Privilege

The user privilege of (0-8) necessary to select the menu entry.

Area

The user area (0-255) necessary to select the menu entry.

See Also

Defining template popup menus
Configuring Page Menus

Defining template popup menus

The Tab style templates menu bar features two pop up menus:

- Login dropdown - brings up a dropdown menu for different login options. You can override the default menu by specifying your own menu in the "Menu Configuration" form. The override menu needs to be assigned with the page name of "Template" and have field "Level 1" set to "Login" in the form.
- Alarm silence option button - clicking this button displays a context menu for selecting different silence options. This button is only enabled if you have sufficient privileges as specified in the parameter [Privilege] SilenceAlarms. You can override the default menu by specifying your own menu in the "Menu Configuration" form. The override menu needs to be assigned with the page name of "Template" and have field "Level 1" set to "Alarm Sound" in the form.

See Also

Defining page menus
Configuring Page Menus

Using Environment Variables in Tab Style Templates

Whereas parameters apply specified rules to every template page, environment variables apply rules to *specific* pages. In this way, you can use environment variables to specify functionality for specific pages.

For example, to display a specific page, you'd specify the page name as the environment variable. To call a specific Cicode function, you'd specify the function name (with a "?" prefix), space, then a list of comma separated arguments, like this: `?WinPrint LPT1,0,0,Trend.pal`. (The "?" indicates that the variable is to be interpreted as a function call rather than a page to display.)

Let's look at a more detailed example: usually the standard **Print** button is used for printing a graphics page in WYSIWYG format. If you have a text/html file displayed on the page, or perhaps a trend, you can create your own print function (for example, `PrintTrend`) and specify it in the environment variable, as `?PrintTrend`. To reduce the amount of ink used, specify the `WinPrint` function with a user-defined palette. In this case you would specify it as a parameter rather than as an environment variable so that it applies to every page.

Note: For pages created based on the Process Analyst templates, you can print the screen using the standard Print button, or print the chart using the Print button on the toolbar of the Process Analyst object. Give the user both alternatives as they both serve different purposes.

Using the CSV_Include Project

This section contains information on the CSV_Include project and describes the following:

Introducing CSV_Include
Using Pages and Templates
Creating a New Project

"CSV_Include Reference" in the CitectSCADA Technical Reference

Chapter: 42 Introducing CSV_Include

The CSV_Include Project is a preconfigured project that includes a set of templates and pages styled for the Windows XP environment. It will help you reduce the time necessary to configure a new project.

When a new CitectSCADA project is created, the CSV_Include project is automatically incorporated as an included project. This means the project's templates and associated content are available for implementation when creating your graphics pages in Graphics Builder.

As well as including a standard graphics page template for creating plant mimics, the project also includes predefined trend and alarm display pages, an administration tools page, a file page for displaying text and Rich Text Format files, and a selection of popup windows. They have identical navigation and alarm menus for consistent "look and feel" across an entire project. The project even supports multimonitor displays, allowing you to simultaneously display several graphics pages across several computer screens.

> **Note:** Do not modify the CSV_Include project for use as a runtime project. It will not compile successfully, and be set aside for use as a template for new projects. CitectSCADA upgrades install a new version of the CSV_Include project, so you will lose changes you make to the project when this happens.

Where to Find Information

Use the following links for details about the preconfigured content in the CSV_Include Project:

- Predefined pages and templates. See Using Pages and Templates.
- Common toolbars. See Common Toolbars.

It also describes the processes used to create a project based on CSV_Include:

- Creating a new project. See Creating a New Project.
- Creating pages. See Creating Pages.
- Creating custom menus. See Creating Custom Menus.
- Creating an alarm group. See Creating an Alarms Group.
- Creating a trend group. Creating a Trends Group.

Reference material is also included, describing the citect.ini parameters and Cicode functions available to customize a project.

Chapter: 43 Using Pages and Templates

When you create a new project based on CSV_Include, the following templates and pages are available for you to use as you create your graphics pages. This section describes the pages and their buttons, menus, and tools.

The CSV_Include project has the following templates:

Template Name	Description
Normal	Standard graphics page
Alarm	Active alarm page
Disabled	Disabled alarm page
Summary	Summary alarm page
Hardware	Hardware alarm page
Trend	Trend page (8 pens)
Double Trend	Split-page trend page (16 pens)
File	Displays a file
Admin Tools	Engineering tools page
Poptrend	Popup trend
Instanttrend	Instant trend
Popup_Small	Small popup window
Popup_Mid	Medium popup window
Popup_Large	Large popup window
Popup_Xlarge	Extra large popup window

The project also includes the following preconfigured pages:

Page Name	Description
CSV_AdminTools	Engineering tools page
CSV_Alarm	Active alarm page
CSV_AlarmDisabled	Disabled alarm page
CSV_AlarmSummary	Summary alarm page
CSV_AlarmHardware	Hardware alarm page
CSV_Trend	Trend page (8 pens)
CSV_TrendDouble	Split-page trend page (16 pens)
CSV_File	File page

See Also

Creating a New Project
Creating Pages

Normal Page Template

A template designed for creating user-necessary content and plant mimics. This page contains the standard navigation and alarm toolbars featured on CSV_Include templates. If you are creating a project based on CSV_Include, use this template for your standard graphics pages.

The Normal template creates a page in 1024 x 768 display format without a title bar, the default size for pages in the CSV_Include project.

See Also

Creating Pages

Alarm Page Templates

Templates are included for the following types of alarm displays:

- **Active Alarm Page (Alarm)**: Used to create a page displaying audible alarms that are unacknowledged or acknowledged and still in alarm state. The preconfigured page CSV_Alarm is based on this template.

- **Hardware Alarm Page (Hardware)**: Used to create a page displaying details of any system errors that are unacknowledged or acknowledged and still in alarm state, for example, if communications are lost, if Cicode can't execute, if a graphics page is not updating correctly, or if a server becomes inoperative, this page will alert you. The preconfigured page CSV_AlarmHardware is based on this template.
- **Disabled Alarm Page (Disabled)**: Used to create a page displaying alarms that are presently disabled in the system. The preconfigured page CSV_AlarmDisabled is based on this template. The alarms will appear on this page when they have been disabled from the normal alarm system due to a maintenance shutdown, nuisance tripping etc.
- **Alarm Summary Page (Summary)**: Used to create a page displaying a historical log of alarms that have occurred. The preconfigured page CSV_AlarmSummary is based on this template. This page can be used for troubleshooting purposes.

Common functionality

The alarm page templates share panels to the left of the alarms list that support the following functionality:

Acknowledge Tasks

Offers the option to:

- acknowledge alarms on the current page
- acknowledge just the selected alarm
- silence an audible alarm.

It is featured on the Active Alarms page and the Hardware Alarms page.

Alarm Page Tasks

Allows the user to navigate through the list of alarms a page at a time. The blue box indicates if more than one page is available to view.

Alarm List Filter Tasks

Allows the user to filter the current list based on plant area or by alarm category. To apply a particular filter, you first configure an alarm group.

Trend Page Templates

The CSV_Include project includes templates for the four following types of trend display:

- **Trend**: An eight-pen trend display. The preconfigured page CSV_Trend is based on this template.

- **DoubleTrend**:A 2 x 8-pen trend display spread across a divided screen. The preconfigured page CSV_TrendDouble is based on this template.
- **PopTrend**: A four-pen pop-up display that can be launched from other graphics pages.
- **InstantTrend**: A popup window for instant display of a variable tag.

> **Note:** To implement the Instant Trend feature in one of your own projects, you need to add the CSV_InstantTrend project as an Included project. (See Including a project in the current project.)

The Trend, Double Trend and PopTrend displays allow you to select and display trend tags from your CitectSCADA project on a color-coded linear chart. Being based on information coming from the CitectSCADA Trends Server, these displays are supported by stored historical data that can be recalled if necessary.

The Instant Trend display is unique as it allows the selection of up to four variable tags for display. This allows you to visually monitor a tag without having to set it up within your CitectSCADA project as a trend tag. You can even load tags directly into the display by hovering the mouse over a tag value on a graphics page and keying in a plus sign (see the parameter `[TrendX]KeySeq`).

There are limitations to this feature, however, as data is only available from the time the display is launched and is lost when the display is closed. The Instant Trend feature has a duration setting that limits the amount of time the display can remain open, you can adjust the default setting for this via the parameter [TrendX]Duration.

For details on implementing instant trending, see Setting Up Instant Trending.

Common functionality

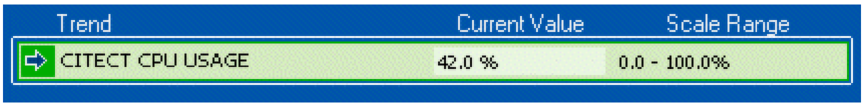
The trend display templates share common controls that support the following functionality:

- Selected Trend Field
- Range/Scale markers
- Span markers
- Set span button
- Trend cursor
- History mode
- Zoom
- Autoscale
- Scale defaults

- Export to file
- Paste to clipboard
- Plot trend
- Trend group

Selected Trend Field

Displays information relating to the trend tag(s) currently being mapped in the trend chart, including a name, the current value and the scale range.

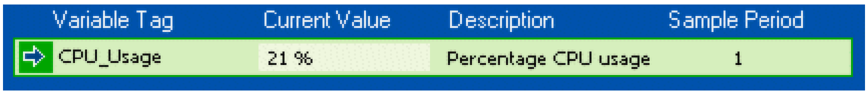


	Trend	Current Value	Scale Range
⇨	CITECT CPU USAGE	42.0 %	0.0 - 100.0%

The field is color coded to match the graphical display, hence the colorful nature of the listed fields.

