

**AVEVA™**  
**Application Server**  
formerly Wonderware

**Application Server User Guide**



© 2020 AVEVA Group plc and its subsidiaries. All rights reserved.

No part of this documentation shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of AVEVA. No liability is assumed with respect to the use of the information contained herein.

Although precaution has been taken in the preparation of this documentation, AVEVA assumes no responsibility for errors or omissions. The information in this documentation is subject to change without notice and does not represent a commitment on the part of AVEVA. The software described in this documentation is furnished under a license agreement. This software may be used or copied only in accordance with the terms of such license agreement.

ArchestrA, Aquis, Avantis, Citect, DYNsIM, eDNA, EYESIM, InBatch, InduSoft, InStep, IntelaTrac, InTouch, OASyS, PIPEPHASE, PRiSM, PRO/II, PROVISION, ROMeo, SIM4ME, SimCentral, SimSci, Skelta, SmartGlance, Spiral Software, Termis, WindowMaker, WindowViewer, and Wonderware are trademarks of AVEVA and/or its subsidiaries. An extensive listing of AVEVA trademarks can be found at: <https://sw.aveva.com/legal>. All other brands may be trademarks of their respective owners.

Publication date: Monday, November 16, 2020

### **Contact Information**

AVEVA Group plc  
High Cross  
Madingley Road  
Cambridge  
CB3 0HB. UK

<https://sw.aveva.com/>

For information on how to contact sales and customer training, see <https://sw.aveva.com/contact>.

For information on how to contact technical support, see <https://sw.aveva.com/support>.

# Contents

Chapter 1 IDE Overview .....	19
What's a Galaxy? .....	19
Start the System Platform IDE .....	20
Install and Activate Your License .....	20
Manage Security Certificate Verification .....	20
Create a Galaxy .....	20
Connect to a Galaxy .....	22
IDE Views and Toolboxes .....	24
Template Toolbox .....	25
Graphics Toolbox .....	27
AVEVA Connect .....	28
IDE Application Views .....	28
Object Icons .....	29
Model View .....	30
Deployment View .....	32
Flex Licenses View .....	33
Derivation View .....	33
IO Devices View .....	34
IO Device Mapping View .....	36
Operations View .....	36
Customizing Your Workspace .....	37
Dock Views .....	37
Float Views .....	37
Hide Views .....	37
Reset the Workspace .....	38
Synchronize Views .....	38
Configure User Information .....	38
Log on with Security Enabled .....	41
Change Users .....	41
Chapter 2 Introduction to Objects .....	43
Introduction to Objects About Templates and Instances .....	43
Instances .....	44
Templates .....	44
Propagation .....	45
About Base Templates .....	45
Application Templates .....	47
Device Integration Templates .....	47
System Templates .....	47
About Derived Templates .....	48
View Object Properties .....	49
Working with Objects .....	51

Manage Toolsets .....	51
Create Toolsets.....	51
Create Child Toolsets .....	52
Deleting Toolsets .....	52
Using AVEVA Connect .....	52
Logging in to AVEVA Connect.....	52
Changing the AVEVA Connect Drive .....	53
Uploading Graphics to AVEVA Connect.....	54
Downloading Graphics to the Local Repository.....	54
Managing Graphics on AVEVA Connect .....	55
Managing Graphics with Multiple Users .....	56
Manage Galaxy Style Libraries .....	56
Import a Galaxy Style Library .....	56
Export a Galaxy Style Library .....	56
Reset a Pre-Defined or User-Defined Galaxy Style Library.....	57
Create Derived Templates .....	57
Derive Templates from Another Derived Template.....	57
Create Contained Templates .....	58
ApplicationObject Containment .....	60
Using Contained Names .....	64
Containment Examples.....	65
View Containment Relationships .....	67
Rename Contained Objects .....	67
Edit Objects.....	68
Object Editor User Assistance.....	69
Help File Structure .....	69
About the General Editor Layout .....	70
Lock and Unlock Template Attributes .....	70
Set Object Security .....	73
Group Locking/Security .....	74
About the Attributes Page .....	74
About the Scripts Page .....	79
About the Object Information Page.....	80
Customize Help .....	81
Locate the Help Folders .....	81
About the Field Attributes, UDAs, and Extensions Pages .....	81
View the Field Attributes Page.....	82
Determine Whether to Convert Field Attributes .....	82
Optimize Performance if You Use Field Attributes .....	83
Convert Field Attributes to Attributes.....	83
Special Considerations when Converting Field Attributes .....	86
Limitations and Exclusions when Converting Field Attributes .....	87
Update Scripts to Reference Converted Attributes .....	87
Reference Objects Using the Galaxy Browser .....	88
Browse for Attributes .....	88
Browse with an ArchestrA OPC UA Client.....	90
View Attribute Details in the Galaxy Browser .....	90
Browse for Graphics.....	92
Browse for Element Properties.....	93

Create a Filter for the Galaxy Browser .....	94
Change How Information is Shown in the Galaxy Browser .....	96
<b>Managing Objects .....</b>	<b>97</b>
Check Out Objects .....	97
Check In Objects .....	97
Validate Objects.....	98
Validate Scripts and Other External Components .....	98
Validate an Object Manually.....	99
Create an Instance.....	99
Rename an Object .....	100
Delete an Object .....	101
Export Objects .....	102
Protecting Objects on Export.....	102
About Protecting Objects on Export .....	102
Exporting Objects as Protected .....	104
Exporting Objects with I/O Auto Assignment .....	104
Exporting Script Function Libraries .....	104
Exporting Objects with Linked Content .....	105
Importing Objects.....	105
Importing Protected Objects.....	107
Importing Objects with I/O Auto Assignment.....	108
Importing Objects with Linked Content.....	108
Importing a SQLData Object .....	108
Importing Script Function Libraries .....	109
Importing Client Controls .....	109
Importing an App from the AVEVA Digital Exchange.....	110
Importing AVEVA Apps.....	110
Add a Control from an AVEVA App to a ViewApp .....	112
Configurable Properties in AVEVA Apps .....	112
AVEVA App Development Guidelines .....	114
AVEVA App Limitations and Restrictions.....	115
Troubleshooting AVEVA or WPF Apps that Fail to Import.....	116
Importing HTML5 Widgets .....	116
Carousel Widget.....	117
Web Browser Widget.....	118
QR Code Scanner .....	118
After You Import .....	119
<b>Configuring Objects .....</b>	<b>121</b>
Add Attributes to an Object .....	121
Configure an Attribute.....	122
Attribute Naming Conventions.....	123
Attributes and Scripting.....	125
Adding Features to Attributes.....	126
About Features Inheritance .....	126
Using the I/O Feature .....	127
Configure I/O as Read-only .....	129
Configure I/O as Read/Write .....	130

Configure I/O as Write-only .....	132
Quality of Read, Read/Write, and Write I/O .....	135
Configure I/O for Use with AVEVA Telemetry Server Communication Drivers .....	135
Using the History Feature .....	136
Using the Limit Alarms Feature .....	139
Using the ROC Alarms Feature .....	140
Using the Deviation Alarms Feature .....	141
Using the State Alarm Feature .....	143
How to Configure an External Alarm Source .....	144
Using the Bad Value Alarms Feature .....	146
Using the Statistics Feature .....	147
Using the Log Change Feature .....	148
Using I/O Auto Assignment .....	148
Assigning Areas and Objects to Scan Groups .....	150
Renaming Application, System, and DI Objects .....	151
Renaming Scan Groups .....	151
Validating and Editing I/O Assignments .....	151
Using the IO Mapping View .....	153
Validating I/O References .....	155
Overriding I/O Auto Assignments .....	155
Overriding Large Numbers of Attributes .....	156
Uploading Run-Time Configuration Changes for Auto Assigned Objects .....	157
I/O Auto Assignment Workflow Example .....	157
Using I/O Auto Assignment with OPC Clients .....	158
Override Scenario 1 — Siemens S7 Controllers with an OPC UA Client .....	159
Override Scenario 2 — Allen-Bradley CIP Controllers with an OPC UA Client .....	160
Writing and Editing Scripts .....	160
About Scripts .....	161
Script Execution .....	162
Script Locking and Change Propagation .....	164
Creating and Working with Content .....	165
Adding Graphics Items .....	166
Errors and Warnings in an Owned Graphic .....	167
Modifying Graphics .....	168
Set Graphics Properties with a Galaxy Style Library .....	168
Use Element Styles to Modify Graphics .....	168
Use Quality and Status Styles to Provide Run-Time Feedback .....	169
Use UI Themes to Set Appearance of WPF Controls .....	169
Using Symbol Wizards .....	170
Creating Symbol Wizards .....	171
Embedding Symbol Wizards into an Application .....	172
Renaming Graphics .....	173
Deleting Graphics .....	173
Rearrange Objects to Improve AVEVA OMI ViewApp Navigation .....	173
Rearrange Objects in the Model View .....	174
More Information About Rearranging Objects in the Model View .....	175
Rearranged Objects in the Export and Import of a Galaxy .....	175
Rearranged Objects in Backup and Restore of a Galaxy .....	175
Rearranged Objects in a GRLoad and GRDump .....	175
Rearrange an Object Programmatically using GRAccess Toolkit .....	175
Assign an Alias Name to an Object .....	176

Steps to Assign or Remove an Alias Name .....	177
More Information About Alias Names .....	178
Alias Retention in Galaxy Operations .....	178
Alias Names in the Export and Import of a Galaxy .....	178
Alias Names in Backup and Restore of a Galaxy .....	179
Migrate a Galaxy .....	179
Alias Names in a GRLoad and GRDump .....	179
Assign an Alias Name Programmatically using GRAccess Toolkit .....	179
<b>Working with the Industrial Graphic Editor .....</b>	<b>180</b>
Industrial Graphic Editor Workflows for Use with HMI/SCADA Applications .....	180
InTouch Managed Applications .....	181
AVEVA OMI ViewApps .....	182
Creating Symbols in the IDE .....	184
Creating Symbols in the Graphic Toolbox .....	184
Creating Symbols in AutomationObject Templates .....	185
Creating Symbols in AutomationObject Instances .....	187
Working with Element Styles.....	188
Understanding Element Styles .....	188
Galaxy Style Library.....	189
Visual Properties Defined by Element Styles .....	190
Element Styles in Animations .....	190
Property Style Order of Precedence.....	190
Updating Element Styles at Application Run Time.....	190
Managing Element Styles .....	191
Importing and Exporting Galaxy Style Libraries .....	191
Change the Visual Properties of an Element Style .....	191
Changing the Visual Properties of User-Defined Element Styles.....	195
Applying Element Styles to Elements.....	195
Using the Element Style List.....	195
Using the Properties Grid.....	195
Using Format Painter .....	196
Clearing an Element Style.....	196
Selecting an Element Style as a Default for a Canvas .....	196
Applying Element Styles to Groups of Elements .....	197
Setting a Group's Run-time Behavior to TreatAsIcon .....	197
Understanding Element Style Behavior with a Group of Elements .....	197
Configuring an Animation Using Element Styles .....	197
Configuring a Boolean Animation Using Element Styles .....	197
Configuring a Truth Table Animation with Element Styles .....	198
Adding Industrial Graphics to AVEVA OMI Panes .....	199
Create Fixed Size Symbols .....	199
Create a Symbol to Fit a Pane of a Known Size.....	200
Create a Fixed Sized Symbol for a Known Pane Size .....	201
Use the Industrial Graphic Editor to Create Fixed Size Symbols .....	202
More Information About Fixed Size Symbols During Configuration .....	203
Fixed Size Symbol Behavior During Run Time.....	204
Using Custom Properties with Application Server.....	204
Examples of Using Custom Properties .....	205
Using Custom Properties to Show Historical Summary Data .....	205
Analog Statistical Summary Data.....	205

