WorkstationST*/CIMPLICITY*
Advanced Viewer Integration
Instruction Guide

These instructions do not purport to cover all details or variations in equipment, nor to provide for every possible contingency to be met during installation, operation, and maintenance. The information is supplied for informational purposes only, and GE makes no warranty as to the accuracy of the information included herein. Changes, modifications, and/or improvements to equipment and specifications are made periodically and these changes may or may not be reflected herein. It is understood that GE may make changes, modifications, or improvements to the equipment referenced herein or to the document itself at any time. This document is intended for trained personnel familiar with the GE products referenced herein.

Public Information – This document contains non-sensitive information approved for public disclosure.

GE may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not provide any license whatsoever to any of these patents.

GE provides the following document and the information included therein as is and without warranty of any kind, expressed or implied, including but not limited to any implied statutory warranty of merchantability or fitness for particular purpose.

For further assistance or technical information, contact the nearest GE Sales or Service Office, or an authorized GE Sales Representative.

Revised: Aug 2019  
Issued: Oct 2008  


* Indicates a trademark of General Electric Company and/or its subsidiaries. All other trademarks are the property of their respective owners.

We would appreciate your feedback about our documentation. Please send comments or suggestions to controls.doc@ge.com
## Document Updates

<table>
<thead>
<tr>
<th>Revision</th>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td><strong>Embed a Trend</strong></td>
<td>Updates to include Trender ActiveX controls methods</td>
</tr>
<tr>
<td></td>
<td><strong>Set Trender Time Axis Duration using a Script</strong></td>
<td>Added a Note directing users to apply a TrenderActiveX controls method instead of a script to set the time axis duration</td>
</tr>
<tr>
<td></td>
<td><strong>Trender ActiveX Control Methods</strong></td>
<td>New section</td>
</tr>
<tr>
<td></td>
<td><strong>Expand the Traces List on Mouse Hover</strong></td>
<td>New section</td>
</tr>
<tr>
<td></td>
<td><strong>ActiveX Alarm Viewer</strong></td>
<td>New section (copied from the WorkstationST Alarm Viewer Instruction Guide, GEI-100620)</td>
</tr>
<tr>
<td></td>
<td><strong>Alarm Symbol ActiveX Control</strong></td>
<td>Updates to include Alarm Symbol ActiveX control methods</td>
</tr>
<tr>
<td>M</td>
<td><strong>CimEdit Browse for Variables</strong></td>
<td>Updated the figure to show Time Axis Duration in seconds</td>
</tr>
<tr>
<td></td>
<td><strong>Example Script to Change Time Axis Duration</strong></td>
<td>Added this section to provide the procedure to add buttons to set the Time Axis duration for a trend object in CimEdit with an example script</td>
</tr>
</tbody>
</table>

## Acronyms and Abbreviations

- **AE** | Alarm and Event
- **DA** | Data Access
- **EGD** | Ethernet Global Data
- **HMI** | Human-machine Interface
- **SCADA** | Supervisory Control and Data Acquisition
Safety Symbol Legend

**Warning**

Indicates a procedure or condition that, if not strictly observed, could result in personal injury or death.

**Caution**

Indicates a procedure or condition that, if not strictly observed, could result in damage to or destruction of equipment.

**Attention**

Indicates a procedure or condition that should be strictly followed to improve these applications.
Contents

1 Introduction .......................................................................................................................... 5

2 Configuration ....................................................................................................................... 7
  2.1 ToolboxST Configuration ...................................................................................................... 7
  2.2 Graphics Without a Project Configuration ........................................................................... 8
  2.3 CIMPLICITY Global Script Configuration ......................................................................... 14

3 CimEdit................................................................................................................................ 21
  3.1 Enable Non-translated Content .......................................................................................... 21
  3.2 Browse for Variables .......................................................................................................... 23
  3.3 Update Rates ....................................................................................................................... 32
  3.4 Embed a Trend .................................................................................................................... 33
  3.5 Trender ActiveX® Control Methods ..................................................................................... 36
  3.6 Expand the Traces List on Mouse Hover ............................................................................ 37
  3.7 Add to Trend ....................................................................................................................... 38
  3.8 Go To Definition In Logic .................................................................................................. 40
  3.9 Go To Display Screen ......................................................................................................... 41
  3.10 Display Variable Attributes .............................................................................................. 42
  3.11 Alarm Status History ........................................................................................................ 43
  3.12 Alarm Status Information ................................................................................................. 44
  3.13 ActiveX Alarm Viewer ....................................................................................................... 56
  3.14 Alarm Symbol ActiveX Control ......................................................................................... 58
  3.15 Print Screen ...................................................................................................................... 62
  3.16 Other Right-click Menus ................................................................................................... 63
  3.17 Drag-and-drop to CimEdit ................................................................................................. 64

4 CimView ................................................................................................................................ 71
  4.1 Start and Stop CimView ...................................................................................................... 72
  4.2 Display Variable Attributes .............................................................................................. 73
  4.3 Enable a Second Language ............................................................................................... 74
  4.4 Select a Measurement System ......................................................................................... 79
  4.5 Alarm Symbol Actions ..................................................................................................... 80

5 Privileges ............................................................................................................................. 81
  5.1 Display Current User ......................................................................................................... 83
  5.2 Log On ............................................................................................................................... 84
  5.3 Log Off .............................................................................................................................. 86
  5.4 Access Roles ..................................................................................................................... 87

6 HMI Tagout .......................................................................................................................... 89
  6.1 System Data Flow .............................................................................................................. 89
  6.2 Operation ............................................................................................................................ 90
  6.3 Initial Display ..................................................................................................................... 90
  6.4 Log On ................................................................................................................................ 93

7 Glossary of Terms ............................................................................................................... 95
1 Introduction

Beginning with the combination of the ControlST* V03.03 and CIMPLICITY* 7.5 HMI/SCADA now supports two approaches for configuration and runtime data flow. The first is the conventional method, where each WorkstationST* HMI includes a CIMPLICITY HMI server and viewer. The WorkstationST Service maintains synchronization of the variable and alarm data between the Mark* Vle components and the CIMPLICITY HMI/SCADA project. Runtime data is used to animate the CimView screens and provide alarm status flows through the CIMPLICITY project point server and alarm manager.

Note Later versions of the ControlST software suite and CIMPLICITY HMI/SCADA continue to support the configuration described in this document. Refer to the ToolboxST* User Guide for Mark Controls Platform (GEH-6700), the section Computer Recommendations for version compatibility.

WorkstationST HMI Project Data Flow

The new method uses each WorkstationST HMI to communicate directly with CimEdit and CimView through OPC, eliminating the need for a CIMPLICITY server. Alarms are monitored using the WorkstationST Alarm Viewer application.
The 7.5 Advanced Viewer product treats OPC DA Servers as pseudo projects. This makes screens developed with prior versions of the CIMPLICITY application more compatible. However, since the 7.5 Advanced Viewer does not operate with a project, the following limitations apply:

- The ability to use the project to connect to alternate data sources through CIMPLICITY devcoms, such as a SRTP connection to a PLC, is not available.
- The ability to use the Basic Script engine that ran as part of the project is not available.
- Other project features are provided in the ControlST software suite. This includes the alarm subsystem and the privileges feature.
- CimEdit allows expressions to drive screen animation. The alarm functions interface with a CIMPLICITY project’s alarm manager and therefore the alarm functions (displayed in the following table) are not supported with CIMPLICITY Advanced Viewer. Using them in an expression results in bad quality data for the expression. Some of these functions are available from analog alarm variables from the ToolboxST configuration.

**Note** Beginning with ControlST V04.05, additional alarm sub-variables (attributes) are available. Refer to the table *Variable Attributes*.

<table>
<thead>
<tr>
<th>CIMPLICITY Expression</th>
<th>Description</th>
<th>ControlST equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>TRUE if the variable is in any Alarm or Warning state</td>
<td>Supported by using analog alarm attributes (.H, .L, .HH, .LL, .HHH, .LLL and so on).</td>
</tr>
<tr>
<td>A1</td>
<td>TRUE if the variable is in a Warning High or Warning Low state</td>
<td>Supported by using analog alarm attributes (.H, .L, .HH, .LL, .HHH, .LLL and so on).</td>
</tr>
<tr>
<td>A2</td>
<td>TRUE if the variable is in an Alarm High or Alarm Low state</td>
<td>Supported by using analog alarm attributes (.H, .L, .HH, .LL, .HHH, .LLL and so on).</td>
</tr>
<tr>
<td>AH1</td>
<td>TRUE if the variable is in a Warning High state</td>
<td>Supported by using analog alarm attributes (.H, .L, .HH, .LL, .HHH, .LLL and so on).</td>
</tr>
<tr>
<td>AH2</td>
<td>TRUE if the variable is in an Alarm High state</td>
<td>Supported by using analog alarm attributes (.H, .L, .HH, .LL, .HHH, .LLL and so on).</td>
</tr>
<tr>
<td>AL1</td>
<td>TRUE if the variable is in a Warning Low state</td>
<td>Supported by using analog alarm attributes (.H, .L, .HH, .LL, .HHH, .LLL and so on).</td>
</tr>
<tr>
<td>AL2</td>
<td>TRUE if the variable is in an Alarm Low state</td>
<td>Supported by using analog alarm attributes (.H, .L, .HH, .LL, .HHH, .LLL and so on).</td>
</tr>
<tr>
<td>ANA</td>
<td>TRUE if the variable is in alarm and the alarm has not been acknowledged</td>
<td>Not supported</td>
</tr>
<tr>
<td>NACK</td>
<td>TRUE if the alarm has not been acknowledged, whether or not the variable is in an alarm state</td>
<td>Not supported</td>
</tr>
</tbody>
</table>
2 Configuration

Much of the integration between CIMPLICITY HMI/SCADA and the ToolboxST application is accomplished by configuration of features within the ToolboxST application. This is done by enabling the WorkstationST HMI feature and configuration of the HMI Graphics without a Project item in the HMI Config tab.

2.1 ToolboxST Configuration

The WorkstationST component of the ToolboxST application allows the user to tailor the configuration of a WorkstationST computer. When the 7.5 Advanced Viewer is installed, and the HMI feature of the WorkstationST component is enabled, the HMI adapts the 7.5 Advanced Viewer to integrate the ControlST variable browser, the CimEdit and CimView right-click menu items, and adds the embedded Trender feature. The WorkstationST HMI feature is enabled on the General tab of the Component Editor.

➢ To enable the WorkstationST HMI feature: from the System Editor select a WorkstationST component.

From the General tab, select Features.

Set the HMI Feature to True.

The HMI Config tab displays.
2.2 Graphics Without a Project Configuration

To tailor the selected WorkstationST computer to interact with the 7.5 Advanced Viewer, you must first enable the WorkstationST Graphics without a Project item.

➢ To enable the HMI Graphics without a Project item: from the ToolboxST Component Editor, select the HMI Config tab, then select the HMI item.

From the Property Editor, select HMI Graphics without a Project and set to True.

The HMI Graphics without a Project item displays in the Tree View.
To configure the HMI Graphics without a Project item: from the HMI Config tab, select the HMI Graphics without a Project item. The HMI Graphics without a Project configuration options display in the Property Editor.
The HMI Graphics without a Project configuration options are provided in the following tables.