To load a trend into a Selected Trend field, right-click the field. A menu appears allowing you to select a trend or clear the field. Once you have loaded your selected trends, you can make a particular pen the focus of the graph by clicking this field. An arrow to the left of the field indicates it is currently the focus trend.

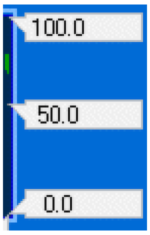
This field appears slightly different when using instant trending, as the graph will be displaying current data for a variable tag, not a trend tag. In this case, the selection field appears as below, with the tag name, current value, description and sample period displayed.



	Variable Tag	Current Value	Description	Sample Period
⇨	CPU_Usage	21 %	Percentage CPU usage	1

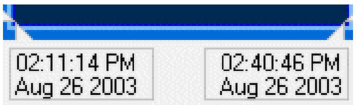
Range/Scale markers

These markers appear along the vertical axis and show the scale of the display for the currently selected pen, highlighting the lowest to highest values for the scale range.



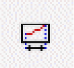
Span markers

These markers appear along the horizontal axis, and show the span of the trend display in time. The left-hand marker shows the current start time and date, the right shows the current end time and date. Typically the right-hand marker shows the current time; however, this can be affected by zooming or displaying in Historical mode.



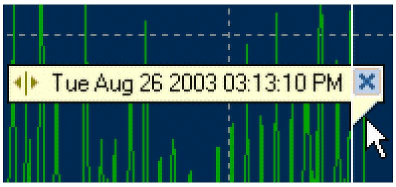
Set span button

Displays an input dialog that sets the span of the current display. When you enter a unit of time, the display will cover the defined period ending with the current time.



Trend cursor

Allows you to select a location on the graphical trend display and determine the time and date for that particular point. The arrows to the left end of the display allow you to move the cursor left or right. The Exit icon to the right closes the trend cursor.

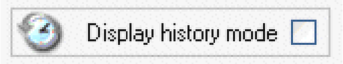


History mode

Allows you to scroll forward or backward through a graph of a trend tag's history. To switch into History Mode, select the **Display History Mode** checkbox. The display changes to lower version of the control shown to the left.

The four buttons allow you to move backwards or forwards through the trend graph, with the two outer buttons shifting a page at a time, and the two inner buttons shifting half a page. The clock icon allows you to edit the end time for the historical display.

Deselecting the box returns to the normal trend view.



Zoom

Zooms in on or out of the current display. Zoom in (+) increases the focus of the display and continues zooming in on subsequent clicks. Zoom out (-) returns to the default.

Autoscale

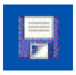
Autoscales the current view, which means the scale adjusts to the lowest and highest values reached.

Scale defaults

Returns the scale for the display to the default for the currently selected trend.

Export to file

Exports the data for the currently displayed trend to a file. A Save As dialog will appear, allowing you to save the data as a D Base III file (.DBF file), a comma separated value file (.CSV file) or a text file (.TXT).

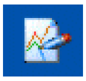


Paste to clipboard

Sends the data for the currently displayed trend to the Windows clipboard.

Plot trend

Plots the trend to a printer. Use this button to print a trend graph instead of the Print Page button. A dialog allows you to configure the printer setup.



Trend group

Allows you to load a trend group. A dialog lists the currently configured groups and allows you to clear currently displayed variable tags. See Creating a Trends Group.

📂

File Page Templates

Use this template to create a page that can display text (.txt) or rich format text (.rtf) files.

To understand how a file page works, you need to distinguish between the file page and the file that is presented on the page; for the file page merely acts as a blank palette to a variety of files.

The file page is displayed whenever the function `CSV_Nav_File` is called. When executed, `CSV_Nav_File` determines the file that is to be displayed on the file page, its location, the title applied to the page, and whether or not the file is editable.

For example, you may configure a menu item that calls up the following:

```
?CSV_Nav_File(MyPageTitle,[Run]:\file.txt,2)
```

This would call up the file page, put the title "MyPageTitle" on the title bar, load the file called File.txt from the Run directory, and allow the file to be edited (with the last argument set to 2, the file can be saved).

The parameter `[Navigation]FilePage` determines the page that's used as the palette for this process. The preconfigured page CSV_File is the default. You can change the setting for `Navigation[FilePage]`, however, you need to verify that any page you specify for this parameter is based on the CSV_File template, otherwise CSV_Nav_File will not be able to execute properly.

Admin Tools Page Template

This template is used to create a page that features network and local machine statistics, and allows the user to launch Citect configuration tools and Windows applications. The preconfigured page CSV_AdminTools is based on this template.

The panels to the left of this page launch the following configuration tools:

- **Windows applications**: Allows the user to launch the listed Windows Applications from within the runtime environment.

- **Citect Configuration**: Allows the user to launch the listed Citect configuration tools from within the runtime environment. This includes viewing and editing the Citect.INI file, running the Computer Setup Wizard, creating alarm groups, trend groups, and launching the Menu Configuration tool.
- **Citect Kernel**: Launches the various components of the Citect Kernel, allowing the system to be debugged from within the runtime environment.
- **System / Hardware**: I/O Device Stats launches a dialog displaying I/O Device statistics. The Next and Previous buttons allow you to step through the I/O Devices connected to the system, while the Search button allows you to view statistics for a particular device.

The Tag Debug tool allows you to browse a list of available tags and read the current value for a selected tag. Depending on your access privileges, you may also be able to write a new value to the tag.

The right-hand side of this page displays statistics about the CitectSCADA system:

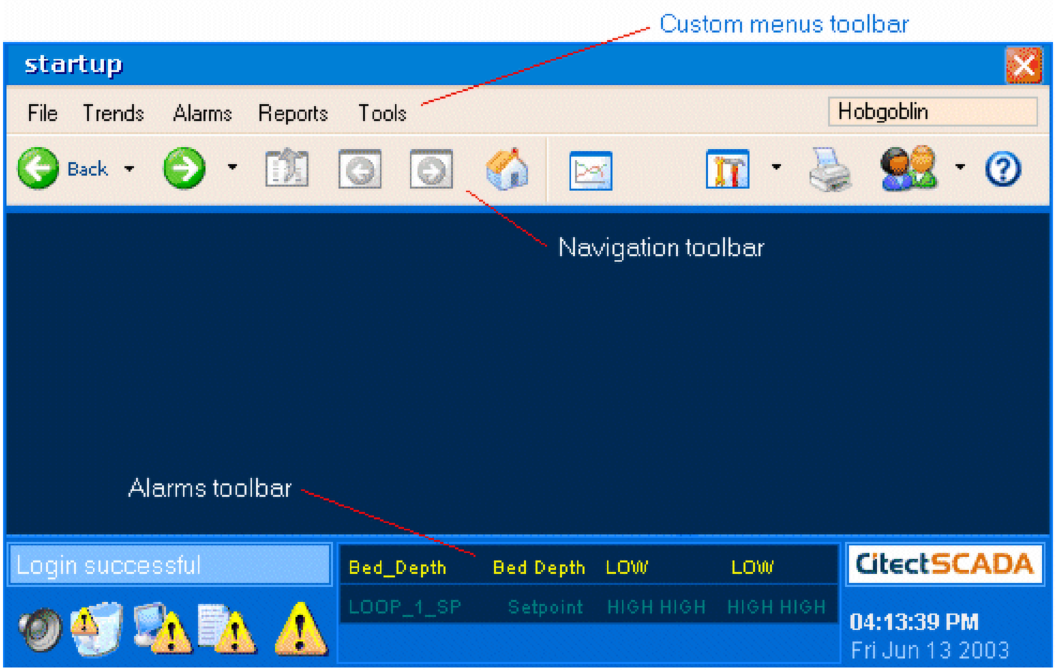
- **System Information**: Provides details of the CPU usage, memory usage and available disk space for the current runtime machine.
- **Citect Information**: Provides information about the CitectSCADA system including version information and the number of trend, alarm and report clients currently connected
- **I/O Server**: Provides information about the status of the CitectSCADA I/O Server your system is connected to. The name of the I/O Server machine appears in the title bar of this panel. The information displayed includes:
  - **Average**: Overall average response time (ms).
  - **Maximum** - the overall maximum response time (ms)
  - **Minimum**: Overall minimum response time (ms).
  - **Read Request**: Total read requests per second.
  - **Read Physical**: Total physical read requests per second.
  - **Write Request**: Total write requests per second.
  - **Write Physical**: Total physical write requests per second.
  - **Reset** button: clears the current values and recalculates them.

Common Toolbars

Pages in the CSV_Include project include common toolbars for easy navigation and feature access, as well as a consistent appearance.

The following three toolbars remain on screen during operation:

- **Navigation toolbar**: Provides navigation buttons and direct access to key pages such as the Trends page and Admin Tools page.
- **Alarm toolbar**: Provides access to Alarms pages and displays the last three active alarms.
- **Custom Menus toolbar**: Provides menus capable of navigating to a specific page or calling a Cicode function. The content of the menus is generated at runtime using a lookup table.


Custom menus toolbar

Navigation toolbar

Alarms toolbar

Login successful

Bed_Depth Bed Depth LOW LOW

LOOP_1_SP Setpoint HIGH HIGH HIGH HIGH

04:13:39 PM

Fri Jun 13 2003

See Also

[Navigation Toolbar](#)

[Alarms Toolbar](#)

[Creating Custom Menus](#)

Navigation Toolbar

The Navigation toolbar includes buttons that allow the user to move between a project's pages. If the current user has insufficient privilege or there is no option configured for a particular button, it is unavailable to the user.