State Statistical Summary Data .....	206
Historical Summary Period .....	207
Showing Statistical Summary Data .....	207
Using Binding in Custom Properties .....	209
Changing the Expression or Reference of a Custom Property at Run Time .....	211
Working with the SignedWrite() Function for Secured and Verified Writes .....	212
SignedWrite() Run-time Behavior .....	212
SignedWrite() Script Execution Sequence at Run Time .....	213
SignedWrite() Scripting Tips .....	213
Examples of Using the Attribute Parameter in the SignedWrite() Function .....	214
Secured and Verified Write Applied Examples .....	215
Working with the Show/Hide Content Script Functions .....	217
About the ShowContent() Function .....	217
Configuring the Show/Hide Content Script Functions .....	217
Best Pane Match Algorithm .....	219
Content Display Rules .....	221
Using Object Wizards .....	223
About Object Wizards .....	223
Object Wizard Features .....	224
Create a Basic Object Wizard .....	225
Object Wizard Configuration Best Practices .....	228
Object Wizard GUI Panes .....	229
Object Wizard User Interface Details .....	231
Object Wizard Components .....	234
Add Content for Use with an Object Wizard .....	234
Add a New Object-Owned Symbol .....	235
Link to Shared Content in the Graphic Toolbox .....	236
Associate Content with a Choice or Option .....	237
Remove an Association from a Choice or Option .....	238
Multiple Content Item Associations .....	239
Map Attribute and Symbol Configurations to a Choice or Option .....	239
About Attribute and Symbol Overrides .....	240
Delete Content from an Object .....	241
Delete an Attribute or Symbol from a Template .....	241
Example 2: Configure a Linked Symbol Associated with Multiple Choices .....	242
Conditional Visibility Expressions for Choice Groups, Choices, and Options .....	244
Add a Basic Visibility Expression .....	245
Add a Visibility Expression with Logical Operators .....	246
Operators, Rules, and Behavior for Visibility Expressions .....	247
Object Wizard Visibility Settings in a Derived Template .....	248
Example 1: Basic Visibility Expressions .....	249
Example 2: Complex Visibility Expressions .....	250
Scenario 1: Visibility of option is contingent on specific choices .....	251
Scenario 2: Visibility of option is contingent on a combination of choices and options .....	252
Example 3: Visibility of Choice Groups, Choices, and Options in a Derived Object Wizard .....	254
Trimming .....	254
Enable Trimming .....	255
Example: Enable Trimming for Attributes and Symbols .....	256
Attribute and Symbol Overrides .....	257
Enter Overrides and Set Visibility for Attribute and Symbol Values .....	259



About Attribute and Symbol Overrides .....	260
Configurable Settings for Attribute and Symbol Values .....	261
Object Wizard Derivation .....	263
Derive a Child Template from a Template with an Object Wizard.....	263
Set Visibility of Choices and Options in a Derived Template .....	263
Reasons to Derive a Template from an Object Wizard .....	264
Changes to Object Wizards in Derived Templates .....	264
Behavior of Object Wizard Components in a Derived Template .....	265
Example: Set Visibility of Derived Wizard Choices and Options .....	266
Object Wizard Test Mode .....	267
Verify Behavior of Visibility Expressions .....	267
About Object Wizard Behavior in Test Mode .....	269
Example 1: Verify Behavior of Associated Attributes .....	269
Example 2: Verify Behavior of Associated Symbols .....	271
Propagation of Object Wizard Changes to Derived Objects.....	276
Object Wizard Behavior when a Default Choice is Deleted .....	277
Limitations Associated with Object Wizard Propagation .....	277
Extend an Inherited Object Wizard .....	277
Add a Choice Group to a Derived Template .....	277
Add an Option to a Derived Template .....	279
Add an Association to an Inherited Object Wizard .....	279
Add a Script to a Derived Template .....	280
Add a Symbol to a Derived Template.....	281
Link to a Symbol in the Graphic Toolbox .....	282
Symbols and Object Wizards .....	284
Example 1: Add a Symbol to the Object .....	284
Example 2: Link to a Symbol in the Graphic Toolbox .....	285
Configure Instances .....	286
Configure an Instance .....	286
View and Configure Symbols .....	287
Guidelines for Configuring Instances with an Object Wizard .....	289
Override Default Settings .....	289
Create a New Instance Graphically .....	290
About Object Options .....	294
About Graphic Options .....	295
Add a Configured Asset to a Symbol .....	296
<b>Configuring and Using the OPC UA Server .....</b>	<b>297</b>
About the Application Server OPC UA Server .....	297
OPC UA Configuration Checklist.....	297
Configuring and Deploying the OPC UA Service .....	298
Client Access Rules and Galaxy Security.....	300
Configuring the Firewall for the OPC UA Service.....	301
Configure the Run-Time Node Firewall .....	301
Firewall Test .....	304
Configuring Server and Client Certificates for Third-Party OPC UA Client Applications .....	305
Export the OPC UA server certificate to the OPC UA client node .....	306
Import the OPC UA Server Certificate on the Client Computer .....	308
Configure OPC UA Client Certificates on the OPC UA Server .....	310

Port Usage .....	310
Using OI Gateway to Configure the Client Security Certificate .....	311
Trusting the Certificate between the OPC UA Server and OPC UA Client .....	312
Verify OPC UA Certificate Installation .....	313
<b>Deploying and Running an Application .....</b>	<b>317</b>
Deployment Overview .....	317
Deploying Objects from the IDE to Run Time .....	318
Allocating Run-time Resources .....	319
Allocating Galaxy Repository Node Memory .....	319
Object, Area, Engine, and Platform Assignments .....	319
Allocating AppEngines to Platforms .....	321
Data Integration Objects .....	323
Auto-Build .....	323
Determining Galaxy Status .....	324
Configuring Advanced Communication Management .....	324
Selecting Advanced Communication Management .....	326
Configuring Scan Modes .....	326
Simulation Server .....	328
Simulation Server and the OPCClient Object .....	329
I/O Auto Assignment in the Simulation Server .....	329
Configure the Simulation Server .....	330
View Simulated Data .....	331
Unknown Data Types .....	332
Deploying Objects .....	333
Deployment Error Messages .....	336
Publishing Managed InTouch Applications .....	336
Redeploying Objects .....	336
Redeploying the WinPlatform Object .....	337
Retentive Attribute Properties and Behavior .....	338
Undeploying Objects .....	338
Uploading Run-time Configuration .....	339
Uploading Restrictions .....	340
Undeployment Situations .....	340
Associating All Galaxy Graphics with an InTouchViewApp .....	340
About Associating All Galaxy Graphics with an InTouchViewApp .....	341
Configuring the Include All Galaxy Graphics Option .....	342
<b>Working with History .....</b>	<b>343</b>
Application Server History Components .....	343
Sending Historical Data Between Application Server and the Historian .....	343
Saving Object Attribute Data to the Historian .....	344
Saving Process Values as Historical Data .....	344
Saving Data Quality as Historical Data .....	345
Additional Quality Data Saved to the Historian .....	345
Saving the Data Timestamp as Historical Data .....	346
Saving Alarms and Events as Historical Data .....	347

Deploying and Undeploying Attributes .....	348
Saving Historical Data During Run Time .....	348
Advanced Communication Management .....	348
Store-and-Forward Mode .....	348
Buffered Behavior.....	349
Configuring Common Historical Attributes .....	349
Configuring System Objects to Store Historical Data .....	351
Configuring the WinPlatform Object to Store Historical Data.....	352
Configuring an AppEngine Object to Store Historical Data .....	354
Configuring an Area Object to Save Alarm Counts as Historical Data .....	355
Configuring Application Objects to Save Historical Data .....	355
<b>Working with Alarms and Events .....</b>	<b>359</b>
Understanding Events .....	359
Types of Events .....	359
Understanding Alarms .....	360
Types of Alarms .....	362
State Alarms .....	362
Limit Alarms .....	363
Target Deviation Alarms .....	364
Rate of Change Alarms .....	365
Statistical Alarms.....	365
Setting Alarm State with Object Attributes .....	366
AlarmModeCmd Attribute .....	366
AlarmInhibit Attribute .....	366
AlarmMode Attribute .....	366
_AlarmModeEnum Attribute .....	367
Alarms and Buffered Data .....	367
Setting Alarm State for Individual Alarms .....	367
Enabling, Silencing, and Disabling Alarms .....	368
Enabling Alarms .....	369
Silencing Alarms.....	369
Disabling Alarms .....	369
Shelving Alarms .....	369
Enabling Alarm Shelving .....	371
Shelving Alarms at Run Time .....	372
Throttling Alarms .....	373
Propagating Timestamps with Alarms and Events .....	373
Alarms or Events Become Active .....	373
Alarms Become Disabled .....	373
Alarms Revert to Normal .....	374
Alarm Acknowledgement .....	374
Acknowledging Alarms with Signature Required .....	374
Configuring Alarms.....	374
Configuring Alarming for System Objects .....	375
Configuring WinPlatform Object Alarms .....	376
Configuring Communication Failure Alarm Priority .....	376
Selecting the Register using "Galaxy_<GalaxyName>" instead of "Galaxy" option .....	376
Configuring Alarms for an AppEngine Object.....	378
Configuring Alarms and Events for Application Objects .....	378
Setting Alarms on the Attributes Page.....	381

Distributing Alarms and Events .....	382
Subscribing to Alarms and Events from a Client .....	383
Using InTouch HMI as the Alarm and Event Client.....	384
Understanding the Syntax of Alarm Queries .....	384
Alarm Query Syntax when Register Using Galaxy_<GalaxyName> is Enabled .....	384
Examples of Alarm Queries.....	385
Alarm Requirements for InTouch Client Applications .....	385
Alarms and Events in InTouch HMI and in Application Server.....	386
Configuring Plant State-Based Alarms.....	387
Mapping Alarm Modes to Plant States .....	388
Configuring State-Based Alarming on an Area Object.....	390
Viewing Plant State-Based Alarms at Run Time .....	390
User Access and Security During Run Time .....	390
Configuring Alarm Historization, Mapping, and Shelving Priority Ranges .....	391
Configuring Historization for Alarms and Events .....	392
Monitoring Alarm Severities at Run Time .....	393
Understanding How Alarms are Ranked at Run Time .....	393
Using Aggregated Alarm Information .....	393
Configuring Alarm State Aggregation .....	394
Aggregating Alarm State Information .....	394
About the AlarmCntsBySeverity Attribute .....	396
Local Alarm Display Display in the Object Viewer .....	396
Aggregated Alarm Display in AVEVA OMI Navigation .....	397
About Alarm Adorner Attributes .....	399
Monitoring Alarm State Information at Run Time.....	400
<b>Working with Multiple GR Nodes and Galaxies.....</b>	<b>403</b>
Understanding the Multi-Galaxy Environment.....	403
Setting up Multi-Galaxy Communication .....	403
Naming Galaxies in a Multi-Galaxy Environment .....	406
Unpairing Galaxies .....	406
Renaming Paired Nodes.....	406
Renaming Galaxies .....	406
Accessing Multiple Galaxies .....	407
Using the Galaxy Browser with Multiple Galaxies .....	407
Using Object Viewer with Multiple Galaxies.....	408
Using InTouch with Multiple Galaxies .....	409
Guidelines for Accessing Multiple Galaxies .....	411
Working with Security in a Multi-Galaxy Environment .....	411
Pairing Galaxies with Multiple Network Interface Cards .....	412
Working with IDE Extensions on a GR Node .....	412
Working with Alarms .....	413
Enhancing Multi-Galaxy Performance.....	414
Remedying Performance Issues After the Fact .....	414
Scaling the Multi-Galaxy Environment for Optimum Performance .....	414
Setting Remote Galaxy References in the Owing Object Property .....	415
Working with ArchestrA Services.....	417
About the ArchestrA Service Bus.....	417
Defining Service Discovery .....	417
Defining User-Configurable ASB Services.....	417