### CimEdit and CimView Global Configuration Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional PTOPC Elements</td>
<td>Add comma separated pairs of element names and values. These elements will be added into ptopc_config.xml under the &lt;config&gt; element as additional elements with the specified names and values.</td>
</tr>
<tr>
<td>CimEdit Global Script Entry Point</td>
<td>The CIMPLICITY editor before context script entry point can be assigned to a global script method which will be executed prior to displaying the right-click menu. The default script method for a WorkstationST system is PreEditorContextMenuSetup. This setting can be changed to an alternate method to tailor the context menu for the CIMPLICITY editor.</td>
</tr>
<tr>
<td>CimView Global Script Entry Point</td>
<td>The CIMPLICITY viewer before context script entry point can be assigned to a global script method which will be executed prior to displaying the right-click menu. The default script method for a WorkstationST system is PreContextMenuSetup. This setting can be changed to an alternate method to tailor the context menu for the CIMPLICITY viewer.</td>
</tr>
<tr>
<td>Default Project</td>
<td>CIMPLICITY CimView has an OPC client that can be configured to obtain data from one or more OPC Servers. Each OPC Server connection is handled as if CimView were connecting to a CIMPLICITY project. Each OPC DA Server connection is treated like a CIMPLICITY project. Variables used on CimView screens can be preceded by an optional project name (such as \MyProject!). Variables not preceded by an optional project uses this default project. If left blank, the GEWORKSTATIONST project name will be used. Additional OPC DA Server projects can be defined in the detail view.</td>
</tr>
<tr>
<td>Default Screen Update Rate</td>
<td>The default update rate, in milliseconds, for live value updates to screens. If individual screen objects need an update rate different from this default, the variable name can be preceded by an update rate. (For example, &lt;50&gt;G1.TNH requests G1.TNH updates at 50 milliseconds, while G1.TNH without the &lt;50&gt; updates the screen object at the default update rate). The OPC DA Server can also limit the update rate with the Maximum Client Rate configured on the OPC DA Server tab. The OPC DA Server’s update rate is effectively the fastest usable rate.</td>
</tr>
<tr>
<td>Global Script File Path List</td>
<td>CIMPLICITY Global Scripts contain entry points available to all other scripts. Enter a comma separated list of script files to be used as global scripts. If a folder path does not precede the script file name, the file is assumed to reside in the working folder where CimView starts. Normally this working folder is the location where the screens reside. The WorkstationST global script is installed by the WorkstationST application and is included in this list by default.</td>
</tr>
</tbody>
</table>

### CimEdit Object Explore Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CimEdit Object Explorer Drag Linked</td>
<td>When set to True, the screen object from the CIMPLICITY CimEdit object explorer is dragged as a linked object, rather than a copied object.</td>
</tr>
<tr>
<td>CimEdit Object Explorer Root Folder</td>
<td>The HMI graphics editor has an object explorer to add library objects to a screen. The explorer displays all objects in CIMPLICITY screen files in sub folders under this root folder setting.</td>
</tr>
</tbody>
</table>

### CimView Compatibility Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Alt key with mouse for zooming</td>
<td>The CIMPLICITY global configuration for using the Alt key with mouse for zooming.</td>
</tr>
</tbody>
</table>
### CimView Navigation Toolbar Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation Bar File Path</td>
<td>CIMPLICITY HMI/SCADA has a built-in navigation toolbar and toolbar editor. This provides the path to a navigation configuration file created from the CIMPLICITY Navigation Bar editor. If blank, the navigation bar will not be displayed.</td>
</tr>
<tr>
<td>Navigation Bar Show</td>
<td>When set to True, the Navigation bar is displayed.</td>
</tr>
<tr>
<td>Navigation Bar Show Child Bar</td>
<td>When set to True, the Navigation bar contains a child bar that is displayed.</td>
</tr>
</tbody>
</table>

**Note** Projects associated with OPC DA Servers can be added.

➢ To add projects associated with an OPC DA Server: from the HMI Config tab, in the Tree View, select the HMI Graphics without a Project item and enter the Project Name in the Summary View.

![Image of GEWORKSTATIONST project](image)

The **GEWORKSTATIONST** project is added by default.

Click the space below and enter the name of a new project.

**Note** Live values can be obtained from additional OPC DA Servers. However, the right-click menu items and the embedded Trender will only have access to the local WorkstationST OPC DA Server.
Configuration options display in the Property Editor.
The new project configuration options are provided in the following tables.

### DCOM Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCOM Threshold</td>
<td>The amount of time, in milliseconds, the CimView OPC client waits to disconnect from the OPC Server after detecting the server has not responded to a ping. The default is 20,000 milliseconds.</td>
</tr>
<tr>
<td>DCOM Timeout</td>
<td>The amount of time, in milliseconds, the CimView OPC client waits for a response from the OPC Server before a communication error is declared. The default is 10,000 milliseconds.</td>
</tr>
<tr>
<td>Hang Protection Timeout</td>
<td>If the CimView OPC client does not have a valid health status within this period, the connection is declared dead. The default is 120,000 milliseconds.</td>
</tr>
<tr>
<td>Ping Interval</td>
<td>The frequency, in milliseconds, at which the CimView OPC client determines the health of the server. The default is 15,000 milliseconds.</td>
</tr>
<tr>
<td>Reconnect Interval</td>
<td>The amount of time, in milliseconds, the CimView OPC client waits, after a forced disconnect, before attempting to reconnect to the server. The default is 15,000 milliseconds.</td>
</tr>
</tbody>
</table>

### General Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Address</td>
<td>The host name or IP address where the OPC DA Server is running.</td>
</tr>
<tr>
<td>OPC DA Server Program ID</td>
<td>The program ID of the OPC DA Server.</td>
</tr>
<tr>
<td>Project Name</td>
<td>The Project Name assigned to the OPC DA Server connection. CimView has an OPC client that can be configured to obtain data from one or more OPC Servers. Each OPC DA Server connection is treated like a CIMPLICITY project.</td>
</tr>
</tbody>
</table>

### Property IDs Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Format Property ID</td>
<td>OPC Servers can provide custom properties for variables. Custom properties have IDs of 5000 and above. If the OPC Server provides a custom property for a format string specifier, this ID allows the CimView OPC client to obtain the value for a given tag, and use the string to format the live value. The default is 5003, the ID used by the GeCssOpcServer.</td>
</tr>
</tbody>
</table>

### Strings Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Array String Length</td>
<td>The maximum length, in characters, of an array of strings for the OPC client connection. The default is 80 characters.</td>
</tr>
<tr>
<td>Maximum String Length</td>
<td>The maximum length, in characters, of a string to be read by the OPC client connection. The default is 80 characters.</td>
</tr>
</tbody>
</table>

Only the GeCssOpcServer OPC DA Server provides an expanded namespace allowing variable attributes to be referenced in CIMPLICITY screen object expressions. (For example, a screen object can refer to \texttt{G1.Var1.Description}.) A list of the available variable attributes is displayed in the CimEdit variable browser.
2.3 CIMPLICITY Global Script Configuration

The 7.5 Advanced Viewer contains CimEdit and CimView global configuration settings driven by the WorkstationST component HMI configuration feature. These settings display using the CimEdit menu for Global Configuration.

**Note** Edits made from CimEdit will be overwritten by the configuration in the HMI tab of the WorkstationST application.

➢➢➢

To display CimEdit and CimView global configuration settings: from the CimEdit **Edit** menu, select **Global Configuration**.

![Global Configuration](image)

The Global Scripts displayed match the list of scripts configured in the Global Script File Path List configured on the WorkstationST component HMI tab, HMI Graphics without a Project item.

**Note** The relative path, *geds.bcl*, is expected to reside in the folder where screens reside. *WorkstationST.cms* provides the methods to display and operate the CimEdit and CimView right-click menus.
2.3.1 Display Global Script Method

The WorkstationST global script contains two methods for displaying the ControlST Trend and the WorkstationST Alarm Viewer using a unique name. The unique name can be used to allow CIMPLICITY to open the same Trend or Alarm Viewer with each repeated call to the global script method.

The global script entry point OnNamedAlarmViewerActivate viewFileList, uniqueAlarmViewerInstanceName can be used to display the Alarm Viewer. The viewFileList is a comma-separated list of view files. If the list contains more than one comma-separated value, a dialog box displays allowing the user to pick from the list. The uniqueAlarmViewerInstanceName is used to display the same Alarm Viewer instance each time. For example, the user could have one named Historical and one named Live. Calling this global script method using the same unique name results in only one instance of the Alarm Viewer. Subsequent calls display the Alarm Viewer as the front window.

**Note** Restarting CimView causes a new instance of the Alarm Viewer to be created.

The global script entry point OnAddToNamedTrend commaSeparatedVariables, uniqueTrenderInstanceName can be used to display the Trend. The variables in the comma-separated list will be displayed on the trend. As with the unique Alarm Viewer instance name, the unique Trend name allows the same Trend instance to be activated.

2.3.2 VariablesConfigured Global Script Method

The VariablesConfigured method is provided in the WorkstationST global script. The global script is available for any script used on any screen or in any screen object in CimEdit. CIMPLICITY HMI/SCADA provides this script feature and allows scripts to be configured to run at various events, such as upon mouse up or upon screen open. The VariablesConfigured script entry point is provided in the WorkstationST.cms global script. This method is useful for quickly determining if a variable has been configured in the ControlST system.

**Example**

```vba
Dim CimObj As Gefobject
Dim ctrlrvar As String
Dim devicevar As String
Dim BQstr As String
Dim DHstr As String
```
Set CimObj = CimGetObject

ctrlrvar = CimObj.GetVariable("ctrlr").value
devicevar = CimObj.GetVariable("device").value

BQstr = ".BQ"

DHstr = ".DH"

Dim varNames(1) As String

Dim ctrlDev As String

Dim configured(1) As Boolean

ctrlDev = ctrlrvar & devicevar

varNames(0) = ctrlDev & BQstr

varNames(1) = ctrlDev & DHstr

VariablesConfigured varNames, configured

Check For BQ - Substitute with 2 if not present

If configured(0) Then
    CimObj.Variables("bq").value = varNames(0)
Else
    CimObj.Variables("bq").value = ".1"
End If

'Check For DH - Substitute with 2 if not present

If configured(1) Then
    CimObj.Variables("dh").value = varNames(1)
Else
    CimObj.Variables("dh").value = ".1"
End If

The WorkstationST.cms default global script has a method called InitializeCimplicity that should be called in a central Screen Open event. InitializeCimplicity could be called from a linked object that is common to all screens. Applications typically contain a tp_Screen file where a navigation object resides. This navigation object is usually linked to all screens. The InitializeCimplicity method configures the Secondary Language selection for WorkstationST components configured to use the second language by default and starts the Logon Security Client. The InitializeCimplicity method performs the initialization and sets a flag, so the initialization will only be done the first time CimView runs, and not with every screen open.
2.3.3 Global Script for Language Selection

The privileges feature allows the changing of regional settings such as language selection from the privileges manager running in the WorkstationST Status Monitor tray icon application or the Logon privileges tray icon application for non-WorkstationST computers. When a language selection is changed, if CimView translation files are used, as described in the section Second Language Use by CimView, CimView must have the application language selection changed. This can be accomplished using the SetCimplicitySecondLanguage global script method. The following illustrates how to configure an expression calculation in a central-linked screen object, such as a navigation object.

➢ To configure an expression calculation in a central-linked screen object: from the CIMPLICITY CimView screen, right-click on the screen and select Properties from the shortcut menu.
From the Actions tab, the Actions for the procedure can be edited.

The Screen Open event can be edited in a similar manner.
➢ To configure an OnScreenOpen Event: from CimView, right-click the CimView screen to display the Properties – Screen dialog box.

2.3.4 Global Script to Browse for Variable

BrowseForVariable is a global script function which can be used to display the ToolboxST Variable browser and return the variable name selected by the user. If the user cancels from the browse dialog, an empty string is returned.

In the following example, a mouseup event (OnMouseUp) calls this script which opens the Variable browser and inserts the result into the string (str). A message box then displays the resulting str.