Icon	Name	Description
Back ▾	**Back**	Goes back to the page displayed before the current page. The arrow to the right of the button allows you

		to select from a menu of recently visited pages. You can set the maximum number of pages included in this menu by adjusting the parameter `[Navigation]LastPageStackSize`.
→ ▾	**Forward**	Goes forward to the page that was displayed prior to the back button being pressed. The arrow to the right of the button allows you to select from menu of pages recently navigated back from. You can set the maximum number of pages included in this menu by adjusting the parameter `[Navigation]LastPageStackSize`.
↑	**Parent Page**	Changes the display to the "parent page" of the current page. You can assign a parent page to a graphics page by setting an environment variable in Graphics Builder. To do this, open the page you want to assign a parent to. Go to the properties dialog for the page (**File | Properties**), and click the **Environment** tab. Add a new variable called "ParentPage" with a value of the page name of the parent page.
← →	**Previous/Next**	Moves backwards or forwards through pages in a browse sequence if a sequence is configured. To configure a browse sequence, go to the Page Properties dialog (**File | Properties**) for each page in the sequence and set the **Next** and **Previous** fields on the **General** tab accordingly.
	Home	Displays the "home" page. By default, this page is the startup page CSV_Start. To change the home page to a different page, use the `[Navigation]HomePage` parameter.
	Trends Page	Displays the "trend" page. By default, the preconfigured page CSV_Trend appears when this button is pressed. To display a different page, adjust the parameter `[Navigation]TrendPage`.
	Network Page	Displays a page called "Network" if one exists. By default, this page does not exist, which means the button will not appear. To use this button to call a function, or to launch a page with a name other than network, adjust the parameter `[Navigation]NetworkPage`.
▾	**Tools**	Displays the Admin Tools page, named CSV_AdminTools. You can adjust the parameter `[Navigation]ToolsPage` to call a function with this button or change the page displayed; however, this is not recommended. This button also features a menu to launch the Tag Debug tool and Instant Trend display.

		It also allows you to switch tool tips on and off.
🖨	**Print Page**	Prints the current page. The parameter `[Printer]Port` needs to be configured correctly for printer selection.
👥	**Login**	Displays the standard CitectSCADA login prompt. The menu to the right offers extra options such as Logout, Change Password, Edit User (restricted by login) and Create User (restricted by login).
?	**Help**	This button can be configured to display user assistance information for your project. By default, it will point to the topic Adding user assistance to a page within the CitectSCADA help. This topic explains how to change this default behavior within your own project, by either disabling the button, or configuring it to connect to your own HTML help file.

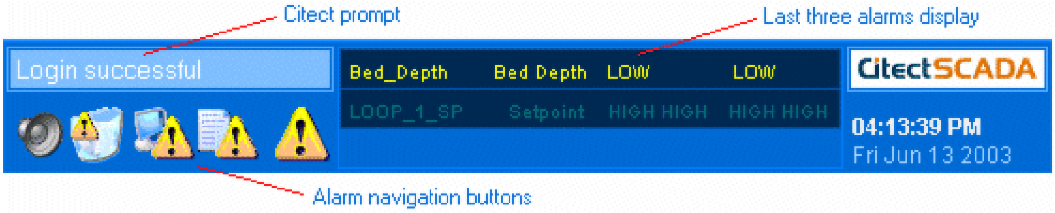
See Also

Alarms Toolbar

Creating Custom Menus

Alarms Toolbar

The alarms toolbar provides access to current alarm information and navigation buttons for single-click access to alarm pages. It also includes standard CitectSCADA prompt and current date and time.



```
Login successful    Bed_Depth    Bed Depth   LOW         LOW
                    LOOP_1_SP    Setpoint    HIGH HIGH   HIGH HIGH
04:13:39 PM
Fri Jun 13 2003
```

The alarms toolbar contains the following features:

Icon	Name	Description
n/a	**Last Alarms**	The center panel of the toolbar displays the last three alarms triggered within the system, providing the operator with an immediate visual cue to alarm occurrences. You can modify the format of this list and control which alarms appear. For example, you can set the list to only display alarms of a particular type, category or priority.

		For details, see the parameters LastAlarmFmt, LastAlarmCategories, LastAlarmPriorities, and LastAlarmType.
⚠	**Active Alarms**	Changes the display to the active alarms page, CSV_Alarm. This button is animated and will blink when there is an unacknowledged alarm in the system. This action performed by this button is determined by the parameter [Navigation]AlarmPage.
⚠	**Alarms Summary**	Changes the display to the alarm summary page, CSV_AlarmSummary, which provides a historical log of alarm occurrences.This action performed by this button is determined by the parameter [Navigation]SummaryPage.
⚠	**Hardware Alarms**	Changes the display to the hardware alarms page, CSV_AlarmHardware. This is an animated button which will blink when an unacknowledged hardware alarm occurs. This action performed by this button is determined by the parameter [Navigation]HardwarePage.
⚠	**Disabled Alarms**	Changes the display to the disabled alarms page, CSV_AlarmDisabled. This action performed by this button is determined by the parameter [Navigation]DisabledPage.
🔊	**Alarm Silence**	Silences the audible alarm buzzer that sounds when an alarm occurs. See Implementing Audible Alarms.

See Also

Creating Custom Menus

Chapter: 44 Creating a New Project

Creating a project based on the CSV_Include project is simple; by default, it is incorporated as an included project in new projects. This means whenever you start a new project, the CSV_Include pages and templates are ready to use as necessary.

To make it easier to configure a project based on the CSV_Include project, follow these steps:

1. Create a privileged user; see Creating a Privileged User.
2. Run the Computer Setup Wizard; see Running the Computer Setup Wizard.
3. Set up instant trending; see Setting Up Instant Trending.
4. Set up multiple monitor display; see Displaying a Project on Multiple Monitors.
5. Implement audible alarms; see Implementing Audible Alarms.

Don't modify the CSV_Include project for use as a runtime project; set it aside for use as a template for new projects.

If creating a project based on the CSV_Include templates, don't include pages based on templates that use a different style, including the earlier Include project. Doing so might affect functionality.

See Also

Creating Pages

Creating a Privileged User

Some CSV_Include project content is restricted via a user login. Without a valid login, some project functionality will be disabled. For example, the Tools page is mostly inactive if you log in as a user with restricted privileges.

By default, the following elements within the CSV_Include project are restricted by global privileges.

Element	Global Privilege	Associated Function
Admin Tools page	8	[Privilege]EngTools
Editing users	8	[Privilege]EditUser

Project shutdown	0	[Privilege]Shutdown
Acknowledge alarms	1	[Privilege]AckAlarms
Disable alarms	8	[Privilege]DisableAlarms

When configuring a CSV_Include project, check that your users have appropriate access to the available functionality. Verify that your users can acknowledge alarms if necessary, and that they have access to the full functionality of the Admin Tools page.

To adjust the global privileges for the elements previously listed for a more complex security architecture, adjust the `[Privilege]` parameters in the `Citect.ini` file. Click the relevant link in the previous table above for details.

See Also

Running the Computer Setup Wizard

Running the Computer Setup Wizard

As with any CitectSCADA project, you need to run the Computer Setup Wizard on any machine where a CSV_Include project will be run. See Running the Computer Setup Wizard for more information.

How you set up a computer using the Wizard is mostly up to you; however, due to some necessary settings, you need to run the Computer Setup Wizard as a custom setup. The Wizard pages you need to configure are:

- Events Setup Page
- Security Setup - Control Menu Page

Events Setup page

The CSV_Include project includes three preconfigured events.

- `CSV_TrendXClient`: enables the Instant Trend feature.
- `CSV_TrendXServer`: enables the Instant Trend feature.
- `CSV_AlarmClient`: enables audible alarm siren.

Select and enable the necessary events listed above. To support the instant trend feature, you need to enable both the `CSV_TrendXClient` and `CSV_TrendXServer` events on the computer acting as your Trends Server. Only `CSV_TrendXClient` needs to be enabled on your runtime machines. To support audible alarms, enable `CSV_AlarmClient`.

select every of the event unless you need to disable the associated functionality.

Security Setup - Control Menu page

Provides a Display Title Bar option. As the CSV_Include pages feature an XP-styled title bar, deselecting this option allows your project to run successfully in full-screen mode (run CSV_Include-based projects in fullscreen mode).

See Also

Creating a Privileged User
Setting Up Instant Trending
Implementing Audible Alarms

Setting Up Instant Trending

Instant trending allows you to monitor a variable tag visually from your CitectSCADA project, without configuring the tag it as a trend tag. Without the support of the CitectSCADA Trends Server to facilitate this functionality, instant trending requires you to perform the following setup:

1. **Include the CSV_InstantTrend project.** To implement the Instant Trend feature in one of your own projects, you need to add the CSV_InstantTrend project as an Included project. (See Including a project in the current project.)
2. **Enable the CSV_TrendX events**. As described in Running the Computer Setup Wizard, you need to enable the events `CSV_TrendXClient` and `CSV_TrendXServer` for instant trending to work. Verify that both events are enabled on the Trends Server computer, and that `CSV_TrendXClient` is enabled on any runtime machines that will use instant trending.
3. **Call the Instant Trend display**. Verify that any buttons or menu items you configure to launch the Instant Trend display call the function `CSV_TrendX_Display()`. call this function with no arguments set, as this will implement the default settings. This loads the correct page with appropriate display settings.
4. **Set the default display duration.** The Instant Trend window only stays open for a set period of time. Use the [TrendX]Duration parameter to adjust the default period of time as appropriate.
5. **Configure the Tag Selection dialog**. You can decide whether or not to load a list of current tags into the Tag Selection dialog when using instant trending. Having no tags appear in the selection dialog might be useful if your project has many tags. See the parameter [TrendX]TagListEnable in the Computer Setup Editor online help.
6. **Check the key sequence used to load tags**. You can load tags directly into the Instant Trend display by putting the mouse cursor on a tag value on a graphics page and pressing the plus (+) key. However, make sure that this key sequence does not

clash with other application functionality. To change the key sequence for this feature, see the parameter[TrendX]KeySeq.