Configuring and Deploying ArcestrA Services.....	418
Creating an Instance of an ASB Service.....	419
Assigning the Service Instance to a Node .....	419
Deploying the Instance .....	419
Configuring ArcestrA Service TCP Ports .....	419
Configuring the ASBGRBrowsing Service .....	421
Configuring the ASBMxDataProvider Service .....	421
Configuring the ASBAuthentication Service.....	422
IOMBLSService .....	423
Configuring and Deploying the OPC UA Service .....	423
Client Access Rules and Galaxy Security .....	425
Managing Galaxies in a Multi-Galaxy Environment .....	426
Uninstalling and Reinstalling Application Server .....	427
Backing Up a Galaxy .....	427
Deleting a Galaxy .....	427
Restoring a Galaxy .....	427
Restoring to an Existing Galaxy .....	428
Restoring to a New Galaxy .....	428
Upgrading a Galaxy.....	428
Migrating a Galaxy .....	428
Upgrading SQL Server .....	428
Troubleshooting a Remote Galaxy Connection .....	429
Connecting to a Remote GR from the IDE .....	435
Fixing Communication Issues with the OS Configuration Utility .....	436
<b>Working with Buffered Data .....</b>	<b>439</b>
About Buffered Data.....	439
Components that Use Buffered Data .....	439
Configuring Buffered Data for an Attribute.....	440
Using Object Viewer with Buffered Data .....	441
Using ArcestrA Scripts to Process Buffered Data .....	443
Scripting Basic Functions.....	443
Sample Script and Output .....	443
Scripting Tips and Best Practices .....	444
Use Asynchronous Scripts for Buffered Data .....	444
Buffered Data Script Syntax and Configuration .....	444
Using BindTo for Buffered Data .....	445
Buffered Data Run-time Behavior.....	446
About Buffered Data and History .....	447
About Buffered Data and Alarms and Events .....	447
AppEngine Reconnect Configuration for Undeploying and Redeploying Objects Linked to the Telemetry Server.....	448
Alarm Functions and Buffered Data .....	449
Alarm and Event Types and Buffered Data .....	450
<b>Working with References .....</b>	<b>453</b>
Using Message Exchange and Attributes .....	453
Reference Strings .....	453
Relative References .....	454

Property References.....	454
Handling Time Zones with the Time Property .....	455
Preserving Time Stamps from the Publishing Source .....	456
Arrays .....	456
Formatting Reference Strings.....	456
Using Literals .....	457
Viewing Attributes in Objects .....	459
Viewing References and Cross References .....	460
Finding Objects.....	461
Using Galaxy References in InTouch.....	462
Telemetry Server Data References .....	464
Telemetry Server Dynamic Tags .....	465
Dynamic Data References vs. Static Data References .....	465
Dynamic Tag Prerequisites .....	465
Naming Rules .....	466
Configuration Errors.....	466
DNP3 Data References.....	467
IEC 60870-5 Dynamic Data References.....	472
Modbus Data References.....	475
Telemetry Server Static Data References .....	478
<b>Configuring Security.....</b>	<b>481</b>
About Security .....	481
Security Model for Galaxies .....	481
About Authentication Modes .....	483
Multiple Accounts Per User .....	484
Changing Security Settings .....	484
About Security Groups.....	484
About Roles .....	485
About Users.....	488
About Credentials.....	489
Create Named Credentials .....	489
Associate an OS Group with a Credential.....	491
Configure Credentials .....	493
About SQL Server Security .....	494
Configuring Security.....	494
Assigning Users to Roles .....	500
Deleting Security Groups .....	502
Deleting Roles .....	502
Deleting Users .....	502
About OS Group-based Security .....	503
Connecting to a Remote Node for the First Time .....	503
Cached Data at Log In .....	503
Mixed or Native Domains .....	503
Using Domain Local Groups.....	504
Using Security and Distribution Domain Configurations .....	504
Using InTouch Access Levels Security .....	504
Using Secured and Verified Writes .....	504
Using Configurable Descriptions and Comments for Secured and Verified Writes .....	505
About Secured and Verified Writes Logged Information.....	506

Working with Languages .....	507
Defining and Configuring Galaxy Languages .....	507
Graphics Language Switching .....	507
Alarm Comment Language Switching .....	507
Workflow .....	507
Configuring Languages for a Galaxy .....	508
Adding a Language to a Galaxy .....	508
Removing a Language from a Galaxy .....	509
Modifying the Font for a Language .....	510
Changing the Default Language for a Galaxy .....	510
Exporting Symbol Text for Offline Translation .....	511
Types of Language Dictionary Files .....	511
Exporting Language Data for All Symbols in a Galaxy .....	512
Exporting Language Data for Specific Objects .....	512
Exporting Symbol Language Data for a Managed InTouch Application .....	514
Exporting Symbol Language Data for a Published InTouch Application .....	514
Exporting Symbol Text to an Existing Dictionary File .....	515
Translating Exported Symbol Language Files .....	515
Translating Exported Symbol Text Dictionary Files .....	515
Importing Translated Symbol Language Files .....	516
Importing Translated Symbol Dictionary Files .....	517
Examples of Symbol or Object Mismatch Handling during Language Imports .....	518
Language Data Handling for Galaxy Operations .....	520
Exporting Alarm Comments for Offline Translation .....	521
Guidelines and Recommendations .....	521
Organizing Your Export .....	521
Translation File Formatting and Editing .....	521
Reimporting .....	521
About the Alarm Comments Language File .....	521
Exporting Alarm Comments from Very Large Galaxies .....	522
Exporting All Galaxy Alarm Comments .....	522
Exporting Alarm Comments by Area .....	523
Using File Names .....	523
Exporting Objects Not Assigned to an Area .....	523
Translating Exported Alarm Comment Language Files .....	524
Importing Translated Alarm Comment Language Files .....	526
Re-exporting Alarm Comments .....	527
Exporting New Untranslated Alarm Comments .....	527
Exporting Modified Existing Alarm Comments .....	527
Testing the Language Switching Functionality at Run Time .....	527
Managing Galaxies .....	529
Backing Up and Restoring Galaxies .....	529
Changing Galaxies .....	530
Deleting a Galaxy .....	531
Galaxy Object Components Synchronization .....	531

---

Definitions.....	532
Typical Out-of-Sync Scenarios .....	532
Using the Synchronization Feature .....	532
Exporting a Galaxy Dump File.....	532
Editing the Galaxy Dump File.....	533
About the Galaxy Dump File Structure .....	533
Host Attributes.....	534
About Quotation Marks and Carriage Returns .....	534
Time Formats in Excel .....	535
Enabling I/O Auto Assignment in the Galaxy Dump File .....	535
Importing a Galaxy Load File .....	536
Hosting Multiple Galaxies in One Galaxy Repository .....	537
Disk Space Requirements .....	537
Managing Communication between Galaxy Nodes .....	537
Mapping Network Drives with User Account Control Enabled.....	537
About ArcestrA User Accounts .....	538
Using Multiple Network Interface Cards .....	539
Defining the Order of the NIC .....	539
Configuring the IP Address and DNS Settings .....	539
Configuring Multiple NICs .....	540
<b>Working with Redundancy .....</b>	<b>545</b>
About Redundancy.....	545
Configuring AppEngine Redundancy .....	545
Redundancy during Run Time .....	546
Engine Restart After Failure .....	546
CPU Load Balancing .....	547
Working with AppEngine Redundancy .....	547
Configuring the Redundancy Message Channel .....	548
Configuring Redundancy .....	549
Configuring Redundancy in Templates .....	550
Deleting Redundant AppEngines .....	550
Deploying AppEngine Objects .....	550
Configuration Requirements.....	551
Undeploying AppEngine Objects .....	551
During Deployment.....	552
Objects at Run Time .....	552
During Run Time .....	552
AppEngine Redundancy States .....	552
Troubleshooting .....	554
Generating Alarms .....	554
Generating History .....	555
Working with Data Acquisition Redundancy .....	556
Configuring Data Acquisition Redundancy .....	556
Deploying Redundant DIOObjects .....	556
What Happens in Run Time .....	557
RedundantDIOObject and PLC Connectivity .....	557



<b>Managing Security for Application Server .....</b>	<b>559</b>
General Considerations for Security .....	559
Introduction.....	559
Securing the Host .....	560
General Guidelines for Securing the Host .....	560
Windows Updates .....	561
Scanning the Host.....	561
Protecting the Applications and Content on the Host.....	562
Securing the Network .....	562
Segmenting the ICS Network .....	562
Managing Network Services and Ports .....	563
Securing Communication between the Client and Server .....	564
Securing Systems through Authentication and Authorization .....	565
Managing Users and Groups through ICS Software.....	565
Managing Users and Groups through Windows.....	566
Contingency Planning .....	566
Business Continuity Planning.....	567
Disaster Recovery Planning.....	567
Security Configuration for InTouch HMI .....	568
Security Configuration for AVEVA Application Server .....	569
<b>Configuring a Multi-User Development Environment .....</b>	<b>571</b>
Best Practice Recommendations .....	571
Development System Architecture .....	571
Local GR Nodes plus Shared GR (Recommended) .....	572
Create a Multi-User Development System.....	573
Development Process.....	575
Production Galaxy Updates.....	575
Benefits of Using a Distributed Multi-User Development System.....	575
Shared GR with Local IDE Nodes (Alternative Architecture) .....	575
Shared GR and IDE Node (Alternative Architecture) .....	576
<b>Glossary.....</b>	<b>579</b>
<b>Index.....</b>	<b>587</b>



# CHAPTER 1

## IDE Overview

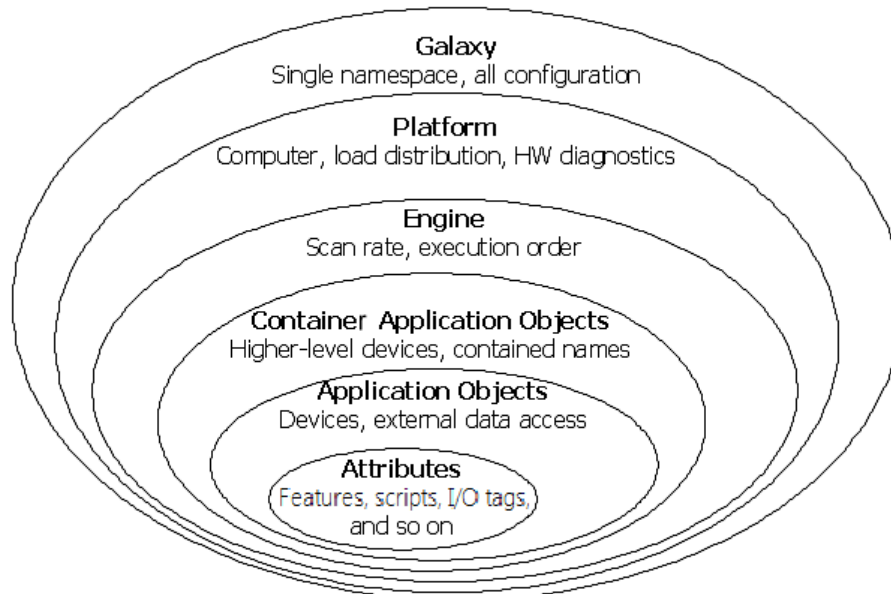
The IDE is the integrated design and development tool for the AVEVA™ Application Server, formerly Wonderware.