Example

```vbscript
Sub OnMouseUp(x As Long, y As Long, flags As Long)
    Dim str As String
    str = BrowseForVariable
    MsgBox str
End Sub
```
2.3.5 Global Script to Browse for a Plant Area

BrowsePlantAreas is a global script function which can be used to display a plant area browser dialog and return the plant area selected by the user. If the user cancels from the browse dialog, an empty string is returned.

In the following example, a mouseup event (OnMouseUp) calls this script which opens the Plant Area browser and inserts the result into the string (str). A message box then displays the resulting str.

Example

Sub OnMouseUp(x As Long, y As Long, flags As Long)
    Dim str As String
    str = BrowsePlantAreas
    MsgBox str
End Sub

2.3.6 Global Script to Browse for a Device Name

BrowseDevices is a global script function which can be used to display a device name browser dialog and return the device name selected by the user. If the user cancels from the browse dialog, an empty string is returned.

In the following example, a mouseup event (OnMouseUp) calls this script which opens the Device name browser and inserts the result into the string (str). A message box then displays the resulting str.

Example

Sub OnMouseUp(x As Long, y As Long, flags As Long)
    Dim str As String
    str = BrowseDevices
    MsgBox str
End Sub
3 CimEdit

CimEdit is used to edit graphic screen documents. When the Graphics without a Project setting has been enabled in the WorkstationST component HMI Config tab, and downloaded to the target WorkstationST computer, the WorkstationST application tailors CimEdit and CimView to work with the ToolboxST configuration. Right-click menus are enabled, a common variable browser is made available, and some global script methods for determining variable configuration are available. Some right-click menu items from CimEdit and CimView, and point browsing for a remote OPC project, are not available.

3.1 Enable Non-translated Content

**Note** The CimView right-click menu is similar in appearance and identical in behavior to the CimEdit menu described in this section.

When you right-click an object on a CimEdit screen the following right-click shortcut menu displays.

```
Cut               Ctrl+X
Copy              Ctrl+C
Paste             Ctrl+V
Menu
Full Screen

Point View
Expression View

Help
Properties      Alt+Enter

Enable Non-Translated Content
Go To Definition In Logic
Add to Trend
Drag Variables To Screen
Go To Display Screen
Alarm Status History
Display Variable Attributes
Print Screen
```

The *Enable Non-translated Content* menu item is available only when the CimEdit or CimView features are running in an operating system with a language pack installed. When the CimEdit or CimView features are first started, the *Enable Non-translated Content* option is not checked, as displayed in the previous figure. When the *Enable Non-translated Content* option is initially enabled (checked), the following dialog box is displayed.
Click **OK** to enable this option, as well as other features, such as the *Go To Definition in Logic* feature. The *Go To Definition in Logic* menu item enables non-translated content.

<table>
<thead>
<tr>
<th>✔ Enable Non-Translated Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go To Definition In Logic</td>
</tr>
<tr>
<td>Add to Trend</td>
</tr>
<tr>
<td>Go To Display Screen</td>
</tr>
<tr>
<td>Display Variable Attributes</td>
</tr>
<tr>
<td>Alarm Status History</td>
</tr>
<tr>
<td>Print Screen</td>
</tr>
</tbody>
</table>

**Note** The Go To Definition In Logic menu item is enabled; however, there are other conditions which can disable this menu item, such as if the ToolboxST application is not installed, or the logged on privileges user does not have the Go To Definition privilege configured.
3.2 Browse for Variables

Within CimEdit, when a variable browser is required, the ControlST variable browser is displayed. The browser allows browsing and searching of all variables in the ControlST system consumed by the HMI.

Example

➢ To configure the Expression for the Display value of an Object: from CimEdit, right-click on any screen object and select Properties from the shortcut menu.

Note Screen objects can be simple text, buttons, or more complex groups of other objects such as a Pump.
The drop-down menu allows browsing by variable name or browsing by alias name.

The upper grid displays variables.

The lower grid allows selection of attributes for the selected upper grid variables.
**Note** Variables can be configured with an alias property (alias name) in the ToolboxST application. The HMI screens can then use the alias name to refer to the variable. When the option *Display Variable Aliases instead of Variable Names* is selected from the WorkstationST User Preferences menu, user-assigned alias names are displayed for the DCS blocks in the ToolboxST Block Editor. For more information, refer to the *ToolboxST User Guide for Mark Controls Platform* (GEH-6700), the section *Variable Names and Aliases*.

### Variable Attributes

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlarmAckCmd&lt;sup&gt;1&lt;/sup&gt;</td>
<td>If the variable is an alarm, write this attribute to acknowledge the alarm.</td>
</tr>
<tr>
<td>AlarmAckNeeded&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True if the variable is an alarm and the alarm needs to be acknowledged.</td>
</tr>
<tr>
<td>AlarmActive&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True if the variable is an alarm and the alarm is active.</td>
</tr>
<tr>
<td>AlarmConfigured&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True to indicate the variable is configured for an alarm.</td>
</tr>
<tr>
<td>AlarmIsOutOfSvc&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True when an alarm is currently out of service.</td>
</tr>
<tr>
<td>AlarmIsShelved&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True when an alarm is currently shelved.</td>
</tr>
<tr>
<td>AlarmLocked&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True if the variable is an alarm and the alarm is locked.</td>
</tr>
<tr>
<td>AlarmOutOfSvcEnabled&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True if Out Of Service has been enabled for this system on the ToolboxST System Editor.</td>
</tr>
<tr>
<td>AlarmPriority&lt;sup&gt;1&lt;/sup&gt;</td>
<td>Priority for the alarm. Analog alarm priority can changed based on the alarm level.</td>
</tr>
<tr>
<td>AlarmResetCmd&lt;sup&gt;1&lt;/sup&gt;</td>
<td>If the variable is an alarm, write this attribute to reset the alarm.</td>
</tr>
<tr>
<td>AlarmResetNeeded&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True if the variable is an alarm and the alarm can be reset.</td>
</tr>
<tr>
<td>AlarmShelvingEnabled&lt;sup&gt;1&lt;/sup&gt;</td>
<td>True if shelving is enabled for this alarm. Shelving is enabled for a ToolboxST system in the properties in the System Editor and additionally each variable's alarm shelving can be enabled.</td>
</tr>
<tr>
<td>AlarmState&lt;sup&gt;1&lt;/sup&gt;</td>
<td>The alarm state text for an alarm variable.</td>
</tr>
<tr>
<td>AlarmSymbolKey&lt;sup&gt;1&lt;/sup&gt;</td>
<td>A string representing the alarm symbol to be used for this alarm. BQ = Bad quality or alarm client not connected to alarm server. OO = out-of-service. AS = Shelved alarm. &lt;alarmClass&gt;AU = active unacknowledged for specified class. &lt;alarmClass&gt;AUB = active unacknowledged for specified class (class configured to blink). &lt;alarmClass&gt;AA = active acknowledged for specified class. &lt;alarmClass&gt;NA = returned to normal and acknowledged for specified class. &lt;alarmClass&gt;NU = returned to normal and unacknowledged for specified class.</td>
</tr>
<tr>
<td>AlarmText&lt;sup&gt;1&lt;/sup&gt;</td>
<td>Alarm text for an alarm variable</td>
</tr>
<tr>
<td>AlarmTimeStamp&lt;sup&gt;1,2&lt;/sup&gt;</td>
<td>Device time stamp for an alarm variable</td>
</tr>
<tr>
<td>Alias&lt;sup&gt;1&lt;/sup&gt;</td>
<td>Customer assigned alias name for the variable</td>
</tr>
<tr>
<td>$Raw_Value</td>
<td>Rw value of the variable. Useful when enumerated values are used</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the variable</td>
</tr>
<tr>
<td>Device_ID</td>
<td>Device name that owns the variable</td>
</tr>
<tr>
<td>DisplayFormat</td>
<td>Display format notation (such as %f.3 for a float with a precision of 3)</td>
</tr>
<tr>
<td>Display_Lim_High</td>
<td>High display limit of the variable data type string</td>
</tr>
<tr>
<td>Display_Lim_High_N</td>
<td>High display limit of the variable data type numeric</td>
</tr>
<tr>
<td>Display_Lim_Low</td>
<td>Low display limit of the variable data type string</td>
</tr>
<tr>
<td>Display_Lim_Low_N</td>
<td>Low display limit of the variable data type numeric</td>
</tr>
</tbody>
</table>
### Variable Attributes (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EU_Label</td>
<td>Engineering units of the variable</td>
</tr>
<tr>
<td>Forced</td>
<td>When True, indicates the variable's value has been forced. This only applies to Mark Vle controllers.</td>
</tr>
</tbody>
</table>

**Warning**

Logic forcing procedures can result in personal injury or death, if not strictly followed. Only adequately trained personnel should modify any programmable machine. Forcing of control logic for an operating process is strongly discouraged.

Forcing of protective functions is never permissible for an operating unit. All safety measures should be strictly enforced in conjunction with this procedure.

---

1. To enable these attributes, the property *Enable Process Alarms* on the OPC DA Server tab must be set to True.

2. In CIMPPLICITY 8.2 the AlarmTimeStamp variable is a long integer (64 bits). It can be displayed on a CIMPPLICITY screen using a Time (Absolute.Sub Seconds) display format. Refer to the following figure.

---

**Screen Object Properties**

---

![Screen Object Properties](image)
## Variable Attributes (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format_Prec</td>
<td>Display precision of a variable. Returns 0 for variables with blank precision.</td>
</tr>
<tr>
<td>FR_ID</td>
<td>HMI resource of the variable</td>
</tr>
<tr>
<td>ItemEUIInfo</td>
<td>Enumeration values</td>
</tr>
<tr>
<td>ItemEUType</td>
<td>Enumeration type (0 = none, 1 = analog, 2 = enumerate)</td>
</tr>
<tr>
<td>Measurement_Unit_ID</td>
<td>Name of the format specification of the variable</td>
</tr>
<tr>
<td>Point_ID</td>
<td>Fully qualified variable name if not in Alias mode, or alias of the variable if in Alias mode (<em>Display Variable Aliases instead of Variable Names</em> selected from WorkstationST User Preferences menu)</td>
</tr>
<tr>
<td>Variable_ID</td>
<td>When not in Alias mode, the value shows the variable name (may not be fully qualified variable name). If the <em>Enable Alias Prefix</em> setting in the ToolboxST Component Editor General tab is set to False, the device prefix name is not displayed in CIMPLICITY screens. If the Alias has not been configured, the value displays the variable name with the prefix applied unless <em>Enable Alias Prefix</em> is set to False.</td>
</tr>
<tr>
<td>AliasConfigured</td>
<td>Indicates if a variable alias is configured (True if Alias is configured, or False if Alias field is empty)</td>
</tr>
<tr>
<td>Quality.Disable_Write</td>
<td>When set to True, the variable is read only, or the user does not have privilege to write the value</td>
</tr>
<tr>
<td>Quality.Is_Available</td>
<td>Variable is configured in the system and the live value is healthy</td>
</tr>
<tr>
<td>Quality.Stale_Data</td>
<td>Variable is configured and the value is not healthy</td>
</tr>
<tr>
<td>Setpoint_High</td>
<td>Entry High limit of the variable data type string</td>
</tr>
<tr>
<td>Setpoint_High_N</td>
<td>Entry High limit of the variable data type numeric</td>
</tr>
<tr>
<td>Setpoint_Low</td>
<td>Entry Low limit of the variable data type string</td>
</tr>
<tr>
<td>Setpoint_Low_N</td>
<td>Entry Low limit of the variable data type numeric</td>
</tr>
</tbody>
</table>