See Also
Trend Page Templates

Displaying a Project on Multiple Monitors

The CSV_Include project can display several pages simultaneously across multiple computer screens. If your hardware can run multiple monitors off a single computer, the CSV_Include project can display a different page on up to six screens at a time.

To successfully run your project across multiple monitors, adjust the following parameters:

- **`[MultiMonitors]Monitors`**. Adjust this parameter to indicate how many monitors you're project will be running on.
- **`[MultiMonitors]StartupPage`***n*. This parameter determines which page will appear on each monitor at startup. You have to duplicate this parameter for each monitor. For example, `StartupPage1` determines the page that appears on the first monitor at startup, `StartupPage2` determines what appears on the second monitor, and so on.

Many types of hardware support multiple-monitor display. The CSV_Include project has been tested on a PC with multiple outputs from a single video card, but not on other architectures. The ability to display project pages on multiple screens might be affected by hardware architectures that have not been tested.

Implementing Audible Alarms

The CSV_Include project offers support for audible alarms. You can configure a project so that a selected wav file is sounded whenever an alarm of a particular priority is triggered. You can even assign different sounds to different alarm priorities, allowing the urgency of an alarm to be determined from its sound.

To implement audible alarms in your project:

1. Verify that the alarms you want to trigger an audible alert are assigned to a category and given a particular priority.
2. Adjust the parameter `[Alarm]Soundn`. This parameter determines the wav file used when an alarm sounds, based on the priority of the alarm (n). For example, `[Alarm]Sound1` identifies the .wav file that will be sounded whenever a priority 1 alarm is triggered.
3. If necessary, adjust the parameter `[Alarm]SoundnInterval`. This parameter determines how long the selected wav file sounds, based on the priority of the alarm (n). For

example, [Alarm]Sound1Interval sets the length of time a priority 1 alarm is sounded for.

4. Enable the CSV_AlarmClient event on any machine you want an audible alert sounded from. You can enable this event from the Events Setup page of the Computer Setup Wizard. See Running the Computer Setup Wizard.
5. Verify that your users have appropriate privileges associated with their login to acknowledge alarms, otherwise they will not be able to silence an alarm. See Creating a Privileged User and the parameter [Privilege]AckAlarms.

See Also

Alarm Page Templates

Creating Pages

When considering the pages necessary for your project, first determine if you can use any of the predefined pages in the CSV_Include project:

Page name	Description
CSV_Trend	Default 8-pen Trend page, called from the Trend button on the Navigation toolbar.
CSV_Alarm	Active Alarms page, called from the Alarms toolbar.
CSV_Alarm-Hardware	Hardware Alarms page, called from the Alarms toolbar.
CSV_Alarm-Disabled	Disabled Alarms page, called from the Alarms toolbar.
CSV_Alarm-Summary	Alarms Summary page, called from the Alarms toolbar.
CSV_Admin-Tools	Admin Tools page, called from the Tools button on the Navigation toolbar.

There are also several pages that are accommodated by the Navigation toolbar that you'll need to create if necessary. Decide if the following pages would be useful in your project and create them using the appropriate name:

Page name	Description

Home	Called from the Home button on the Navigation toolbar.
Network	Called from the Network button on the Navigation toolbar. This button does not display if a page called "Network" does not exist.

The links between the pages listed and the buttons that call them are defined by the [Navigation] parameter settings within the `Citect.ini` file. If you want a button to call a page with a different name, adjust these parameter settings. For details see "CSV Include Parameters" in the CitectSCADA Technical Reference.

Creating new pages

Normal and Popup templates are designed with minimal content to enable the presentation of customer-necessary information and plant mimics.

Pages are created based on these templates within Graphics Builder by choosing the necessary template from the Use Template dialog (**File** | **New Page**). You can access the CSV_Include templates by selecting **csv_xp_01** from the **Style** field.

Note: If you try to launch a CSV template from within Citect Explorer, you need to drill down to `[Project Name]/Graphics/Templates/csv_xp_01/XGA` to locate it.

The CSV_Include project can add rollover buttons to pages without drawing and configuring multiple elements. You can add text to a page that has a button appear behind it whenever the cursor passes by adding the function `DspButtonRollOver()` to a button's visibility property. No arguments are necessary.

See Also

Adding user assistance to a page

Adding user assistance to a page

The CSV_Include project includes a Help button on the Navigation toolbar. As you create your pages, you can configure this button to provide information to the user about the content of a page, and its supported functionality.

The information provided to your operators needs to be be created as individual HTML pages and compiled into a HTML help file (.chm). The compiler necessary to create a .chm file is available for download from the Microsoft Developer Network (MSDN). Go to http://msdn.microsoft.com and search for "HTML Help Workshop". An SDK is available that includes the necessary downloads and information about creating an HTML help project.

Once you have created a .CHM file, it is recommended you place it in the project directory.

To configure the help button for a project page

1. Select the project you've created in Project Explorer.
2. Go to Project Editor and select **Parameters** from the **System** menu. The Parameters dialog will appear.
3. In the **Section Name** field, type in "HelpButton".
4. In the **Name** field, identify the page you would like to add help to using the following format:

```
SetTopic:<PageName>
```

where <PageName> is the name of the page included in your CitectSCADA project. Do not include a space after the colon.

5. In the **Value** field, identify the associated HTML Help file and topic using the following format:

```
<HelpFileName>|<HTMLPageName>
```

where:

- <HelpFileName> is the name of the compiled HTML Help file you've created (for example MyHelpFile.chm)
- <HTMLPageName> is the name of the compiled HTML page you'd like displayed as a help topic (for example Help_Topic.html).

6. If necessary, add a **Comment**.
7. Click **Add**.

The help button will now launch the identified HTML file with the selected topic showing.

To disable the help button for a project page

1. Select the project you've created in Project Explorer.
2. Go to Project Editor and select **Parameters** from the **System** menu. The Parameters dialog will appear.
3. In the **Section Name** field, type in "HelpButton".
4. In the **Name** field, identify the page you would like to disable the help button for using the following format:

```
SetTopic:<PageName>
```

where <PageName> is the name of the page included in your CitectSCADA project. Do not include a space after the colon.

5. In the **Value** field, type in DISABLED.
6. If necessary, add a **Comment**.
7. Click **Add**.

The help button on the identified page will now be unavailable during runtime.

Note: Do not duplicate this parameter for a single project page.

See Also

Using Pages and Templates

Creating Custom Menus

The Custom Menu Toolbar, located directly beneath the page title bar, allows you to create drop-down menus capable of calling a Cicode function or navigating to a specific page (i.e. to a specific plant mimic).

You can disable the menu bar on pages by adjusting the parameter `[Page]MenuDisable`. If you need to disable the menu on a particular page, you can apply this parameter as an environmental variable. To do this, open the necessary page in Graphics Builder, go to the Properties dialog (**File** | **Properties**), and insert the parameter on the Environment tab.

Menu Configuration tool

The content of the menus can be configured via the Menu Configuration tool, which is launched from the Citect Configuration panel of the Admin Tools page.

The Menu Configuration tool has two panels. The panel on the left represents the menus configured for the current project in a directory structure. The panel on the right includes information about the item currently selected in the left panel.

The directory in the left panel is a graphical representation of a DBF lookup table that forms the basis of the menus displayed at runtime. The hierarchical structure is determined by the following fields within this table:

Page	The page field is defined as either "Generic", or the name of a page within the project. Generic specifies that the menu is associated with every page, a specific page name indicates the menus that will appear just on that particular page.

Menuname	Name(s) of the menus included on the specified page.
Menuitem	Item(s) that appear within each menu.
Submenu	Submenus that appear in a menu (optional). Adding a sub menu automatically removes the action defined for the menu item it is branched from, as the parent becomes a placeholder for the list of sub menus.

The Menu Names that appear are the five defaults: Pages, Trends, Alarms, File and Tools.

The fields you can expect to see associated with an item, as defined in the DBF table, are as follows:

Action	Either the name of the page to display or a Cicode function. If specifying a Cicode function, it needs to be prefixed by a question mark ("?"). To invoke a Cicode function that has arguments from the Menu System, use the following format: function name, space, comma-separated list of string arguments. For example, ?LoadPav dP45.pav.
Privilege	The login privilege necessary to trigger the associated action.
Disabled	Indicates if the item is currently disabled (embossed). For example, if the current login does not meet the necessary privilege set for the item (see above), this field will be set to True and the menu item will appear embossed to indicate its currently not active.
Checked	Indicates if the menu item is currently checked (with a tick). True indicates
Btnwidth	Specifies the width of the button (optional).

See Also

Building custom menus

Building custom menus

Creating a custom menu entails adding the necessary pages, menus, and items to the Configure Menus directory structure, and then assigning the necessary action and privileges to each item. This is achieved via the right-click menu.

Right-click a branch in the directory to display the context menu, which shows the available actions for the current selection. The right-click menu includes the following actions:

Edit Item	Available when an Item is selected. This option displays the Edit Item dialog, which allows you to define the fields associated with the current Item. In particular, it allows you to specify the action associated with the item. (See Editing an item)
New Page	Adds a new page to the menu configuration. Use this option to create new custom menus designed specifically for a particular page in your project. The name you give the new page in the menu configuration needs to be identical to the name of the page in your project that you want the menus to be applied to.
New Button	Adds a new menu button to the current page.
New Item	Adds a new item to the currently selected menu.
New Sub Item	Adds a sub item to the currently selected item. A subitem overwrites the action configured for its parent item. The parent item becomes a label identifying the list of Sub Items assigned to it.
Delete Page	Deletes the currently selected Page from the menu configuration.
Delete Button	Deletes the currently selected Menu Button from the menu configuration.
Delete Item	Deletes the currently selected Item from the menu configuration.
Delete Sub Item	Deletes the currently selected Sub Item from the menu configuration.
Copy page	Copies the currently selected page and pastes it to the end of the menu configuration. This option is useful if you want a page to include the generic page menu configuration, but with additional menus that only appear when the particular page is displayed. To do this, copy the generic page, rename it to match the page you want to customize the menus for, then add the necessary menus and items.
Save	Saves the current menu configuration.