You work from the IDE to create, configure, and maintain the objects that comprise your application and the underlying infrastructure that supports your application.

From the IDE, you can import new types of objects in to the Galaxy Repository, configure new ones, and deploy them to computers in your network. Multiple users can work concurrently on different sets of objects from different System Platform IDEs.

## What's a Galaxy?

Within Application Server, a Galaxy represents your entire production environment, including all computers and components that run your application. A Galaxy is a collection of platforms, engines, templates, instances, and attributes you define as the parts of your specific application. Persistent information about this collection of objects is stored in a Galaxy database.



A Galaxy database resides on a single network computer. A Galaxy database can reside on any computer on your network with the SQL Server, Bootstrap, and Galaxy Repository software installed. But, you cannot store parts of a Galaxy database on several computers.

A Galaxy Repository (GR) is the name of the single computer where the Galaxy database is located.

You can deploy Galaxy components, such as platforms and engines, on multiple computers to share the work load while applications are running. For more information, see *Deploying and Running an Application* on page 317.

A Galaxy's namespace is the set of unique object and attribute identifiers. The namespace and the values of each of its identifiers define an Application Server application, and can be accessed by clients of the configuration system as well as the Application Server Message Exchange in a deployed system.

A key benefit of the Application Server namespace is that it allows Application Server objects and process data to be referenced by scripts and animation links from any computer in the Galaxy without the reference needing to specify the object's location.

Galaxies also include security, which is turned off by default. Using security allows you to limit what users can do. You can add more users, security roles, and security groups later if you want. For more information, see *Configuring Security* on page 481.

## Start the System Platform IDE

When you start the IDE, you must select an existing Galaxy or create a new Galaxy. You cannot open the IDE without opening a Galaxy.

## Install and Activate Your License

Before you can open the IDE, you must have a valid license. For information about licensing, see "License Installation and Activation" in the *AVEVA System Platform Installation Guide*. If you need additional information, see the *AVEVA Enterprise Licensing Guide*.

If the Galaxy Repository process (aaGR) cannot obtain a license for any reason, it will send requests periodically to the license server. These attempts to acquire a license will continue until either a license is acquired successfully, or the aaGR process is stopped. This retry process also applies if access to a license is lost after it was acquired.

The IDE supports both perpetual and Flex licensing. See *Flex Licenses View* on page 33 and *Allocating AppEngines to Platforms* on page 321 for information about using Flex licenses.

## Manage Security Certificate Verification

The System Platform IDE may take 30 seconds to open on a system that is not connected to the Internet. When the IDE starts, the operating system attempts to verify the digital certificates for internal components against a Certificate Revocation List (CRL) located on a public website. If your system cannot access the public site within 30 seconds, the IDE startup process resumes and completes.

To maintain a better security profile by checking components against a CRL, keep your internet connection enabled and allow the verification to proceed.

To avoid the potential delay, in **Internet Explorer** or through **Control Panel**, open **Internet Options**, **Advanced Options**, and uncheck the **Security** option to **Check for publisher's certificate revocation**.

## Create a Galaxy

Each time you start the System Platform IDE, you must connect to a Galaxy. You can either create a new Galaxy or select an existing Galaxy before opening the IDE. You must have activated a valid license before opening the IDE.

Creating a new Galaxy requires you to specify a Galaxy Repository (GR) node name and the name of the Galaxy. The Galaxy database is created and is ready for you to connect to and use.

### Guidelines for Creating a New Galaxy

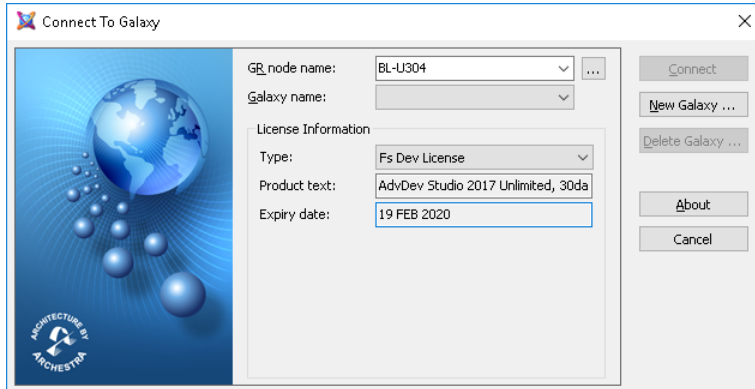
- Activate a valid license before opening the IDE.
- You can only create a new Galaxy on a computer with the Bootstrap and the Galaxy Repository software installed.
- To be able to create a new Galaxy, TCP/IP must be enabled on the computer hosting the SQL Server database.

The TCP/IP protocol setting can be verified with **SQL Server Configuration Manager**. See the *System Platform Installation Guide* for instructions on enabling the TCP/IP protocol for SQL Server.

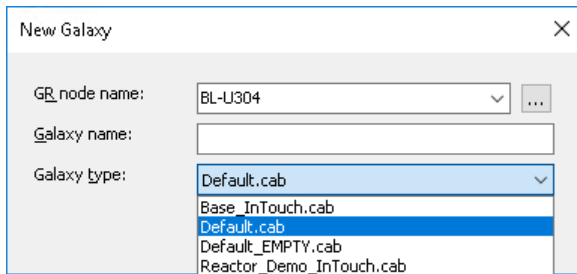
- New Galaxies are created without security. To learn more about setting security for your Galaxy, see *Configuring Security* on page 481.
- A computer name, as well as the name of the domain on which the computer exists, should include only English language characters, alphabetic and numeric. Use of characters from other language character sets that are not also included in the English language character set can result in a failure to create the new Galaxy.


**To create a new Galaxy**

1. From the **Start** menu, go to the **AVEVA System Platform** folder, and then select **System Platform IDE**. The **Connect to Galaxy** dialog box appears.



2. Click **New Galaxy**.



3. Specify the properties of your new Galaxy:
  - In the **GR Node Name** list, type, or select the name of a computer that has the Galaxy Repository software installed. If necessary, click the **Browse** button  to locate the node where the Galaxy Repository is installed.
  - In the **Galaxy Name** box, type the name of the Galaxy you want to create within that Galaxy Repository. A Galaxy name can be up to 32 alphanumeric characters, including **\_** (underscore), **\$**, and **#**. The first character must be a letter. A Galaxy name cannot contain a blank space.

---

**Note:** You cannot use the following reserved names as Galaxy names: Me, MyContainer, MyArea, MyHost, MyPlatform, MyEngine and System. You cannot use a name that conflicts with an existing object in the backup Galaxy file.

---

- In the **Galaxy Type** list, select the galaxy type from the available standard galaxy cab files, which are located in the BackupGalaxies folder. The following cab files are included with Application Server:
 

**Base\_InTouch.cab:** Use the Base\_InTouch cab file to create InTouch HMI Galaxies. This is mainly for use if you are already using InTouch HMI applications, and can be used to convert legacy InTouch applications to managed applications.

**Default.cab:** The Default cab file is the *recommended* starting point for creating a new Galaxy. The Default.cab file contains default screen profiles, templates, a basic equipment model, and an example AVEVA OMI ViewApp that provides an introduction to key features.

**Default\_EMPTY.cab:** Use the Default\_EMPTY.cab file to create a baseline Galaxy that contains only base template objects. It is intended primarily as a recovery mechanism for recreating a damaged galaxy. After creating the baseline galaxy, you can import existing objects into it from aaPKG files.

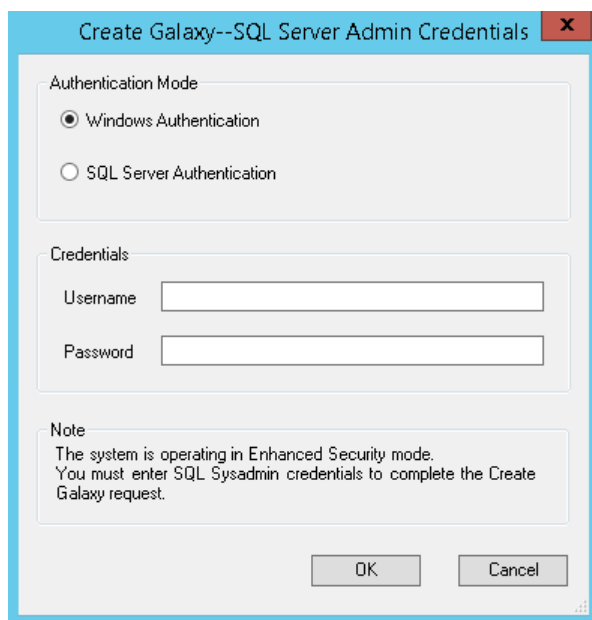
**Reactor\_Demo\_InTouch.cab:** This is the same as the Base\_InTouch.cab file, but includes a demonstration run-time application, "\$ReactDemo," and all the configuration objects and graphic files used to create the demo. \$ReactDemo is derived from the InTouchViewApp object.

- If you have created Galaxy backups, your cab files will also appear, in addition to the cab files included with Application Server.

The selected backup file is used as a template to create a new Galaxy. The system restores the selected backup Galaxy and renames it to the Galaxy name that you provided in the Galaxy Name box.

4. Click **Create**. The **Create Galaxy** dialog box opens, showing the Galaxy database being created.

**Note:** If you use a backup cab file that was created in a previous version of Application Server, and if Enhanced Security mode is in effect, you will be prompted to enter SQL SysAdmin credentials before proceeding.



5. After the Galaxy database is created, click **Close**. You are ready to open the Galaxy and begin creating your application. You must be a member of the aaConfigTools OS group to connect to a Galaxy, or you must be running the IDE as administrator. For more information about opening a Galaxy, see *Connect to a Galaxy* on page 22.

## Connect to a Galaxy

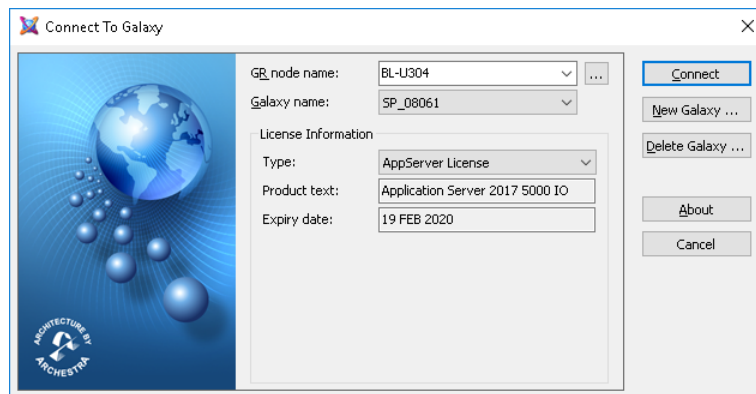
You must connect to a Galaxy before you can perform work in it. To connect to a Galaxy, you must be a member of the **aaConfigTools** OS group *on the GR node*. To add a user to the aaConfigTools group, use the Windows Control Panel. If you are not a member of aaConfigTools, you can instead start the IDE as administrator and then connect to the Galaxy.