### WorkstationST Intrinsic Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Project</td>
<td>Name of the control system</td>
</tr>
<tr>
<td>$Role</td>
<td>User role</td>
</tr>
<tr>
<td>$User</td>
<td>User ID</td>
</tr>
<tr>
<td>Customer_Name</td>
<td>Customer name</td>
</tr>
<tr>
<td>MeasurementSystem</td>
<td>Type of Active Measurement System set by each WorkstationST live data client to select values only in that measurement system.</td>
</tr>
<tr>
<td>Site_Name</td>
<td>Name of the Site</td>
</tr>
<tr>
<td>UseSecondLanguage</td>
<td>Set by each WorkstationST live data client. If true, varName.Description contains Second Language Descriptions.</td>
</tr>
</tbody>
</table>
The CIMPLICITY 6.1 project provided the following intrinsic $Project variables.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Project.Computer</td>
<td>Name of the computer</td>
</tr>
<tr>
<td>$Project.Date.AMPM</td>
<td>0 = AM, 1 = PM</td>
</tr>
<tr>
<td>$Project.Date.Day</td>
<td>Current day in the month</td>
</tr>
<tr>
<td>$Project.Date.DayOfWeek</td>
<td>Current day in the week (1 = Sunday)</td>
</tr>
<tr>
<td>$Project.Date.DayOfYear</td>
<td>Current day in the year</td>
</tr>
<tr>
<td>$Project.Date.Hour</td>
<td>Current hour in the day (0 - 24)</td>
</tr>
<tr>
<td>$Project.Date.Hour12</td>
<td>Current hour in the day (1 - 12)</td>
</tr>
<tr>
<td>$Project.Date.Minute</td>
<td>Minutes past the hour (0 - 59)</td>
</tr>
<tr>
<td>$Project.Date.Month</td>
<td>Current month of the year (1 = January)</td>
</tr>
<tr>
<td>$Project.Date.Second</td>
<td>Seconds past the minute (0 - 59)</td>
</tr>
<tr>
<td>$Project.Date.SecondOfDay</td>
<td>Seconds past midnight</td>
</tr>
<tr>
<td>$Project.Date.Week</td>
<td>Current week in the year</td>
</tr>
<tr>
<td>$Project.Date.Year</td>
<td>Current year</td>
</tr>
<tr>
<td>$Project.Datetime</td>
<td>Local date and time in seconds from January 1, 1970</td>
</tr>
</tbody>
</table>

The CIMPLICITY 6.1 project provided the following intrinsic $Local variables.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Local.Computer</td>
<td>Name of the computer</td>
</tr>
<tr>
<td>$Local.Date.AMPM</td>
<td>0 = AM, 1 = PM</td>
</tr>
<tr>
<td>$Local.Date.Day</td>
<td>Current day in the month</td>
</tr>
<tr>
<td>$Local.Date.DayOfWeek</td>
<td>Current day in the week (1 = Sunday)</td>
</tr>
<tr>
<td>$Local.Date.DayOfYear</td>
<td>Current day in the year</td>
</tr>
<tr>
<td>$Local.Date.Hour</td>
<td>Current hour in the day (0 - 24)</td>
</tr>
<tr>
<td>$Local.Date.Hour12</td>
<td>Current hour in the day (1 - 12)</td>
</tr>
<tr>
<td>$Local.Date.Minute</td>
<td>Minutes past the hour (0 - 59)</td>
</tr>
<tr>
<td>$Local.Date.Month</td>
<td>Current month of the year (1 = January)</td>
</tr>
<tr>
<td>$Local.Date.Second</td>
<td>Seconds past the minute (0 - 59)</td>
</tr>
<tr>
<td>$Local.Date.SecondOfDay</td>
<td>Seconds past midnight</td>
</tr>
<tr>
<td>$Local.Date.Week</td>
<td>Current week in the year</td>
</tr>
<tr>
<td>$Local.Date.Year</td>
<td>Current year</td>
</tr>
<tr>
<td>$Local.Datetime</td>
<td>Local date and time in seconds from January 1, 1970</td>
</tr>
</tbody>
</table>


28 GEI-100697N

GEI-100697 WorkstationST/CIMPLICITY Advanced Viewer Integration

Public Information
3.2.1 $Alarm OPC DA Variables

The ability to view active alarm totals, un-acknowledged alarm totals, out-of-service alarm totals, and shelved alarm totals is provided in the OPC DA server’s namespace and therefore available for use on CIMPLICITY screens. The totals are provided for all alarms (such as $Alarms.TotalActiveU), by plant area (such as $Alarms.IGCC.Common.TotalActiveU), and by device (such as $Alarms.G1.TotalActiveU). Priority sub variables represent these area alarm totals by priority. Additionally, each plant area owns an IncludeChildren variable that can be written by a client to make the calculations for the alarm totals that include alarms for all child plant areas. For example, in the following figure, if the variable $Alarms.IGCC.IncludeChildren were set to True by a client, then $Alarms.IGCC.TotalActiveU would include alarm counts from IGCC and its children (Common, Train1, and Train2) as well as recursively all the children below Common, Train1, and Train2.

**Note** Using IncludeChildren only impacts a single client connection. Each client’s connection to the server has its own unique setting for IncludeChildren.

Alarms residing in the Alarm Server in use are used to calculate the $Alarms totals. The Alarm Server in use is either the local Alarm Server, or the Alarm Server configured in the WorkstationST Component Editor, General tab, Alarm Server to Use property.

**Note** The Enable Process Alarms property on the WorkstationST OPC DA Server tab must be set to True.

**Note** Device cared about by proxy will not be available to calculate the $Alarms totals as alarms with the by proxy feature are not included.
The following table represents the available alarm variables.

<table>
<thead>
<tr>
<th>All Alarm Totals</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Alarms.SymbolKey</td>
</tr>
<tr>
<td>A string representing the highest sort order alarm symbol for all alarms. BQ = Bad quality or alarm client not connected to alarm server. OO = out-of-service. AS = Shelved alarm. &lt;alarmClass&gt;AU = active unacknowledged for specified class. &lt;alarmClass&gt;AUB = active unacknowledged for specified class (class configured to blink). &lt;alarmClass&gt;AA = active acknowledged for specified class. &lt;alarmClass&gt;NA = returned to normal and acknowledged for specified class. &lt;alarmClass&gt;NU = returned to normal and unacknowledged for specified class.</td>
</tr>
<tr>
<td>$Alarms.TotalActiveA</td>
</tr>
<tr>
<td>Total number of active and acknowledged alarms in the alarm server (alarms not reset).</td>
</tr>
<tr>
<td>$Alarms.TotalActiveU</td>
</tr>
<tr>
<td>Total number of active and unacknowledged alarms in the alarm server.</td>
</tr>
<tr>
<td>$Alarms.TotalNormalA</td>
</tr>
<tr>
<td>Total number of returned to normal, acknowledged alarms in the alarm server (alarms not yet reset).</td>
</tr>
<tr>
<td>$Alarms.TotalNormalU</td>
</tr>
<tr>
<td>Total number of returned to normal, unacknowledged alarms in the alarm server.</td>
</tr>
<tr>
<td>$Alarms.TotalOutOfSvc</td>
</tr>
<tr>
<td>Total number of out of service alarms for the system.</td>
</tr>
<tr>
<td>$Alarms.TotalShelved</td>
</tr>
<tr>
<td>Total number of shelved alarms for the system.</td>
</tr>
<tr>
<td>$Alarms.$# namespace is provided for filtering of alarms by priority (such as $Alarms.$1.TotalOutOfSvc). Each of the sub variables under $Alarms are available under $Alarms.$#.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Alarm Totals by Plant Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Alarms.&lt;plantArea&gt;.IncludeChildren</td>
</tr>
<tr>
<td>A boolean which can be set independently by each live client to determine if child plant area alarms are considered when creating the value for alarm total and symbol key variables.</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.SymbolKey</td>
</tr>
<tr>
<td>A string representing the highest sort order alarm symbol for this plant area (child plant areas are also considered if IncludeChildren variable is true). &lt;alarmClass&gt;AU = active unacknowledged for specified class. &lt;alarmClass&gt;AUB = active unacknowledged for specified class (class configured to blink). &lt;alarmClass&gt;AA = active acknowledged for specified class. &lt;alarmClass&gt;NA = returned to normal and acknowledged for specified class. &lt;alarmClass&gt;NU = returned to normal and unacknowledged for specified class.</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.TotalActiveA</td>
</tr>
<tr>
<td>Total number of active and acknowledged alarms for this plant area (and child plant areas if IncludeChildren variable is true).</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.TotalActiveU</td>
</tr>
<tr>
<td>Total number of active and unacknowledged alarms for this plant area.</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.TotalNormalA</td>
</tr>
<tr>
<td>Total number of returned to normal, acknowledged alarms for this plant area (alarms not yet reset).</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.TotalNormalU</td>
</tr>
<tr>
<td>Total number of returned to normal, unacknowledged alarms for this plant area.</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.TotalOutOfSvc</td>
</tr>
<tr>
<td>Total number of out of service alarms for this plant area (and child plant areas if IncludeChildren variable is true).</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.TotalShelved</td>
</tr>
<tr>
<td>Total number of shelved alarms for this plant area (and child plant areas if IncludeChildren variable is true).</td>
</tr>
<tr>
<td>$Alarms.&lt;plantArea&gt;.$# namespace is provided for filtering of alarms by priority (such as $Alarms.GreyWater.$1.TotalOutOfSvc where GreyWater is a plant area). Each of the sub variables under $Alarms.&lt;plantArea&gt; are available under $Alarms.&lt;plantArea&gt;.$#</td>
</tr>
</tbody>
</table>

1 Some symbols are class dependent.
<table>
<thead>
<tr>
<th>All Alarm Totals</th>
<th>Alarm Totals by Device</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$Alarms.&lt;device&gt;.SymbolKey</code></td>
<td>A string representing the highest sort order alarm symbol for this plant area (child plant areas are also considered if IncludeChildren variable is true). <code>&lt;alarmClass&gt;AU = active unacknowledged for specified class. </code>&lt;alarmClass&gt;AUB = active unacknowledged for specified class (class configured to blink). <code>&lt;alarmClass&gt;AA = active acknowledged for specified class. </code>&lt;alarmClass&gt;NA = returned to normal and acknowledged for specified class. `&lt;alarmClass&gt;NU = returned to normal and unacknowledged for specified class.¹</td>
</tr>
<tr>
<td><code>$Alarms.&lt;device&gt;.TotalActiveA</code></td>
<td>Total number of active and acknowledged for this device (alarms not reset).</td>
</tr>
<tr>
<td><code>$Alarms.&lt;device&gt;.TotalActiveU</code></td>
<td>Total number of active and unacknowledged alarms for this device.</td>
</tr>
<tr>
<td><code>$Alarms.&lt;device&gt;.TotalNormalA</code></td>
<td>Total number of returned to normal, acknowledged alarms for this device (alarms not yet reset).</td>
</tr>
<tr>
<td><code>$Alarms.&lt;device&gt;.TotalNormalU</code></td>
<td>Total number of returned to normal, unacknowledged alarms for this device.</td>
</tr>
<tr>
<td><code>$Alarms.&lt;device&gt;.TotalOutOfSvc</code></td>
<td>Total number of out of service alarms for this device.</td>
</tr>
<tr>
<td><code>$Alarms.&lt;device&gt;.TotalShelved</code></td>
<td>Total number of shelved alarms for this device.</td>
</tr>
<tr>
<td><code>$Alarms.&lt;device&gt;.$#</code> namespace is provided for filtering of alarms by priority (such as <code>$Alarms.G1.$1.TotalOutOfSvc</code> where G1 is a controller device name). Each of the sub variables under <code>$Alarms.&lt;device&gt;</code> are available under <code>$Alarms.&lt;device&gt;.$#</code>.</td>
<td></td>
</tr>
</tbody>
</table>

¹ The sorting order for alarm symbols can be configured using the ToolboxST System Information Editor. Refer to the following figure.