Editing an item

You configure a menu item's functionality by using the **Edit Item Menu** dialog.

Use this dialog to identify the action an item will trigger, be it navigation to a specific page, or execution a function. To launch this dialog, right-click the item to configure and choose **Edit Item** from the menu.

fill out the fields as follows:

Action	Indicates the name of the page to display, or a Cicode function. If specifying a Cicode function, it needs to be prefixed by a question mark ("?").
Privilege	The login privilege necessary to trigger the action. Only users with appropriate access privileges will be able to use this item.
Disabled	As this setting is determined automatically depending on the current user's access privileges, leave this set to **False** unless you want the item disabled by default.
Checked	Leave this set to **False** unless you want the item checked by default.
Button Width	Leave this set to zero (**0**) as this sets the menu button to the appropriate width. To set a specific width for the button, input the width in pixels.

See Also

Common Toolbars

Creating an Alarms Group

The CSV_Include project allows you to use "alarm groups" to display a specific set of tags defined by the alarm category and area settings configured within the runtime CitectSCADA project.

For example, you could create a group containing category one alarms. This group could then be used as a filter to create a list of the category one alarms currently displayed on the Active Alarms page.

Alarm groups are configured by clicking the appropriate link on the Citect Configuration panel to the left of the Admin Tools page. The functionality can also be added to a custom menu for easier access.

To configure an alarm group:

1. Go to the Admin Tools page of a runtime project by clicking the **Tools** button.
2. Select **Configure an Alarm Group** from the Citect Configuration panel. The Configure Alarm Groups dialog appears.

3. In the **Alarm Group Description** box, type the name you want to use to identify the group.
4. In the **Categories** box, list the alarm categories configured in the CitectSCADA runtime project that you want to use to define the group.
5. In the **Area** box, type the areas defined in the CitectSCADA project you want to use to define the group.
6. Click **Add**.

The information to the right of the dialog indicates how many alarm groups are configured and where you are currently positioned in this list. You can scroll through the list of configured groups by using the up and down arrows.

To change a group, locate it in the list, change it as appropriate, then click **Replace**.

See Also

Creating a Trends Group

Creating a Trends Group

The CSV_Include project allows you to use "trend groups" to display a specific set of trend tags. A trend group includes a set of up to eight variable tags that can be automatically loaded into a trend display without having to select each tag individually.

Trend groups are configured by clicking the appropriate link on the Citect Configuration panel to the left of the Admin Tools page. The functionality can also be added to a custom menu for easier access.

To configure a trend group:

1. Click **Tools**.
2. Select **Configure a Trend Group** from the Citect Configuration panel. The Configure Trend Groups dialog appears.
3. In the **Description** text box, type the name you want to use to identify the group.
4. In the **Trend Pen** text boxes, select the eight variable tags you want the group to trend. Click the button on the right of each field to view the available tags within the CitectSCADA runtime project.
5. In the **Area** text box, type the areas defined in the CitectSCADA project you want to use to define the group.
6. Click **Add**.

The information to the right of the dialog indicates how many trend groups are configured and where you are currently positioned in this list. You can scroll through the list of configured groups by using the up and down arrows.

To change a group, locate it in the list of groups or browse to it using the selection button to the right of the **Description** text box, make the necessary changes, then click **Replace**.

See Also

Creating an Alarms Group

Using Environment Variables

Whereas parameters apply specified rules to every template page, environment variables apply rules to *specific* pages. In this way, you can use environment variables to specify functionality for specific pages.

For example, to display a specific page, you'd specify the page name as the environment variable. To call a specific Cicode function, you'd specify the function name (with a "?" prefix), space, then a list of comma separated arguments, like this: `?WinPrint LPT1,0,0,Trend.pal`. (The "?" indicates that the variable is to be interpreted as a function call rather than a page to display.)

Let's look at a more detailed example: usually the standard **Print** button is used for printing a graphics page in WYSIWYG format. If you have a text/html file displayed on the page, or perhaps a trend, you can create your own print function (for example, `PrintTrend`) and specify it in the environment variable, as `?PrintTrend`. To reduce the amount of ink used, specify the `WinPrint` function with a user-defined palette. In this case you would specify it as a parameter rather than as an environment variable so that it applies to every page.

> **Note:** With the standard CSV_Include trend pages, you can print the screen using the standard **Print** button, or plot the trend using the **Trend Plot** button. give the user both alternatives as they both serve different purposes.

Glossary

1

10base2

Ethernet implementation on thin coaxial cable. Typically uses a BNC connection.

10base5

Ethernet implementation on thick coaxial cable.

10baseT

Ethernet implementation on unshielded twisted pair. Typically uses as RJ45 connection.

A

Accredited - Level 1

Drivers developed under the CiTDriversQA96 Driver Quality and Accreditation System, which ensures the driver was designed, coded, and tested to the highest possible standards.

Accredited - Level 2

Drivers developed using the CiTDriversQA92 Driver Quality and Accreditation System.

accumulator

A facility that allows you to track incremental runtime data such as motor run hours, power consumption, and downtime.

active alarm

An active alarm is an alarm in one of the following states: ON and unacknowledged; ON and acknowledged; OFF and unacknowledged.

advanced alarm

Triggered when the result of a Cicode expression changes to true. Use advanced alarms only when alarm functionality cannot be obtained with the other alarm types. If you configure too many advanced alarms, your system performance can be affected.

alarm categories

You can assign each alarm to a category, and then process each category as a group. For example, for each category, you can specify the display characteristics, the action to be taken when an alarm in the category is triggered, and how data about the alarm is logged. You can also assign a priority to the category, which can be used to order alarm displays, filter acknowledgments, and so on.

alarm display page

The alarm display page displays alarm information in the following format: Alarm Time, Tag Name, Alarm Name, Alarm Description.

alarm summary page

Displays alarm summary information in the following format: alarm name, time on, time off, delta time, comment.

Alarms Server

Monitors all alarms and displays an alarm on the appropriate control client(s) when an alarm condition becomes active.

analog alarms

Triggered when an analog variable reaches a specified value. supports four types of analog alarms: high and high high alarms; low and low low alarms; deviation alarms; and rate of change alarms.

animation number files (.ANT)

ASCII text files that contain a list of animation points (ANs) and the coordinate location (in pixels) of each point.

animation point

The points on a graphics page where an object displays. When you add an object to your page, automatically allocates a number (AN) to the animation point, (i.e., the location of the object).

area

A large application can be visualized as a series of discrete sections or areas. Areas can be defined geographically (where parts of the plant are separated by vast distances) or logically (as discrete processes or individual tasks).

arguments

Values (or variables) passed in a key sequence to a keyboard command in runtime (as operator input). Arguments can also be the values (or variables) passed to a Cicode function when it executes.

Association

An association is the name or number you use when defining a Super Genie substitution, the value or values of which are dynamically generated at runtime.

attachment unit interface (AUI)

Typically used to interface to a transceiver through what is often known as a drop cable.

automation component (ActiveX object)

ActiveX objects typically consist of a visual component (which you see on your screen) and an automation component. The automation component allows the interaction between the container object and the ActiveX object.

B

baud rate

The number of times per second a signal changes in a communication channel. While the baud rate directly affects the speed of data transmission, the term is often erroneously used to describe the data transfer rate. The correct measure for the data rate is bits per second (bps).

BCD variable (I/O device)

BCD (Binary Coded Decimal) is a two-byte (16-bit) data type, allowing values from 0 to 9,999. The two bytes are divided into four lots of four bits, with each lot of four bits representing a decimal number. For example the binary number 0010 represents decimal 2. Thus the BCD 0010 0010 0010 0010 represents 2,222.

bottleneck

A bottleneck occurs when too many requests are being sent to a PLC communication link/data highway. It can occur with all types of protocols, and is dependent on several factors, including the frequency of requests, the number of duplicated (and hence wasteful) requests, whether the protocol supports multiple outstanding requests, as well as other network traffic.

browse sequence

A series of graphics pages linked by a browse sequence, which is a linear navigation sequence within your runtime system that uses Page Previous and Page Next commands.

byte variable (I/O device)

Byte is a one-byte data type, allowing values from 0 to 255. One byte consists of 8 bits. Each ASCII character is usually represented by one byte.

C

cache (I/O device data cache)

When caching is enabled, all data read from a I/O device is stored temporarily in the memory of the I/O server. If another request is made (from the same or another control client) for the same data within the cache time, the I/O server returns the value in its memory, rather than read the I/O device a second time.

callback function

A function that is passed as an argument in another function. Callback functions must be user-written functions.

Cicode

Programming language designed for plant monitoring and control applications. Similar to languages such as Pascal.

Cicode blocking function

A Cicode function that blocks, or waits, for an asynchronous event to complete before returning.

CiNet

CiNet is no longer supported. CiNet was designed as a low speed wide area network (for remote monitoring applications). If you have a widely-distributed application where computers are separated by vast distances, using a LAN to connect your control clients can be expensive. To connect control clients in this instance, use Microsoft's remote access server (RAS) or a Microsoft-approved solution, such as Shiva LanRover.

citect.ini file

A text file that stores information about how each computer (servers and control clients) operates in the configuration and runtime environments. The Citect.INI file stores parameters specific to each computer and therefore cannot be configured as part of the project.