**Important:** In a domain-based network, the **aaConfigTools** OS group must exist on the GR node. You will not be able to connect to the Galaxy if you are not a member of the **aaConfigTools** group on the GR node.

If security is enabled for an existing Galaxy, you cannot open it without logging in, even if you are a system administrator or member of the **aaConfigTools** group. If you do not have log on rights to a Galaxy, you cannot log in to that Galaxy. For more information about Galaxy security, see [01Standardize\\_Security\\_Configure Security](#).

### To connect to a Galaxy

1. From the **Start** menu, go to the **AVEVA System Platform** folder, and then select **System Platform IDE**. The **Connect to Galaxy** dialog box appears.

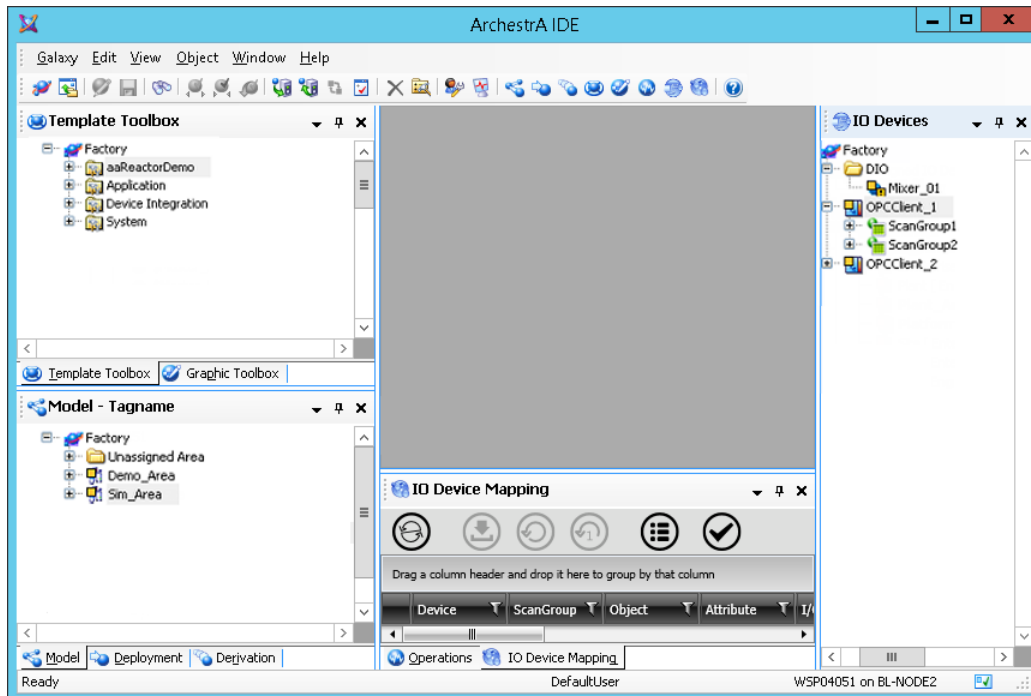


2. Do the following:
  - a. In the **GR node name** list, select the name of a computer you previously connected to. Click the **Browse** button  to browse and select from the available domains and nodes on your network.
  - b. In the **Galaxy name** list, select the name of the Galaxy on that GR node.
  - c. Click **Connect**.
    - If you are a member of the **aaConfigTools** OS group on the GR node or running the IDE as administrator, AND the Galaxy you select does NOT have security enable, you are connected to the Galaxy.
    - If you are a member of the **aaConfigTools** OS group or running the IDE as administrator, AND the Galaxy you select has security enabled, the **Login** dialog box appears. Type your user name and password and click **OK**.
    - If you are not a member of the **aaConfigTools** OS group or running the IDE as administrator, you are blocked from connecting to the Galaxy and a message about insufficient permissions is displayed. Take the corrective action described in the message and relaunch the IDE.

Once the Galaxy opens in the IDE you can start working with your Galaxy. The IDE provides different ways to view the structure of the Galaxy and how it will be deployed to run time. See [IDE Application Views](#) on page 28 for more information.

## IDE Views and Toolboxes

After you open a Galaxy, the IDE opens and shows the different views of your Galaxy.



For a complete discussion of items such as templates and instances, see *About Templates and Instances* on page 43.

Views in the IDE include:

- |                  |  |
|------------------|--|
| Template Toolbox | Expand the top level folders to see the different templates in the toolboxes.  |
| Graphic Toolbox  | Contains Industrial Graphics and Situational Awareness Library symbols that can be used in the Galaxy. For more information, see the <i>Creating and Managing Industrial Graphics User Guide</i> . |



Application views	<p>Click the tabs at the bottom or icons at the top to open:</p> <ul style="list-style-type: none"><li>• <b>Model view</b> - the object relationship to the automation scheme layout. The objects are organized into Areas that typically represent the physical plant layout.</li><li>• <b>Deployment view</b> - the object relationship to the computers that comprise the deployed system that the objects run on.</li><li>• <b>Derivation view</b> - the derivation path from base template to the instances. This view allows a user to see all object instances that were based on a given template. All templates and instances appear in this view.</li><li>• <b>IO Devices view</b> - the object relationship to scan groups and DI objects. This view lets you assign objects that were configured for I/O automatic assignment in the Object Editor to scan groups, and assign scan groups to Device Integration (DI) objects. The IDE then builds the I/O references for each object.</li><li>• <b>IO Device Mapping view</b> - shows the relationships of objects and attributes to scan groups and DI objects. This view lets you see, validate, and edit the I/O references of object attributes that have been automatically configured.</li><li>• <b>Operations view</b> - shows the results of validating the configuration of objects.</li></ul>
Status bar	<p>Shows messages, user name, Galaxy name and node, and license information. Turn off the Status bar by clicking <b>Status</b> bar on the <b>View</b> menu.</p>

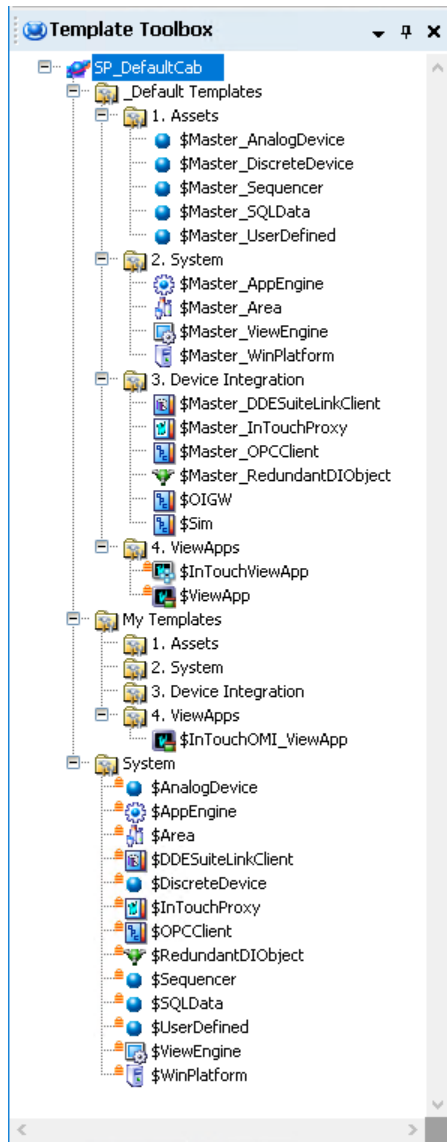
## Template Toolbox

The Template Toolbox lists template toolsets, which contain object templates. The Template Toolbox shows a tree view of template categories in the Galaxy. Expand the toolsets to see their contents.

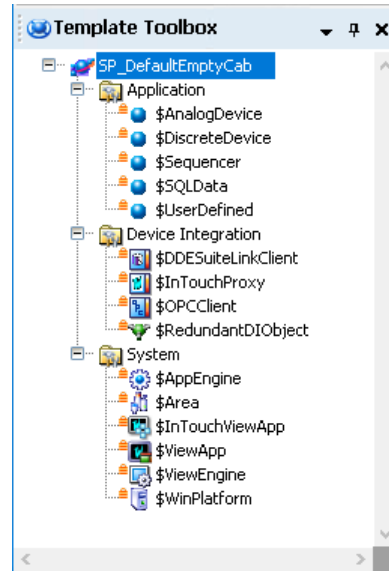
A new Galaxy is automatically populated with the content defined by the cab file that was used to create it. At a minimum, the newly-created Galaxy will include base templates.

- The Default.cab includes first level derived templates in addition to base templates, to simplify the process of building your Galaxy and a sample AVEVA OMI ViewApp.
- The Default\_EMPTY.cab includes only base templates. See *About Base Templates* on page 45 for additional information.
- The Base\_InTouch.cab and the Reactor\_Demo\_InTouch.cab files include only the base templates needed to support an InTouch HMI application.

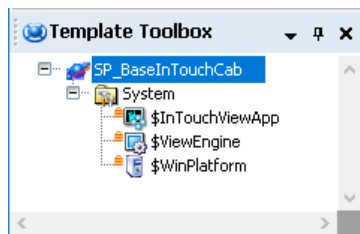
**Default.cab Template Toolbox**



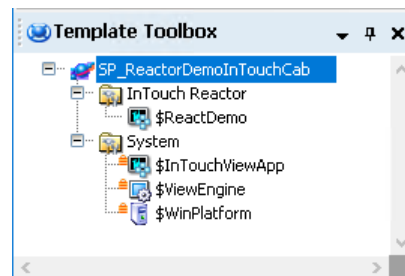
**Default\_EMPTY.cab Template Toolbox**



**Base\_InTouch.cab Template Toolbox**



**Reactor\_Demo\_InTouch.cab Template Toolbox**

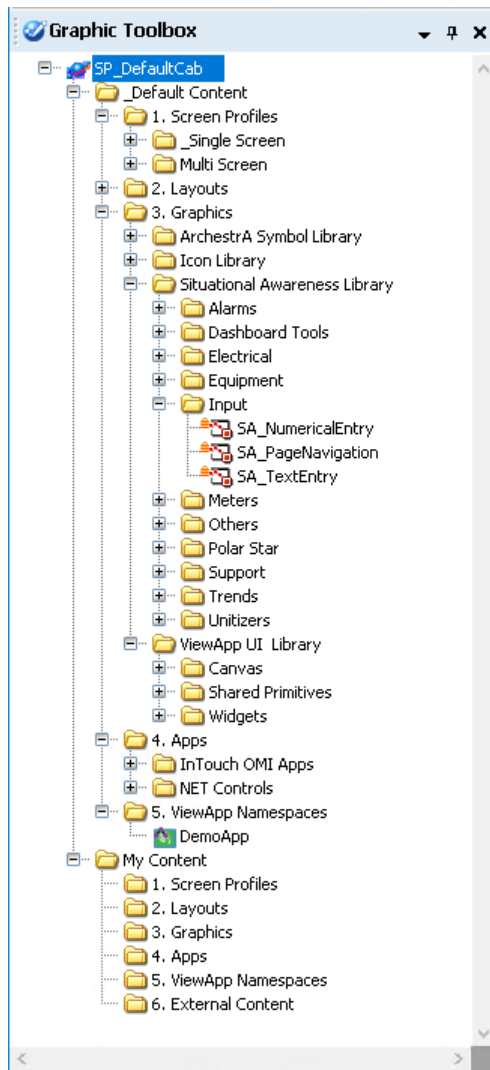


## Graphics Toolbox

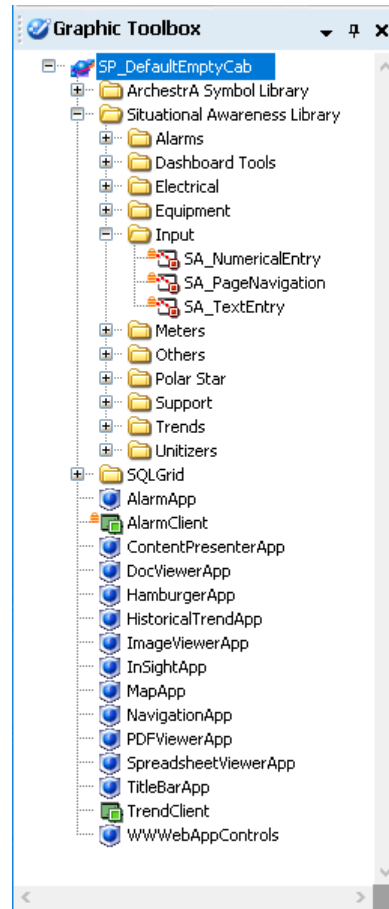
The Graphic Toolbox lists symbols and other graphic-related content. The Graphic Toolbox shows a tree view of the symbol/graphic toolsets. As with the Template Toolbox, new Galaxies are populated with the content defined by the cab file used to create it. Expand the toolsets to see their contents.