![Alarm Symbol Sorting](image-url)
### 3.3 Update Rates

The Default Screen Update Rate is set in the WorkstationST component HMI Tab, HMI Graphics without a Project item. All variable values displayed on a screen are requested at this update rate.

**Note** This will not change the rate at which any particular variable is made available to the WorkstationST application. For example, if a variable resides on a Mark VIe EGD page produced at 1000 milliseconds, then the fastest the variable changes on a screen will be 1000 milliseconds. As noted in the help for the Default Screen Update Rate setting, the OPC DA Server's maximum client rate also has an effect on update rates of variables on a screen.

For applications where it is necessary to have particular screen variable values updated at a different rate than the configured default, a special syntax can be used when specifying the variable on a CIMPLICITY screen object. If the variable name is preceded by `<###>` the numeric value is used as an update rate.

**Example**

![Diagram](image)

The Expression is requesting the variable be updated at a 50 millisecond rate.

---

**Public Information**
3.4 *Embed a Trend*

Trend can be embedded into a CIMPLICITY screen.

➢ To embed a trend

1. From the **ToolboxST Trend**er, create and save a configured trend file.

2. From CimEdit, click the **Trend**er button located on the trend insert toolbar.
3. Right-click the inserted trend object, select **ControlSTTrendActiveX Control Object**, then click **Properties** to display the *ControlSTTrendActiveX Control Properties* dialog box.

4. Click **Browse** and navigate to the saved trend configuration file.

5. (Optional) Set the **Time Axis Duration in Seconds**. Refer to the section *Set Trender Time Axis Duration using a Script* or the section *Trender ActiveX Control Methods*, the method *SetXAxisDuration*, for instructions.

6. Click **OK** to upload the trend configuration file.

When viewed in CimEdit, the trend opens the saved configuration file and displays live data. Variables configured with Historian or Recorder deadbands will backfill with historical data.
Set Trender Time Axis Duration using a Script

**Note** The Trender ActiveX object contains method calls that can be called from CIMPLICITY procedures. This is preferred to the use of a script. Refer to the section *Trender ActiveX Control Methods*, the method SetXAxisDuration.

➢➢ To set the Time Axis using a script

1. Verify that the Trender’s ActiveX object is named.
2. Right-click on the ActiveX trender object in CimEdit and select **Properties**.
3. Enter a unique name for the object (*MyTrend* for this example).

![Properties - Object (MyTrend)](image)

4. Select the **Object help** type as **Text**, **Text file**, or **Help file**.
5. In the **Help Text** field, enter a script to set the Time Axis duration, then click **OK**.

The following example script sets the Time Axis duration to 5 minutes:

```vba
Sub OnMouseUp(x As Long, y As Long, flags As Long)
    Dim oScrnObj As GefObjectZ
    Set oScrnObj = CimGetScreen.Object
    Dim OleObj As GefObject
    Set OleObj = oScrnObj.Objects.Item("MyTrend")
    OleObj.OleObject.TimeAxisDuration = "300"
End Sub
```
3.5  **Trender ActiveX® Control Methods**

Beginning with ControlST V07.06, the ActiveX controls used in CIMPLICITY CimView offer methods that can be called from CimView procedures. Applying a method eliminates the need to use scripts to configure the properties for the ActiveX control. In the following figure, a Trender ActiveX control has been given a name on the general tab, *TrenderActiveX*, which enables other CIMPICITY objects to have procedures that invoke a method.

The following methods are available for the Trender ActiveX control:

- **AddVariables** adds one or more variables specified by the comma separated list of variable names.
- **RemoveVariables** removes one or more variables specified by the comma separated list of variable names.
- **PauseUpdates** pauses or restarts live scrolling depending on the enable argument.
- **Record** starts or stops recording of live data depending on the enable argument.
- **SetXAxisDuration** sets the time axis duration to the value specified in seconds.
- **SetStackedTraceMode** sets the stacked trace mode to Auto when specified with a value of 1 and to Off when specified with a value of 0.
- **SetNumberOfChaptersToSave**, when set to a non-zero value and the Trender ActiveX control goes out of the active screen cache, the associated *.Trend* file is saved with the configured number of chapters of data, starting with the most recent chapter.
3.6 Expand the Traces List on Mouse Hover

If the Trender Trace items detail splitter bar is collapsed all the way to the bottom, when the Expand the traces list on mouse hover option is set to True (enabled), the traces list will automatically unhide when the mouse hovers over the toolbar.

The following figure displays the toolbar and slider fully collapsed.

*Trender Toolbar and slider Collapsed*

The following figure displays the toolbar and slider on mouse hover when this option is enabled.

*Trender Toolbar and slider Expanded*
3.7 Add to Trend

Both CimEdit and CimView support a right-click Add to Trend menu item.

➢➢ To add variables to a trend: from CimEdit, right-click any object on the screen and select Add to Trend from the shortcut menu.

The variables referenced within the screen object where the mouse is right-clicked are added to a Trender window. If the variables have been configured with a Recorder or Historian Deadband, the trend will attempt to back fill the values after starting the live value update. By default, every variable behind the right-clicked screen object is added to a trend. If the right-clicked screen object, or any of that object’s parent objects, including the screen itself, has a CIMPLICITY AddToTrendVars variable configured, then the variables configured on the AddToTrendVars variable are used.

An optional variable can be added to a CIMPLICITY object to allow inclusion of additional Trender command line arguments when Trender runs. A variable named TrenderCmdArgs can be set to a comma-separated list of arguments. For example, setting TrenderCmdArgs to /DisableBackFill results in launching Trender without providing the historical backfill of data for each trended variable. Additionally, setting a global Boolean variable, DisableTrenderBackfill to True enables Trender to be run with a /DisableBackFill variable.

Note For more information on command line arguments, refer to the Trender Instruction Guide (GEI-100795), the section Command Line Arguments.

Example

A group of screen objects that make up a tank are configured with an AddToTrendVars variable.

➢➢ To display the variables for an object

1. From CimEdit, right-click the object and select Properties from the shortcut menu.
2. Select the Variables tab to display the variables.
The value (G1.TankC_Level) configured for the GoToDefVar variable is displayed.

The variables configured for the AddToTrendVards variable (G1.Var1, G1.Var2) are added to the trend.
3.8 Go To Definition In Logic

Similar to the function described in the Add to Trend section, both CimEdit and CimView also support a right-click on the Go To Definition In Logic menu item.

**Note** The Go To Definition In Logic menu item displays either a block diagram with the variable highlighted, or, if the variable is an input point, an input module.

If the Go To Definition In Logic menu item is not enabled, refer to the section Enable Non-translated Content for additional information.

➢➢ To display a variable: from CimEdit, right-click any object on the screen and select Go To Definition In Logic from the shortcut menu.

The variables referenced within the screen object located where the mouse is right-clicked are collected. If more than one variable is present, a dialog box is presented allowing the user to select a single variable. Once the selection is narrowed to a single variable, the ToolboxST application navigates to the first write location of the variable. If the right-clicked screen object, or any of that object’s parent objects, including the screen itself, has a CIMPLICITY GoToDefVar variable configured, the variable configured on the GoToDefVar variable is used.

**Example**

A group of screen objects that make up a tank are configured with a GoToDefVar variable.

➢➢ To display the variables for an object

1. From CimEdit, right-click the object and select Properties.
2. Select the Variables tab to display the variables.

![Properties - Group](image_url) The value (G1.TankC_Level) configured for the GoToDefVar variable is displayed.
3.9 Go To Display Screen

If a variable behind a CIMPLICITY screen object has a configured Display Screen attribute, the right-click Go To Display Screen menu from CimView or CimEdit displays the defined screen if that screen exists. If the right-clicked CIMPLICITY screen object contains more than one variable with a configured Display Screen attribute, the user is presented with a list of variables to narrow the search.

Example

In this example, the variable D1.ThirtySec has a configured Display Screen attribute.

There are two variables that can override the display screen setting for a CIMPLICITY screen object. If a Display Screen variable or a parent object’s Display Screen variable exists, this screen is used.
The GoToScreenVar CIMPLICITY screen object variable can be used to override the Go To Display Screen behavior. In the following example, the Display Screen attribute of the variable D1.TankLevel defines the screen to be opened.

![GoToScreenVar CIMPLICITY screen object variable](image)

### 3.10 Display Variable Attributes

The variable, or variables, behind the right-clicked CIMPLICITY screen object are displayed in the Display Variable Attributes dialog box. The dialog box displays the attributes and live values of the variables.

➢ To display variable attributes: from CimEdit, right-click any object on the screen and select **Display Variable Attributes** from the shortcut menu.

![Display Variable Attributes](image)
3.11 Alarm Status History

If a variable on a CIMPLICITY screen is defined as an alarm, the CIMPLICITY\texttt{AlarmStatusHistory} screen displays when the CimView/CimEdit right-click \textbf{Alarm Status History} menu item is selected.

A CIMPLICITY screen object with the Variable from Device D1 and a \textbf{One Minute Alarm} was selected.

The previous screen uses a view named \textit{CimplicityAlarmStatusHistory.AVView}. The WorkstationST Alarm Viewer allows you to create a view and configure it to your requirements. The view is saved as \textit{CimplicityAlarmStatusHistory.AVView} by right-clicking on the Alarm Status History tab to display a shortcut menu.

Select \textbf{Save View} or \textbf{Save View As}.

The \textbf{Save As} dialog box displays.
Select the **CimplicityAlarmStatusHistory.AVView** file and click **Save**.

**Note** Refer to the *WorkstationST Alarm Viewer Instruction Guide* (GEI-100620) for more information.

### 3.12 Alarm Status Information

A CIMPPLICITY screen script can be written to obtain the total number of active alarms and the total number of unacknowledged alarms (active alarms or normal alarms). The Com2ControlST COM callable dynamic-link library, included with the WorkstationST application, provides a method that can be called to obtain these alarm totals.

The **Alarm Viewer** is displayed with one alarm.

The **CimView** screen displayed has unacknowledged alarms.
3.12.1 *CimEdit Configuration*

The button in this example is part of the Screen Standards group's Navigation library screen object. The Alarm button consists of a group containing a button and a label.

➢ *To display the Alarm button properties*

From the *CimEdit* screen, right-click *Alarms* and select *Properties* to display the *Properties - Group* dialog box.

From the *Group* tab, expand the *Alarms* object to display the button and the label.

Select the *Variables* tab to display the variables added to the group.
The AlarmStatusTotalsUpdate displays as follows:

Sub AlarmStatusTotalsUpdate()
    If com2ControlSTApp Is Nothing Then
        Set com2ControlSTApp = CreateObject("GeCss.Util.Com2ControlST")
    End If

    Dim totAlm As Integer
    Dim totUnack As Integer
    Dim almBlink As Boolean
    Dim almPresent As Boolean

    com2ControlSTApp.GetProcessAlarmQtys totAlm, totUnAck

    Dim almBlinkVar As GefObjectVariable
    Dim almPresentVar As GefObjectVariable
    Dim oCimObj As GefObject

    Set oCimObj = CimGetObject
    Set almBlinkVar = oCimObj.GetVariable("AlmBlink")
    Set almPresentVar = oCimObj.GetVariable("AlmPresent")
    almBlink = almBlinkVar
    almPresent = almPresentVar

    If totUnack > 0 And Not almBlink Then
        almBlinkVar = True
    ElseIf totUnack = 0 And almBlink Then
        almBlinkVar = False
    End If

    If totAlm > 0 And Not almPresent Then
        almPresentVar = True
    ElseIf totAlm = 0 And almPresent Then
        almPresentVar = False
    End If

End Sub

Note: The existing AlarmViewerActivate script entry point used the global com2ControlSTApp and therefore contained the following line. The Global declaration of com2ControlSTApp is required in order for this script to execute.