CiUSAFE

CiUSAFE is the application used to manage the hardware key that authorizes use of your software within the agreed limitations.

client

A computer that accesses shared network resources provided by another computer called a server. 's client-server based architecture is designed to distribute the processing tasks and optimize performance.

cluster

A discrete group of alarms servers, trends servers, reports servers, and I/O servers. It would usually also possess local control clients. For a plant comprising several individual sections or systems, multiple clusters can be used, one cluster for each section.

command

A command performs a particular task or series of tasks in your runtime system. A command is built from Cicode and can consist of just a function or a statement.

communications link

A connection between computers and peripheral devices, enabling data transfer. A communications link can be a network, a modem, or simply a cable. .

communications port

PC port used for sending and receiving serial data (also called serial or COM ports).

computer

A computer running . Other common industry terms for this computer could be node, machine or workstation.

Control Client

The interface between the runtime system and an operator. If you are using on a network, all computers (on the network) are control clients.

control inhibit mode

Prohibits writing to the Field VQT tag element of a tag extension.

custom alarm filter

Custom alarm filters provide a way to filter and display active alarms. Up to eight custom filter strings can be assigned to a configured alarm. In conjunction with a user-defined query function, the custom filters enable operators to identify and display active alarms of interest.

D

data acquisition board

Data acquisition boards communicate directly with field equipment (sensors, controllers, and so on). You can install a data acquisition board in your server to directly access your field equipment.

data bits

Group of binary digits (bits) used to represent a single character of data in asynchronous transmission.

data communications equipment (DCE)

Devices that establish, maintain, and terminate a data transmission connection. Normally referred to as a modem.

data terminal equipment (DTE)

Devices acting as data source, data sink, or both.

data transfer

Transfer of information from one location to another. The speed of data transfer is measured in bits per second (bps).

data type (I/O device)

Type of I/O device variable. I/O devices may support several data types that are used to exchange data with . You must specify the correct data type whenever I/O device variables are defined or referenced in your system.

DB-15

Often called a `D' type connector due to the vague D shape of the casing. Has 15 pins arranged in two rows of 8 and 7 pins. While not as common as DB-9 or DB-25 they may be found on some computers and data communication equipment. Comes in both male (pins protruding) and female (pin sockets) configurations.

DB-25

Often called a `D' type connector due to the vague D shape of the casing. Has 25 pins arranged in two rows of 13 and 12 pins. This kind of connection is a part of the standard for RS-232-D and is found on many computers, modems and other data communication equipment. Comes in both male (pins protruding) and female (pin sockets) configurations.

DB-9

Often called a `D' type connector due to the vague D shape of the casing. Has 9 pins arranged in two rows of 5 and 4 pins. This kind of connection is common and is often used as the serial (com) port in computers. Often used in modems and other data communication equipment. Comes in both male (pins protruding) and female (pin sockets) configurations.

debug.log

The debug.log file stores information about an unexpected system shut down or other internal issues. If an unexpected shutdown occurs, it will identify the version and path of each DLL being used at the time.

deviation alarm

Triggered when the value of a variable deviates from a setpoint by a specified amount. The alarm remains active until the value of the variable falls (or rises) to the value of the deadband. .

dial-back modem

Only returns calls from remote I/O devices.

dial-in modem

Only receives calls from remote I/O devices, identifies the caller, then hangs up immediately so it can receive other calls. then returns the call using a dial-back modem.

dial-out modem

Makes calls to remote I/O devices in response to a request; e.g., scheduled, event-based, operator request, and so on. Also returns calls from remote I/O devices.

Digiboard

A high-speed serial board manufactured by the Digiboard Corporation.

digital alarms

Triggered by a state change in a digital variable. Use these alarms when a process has only one of two states. You can use either the on (1) state or off (0) state (of a digital variable) to trigger the alarm.

digital variable (I/O device)

Usually associated with discrete I/O in your I/O device, a digital variable can only exist in one of two states: on (1) or off (0). Allowed values for the digital data type are therefore 0 or 1. Discrete inputs (such as limit switches, photoelectric cells, and emergency stop buttons) and discrete outputs are stored as digital variables.

disk I/O device

A disk file that resides on the hard disk of a computer and emulates a real I/O device. The value of each variable in the disk I/O device is stored on the computer hard disk. The disk I/O device is not connected to any field equipment in the plant.

display period

Defines the rate at which trend data is displayed on the trend page.

distributed processing

For large applications with large amounts of data, you can distribute the data processing to reduce the load on individual computers.

distributed servers

If your plant consists several sections or systems, you can assign a cluster to each individual section, and then monitor all sections using one control client.Note: Don't use distributed servers to split up a single section or process into discrete areas. A single cluster system with distributed processing would be better used here since it would not be hampered by the maintenance overhead of a distributed server system (such as extra project compilations, and so on).

dither (imported bitmaps)

A method of approximating colors in imported or pasted bitmaps that involves combining pixels of different or colors from a color palette.

domain name server (DNS)

Database server that translates URL names into IP addresses.

dot notation

Used for Internet addresses. Dot notation consists of four fields (called octets), each containing a decimal number between 0 and 255 and separated by a full stop (.).

driver

A driver is used to communicate with control and monitoring devices, allowing the run-time system to interact directly with different types of equipment. Communication with an I/O device requires a device driver which implements the communication protocol(s).

driver logs

Driver logs relate to the operation of a particular driver and are named accordingly. For example, the OPC driver is logged in 'OPC.dat'.

duplex

The ability to send and receive data over the same communication line.

dynamic data exchange (DDE)

A Microsoft Windows standard protocol set of messages and guidelines that enables communication between Windows applications on the same Windows computer.

dynamic data exchange (DDE) Server

A Windows standard communication protocol supported by . The I/O server communicates with the DDE server using the Windows standard DDE protocol. DDE servers are appropriate when data communication is not critical as DDE servers are not designed for high-speed data transfer.

E

empty value

Indicates that the variant has not yet been initialized (assigned a value). Variants that are empty return a VarType of 0. Variables containing zero-length strings (" ") aren't empty, nor are numeric variables having a value of 0.

Ethernet

Widely used type of local area network based on the CSMA/CD bus access method (IEEE 802.3).

Event data displayed by time

As an alternative to viewing event trend data by event number, it is possible to see event trends across a timeline. When event trends are shown by time, the trend graph includes a start and end time and enables operators to see both the time of a triggered event, and the elapsed period between events. This data can also be displayed on the same graph as a periodic trend.

event trend/SPC

To construct an event trend/SPC, takes a sample when a particular event is triggered (in the plant). This sample is displayed in the window. The event must then reset and trigger again, before the next sample is taken. Events are identified by the event number. .

expression

A statement (or group of statements) that returns a value. An expression can be a single variable, a mathematical formula, or a function.

F

Field element

The latest tag field data received from a device.

file server

A computer with a large data storage capacity dedicated to file storage and accessed by other client computers via a network. On larger networks, the file server runs a special network operating system. On smaller installations, the file server may run a PC operating system supplemented by peer-to-peer networking software.

full duplex

Simultaneous two-way (in both directions) independent transmission (4 Wires).

G

generic protocol

A pseudo-protocol supported by disk I/O devices that provides a convenient way to represent disk data. The generic protocol is not a real protocol (communicates with no physical I/O device).

Genie

If you have numerous devices of the same type (e.g., 100 centrifugal pumps), the display graphics for each will behave in much the same way. Using Genies, you only have to configure common behavior once. The graphics can then be saved as a Genie and pasted once for each device.

global Cicode variable

Can be shared across all Cicode files in the system (as well as across include projects).

global client

A control client used to monitor information from several systems or sections (using clusters).

graphics bounding box

A faint (grayed) dotted rectangular box outline defining the exterior boundary region of a graphic object. Only visible and active when the graphics object is selected and being resized. Contains sizing handles in each corner and (if sized large enough to display) one in the centre of each side.

graphics page

A drawing (or image) that appears on a workstation to provide operators with control of a plant, and display a visual representation of conditions within the plant.

group (of objects)

allows you to group multiple objects together. Each group has a unique set of properties, which determine the runtime behavior of the group as a whole.

H

half duplex

Transmission in either direction, but not simultaneously.

hardware alarm

A hardware alarm indicates that an error has been detected in your system. Typically displayed on a dedicated hardware alarms page, this type of alarm may indicate that a loss of communication has occurred, that Cicode can not execute, that a graphics page is not updating correctly, or that a server has become inoperative. A description and error code are provided to help decipher the cause of the problem.

histogram

A bar graph that shows frequency of occurrence versus value. Quite often the data is fitted to a distribution such as a normal distribution. .

I

I/O Device

An item of equipment that communicates with plant-floor control or monitoring equipment (sensors, controllers, and so on). The most common I/O devices are PLCs (programmable logic controllers); however, supports a wide range of I/O devices, including loop controllers, bar code readers, scientific analyzers, remote terminal units (RTUs), and distributed control systems (DCS). can communicate with any I/O device that has a standard communications channel or data highway.

I/O device address

The (logical) location of the I/O device in the system. Each I/O device must have a unique address in the system, unless the I/O device is defined in other servers (to provide redundancy). If redundancy is used, the I/O device must then have the same I/O device name, number, and address for each server.

I/O device variable

A unit of information used in . Variables are stored in memory registers in an I/O device. exchanges information with an I/O device by reading and writing variables. refers to I/O device variables by their register addresses. I/O devices usually support several types of variables; however, the most common are digital variables and integer variables.