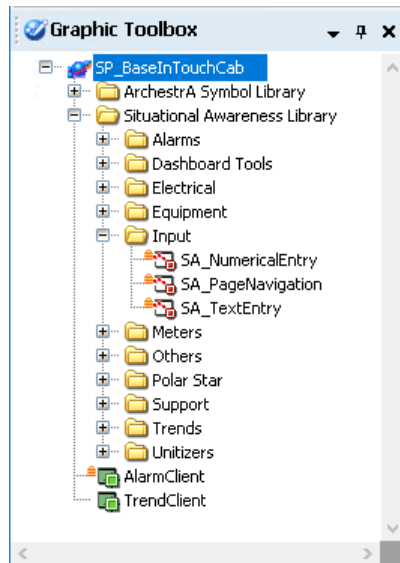
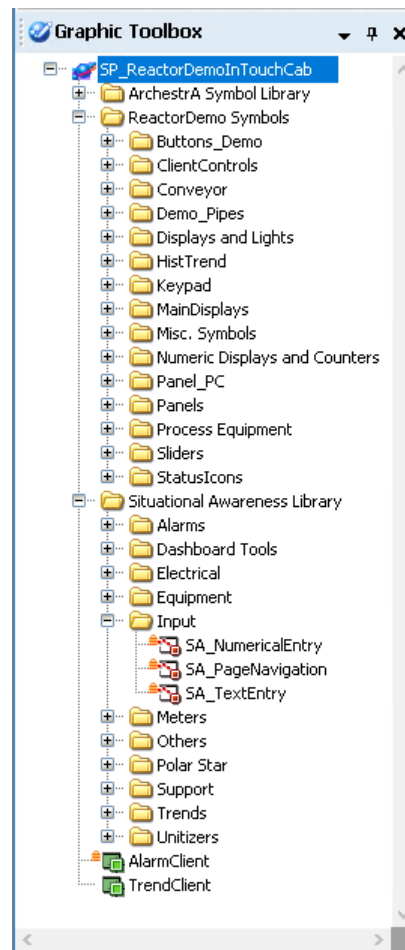
- The Default.cab includes several symbol and graphic libraries, including the Industrial Graphics Library, the Situational Awareness Library, and an icon library. The Default.cab also includes various Screen Profiles, Layouts, Apps and Controls.
- The Default\_EMPTY.cab includes only a basic set of Apps and Controls.
- The Base\_InTouch.cab and the Reactor\_Demo\_InTouch.cab include the Industrial Graphics and Situational Awareness Libraries. The Reactor\_Demo-InTouch.cab also includes additional symbols used for the InTouch HMI reactor demo app.

**Default.cab Graphic Toolbox**



**Default\_EMPTY.cab Graphic Toolbox**



**Base\_InTouch.cab Graphic Toolbox****Reactor\_Demo\_InTouch.cab Graphic Toolbox**

## AVEVA Connect

AVEVA Connect is a cloud-based symbol repository that you can add to the Graphic Toolbox. Adding AVEVA Connect to the Graphic Toolbox provides a centralized storage and access point for reusable Industrial Graphics, which you can use as templates to simplify building applications. For more information, see *Using AVEVA Connect* on page 52.

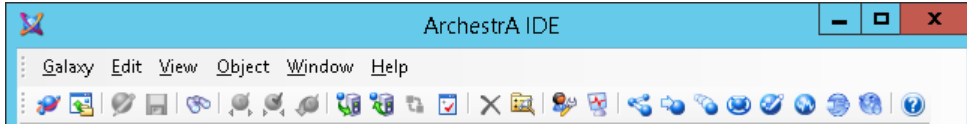
## IDE Application Views

Base templates and non-base templates are included with Application Server. The templates are automatically imported into the IDE when you first create a Galaxy.

Base templates appear in the Template Toolbox and in the Derivation view with a \$ as the first character of their name. You cannot directly modify base templates but you can use them to create your own objects or derived templates.

When you move from one view to another, the selected object in the first view is selected in the second view, if the object exists in that view. For example, templates are not shown in the Deployment view, Model view and IO Devices view.

You can open views from the **View** menu, or click the icons in the main toolbar. Keyboard shortcuts can also be used to open the views.








View	Shortcut	Icon
Model View	Ctrl+Shift+M	
Deployment View	Ctrl+Shift+D	
Derivation View	Ctrl+Shift+R	
Template Toolbox	Ctrl+Shift+T	
Graphics Toolbox	Ctrl+Shift+P	
Operations View	Ctrl+Shift+O	
IO Devices View	Ctrl+Shift+I	
IO Device Mapping view	Ctrl+Shift+G	











## Object Icons

In the Model, Deployment, Derivation, and IO Devices views, objects are listed in a tree structure that functions like a standard Microsoft Windows Explorer tree. Under each branch of the tree, objects are listed in alphabetical order. Default objects are shown in bold.

For objects shown in any view, icons to the left of the object name are used to show:

- Object deployment status
- Error and warning conditions
- If the object is checked out
- If a template contains an Object Wizard

-  Not deployed
-  Deployed
-  Deployed, but pending configuration changes exist that have not been deployed.
-  Deployed, but software modifications exist that have not been deployed.
-  Applies only to redundant AppEngines. An AppEngine is undeployed, but its redundant pair is deployed.

-  Applies only to redundant AppEngines. An AppEngine is deployed, but its redundant pair is not deployed.
-  Applies only to redundant AppEngines. An AppEngine is deployed, its redundant pair is not deployed, and pending configuration changes exist that have not been deployed.
-  Applies only to redundant AppEngines. An AppEngine is deployed, its redundant pair is not deployed, and software modifications exist that have not been deployed.
- [No icon] Good. No additional icon is displayed for deployed objects running on-scan or off-scan with a good status.
-  Warning
-  Error. The object is in an Error state and cannot be deployed.
-  The process of asynchronously transferring files to the target node is not yet complete. This icon is normally visible for only a few moments at the end of a deployment operation, unless the object is deployed on a slow network. If the icon remains for more than a few moments, check if the application is OnScan, as the files will not transfer if the application is OffScan. This icon completely replaces the original while it is shown.
-  The object is a base template (read-only).
-  The object is checked out.
-  The template contains an Object Wizard.
-  To assign an object to another, drag it onto the host object. If that object is an inappropriate assignment match, the international Not symbol appears. To unassign an object, drag it to the **Unassigned Host** folder.

---

**Note:** You must undeploy an object that is currently deployed before reassigning it from one object to another.

---

## Model View

The Model view shows objects in terms of their physical or containment relationships, which can be organized through a folder structure. This Model view most accurately represents an application perspective of the processes that users are emulating: for instance, specific process areas, tanks, valves, pumps and their relationships based on containment.

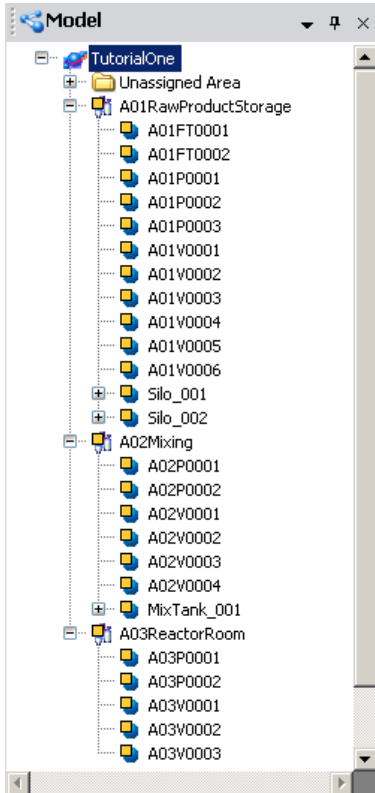
For more information about containment, see *Create Contained Templates* on page 58.

---

**Note:** You must undeploy an object before reassigning it to another object.

---

The tree structure acts like a standard Microsoft Windows Explorer tree. Initially, it shows a simple hierarchy: <Galaxy name> and the Unassigned Area folder.



In the Model view, all objects are grouped by areas and by containment relationship. The Model view shows these relationships in the following ways:

- The top of the tree is the Galaxy.
- Top-level Areas are shown under the Galaxy.
- Within each Area, contained Areas are listed. Areas support hierarchical composition; that is, they support sub-Area construction. Areas can be nested only 10 levels (after the sub-area is 10-levels deep, you cannot add another sub-level).
- Objects that belong to an Area are listed under the Area.
- Objects contained by other objects are listed under their respective containers. Multiple levels are allowed. For more information about containment, see *Create Contained Templates* on page 58.

---

**Note:** Contained objects belong to the same Area as the object that contains them.

---

Some object's hierarchical, or contained, names are truncated if you have multiple levels shown. To view the entire hierarchical name, select the object and click **Properties** on the **Galaxy** menu. The entire hierarchical name is shown in the **Properties** dialog box. For more information about hierarchical names, see *Using Contained Names* on page 64.

- Objects that currently do not belong to an Area are listed under Unassigned Area. Containment relationships between parent and child objects are shown there.

In each branch of the tree, objects are listed in alphabetical order. Default objects are shown in bold.

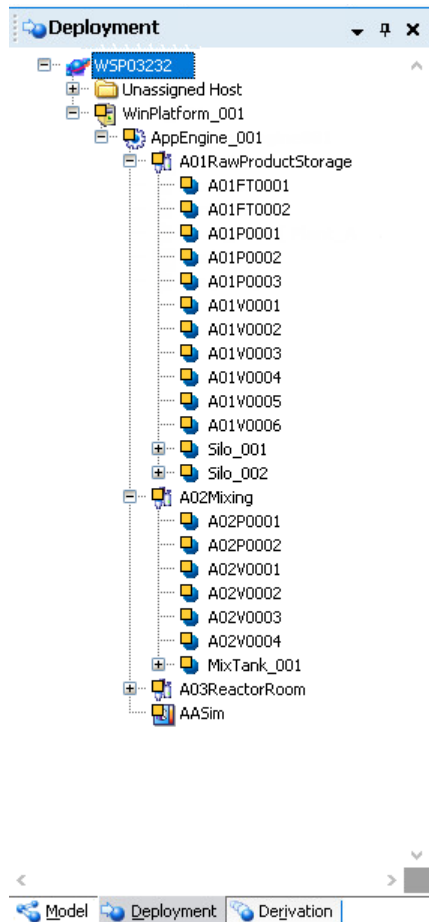


To assign an object to another, drag it onto the host object. If the object is an inappropriate assignment match, the international Not symbol appears. To unassign an object, drag it to the **Unassigned Host** folder.