' Globals
Global com2ControlSTApp As Object
The script to open the named alarm viewer is displayed below:

Sub OpenNamedAlarmViewer
    If com2ControlSTApp Is Nothing Then
        Set com2ControlSTApp = CreateObject("GeCss.Util.Com2ControlST")
    End If
    com2ControlSTApp.OnGoToNamedAlarmViewer
    "CimplicityAlarmStatusHistory.AVView", "AlarmViewerNav"
End Sub
From the Actions tab, the Actions for the procedure can be edited.
➢ To display the AlarmPb button properties

From the Group tab, right-click the AlarmPb object and select Properties to display the Properties-Object dialog box.
From the Color Animation tab, click Edit to display the Expression List Attribute Animation dialog box.
➢ To display the label properties

From the **Group** tab, right-click the **label** object and select **Properties** to display the **Properties-Object** dialog box.
From the Color Animation tab, click **Edit** to display the Expression List Attribute Animation dialog box.
3.12.2 Alarm Viewer View Configuration

The `CimplicityAlarmStatusHistory.AvView` file is used to tailor the behavior of the alarm totals returned by the `GetProcessAlarmQtys` com2ControlSTApp call. The `CimplicityAlarmStatusHistory.AvView` file is the view used for the CIMPLICITY right-click `Alarm Status History` menu item. The `CimplicityAlarmStatusHistory.AvView` file provides the embedded alarm status client information necessary to determine the WorkstationST Alarm Server host (or redundant Alarm Server host pair), and the alarm filter to use. The following example shows how the user would set up a computer to provide alarm totals from a redundant Alarm Server configuration and filtered by alarms from controllers D1 and B1.

➢➢ To set up a computer to provide alarm totals from a redundant Alarm Server configuration

1. From the `Start` menu, select `Programs, GE ControlST, WorkstationST Alarm Viewer`, and `WorkstationST Alarm Viewer`.

2. Enter Remote Mode and connect to the Primary Alarm Server.

   From the `ToolboxST System Editor`, select the system item.

   From the `Property Editor`, expand `Redundant Alarm Server` and select a `Primary Alarm Server` and `Secondary Alarm Server`.

   From the `View menu`, select `Remote Mode` to display the `Remote Connection to WorkstationST Alarm Server` dialog box.
3. Create a New Filter Collection.

From the **File** menu, select **New Filter Collection**.

A new collection displays in the **Filter Definitions** tab **Tree View**.

From the **Filter Definitions** tab **Tree View**, right-click the new filter and select **Rename**.

The filter must be named **AlarmStatusFilter**.

4. Select the devices for display.
5. From the toolbar, click **Save** to save the currently defined filters.

6. From the file menu, select **Save View As** to display the **Save As** dialog box.

7. Enter **CimplexityAlarmStatusHistory.AvView** and click **Save**.

If CimView is open, use the CIMPLICITY options dialog available through Start/Programs/ Proficy*/HMI SCADA CIMPLICITY to stop the CIMPLICITY viewer. The script call now reflects the total counts that are seen by the CimplexityAlarmStatusHistory.AvView alarm viewer.
3.13 **ActiveX Alarm Viewer**

The WorkstationST Alarm Viewer can be added in CIMPLICITY CimEdit as an ActiveX control.

➢➢

**To configure the Alarm Viewer as an ActiveX control**

1. Right-click the ActiveX Alarms object and select the **Control Properties** tab.

![ActiveX Alarm Viewer Properties](image)

2. Set the **View State File Path** and **Active Filter Name** properties as described in the following table.

### ActiveX Alarm Viewer Configuration Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>View State File Path</td>
<td>Set to a previously saved alarm View file. The View file settings include which tabs are visible (such as Live Alarms and Short Term Historical Alarms), an active filter, the Visible column, and column order.</td>
</tr>
<tr>
<td>Active Filter Name</td>
<td>Set to change the display filter for the Live Alarms tab. The following script, <em>Example Active Filter CIMPLICITY Script</em>, provides an example CIMPLICITY script that sets the Active Filter to <em>Device</em> after locating an ActiveX alarms object that has been named <em>ActiveXAlarms</em>.</td>
</tr>
</tbody>
</table>
Example Active Filter CIMPLICITY Script

Sub OnMouseUp(x As Long, y As Long, flags As Long)
    Dim cimScreen As GefScreen
    Dim cimClassObj As GefObject
    Dim oCimOLEObj As Object
    Set cimScreen = CimGetScreen()

    Set cimClassObj = cimScreen.Object.Objects.Item("ActiveXAlarms")
    If Not cimClassObj Is Nothing Then
        Set oCimOLEObj = cimClassObj.OLEObject
        If Not oCimOLEObj Is Nothing Then
            oCimOLEObj.ActiveFilterName = "Device"
            End If
        End If
    End Sub

Note  The CIMPLICITY object name is configured by setting the Object name property from the object’s Properties General tab.

The following methods are available for the Alarm Viewer ActiveX control:

- **SetViewStateFile** allows setting the view state file for use by the Alarm Viewer.
- **LoadFilterFile** allows setting the path to the alarm filter file.
- **SetFilterName** allows the setting of a filter name. The filter must exist in the current list of filters.
- **AddFilterDevices** allows adding the specified devices in a comma separated string to the current filter’s device property.
- **RemoveFilterDevices** allows removing the specified devices in a comma separated string from the current filter’s device property.
- **GetFilterDevices** returns the current comma separated list of devices in the current active filter.
- **ShowMenu** makes the main menu visible.
- **HideMenu** makes the main menu invisible.
3.14 **Alarm Symbol ActiveX Control**

The ControlST Alarm Symbol ActiveX object can be added to a CIMPLICITY screen by using the default tool icon on the CIMPLICITY CimEdit Drawing tool ribbon displayed as follows,

![Insert Object](image1)

OR

by browsing for OLE objects and selecting the GeCssAlarmImageActiveXControl/

![Insert Object](image2)

The alarm symbol displays on the screen.

![Alarm Symbol](image3)

Alarms residing in the *Alarm Server in use* are used to drive the alarm symbols animation. The Alarm Server in use is either the local Alarm Server, or the Alarm Server configured in the WorkstationST Component Editor, General tab, *Alarm Server to Use* property.

---

*Note*  The *Enable Process Alarms* property on the WorkstationST OPC DA Server tab must be set to True.

---

*Note* Device cared about by proxy will not be available to drive alarm symbol animation as alarms with the by proxy feature are not included.
➢➢ To configure an alarm symbol in CimEdit

From the CIMPLICITY screen, right-click the alarm symbol, select GECssAlarmImageActiveX Control Object and Properties. The GECssAlarmImageActiveXControl Object Properties dialog box displays.

Enter the configuration and click OK.

In this example, the symbol will display priority 1 alarms from plant area IGCC.Common and will be invisible if all priority 1 alarms in this area are normal and acknowledged.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plant Area</td>
<td>If configured, the device name and variable name are ignored. Any alarm configured in this plant area will be evaluated in the symbol's presentation.</td>
</tr>
<tr>
<td>Include alarms from child plant areas</td>
<td>If checked, the evaluation will include the specified plant area and all child plant areas.</td>
</tr>
<tr>
<td>Device Name</td>
<td>If plant area is blank, all alarms from this device are used when evaluating the symbol’s presentation.</td>
</tr>
<tr>
<td>Filter by Priority</td>
<td>If not zero, only alarms of this priority from the Plant Area or Device will be included when evaluating the symbol’s presentation.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>If the Plant Area and Device Name settings are blank, the variable name can be used to allow the symbol to reflect a single variable’s current symbol.</td>
</tr>
<tr>
<td>Make invisible for the Normal and Acknowledged alarm state</td>
<td>If selected, the symbol is not visible when the evaluation results in a symbol for the normal and acknowledged state. When not selected, the normal background color for the alarm’s class is used with a text of N.</td>
</tr>
</tbody>
</table>
These properties may also be configured from the CIMPLICITY screen.

From the CIMPLICITY screen, right-click the alarm symbol, select Properties and Control Properties to display the Control properties.

Edit the properties as required.
CIMPLICITY scripting can be used to modify the properties on the ActiveX alarm symbol. In the following procedure an on screen open script is used to configure the properties of the alarm symbol. All the properties can be set. Any that are not set keep the values specified from one of the previous procedure.

➢➢➢ To configure an alarm symbol with CIMPLICITY scripting

1. Create an Edit Script for an alarm symbol and enter the CIMPLICITY script.

2. From the Edit Script File menu, click Save.
The following methods are available for the Alarm Symbol ActiveX control:

- **SetDeviceName** allows a device name filter to be set.
- **GetDeviceName** returns the name of the device currently being used for the device name filter.
- **SetIncludePlantAreaChildren** allows a Boolean setting to include child plant areas or exclude them.
- **GetIncludePlantAreaChildren** allows the reading of the current setting for including child plant areas.
- **SetInvisibleForNormal** allows a Boolean setting to show the symbol or hide the symbol when the alarm state is normal and acknowledged.
- **GetInvisibleForNormal** allows the reading of the current setting for invisible for normal.
- **SetPlantArea** allows the plant area filter to be set.
- **GetPlantArea** returns the current plant area filter setting.
- **SetPriority** allows an alarm priority filter to be set.
- **GetPriority** returns the current priority filter setting.
- **SetVariableName** allows the variable name filter to be set.
- **GetVariableName** returns the current variable name filter setting.

### 3.15 Print Screen

This menu item will print the current screen to the current Windows® default printer. CimView and CimEdit also contain standard menu items for printing.

CIMPLICITY scripting can be used to modify the properties on the ActiveX alarm symbol. In the following example, an on screen open script is used to configure the properties of the alarm symbol. Any of the properties can be set. Any properties not set keep the values specified from one of the two previous dialogs.
3.16 Other Right-click Menus

Refer to CIMPLICITY Advanced Viewer documentation for information on the standard right-click menu items displayed as follows.

For CimView:

For CimEdit:
3.17 Drag-and-drop to CimEdit

The ToolboxST application supports copy-and-paste, and drag-and-drop functions for blocks. This support includes the ability to create HMI screen objects on HMI screens created with the Proficy HMI/SCADA CIMPLICITY 7.5 product. If you drag a block from the ToolboxST Tree View onto the CimEdit screen editor, CimEdit uses data from the block’s attribute list to create a screen graphic using a linked object.

The block’s attribute list must include HMLinkSource and HMLinkedObject, since these specify the path to the link source file and link source object needed to create the linked object. The path specified by the HMLinkSource must be fully qualified, or must be relative to the screen being edited. CimEdit names the newly created link container based on the block’s HMLinkContainer attribute, if present, or will give it a default name if the block’s HMLinkContainer attribute is missing or empty.