I/O server

A dedicated communications server that exchanges data between I/O devices and control clients. No data processing is performed by the I/O server (except for its local display). Data is collected and passed to the control clients for display, or to another server for further processing. All data sent to an I/O device from any computer is also channelled through the I/O server. If data traffic is heavy, you can use several I/O servers to balance the load.

imestamp (T)

The timestamp of when the element was last updated on a tag extension.

include file (.CII)

There is a maximum number of characters that you can type in a Command or Expression field (usually 128). If you need to include many commands (or expressions) in a property field, you can define a separate include file that contains commands or expressions. An include file is a separate and individual ASCII text file containing only one sequence of commands or expressions that would otherwise be too long or complicated to type into the command or expression field within . The include file name is entered instead, and the whole file is activated when called.

integer variable (Cicode)

A 4-byte (32-bit) data type allowing values from 2,147,483,648 to 2,147,483,647.

integer variable (I/O device)

A 2-byte data type, allowing values from -32,768 to 32,767, that is used to store numbers (such as temperature or pressure). Some I/O devices also support other numeric variables, such as real (floating point) numbers, bytes, and binary-coded decimals.

Internet Display Client

Allows you to run projects over the Internet from a remote location. It is basically a "runtime-only" version of : you can run your project from that computer, just as you would from any normal client.

interrupt

An external event indicating that the CPU should suspend its current task to service a designated activity.

IP address

A unique logical address used by the Internet Protocol (IP). Contains a network and host ID. The format is called dotted decimal notation, and is written in the form: w.x.y.z.

K

Kernel

The Kernel allows you to perform low-level diagnostic and debugging operations for runtime analysis of your system. A set of diagnostic windows display low-level data structures, runtime databases, statistics, debug traces, network traffic, I/O device traffic and so on.

keyboard command

Consist of a key sequence that an operator enters on the keyboard, and an instruction (or series of instructions) that executes when the key sequence is entered. Keyboard commands can be assigned to an object or page, or they can be project-wide.

knowledge base

Provides high-level technical information beyond the scope of standard technical documentation that is updated regularly and available at http://www.citect.com.

kurtosis

An index indicating the degree of peakedness of a frequency distribution (usually in relation to a normal distribution). Kurtosis < 3 indicates a thin distribution with a relatively high peak. Kurtosis > 3 indicates a distribution that is wide and flat topped.

L

language database

When a project is compiled, creates a language database (dBASE III format) consisting of two fields: native and local. Any text marked with a language change indicator is automatically entered in the native field. You can then open the database and enter the translated text in the local field.

link

A copy of a library item, possessing the properties of the library original. Because it is linked, the copy is updated whenever the original is changed.

local area network (LAN)

A system that connects computers to allow them to share information and hardware resources. With real-time LAN communication, you can transfer data, messages, commands, status information, and files easily between computers.

local Cicode variable

Only recognized by the function within which it is declared, and can only be used by that function. Local variables must be declared before they can be used. Any variable defined within a function (i.e., after the function name) is a local variable, therefore no prefix is needed. Local variables are destroyed when the function exits and take precedence over global and module variables.

local language

The language of the end user. Runtime display items such as alarm descriptions, button text, keyboard/alarm logs, graphic text, Cicode strings and so on can be displayed in the local language, even though they may have been configured in the language of the developer (native language).

local variable

Local variables allow you to store data in memory when you start your runtime system. They are created each time the system starts, and therefore do not retain their values when you shut down.

log files

Log files are a record of time-stamped system data that can be analyzed to determine the cause of a problem. The available log files include syslog.dat, tracelog.dat, debug.log, kernel.dat, and dedicated driver logs.

long BCD variable (I/O device)

A 4-byte (32-bit) data type, allowing values from 0 to 99,999,999. The four bytes are divided into eight lots of four bits, with each lot of four bits representing a decimal number. For example the binary number 0011 represents decimal 3. Thus the BCD 0011 0011 0011 0011 0011 0011 0011 0011 represents 33,333,333.

long variable (I/O device)

A 4-byte (32-bit) data type allowing values from 2,147,483,648 to 2,147,483,647.

low and low low alarms

Defined by specifying the values of the variable that trigger each of these alarms. As a low alarm must precede a low low alarm, the low alarm no longer exists when the low low alarm is triggered. Note that the variable must rise above the deadband before the alarm becomes inactive. .

M

maximum request length

The maximum number of data bits that can be read from the I/O device in a single request. For example, if the maximum request length is 2048 bits, the maximum number of integers that can be read is: 2048/16 = 128.

Metadata

Metadata is a list of names with corresponding values that is attached to an objects animation point.

millisecond trending

Allows you to use a trends sample period of less than one second.

mimic

A visual representation of a production system using an organised set of graphical pages. .

minimum update rate

A pre-defined period of time after which tag update value notifications are sent to subscription clients

module Cicode variable

Specific to the file in which the variable is declared. This means that it can be used by any function in that file, but not by functions in other files. By default, Cicode variables are defined as module, therefore prefixing is not required (though a prefix of MODULE could be added if desired). Module variables should be declared at the start of the file.

multi-digital alarms

Use combinations of values from three digital variables to define eight states. For each state, you specify a description (e.g., healthy or stopped), and whether or not the state triggers an alarm.

N

native language

Generally the language of the project developer. Display items such as alarm descriptions, button text, keyboard/alarm logs, graphic text, Cicode strings and so on can be configured in the native language, and displayed, at runtime, in the language of the end-user (local language).

network

A group of computers and peripheral devices, connected through a communications link. Data and services (e.g., printers, file servers, and modems) can be shared by computers on the network. A local network of PCs is called a LAN.

network computer

A computer running that is connected to a LAN through a network adaptor card and network software. .

Network Dynamic Data Exchange (NetDDE)

Enables communication between Windows applications on separate computers connected across a common network.

nodes

A structural anchor point for a graphic object, usually visible as a small square box superimposed over a graphic. Nodes will be located separately at the start, at the end, and at every change in direction within a graphic object. .

normal distribution

Also known as a `bell' curve, the normal distribution is the best known and widely applicable distribution. The distribution is symmetrical and popularly represents the laws of chance. 68.27% of the area lies between -1 sigma and +1 sigma, 95.45% between -2 sigma and+2 sigma, and 99.73% between -3 sigma and +3 sigma. The values of skewness and kurtosis are used to provide quantitative measures for normality. Assuming that at least 20 samples are used to construct a distribution, a good rule of thumb is to accept the data as a normal distribution when, -1.0 = skewness = 1.0 2 = kurtosis = 4.

null value

Indicates that a variant contains no valid data. Variants that are null return a VarType of 1. Null is not the same as empty, which indicates that a variant has not yet been initialized. It is also not the same as a zero-length string (" "), which is sometimes referred to as a null string. Null is not equivalent to zero or blank. A value of null is not considered to be greater than, less than, or equivalent to any other value, including another value of null. A boolean comparison using a null value will return false.

O

object

Basic building blocks of a graphics page. Most objects possess properties that allow them to change dynamically under user-definable runtime conditions allowing them to provide animated display of conditions within the plant.

object ID (OID)

An object ID associated with every tag in a project that uniquely identifies the tag for use by tag-based drivers, automatically generated at compile. It is used instead of the actual address of the register (which is what most other drivers use to read from and write to I/O devices).

object variable (Cicode)

An ActiveX control that can only be declared with local, module, or global scope.

open database connectivity (ODBC)

Allows applications to access data in database management systems using structured query language (SQL) to access data.

override mode

A state where an invalid tag quality value is overridden by a manually added value.

P

pack

Packing a database re-indexes database records and deletes records marked for deletion. If you edit your databases externally to , you should pack the database afterwards.

page environment variable

A read-only variable associated with a particular page When you make the association, you name the variable, and assign it a value. When the page is opened during runtime, creates the variable. Its value can then be read. When the page is closed, the environment variable memory is freed (discarded).

parity

A communications error-checking procedure. The number of 1's must be the same (even or odd) for each group of bits transmitted without error.

periodic trend

A trend that is sampled continuously at a specified period. You can also define a trigger (an event) to stop and start the trend (when a specified condition occurs in the plant).

persistence cache

Cache data saved to a computer hard disk that allows an I/O server to be shut down and restarted without having to re-dial each I/O device to get its current values. This cache consists of all the I/O device's tag values.

PLC interface board

You can sometimes install a PLC interface board in your server. A proprietary interface board is usually supplied by your PLC manufacturer, and you can connect it to a PLC or a PLC network. You can only use proprietary interface boards with the same brand of PLC.

point limit

An individual digital (or analog) variable read from an I/O device. only counts physical points (and counts them only once, no matter how many times they are used). The point limit is the maximum number of I/O device addresses that can be read and is specified by your license. When you run the point count of your project is checked against the point limit specified by your Hardware Key.

port(s)

Provide the communication gateway to your I/O device(s).

primary Alarms Server

The server that normally processes alarms.

primary Reports Server

The server that normally processes reports.

primary Trends Server

The server that normally processes trends.

Privileges

Level of access applied to system elements within your project. A user assigned a role that possesses the matching privilege can control it.

project

The elements of a monitoring and control system, such as graphics pages, objects, and so on. These elements are stored in files of various types; for example, graphics files for graphics pages, databases for configuration records, and so on. You use the compiler to compile the project into a runtime system.

properties, object

Describes the appearance of an object (size, location, color, and so on.) and its function (the command or expression executed by the object, the privilege required to gain access to the object, and so on).

protocol

Messaging format consisting of a set of messages and guidelines used for communication between the server and an I/O device. The communication protocol determines how and the I/O device communicate; the type of data to exchange; rules governing communication initiation and termination; and error detection.

proxi/proxy server

Caches internet transactions to improve performance by reducing the average transaction times by storing query and retrieved information for re-use when the same request is made again. When an Internet display client (IDC) connects to a proxy server, that server provides the TCP/IP addresses necessary to access report server session information.

PSTN

A public switched telephone network is the network of all the world's public switched telephone networks. It is now primarily digital and includes mobile as well as fixed telephones.