## Deployment View

This view shows instances only in terms of their assignment relationships. This view enables you to organize those objects through a folder structure.

The tree structure acts like a standard Windows Explorer tree. It is initially divided into two hierarchical levels: <Galaxy Name> and the Unassigned Host folder.



In the Deployment view, objects appear in a tree according to their distribution relationships in a multi-node system in the following ways:

- The top of the tree is the Galaxy.
- WinPlatforms are shown under the Galaxy.
- Under each WinPlatform, assigned AppEngines are listed.
- Under each AppEngine, assigned Areas and DIObjects, such as DINetwork Objects, are listed.
- Under each Area, assigned ApplicationObjects are listed.
- Under each ApplicationObject, contained ApplicationObjects are listed. Multiple levels are allowed.
- Under each DINetwork Object, assigned DIDevice Objects are listed.
- Unassigned objects are grouped together in the **Unassigned Host** folder. Area and containment relationships are maintained in this view.



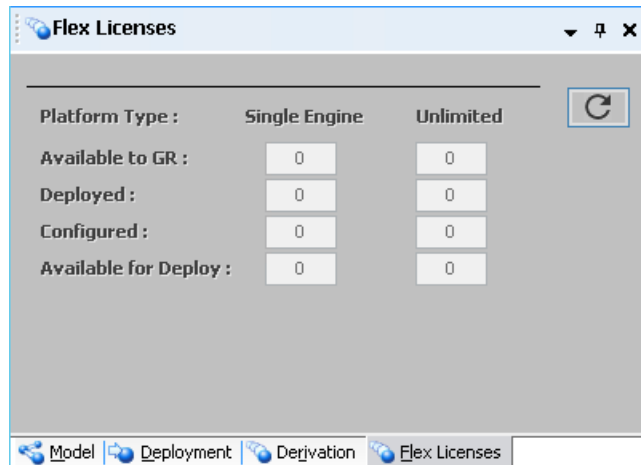
---

**Important:** DInetwork objects have specific configuration limits such as whether more than one object can be deployed to a single WinPlatform. The IDE does not check for these limits. For more information about configuration limits, see the online help for the DInetwork object.

---

## Flex Licenses View

Flex licenses are subscription-based licenses that can be used in place of traditional, perpetual licenses. When used with Application Server, Flex licenses are configured for Platform objects. There are three choices when configuring Flex licenses for a Platform object: None, Single Engine and Unlimited. See *Allocating AppEngines to Platforms* on page 321 for more information.



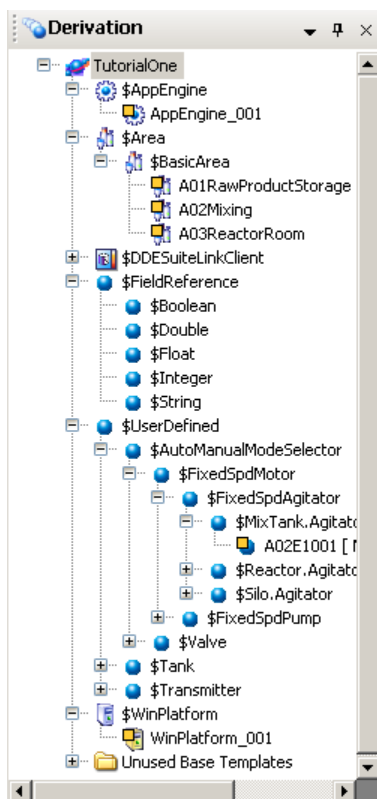
The Flex Licenses view shows how many licenses of each Platform type (Single Engine and Unlimited) are available for configuration and deployment. The count is automatically updated before and after deployment/undeployment, upon connection to the galaxy, and whenever platform assignments are changed. If you have a perpetual license, the Flex License view shows "0" for all licenses.

- Available to the GR: This is the total number of Single Engine and Unlimited licensed platforms available in the galaxy.
- Deployed: Licenses that have been deployed. You cannot deploy an AppEngine that does not have a license.
- Configured: The number of AppEngines that have been assigned to a platform. Note that you can configure an AppEngine without a license, but you cannot deploy it. The platform will show a configuration error if there are not enough licenses.

## Derivation View

The Derivation view shows objects and templates in terms of their parent/child relationship. An object derived from another object appears in a hierarchy level under it.

The tree structure acts like a standard Windows Explorer tree, and initially is divided into three hierarchical levels: <Galaxy Name>, <Used Base Templates>, and the Unused Base Templates folder.



In the Derivation view, objects appear according to their parent-child relationship in the following ways:

- The top of the tree is the Galaxy.
- Base templates with associated child objects, either derived templates or instances, are shown under the Galaxy.
- Under each base template, derived templates and instances created from the base template are listed. Multiple levels are allowed. Instances created from derived templates are listed under their parents.
- Templates with no associated derived templates or instances are grouped together in the **Unused Base Templates** folder.

Objects with names that start with a "\$" are templates, either base or derived. Under each branch of the tree, child objects are listed in alphabetical order.

As in other views, dragging one object onto another in the **Derivation view** associates the two objects based on the predefined rules of the object types. For example, you can drag ApplicationObjects onto other ApplicationObjects but you cannot drag ApplicationObjects to an Engine.

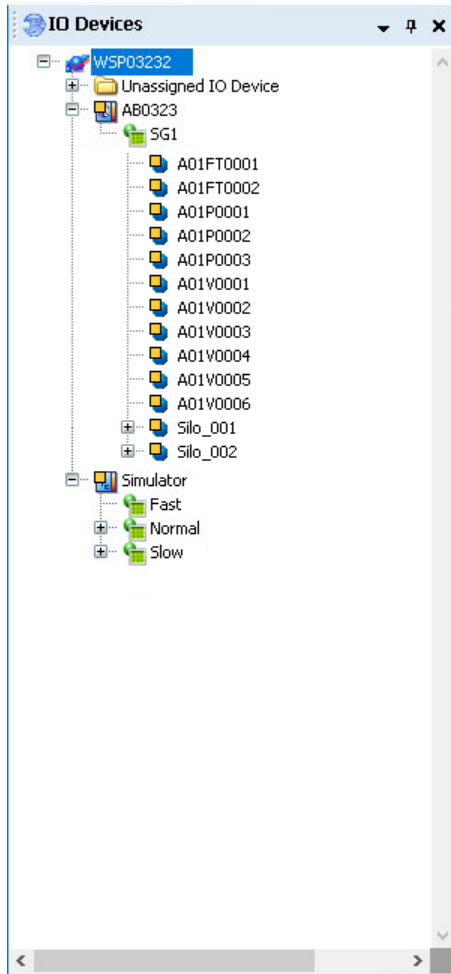
## IO Devices View

The **IO Devices** view is used for I/O auto assignment and shows the relationship of system objects and application objects to scan groups and Device Integration objects. Before using this view, system and application objects must be configured to use I/O auto assignment by adding one or more attributes to the object and activating the I/O feature. See *Edit Objects* on page 68 and *Using I/O Auto Assignment* on page 148 for additional information.

You will also use the Object Editor to add scan groups to DI objects (OPCCClient, DDESuiteLinkClient, or RedundantDIObject) before using the **IO Devices** view.

Once you have configured system and application objects for I/O auto assignment, and added scan groups to at least one DI object, you will assign the system and application objects to scan groups in the **IO Devices** view. This links (auto-binds) the system and application objects to PLCs, through the DI objects.

To open **IO Devices** view, click **IO Devices** on the **View** menu. You can also open the **IO Devices** view by pressing **Ctrl+ Shift+I**, or click the IO Devices icon in the main toolbar of the IDE.



System and application objects appear as child objects to scan groups, and scan groups appear as child objects to DI objects.

---

**Note:** Only instances, and not templates, appear in the **IO Devices** view.

---

Device linkage, which is established by assigning a system or application object to a scan group, ensures that all of the object’s attributes that have been prepared for automatic I/O assignment will match up with a corresponding input source or output destination in the scan group. The I/O references are then referred to as auto-bound.

---

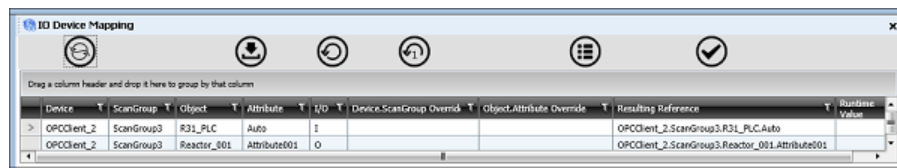
**Important:** Objects must be undeployed before they can be assigned to a scan group.

---

System and application objects that are not yet assigned to a scan group are placed under the folder "Unassigned IO Devices." To assign objects to a scan group, and thus allow I/O automatic assignment to occur, simply select objects in the Unassigned area, and drag and drop them to the applicable scan group. Once an object has been assigned to a scan group, the I/O reference for each I/O attribute is automatically configured.

## IO Device Mapping View

The **IO Device Mapping** view shows I/O references for attributes that have been configured for I/O auto assignment, and that have been assigned to a scan group through the **IO Devices** view. To open this view, click **IO Device Mapping** in the **View** menu. The **IO Device Mapping** view will also open if you select a DI object, scan group, or object assigned to a scan group in the **IO Devices** view. Other ways to open this view are by pressing **Ctrl+Shift+G**, or by clicking the IO Device Mapping icon in the main toolbar.



The **IO Device Mapping** view shows I/O auto assigned attributes with resolved I/O references for objects that have been assigned to scan groups. The **IO Device Mapping** view lets you enter override values to change the I/O reference. You can also validate the I/O references.

You select the references that are displayed in the **IO Device Mapping** view by selecting objects in the **IO Devices** view. Manually configured references are not displayed in the **IO Device Mapping** view, nor are attributes associated with objects that are not assigned to a scan group. Only I/O references for the object or group of objects currently selected in the **IO Devices** view are displayed. You can select DI objects, scan groups, and application and system objects in the **IO Devices** view. If only the top level item in a device hierarchy is selected, all subordinate items are automatically selected. If, however, you select a subordinate item along with the top level item, only I/O references for the selected items are displayed. You can select multiple items at different hierarchical levels.

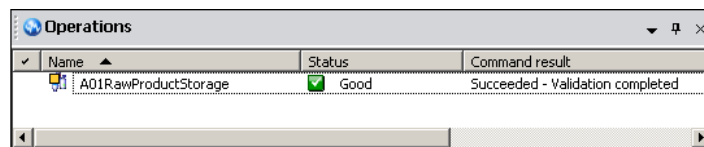
---

**Note:** Selecting a subordinate object will exclude non-selected objects within the device hierarchy, even if the parent object is selected.

---

## Operations View

The **Operations** view shows the results of validating the configuration of objects. You may need to open it before you see it. On the **View** menu, click **Operations**.



During the validation of an object, its icon and name appear with the status of the operation.

---

**Important:** You can validate both templates and instances if they are checked in.

---

The status of the object (**Status** column) is shown with an icon and a descriptive word or phrase.

When validation is complete, the **Command Result** column shows a "Succeeded" or "Failed" message and additional information about the validation results. For more information about validating objects, see *Validate Objects* on page 98.