CimEdit populates public variables of the linked object with the values from the block’s attributes. Each attribute is used to populate the public variable of the same name, or is ignored if no public variable with that name exists. The exact value from each attribute is used during this process, with the following exceptions:

- If an attribute’s values contain text in braces {braces}, that text is expanded based on the value of other attributes of the block or its parent. This expansion is identical to that done during block instancing.
- If there is no attribute named Caption, the ToolboxST application automatically creates one for CimEdit, and sets its value to that of the Device attribute. By convention, the Device attribute contains the name of the field device, such as the pump or motor, being controlled by the block.
- If there is no attribute named ctrlr, the ToolboxST application automatically creates one for CimEdit, and sets its value to the name of the controller followed by a period. For example, if the block is being used by a controller named G1, the ctrlr attribute will be set to a value of G1 unless the block actually has a ctrlr attribute.
- The optional HMLinkBlockPinsToParameters block attribute can be used to further tune the drag-and-drop or copy-and-paste behavior. If this attribute is set to True, for each block pin with a connected global variable, the ToolboxST application will create an attribute named after the pin, with a value equal to the short name of the connected variable. If the HMLinkBlockPinsToParameters block attribute is set to a value of FullNames, the full name of the attached variable is used.
- Occasionally, there is a need to create multiple objects on the CimEdit screen editor when dragging a single block from the ToolboxST application. This can be done by putting comma separated values for the HMLinkedObject block attribute, causing the ToolboxST application to split that attribute value into multiple values. If this is done, the ToolboxST application also checks the HMLinkSource, HMLinkContainer, Device, and Caption attributes for commas, and splits each of them if appropriate.

Example

A Software block is being dragged to CimEdit. The acv_horiz screen object found in tp_dcs_objects.cim will be added to the target CimEdit design surface when the 212FCV50219:PID_MA item is dragged to CimEdit.
➢ To link a ToolboxST block to an HMI block

From the Software tab, select the 212FCV50219:PID_MA item and drag it to the CimEdit design surface.

The result of the drop onto the CimEdit design surface is displayed as follows.

The parent task has a device attribute of Value 212FCV50219.
You can also drag a variable to CimEdit. When dragging from a variable grid, or the variable browser, in the ToolboxST application, a text object is created on the CimEdit screen with the text expression set to the value of the variable. Alternately, there is a way to map the drag of a variable to a CIMPLICITY linked screen object. If the file HMIBlockMappings.xml exists in the ToolboxST application’s HMIScreens folder, when a variable is dragged, the entry for $Variable allows the mapping of the variable to a screen object in a source screen. The value of the variable being dragged is assigned to the screen object’s varname variable.

Example

An HMIBlockMappings.xml file looks like this:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <Blocks>
    <Block Name="$Variable"
        HMIScreenObj="variableDisplay"
        HMILinkSource="C:\GEWorkstationST\HMIFiles\Library\Screen objects\General\tp_WorkstationST_objects.cim" />
  </Blocks>
</root>
```

The CIMPLICITY screen object variableDisplay, referenced in the above HMIBlockMappings.xml, contains the variable varname which will be assigned a value matching the dragged variable name. In this example, the variableDisplay screen object is a group of three text objects (<varname>.point_ID, <varname>. <varname>.description).
The General tab displays the Object type (Group) and the assigned group Object Name (variableDisplay).

In addition to the drag-and-drop function available in the ToolboxST application, there is a similar Drag Variables To Screen option available from the right-click menu in CimEdit. The same rules that apply to the drag-and-drop of variables from the ToolboxST application apply here.

➢ To drag a variable onto the CimEdit design surface: from CimEdit, right-click anywhere on the design surface and select Drag Variables To Screen from the shortcut menu.
Beginning with ToolboxST V3.05 and later, you can drag and drop hardware I/O modules to CimEdit. The ToolboxST Hardware tab’s Distributed I/O can contain groups and modules, and each group can also contain modules. Groups and modules can be dragged onto a CimEdit screen.

**Example**

If the target HMI Linked Object contains a variable corresponding to the following listed variables, the drag results in the update of the value of the variable. The following is an example of a group with two child modules:

**HMI Link Source** is the path to the CIMPLICITY screen file containing the HMI Linked Object. This path can be relative or absolute.

**HMI Linked Object** is the name of the library screen object found in the HMI Link Source file. This ties this hardware group to a screen object when dragging the group and its child modules onto a CimEdit screen.
When dragging a Group, the following Group properties are updated to corresponding variables of the target HMI Linked Object:

- Description
- Icon
- Name
- Type

The following is an example of a module:

![Module Example](image)

**HMI Link Source** is the path to the CIMPLICITY screen file containing the HMI Linked Object. This path can be relative or absolute.

**HMI Linked Object** is the name of the library screen object found in the HMI Link Source file. This ties this module to a screen object when dragging the module onto a CimEdit screen.

When dragging an I/O pack, the following pack properties are updated to corresponding variables of the target HMI Linked Object:

- Module (the module name)
- Barcode
- Location
- Group
- ModuleID
- Required (true or false)
- Description
- Position
- Type
- ctrlr (the Controller Name)
- Redundancy

**Note** When I/O packs are nested within a group, and copied with that group, the pack Data Fields are added to the group and prefixed with PACK_Position_ (for example PACK_1A1A_).

You can also override the HMI Linked Object and HMI Link Source. If the file `HMIBlockMappings.xml` exists in the ToolboxST system's HMIScreens folder, when an I/O pack is dragged, the entry in `HMIBlockMappings.xml` matching the Name property of the I/O module allows the mapping of a screen object in a source screen. When a hardware group is dragged, the entry in `HMIBlockMappings.xml` matching the Type property of the hardware group (Cabinet or General), allows the mapping of a screen object in a source screen.
Example

<?xml version="1.0" encoding="utf-8" ?>

<root>

<Blocks>
  <Block Name="Cabinet" HmiScreenObj="cab_module" HMILinkSource="tp_dcs_objects.cim" />
  <Block Name="General" HmiScreenObj="cab_module" HMILinkSource="tp_dcs_objects.cim" />
  <Block Name="IS220PAIC" HmiScreenObj="io_module" HMILinkSource="tp_dcs_objects.cim" />
</Blocks>

</root>

In this example, any hardware group with a blank HMI Link Source and HMI LINKED Object defaults to using cab_module and tp_dcs_objects.cim. All PAIC modules will use io_module and tp_dcs_objects.cim.
4 CimView

CimView is used to view graphic screen documents. When the Graphics without a Project setting has been enabled and downloaded to the target WorkstationST computer, the WorkstationST application tailors CimEdit and CimView to work with the ToolboxST configuration.

The right-click shortcut menu for CimView is similar in appearance and identical in behavior to the CimEdit menu.

![Menu](image)

**Note** Refer to the section *Enable Non-translated Content* for additional information.
4.1 Start and Stop CimView

For CimView to communicate to an OPC DA Server, the CIMPLICITY Service must be started. If this service is not running when CimView is started, the following dialog box displays:

If the HMI Config feature is enabled, when the WorkstationST Service starts, the CIMPLICITY Service startup type is set to Automatic and the CIMPLICITY Service starts.

The CIMPLICITY Options dialog box can be accessed through the start program’s Proficy HMI/SCADA - CIMPLICITY 7.5 menu.

Click Start as Viewer. The CIMPLICITY service starts.

Click Stop. The Viewer project stops. This is equivalent to stopping the CIMPLICITY service.
4.2 Display Variable Attributes

The variables referenced within the screen object located where the mouse is right-clicked are displayed in a variable attribute window.

➢ To display variable attributes: from CimView, right-click an object and select Display Variable Attributes from the shortcut menu.
4.3 Enable a Second Language

The Primary and Second languages can be configured in the ToolboxST System Editor. Configuring the names of the languages allows HMI applications to provide translation support. CimView can be configured to use translation files which index translated text by language name.

➢➢➢ To configure a primary and second language

From the ToolboxST System Editor Tree View, select the system item.

From the Property Editor, expand Languages and select the Primary Language and Secondary Language.

Within the ToolboxST configuration there are several places where a primary and second description can be set:

• Each variable has a Description and a Second Language Description

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Second Language Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TemperVar</td>
<td>Temperature in Celsius</td>
<td>Temperatur i Celsius</td>
</tr>
</tbody>
</table>

• Variable Enumeration Definitions accessed from the variable’s enumerations property, or from the variable’s Initial Value property editor.

• Alarm States that can be associated with a variable can have a second language description.
• Each WorkstationST component can enable the default language to be used by HMI applications.

When **Use Second Language** is set to **True**, the second language is selected by CimView at initial startup.
4.3.1 Second Language Use by HMI Applications

Applications that display the variable description in a data grid display both the primary and the second descriptions. You can choose which columns, and their order, to be displayed. These HMI applications are:

- WorkstationST Alarm Viewer
- Trender
- ToolboxST Watch Window
- ToolboxST Live View

The CimView application allows switching of the selected language.

4.3.2 Second Language Use by CimView

From a CimView screen, you can right-click and change regional settings. The Use Second Language option uses a toggle key to turn the second language on or off.

If the Use Second Language option is selected, the Enumeration text and the variable Description attributes display the configured second language for the associated enumeration or description. In addition to displaying descriptions and enumeration text in a second language, CimView can be configured to use translation files. The use of translation files is documented in the CIMPLICITY product documentation. The files contain one or more language translations of text. Using this CIMPLICITY feature allows static text on screens to be translated. This requires creation of a new language mapper file and the addition of language keys to the translation file.

➢➢ To create a new language mapper file: from the CIMPLICITY screens folder, right-click anywhere, select New, and CIMPLICITY Language Mapper UI from the shortcut menu.
You must add language keys to the translation file for each language. The names of the languages must match the names used in the ToolboxST system overview for the primary and second languages. (An earlier example used English and Norwegian.)

➢➢

To add language keys to the translation file

1. From the CIMPLICITY Language Mapper, select Translations and click Language Editor.

2. From the Language Editor, click Add, select the Language to add, click OK, and Done.
Various items display in the second language as displayed below.
When you right-click a CimView screen and select Use Second Language, the WorkstationST global script sets the Application.Language attribute to match the primary or second language. The script will issue the command:

```plaintext
Set scrnGef = CimGetScreen()
scrnGef.Application.Language = langName
```

Where `langName` has a value equal to the ToolboxST application configured Second Language.

**Note** If the WorkstationST component, Use Second Language option, is set to True, you must add scripting to initially set the second language upon the first screen open. The installed WorkstationST global script method, SetLanguageDefault, can be called in a common screen open event. Most applications have a navigation object that is common to all screens. The OnScreenOpen event can be used to execute the SetLanguageDefault method. When the option, Use Second Language, is set to True, this method will set the translation language to match the configured second language.

### 4.4 Select a Measurement System

Similar to the Use Second Language option, CimView has a setting for a measurement system. Each variable in the system can be configured with a format specification. Format specifications are contained in a Format Specification Set, which can contain multiple format specifications.

**Note** Each format specification in the set has a unique measurement system assigned.

**Example**

A format specification set can contain a format specification (such as DegreesF) with a US measurement system, and a format specification (such as DegreesC) with a metric measurement system. Assume a variable `B1.Var1` is assigned a format specification DegreesF. By default, CimView will display values in the native units of the variable (which may be Centigrade).

➢➢ **To switch from the native units to one of the configured measurement systems**

1. From CimView, right-click an object, select **Regional Settings** and **Select Active Measurement System** from the shortcut menus.

2. Select one of the configured measurement systems.

A script can be called to initialize the default measurement system. In a common On Screen Open event, the following script could be used:

```plaintext
Global MeasurementSystemSet As Boolean

If Not MeasurementSystemSet Then
    MeasurementSystemSet = true
    PointSet "MeasurementSystem", "Metric"
End If
```
4.5 Alarm Symbol Actions

Action can be taken from the CimView screen on active alarms configured in an alarm symbol ActiveX object.

➢ To take action on an alarm

From the CimView screen, click the alarm symbol to display the AlarmInfo screen.

This contains a list of all active alarms with symbols, for the alarm group configured in the ActiveX object (such as entire plant device, plant area, or single variable).

Select the alarm(s) to take action on by clicking on the text (such as LP DRUM LEVEL Low Low Low) and click on the symbol to display the AlarmAction screen.

Alarm Help and Comments, Alarm Attributes, and Go to Definition in Logic are disabled if multiple alarms are selected.

From the AlarmAction screen, click to select the action (such as Acknowledge Alarm) for the alarm(s).

One or more actions may be disabled based on the system alarm parameters and user privilege configuration.
5 Privileges

The Privileges feature defines system access rights. System users and their roles are configured using the System Information Editor. Each user is assigned a role and an optional set of resources. User names correspond to Windows users (work groups or domain users). The following sections describe Privilege feature functionality:

- Display current user
- Log on
- Log off
- Set user access roles

Note: The Privileges feature is accessible from the WorkstationST application or WorkstationST Status Monitor icon in the taskbar notification area.

The following figure illustrates how the ToolboxST application, CimView, and CimEdit authenticate system access using the Privileges feature.
1. Request for the current logged-on user update message. The requestor sends a message: “LogonSecurityGetCurrentUserIntercomMsg” (send to master)
2. When a user is logged on or off, or a request for the current logged-on user is sent, the logon manager application sends a “logonSecurityGetCurrentUserIntercomMsg” intercom message (send to all clients)

Logged-on response messages contain the Log On Security object, as well as the current username. The new user is sent as Logged-on. When that user is logged off, the previous user is sent as a logged-on intercom. Messages also contain the current Windows session ID.
5.1 Display Current User

**Note** Privileges can only be modified from the System Information Editor.

➢ To display the current user

From the taskbar notification area, right-click the WorkstationST icon and select **Show Current Privileges User**.

The current user and assigned privileges display.
5.2 Log On

From the Privileges User Logon Manager, you can:

- Log on and replace the current user
- Log on and temporarily override the current user

**Note** An event message indicating the logged-on user (and role) is generated in the Alarm Viewer.

➢➢➢

To replace the current user: right-click the WorkstationST Status Monitor icon and select **Privileges Log On**.

![Privileges User Logon Manager](image)

From the **User** drop-down list, select the user, enter the password, and click **OK**.

The user selected (with defined privileges) replaces the current user.
➢ To temporarily override the current user

The user `maint` is now logged on over the current user (now sub-user). The previously defined user privileges for `maint` take effect.
5.3 Log Off

The feature *Auto LogOff Timeout* is activated when one user is logged on over another user. This feature is enabled in the Property Editor when the Users and Roles item is selected from the Tree View.

**Note** An event message indicating that the user logged off is generated in the Alarm Viewer.

➢➢ To set the *Auto LogOff Timeout*: from the **System Editor Tree View**, double-click the system item to display the **System Information Editor**.

![System Information Editor](image)

If the temporary user remains inactive through the timeout period, that user is logged off and message displays that the user administrator is again active.

➢➢ To log off the **Current User**: right-click the WorkstationST Status Monitor icon and select **Privileges Log Off**.

The current user is now logged off. If the current user was the temporary user, the sub-user now becomes the current user and previously defined user privileges now apply.
5.4 Access Roles

The Access Roles property in the Component Property Editor allows you to set access rights for modifying data or design according to user names and roles pre-defined in the System Information Editor.

**Note** Only users with roles assigned to a protected object can access that object.

➢➢ To set access roles

From the General tab Property Editor, select Access Roles and click ellipsis button.
By default, when a role is assigned to an access right, only users with that role can access the protected object (programs, tasks, and such). Permissions are also inherited from parent objects. For example, if a component has the *Modify Design* role assigned to *Operator*, all objects in that component require the *Operator* role before allowing design changes. However, each Role name can optionally be applied with the *Grant* keyword. This reverses the normal behavior so that users with the *Grant* keyword have access to the object regardless of any parent objects. As an example, the *Modify Design* access could be restricted for an entire component to users with the *Maintenance* role, but one particular program could be modifiable by users with the *Operator* role.

<table>
<thead>
<tr>
<th>Access Right</th>
<th>Role Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modify Design</td>
<td>Operator</td>
</tr>
<tr>
<td>View Design</td>
<td>User</td>
</tr>
<tr>
<td>Modify Data</td>
<td>operator</td>
</tr>
</tbody>
</table>

Modify Design Grant Admin Access

From the drop-down list, select an **Access Right** and the **Role Name** to be assigned.
The Tagout application provides the means to lock out the control of plant equipment from the HMI screen. This capability is embedded in the HMI screens and is available for components that interface with plant equipment such as motor operated valves.

**Warning**
This function is provided for indication only. This action will not lockout this piece of equipment, it will only indicate lockout status. The Owner's lockout procedures must be followed to safely lock this equipment out.

### 6.1 System Data Flow

The following diagram displays the data flow when the tagout application is used. In the example, user1 has requested to tagout an S-O-V object on the HMI screen. When the user applies the tagout, a lock request is sent to the controller that is executing the S_O_V block. If the tagout request is successful, the tagout status variable transitions to True to indicate success. This status value is displayed in the tagout application.
6.2 Operation

The tagout of equipment is managed by a popup application attached to each HMI screen object that implements the Tagout feature. Each user that tags out the equipment must have Tagout Privileges enabled and must log on with a user name and a password. The user name and password entered can be anything the user can remember and is not tied to a Windows account name.

6.3 Initial Display

The Tagout dialog box on the HMI displays the following:

- A Tree View of all Lock Status variables that have content defined. If no tagout entries exist, only the current Lock Status variable being edited is displayed in the Tree View.
- For a selected Status variable, a list of users and the entered tagout information for each user is displayed.
- The Lock Status variable that is used.
- The Tagout State of the variable and the communications status to the controller.
- Any error conditions that may be present.
- The current language used on the display.
- The User that is currently editing the tagout file if any.

Information about the tagout status variable TMRM6E.TagoutSOV_1.lock is displayed in the following Tagout dialog box:

![Tagout dialog box with information]

No user is logged in editing this status variable.

No remote user is currently editing the content.

Live lock status value from the controller. The current state is not tagged out (Unlocked), as displayed in the Controller Communications status window.
# Tagout Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="printer" /></td>
<td>Prints the displayed tagout content to the printer.</td>
</tr>
<tr>
<td><img src="image" alt="delete" /></td>
<td>Allows the logged in user to delete the record being edited.</td>
</tr>
<tr>
<td><img src="image" alt="toggle" /></td>
<td>Toggles the tagout state of the variable in the selected row.</td>
</tr>
<tr>
<td><img src="image" alt="save all" /></td>
<td>Applies all changes to the controller, Alarm Server, Master Workstation, and other open tagout applications displaying the same Lock Status variable.</td>
</tr>
<tr>
<td><img src="image" alt="log on" /></td>
<td>Allows a user to log on and perform editing.</td>
</tr>
<tr>
<td><img src="image" alt="log off" /></td>
<td>Click to log off a user. Prevents editing of the content.</td>
</tr>
<tr>
<td><img src="image" alt="user logged in" /></td>
<td>Identifies the user currently logged on. The format is Computer Name:UserName.</td>
</tr>
<tr>
<td><img src="image" alt="tagout being edited by" /></td>
<td>Identifies the user currently editing the tagout information. The format is Computer Name:UserName.</td>
</tr>
<tr>
<td><img src="image" alt="change password" /></td>
<td>Change the password used with the selected tagout record. The user currently logged on can change their password. Also, the password can be changed by any user that has a Windows log on account with administrative privileges.</td>
</tr>
<tr>
<td><img src="image" alt="unlock file" /></td>
<td>Unlocks a file currently being edited by someone else. Forces the user to log off and discards all pending changes. Requires Windows log on account with administrative privileges to unlock.</td>
</tr>
<tr>
<td><img src="image" alt="language" /></td>
<td>Identifies the current language selection in use (English). This affects the description displayed in the dialog box title.</td>
</tr>
<tr>
<td><img src="image" alt="controller communications" /></td>
<td>Displays the current state of the tagout Lock Status variable from the controller. This is a live status update.</td>
</tr>
</tbody>
</table>
A user has the following capability from the Tagout dialog box:

- Select items in the Tree View to display other tagout entries.
- Print the displayed tagout contents for a specific Lock Status variable.
- Select the Tagout item (root node) in the Tree View and print all tagout entries.
- Log on with a valid user name and password to edit the contents of the selected tagout record.
- Reset the selected tagout records password if the user is authenticated as an administrator on this computer.
- Select each cell and edit the contents, except the Lock Status variable field. It is read only. (No editing is allowed when the cell displays black on dark gray to indicate the user does not have edit capability.)

- Toggle the tagout state using the Toggle button and apply changes. The Controller Communication window will reflect the actual state from the controller.
- Edit other fields and apply changes. The description field is edited in the context of the current display language. The other text fields are not language qualified.

- Reset the password for this record by selecting the Reset Password button.
- Click the Log off button to prevent unauthorized changes.
- Delete the current record being edited. If the record is deleted, the user is automatically logged off.

When the Use Second Language option is enabled, the description displays in the selected second language. Refer to GHT-200009, How to Configure a Second Language for Use by ControlST* HMI Applications for additional information.
6.4 Log On

If the user has tagout privileges enabled, the user is allowed to log on and perform editing. Otherwise, the user is denied edit access to the tagout content.

➢➢ To log on to the Tagout application

1. Click the Logon button to display the Tagout Login dialog box.

![Tagout Login dialog box](image)

Note If the user name and password already exists, the Tagout dialog box displays the user tagout record enabled for editing.

![Tagout dialog box](image)

![Tagout State cell](image)

The color of the row owned by the user is changed to indicate edit capability allowed. The user can select each cell and edit the contents.

When the Tagout State cell is selected, the user can double-click to toggle the state or click the Toggle button.
2. Click the **Apply** button to write the resolved state to the controller, send an event notification to all Alarm Servers, and write the contents to the master workstation.

**Note** If the deleted record is the last record in the list that is locked, the state is forced to *Not Tagged* out and this state is written to the controller automatically. If the lock is to be maintained, the user must maintain a record and the lock status must indicate *Locked*.

**Note** All changes are logged in the Alarm/Event subsystem and can be seen using the WorkstationST Alarm Viewer. The user entered information is written back to the master workstation defined in the system.
7 Glossary of Terms

Bind  To establish the correspondence between the data in an exchange and variables in a device.

Bind/Build  To bind the configuration for each consumed exchange and create/update the configuration for any produced exchange.

Collection  More formally, an EGD Collection. A group of devices that constitutes a formal subset of the devices participating in a particular EGD installation. This arbitrary grouping allows users to subdivide the system to make some tasks easier.

Consume  To receive an EGD data message (exchange).

Consumer  An EGD node configured to receive an EGD data message.

EGD  A mechanism that provides access to global data between nodes supporting the EGD protocol.

Exchange  An EGD data message consisting of a header and a body of data. The header contains the producer ID and the exchange ID that uniquely identifies the message. The body of data is a block of bytes in a format agreed upon by the producer and all consumers.

Feature  An element of the WorkstationST runtime system, which can be optionally enabled through the ToolboxST application. Examples include OPC Server, Recorder, and Alarm Viewer.

Global Data  A concept in which multiple controllers on a network can share information by exchanging portions of their local memory with peer controllers.

OPC  A standard for data exchange in the industrial environment. The OPC foundation provides specifications for various OPC standards such as OPC DA (Data Access) and OPC AE (Alarm and Event).

Produce  To send an EGD data message (exchange).

Producer  The EGD node configured to send data messages. The source of the data samples for an exchange.

Refresh  To bind the configuration for each consumed exchange for a particular consumed device.

Runtime  Software stored in the controller’s Flash memory that converts application code (pcode) to executable code.

Unbound Variables  Variables required by a consumer that were not found in the producer configuration during the bind.