Q

qualified tag reference

Referencing tag data by using the tag name, element name and the item name.

Quality (Q)

The quality of the value of a tag extension.

QualityTimestamp (QT)

The timestamp of when the quality last changed on a tag extension

R

rate of change alarms

Triggered when the value of the variable changes faster than a specified rate. The alarm remains active until the rate of change falls below the specified rate. Deadband does not apply to a rate of change alarm.

real variable (Cicode)

Real (floating point) is a 4-byte (32-bit) data type allowing values from 3.4E38 to 3.4E38. Use a real variable to store numbers that contain a decimal place.

real variable (I/O device)

Real (floating point) is a 4-byte (32-bit) data type, allowing values from 3.4E38 to 3.4E38. Use a real variable to store numbers that contain a decimal place.

record name

Usually the primary property of a database record, referenced in system through its name. Database record names must be unique for each type of database record. Sometimes you can use identical names for different record types. However, to avoid confusion, you should use a unique name for each database record in your application.When you specify a name for a database record, the name must begin with an alphabetic character (A-Z, a-z) and cn only include alphanumeric characters (A-Z, a-z, 0-9) and the underscore character (_). For example, "Pressure," "Motor_10," and "SV122_Open" are all valid database record names. Each database record name can contain up to 16 characters.Database record names are not case-sensitive, so "MOTOR_1," "Motor_1" and "motor_1" are all identical database record names. For this reason use a meaningful name for any database record as well as the necessary naming conventions.

redundancy

A method of using the hardware in a system such that if one component in the system becomes inoperative, control of the system is maintained, and no data is lost.

remote communications

Interaction between two computers through a modem and telephone line.

remote terminal

A terminal remote from the computer that controls it. The computer and remote terminal communicate via a modem and telephone line.

report

A statement or account of plant-floor conditions. reports can be requested when required, on a periodic basis, or when an event occurs.

report format file

Controls the layout and content of reports. The format file is edited using a text editor and can be in either ASCII or RTF format.

Reports Server

Controls report processing. You can request reports at any time or when specific events occur.

reserved words

Words that cannot be used as a name for any database record or Cicode function.

RJ11

A type of IDC plug commonly used in data communications. Recognizable as the style of data plug used in phone line and handset connectors. RJ11 is a 6/4 plug with 6 contacts but only 4 loaded.

RJ12

A type of IDC plug commonly used in data communications. Recognizable as the style of data plug used in phone line and handset connectors. RJ12 is a 6/6 plug with 6 contacts.

RJ45

A type of IDC plug commonly used in data communications. Recognizable as the style of data plug used in phone line and handset connectors. RJ45 is often used with 10baseT and is an 6/8 plug with 8 contacts.

Roles

A defined set of permissions (privileges and areas) that are assigned to users.

RS-232

An industry standard for serial communication. The standard specifies the lines and signal characteristics that are used to control the serial transfer of data between devices.

RS-422

An industry standard for serial communication. The standard specifies the lines and signal characteristics that are used to control the serial transfer of data between devices. RS-422 uses balanced voltage interface circuits.

RS-485

An industry standard for serial communication. The standard specifies the lines and signal characteristics that are used to control the serial transfer of data between devices. RS-485 uses balanced voltage interface circuits in multi-point systems.

runtime system

The system that controls and monitors your application, process, or plant. The runtime system is sometimes called the Man-Machine Interface (MMI), and is compiled from a project.

S

scalable architecture

A system architecture that can be resized without having to modify existing system hardware or software. lets you re-allocate tasks as more computers are added, as well as distribute the processing load.

schedule period

Determines how often the I/O server contacts a scheduled I/O device to read data from it. .

serial communication

Uses the communication port on your computer or a high speed serial board (or boards) installed inside your computer.

server

A computer connected to an I/O device (or number of I/O devices). When is running, the server exchanges data with the I/O device(s) and distributes information to the other control clients as required. A local area network (LAN) computer that perform processing tasks or makes resources available to other client computers. In , client-server architecture distributes processing tasks to optimize performance.

simplex transmission

Data transmission in one direction only.

skewness

An index indicating the degree of asymmetry of a frequency distribution (usually in relation to a normal distribution). When a distribution is skewed to the left (for example), then the tail is extended on that side, and there is more data on the left side of the graph than would be expected from a normal distribution. Positive skew indicates the distribution's mean (and tail) is skewed to the right. Negative skew indicates the distribution's mean (and tail) is skewed to the left.

slider control

Allow an operator to change the value of an analog variable by dragging an object (or group) on the graphics page. Sliders also move automatically to reflect the value of the variable tag.

soft PLC

A pure software (virtual) PLC created by software and existing only within the computer memory. Usually provides a software interface for communication (READ and WRITE) operations to take place with the soft PLC. Also known as a `virtual field unit' or `virtual I/O device'.

software protection

uses a hardware key that plugs into the printer port of your computer to protect against license infringement. The hardware key contains the details of your user license. When you run , the point count in your project is checked against the point limit specified in the hardware key.

staleness period

Represents the total number of seconds that will elapse after the last update before extended quality of the tag element is set to “Stale”.

standby Alarms Server

The Server that processes alarms if the primary alarms server is unavailable.

standby Reports Server

The server that processes reports if the primary reports server is unavailable.

standby Trends Server

The server that processes trends if the primary trends server is unavailable.

stop bits

The number of bits that signals the end of a character in asynchronous transmission. The number is usually 1 or 2. Stop bits are required in asynchronous transmissions because the irregular time gaps between transmitted characters makes it impossible for the server or I/O device to determine when the next character should arrive.

substatus value

The underlying details of a QUALITY tag.

Substitution

A Super Genie substitution is comprised of the data type (optional) and association that you use to define an object or group of object’s properties when creating a Super Genie.

Super Genies

Dynamic pages (usually pop-ups), to which you pass information when the page displays at run-time. You can use Super Genies for pop-up type controllers (to control a process, or a single piece of plant floor equipment).

symbol

An object (or group of objects) stored in a library for later retrieval and use. By storing common objects in a library, you reduce the amount of disk space required to store your project, and reduce the amount of memory required by the run-time system.

syslog.dat

Syslog.dat is the primary log file. It contains useful system information, from low-level driver traffic and Kernel messages, to user defined messages. Trace options (except some CTAPI traces) are sent to this file.

T

tag extension

Additional information for a tag that represents data as a collection of elements, and a collection of items in a tag.

task

Includes operations such as I/O processing, alarm processing, display management, and Cicode execution. Any individual `instance' of Cicode is also a `task'.

template

A base drawing or time-saving pattern used to shape a graphics page. Each template contains base information for the page, such as borders and common control buttons. provides templates for all common page types.

text box

When text is added to a graphics page, it is placed in a text box. A text box has a number of handles, which can be used to manipulate the text object.

thread

Used to manage simultaneous execution of tasks in multitasking operating systems, enabling the operating system to determine priorities and schedule CPU access.

timeout

The period of time during which a task must be completed. If the timeout period is reached before a task completes, the task is terminated.

time-stamped alarms

An alarm triggered by a state change in a digital variable. Time-stamped alarms have an associated register in the I/O device to record the exact time when the alarm changes to active. Use time-stamped alarms when you need to know the exact order in which alarms occur.

time-stamped analog alarms

Time stamped analog alarms work in the same way as analog alarms except that they are time stamped (with the Alarm On and Alarm Off times) using millisecond precision from the time kept by the field device (i.e. the RTU or PLC). The configuration details for time stamped analog alarms are exactly the same as for analog alarms.

time-stamped digital alarms

Time stamped digital alarms work in the same way as digital alarms except that they are time stamped (with the Alarm On and Alarm Off times) using millisecond precision from the time kept by the field device (i.e. the RTU or PLC). The configuration details for time stamped digital alarms are exactly the same as for digital alarms.

tool tip

A help message that displays in a pop-up window when an operator holds the mouse stationary over an object.

touch (object at runtime)

An object is considered touched if an operator clicks it.

Touch command

Can be assigned to objects on graphics pages. Touch commands allow you to send commands to the runtime system by clicking an object.

tracelog.dat

The tracelog.dat file contains managed code logging, mainly in relation to data subscriptions and updates. Note that field traces and requests to native drivers go to the syslog.dat or a specific driver log file.

trend

A graphical representation of the changing values of a plant-floor variable (or expression), or a number of variables. .

trend line

The actual line on a trend that represents the changing values of a plant-floor variable (or expression). .

trend plot

Consists of a trend (or a number of trends), a title, a comment, scales, times and so on.

Trends Server

Controls the accumulation and logging of trend information. This information provides a current and historical view of the plant, and can be processed for display on a graphics page or printed in a report.

U

UAC

User Account Control. Security technology introduced in Windows Vista to enable users to run with standard user rights more easily. .

unqualified tag reference

Reference to tag data by using only the tag name.

unsigned integer variable (I/O device)

A 2-byte (16 bit) data type, representing an integer range from 0 to 65,535. This is supported for all I/O devices that can use INT types. This means you can define any integer variable as an unsigned integer to increase the positive range.

Users

A person or group of persons that require access to the runtime system

V

Valid element

The last field data which had “Good” quality in a tag extension.

Value (V)

The value of the extension of a tag.

ValueTimestamp (VT)

The timestamp of when the value last changed on a tag extension

variable type (Cicode)

The type of the variable (INT (32 bits), REAL (32 bits), STRING (256 bytes), OBJECT (32 bits)).

view-only client

A computer configured with manager-only access to the runtime system. No control of the system is possible, but full access to data monitoring is permitted.

virtual

Behavioral identification rather than a physical one. For example, Windows 95 is a virtual desktop.

W

wizard

A facility that simplifies an otherwise complex procedure by presenting the procedure as a series of simple steps.

Index