---

**Note:** You can validate all objects in the Galaxy by running the Validate operation on the Galaxy object. In that case, Command Result messages are shown after all objects in the Galaxy are validated.

---

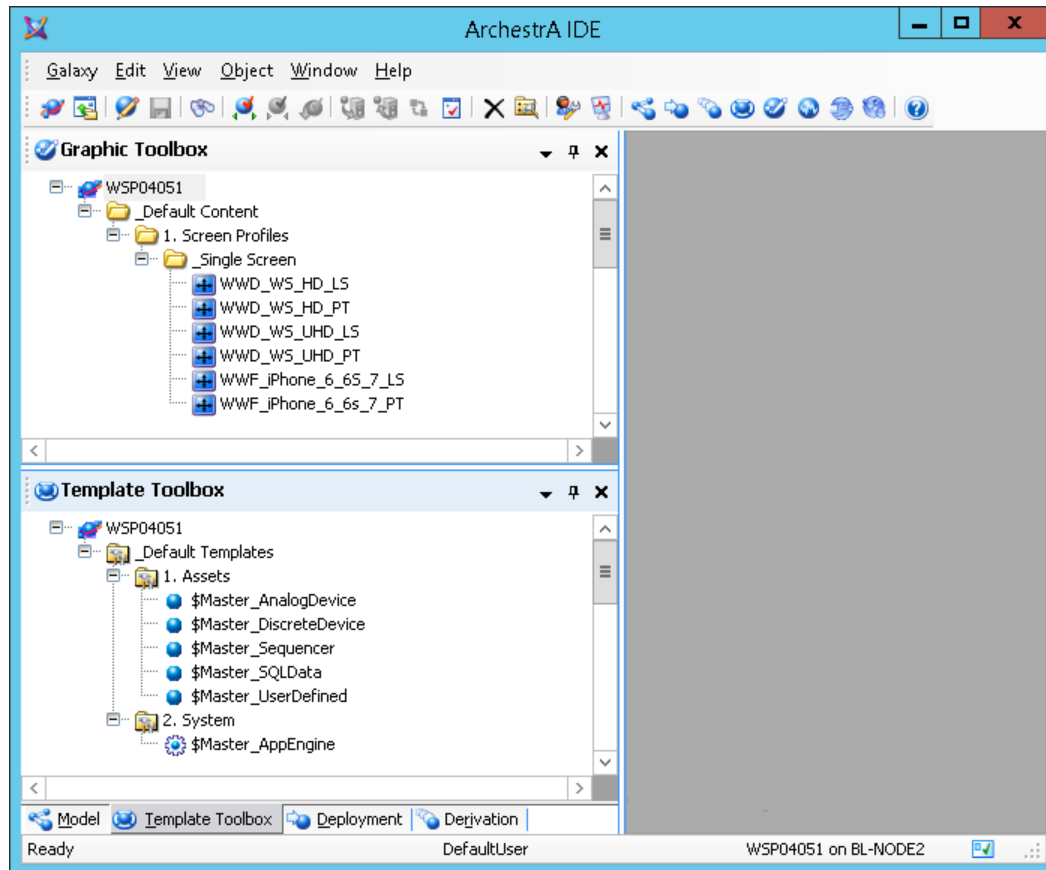
The **Operations** view, like the **Template Toolbox** and **Applications** views, is also updated as the status and conditions of objects in the Galaxy change.

## Customizing Your Workspace

You can customize your workspace by docking and floating the IDE's views. You can also hide some of the views.

### Dock Views

To dock a view, drag the view to the location you want it. For example, drag the title bar of the Template Toolbox and dock it under the Application views. You can also undock it by dropping it anywhere on your desktop. Drag it back to dock it again.

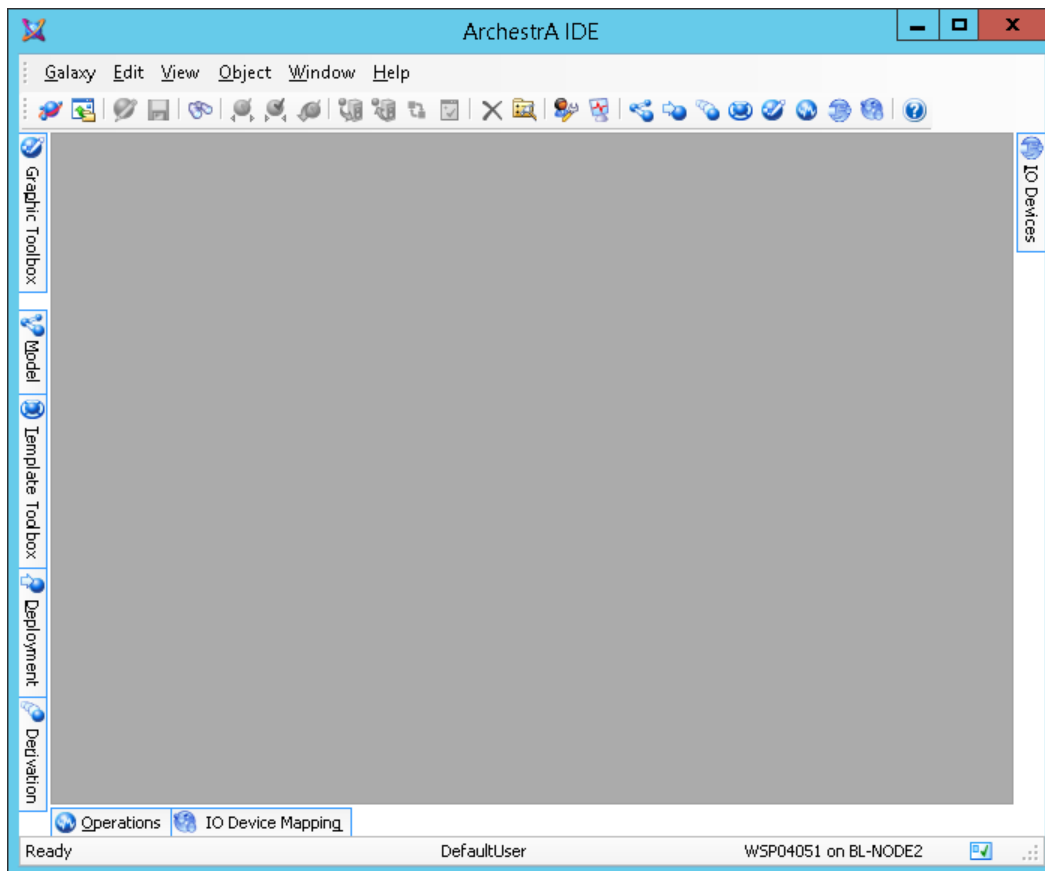


### Float Views

- ▼ You can also float a view. With your cursor over the view you want to float, click the arrow. On the menu that appears, click **Floating**.
- ▼ You can also float just one view. To float the **Derivation** view, click the arrow. On the menu that appears, click **Floating**. The view floats on your desktop. You can move it to another location or dock it.

### Hide Views

- 📌 To hide a view, click the **Pin** icon. The views "hide" as tabs on the side of the window. The Operations and IO Device Mapping views hide at the bottom of the window, and the IO Devices view hides at the right side. The Graphic and Template Toolboxes, and Model, Derivation, and Deployment views are at the left side of the window.



When you move the mouse over the tab, the view expands into the workspace.

## Reset the Workspace

You can easily reset the IDE workspace and restore views back to their default locations.

### To return the views to the default docking

- On the **View** menu, click **Reset Layout**.

## Synchronize Views

You can specify that a selected object stay selected as you move through the views. In any of the views, select the object you want to synchronize. On the **View** menu, click **Synchronize Views**. Now as you move from one view to another, that object stays selected.

You can also configure an option in your user information to automatically keep the same object selected as you switch between the **Model**, **Deployment**, and **Derivation** views. See *Configure User Information* on page 38 for details.

## Configure User Information

You can configure options for a Galaxy, including prompts for check-in comments and user defaults. These user options apply to the currently open Galaxy and do not change options for other Galaxies.

If you specify a security group, that security group must already exist. For more information about security and security groups, see 01Standardize\_Security\_Configure Security.

## To configure user information

1. On the **Edit** menu, click **User Information**. The **Configure User Information** dialog box appears.

Configure User Information

**Prompts**

- Ask for check-in Comments
- Warn if check-in operation will block other users
- Warn before launching an editor for a read-only object
- Warn for insufficient permissions
- Warn before launching an InTouchViewApp editor
- Warn before launching a Linked Content editor
- Warn before launching ViewApp in Preview Mode
- Warn for Platform issues in Preview Mode

Initial scan state for deployed objects

On Scan

Off Scan

Scan state defaults

Force Off Scan

Don't Force Off Scan

Auto context selection

Automatically synchronize the current single object selection when opening a new view

Field Attributes

Show the Field Attributes tab

**IMPORTANT: Field Attributes are deprecated. Convert these attributes to User Defined Attributes.**

User Defaults

Platform:

Application Engine:

Area:

View Engine:

Security Group:

OK Cancel

2. In the **Prompts** area, do one or more of the following:
  - Select the **Ask for Check In Comments** check box if you want to be prompted to type comments when checking in templates and objects.
  - Select the **Warn before launching an editor for a read-only object** check box if you want to see a prompt that informs you if you are opening an instance or template as read-only. The prompt warns you if you open an instance or template while someone else is working on it. If someone else is working in the instance or template, you cannot make changes.
  - Select the **Warn for insufficient permissions** check box if you want to see a prompt that tells you if you have permission to create or modify InTouchView applications. These permissions authorize or prevent you from creating and modifying InTouchView Applications.

- Select the **Warn before launching an InTouchViewApp editor** check box if you want to see a prompt each time you attempt to edit an InTouchView application instance. You can edit the associated template or cancel the operation. If you do not select this check box, the request to edit the InTouchViewApp instance is automatically redirected to the associated template.
  - Select the **Warn before launching a Linked Content editor** check box if you want to see a prompt each time you attempt to edit a Graphic Toolbox item, such as a symbol or layout, linked to an object. You can edit the linked content or cancel the operation. If you do not select this check box, the linked content opens in its editor (symbols in the Graphic Editor, layouts in the Layout Editor, etc.) without a warning, and you will be able to make changes to the content. If you then save the content, the item in the Graphic Toolbox is updated with your edits, and the changes propagate to all objects that link to the content.
  - Select the **Warn before launching ViewApp in Preview Mode** check box to restore the first warning message that appears after requesting a preview of a ViewApp. You need to select this prompt to display the message again after the user selected the check box on the warning message to hide the message.
  - Select the **Warn for Platform issues in Preview Mode** check box to restore the warning message that appears after requesting a preview without the WinPlatform deployed. You need to select this prompt to display the message again after the user selected the check box on the warning message to hide the message.
3. Select the **Initial scan state for deployed objects**. You can select **On Scan** or **Off Scan**. You can change this setting on an individual basis in the **Deploy** dialog box when you deploy.
  4. Select the **Scan state defaults** when undeploying or redeploying instances. **Force Off Scan** will attempt to take the target object offscan when an already deployed object is redeployed. **Don't Force Off Scan** does not force the target to go off scan when you deploy.

---

**Note:** Redeployment of objects that are currently deployed on-scan will be canceled unless this option is selected.

---

5. Enable or disable **Auto context selection**. This setting is enabled by default.
  - When **Auto context select is enabled** (checked), an object selected in the **Model**, **Derivation**, or **Deployment** view remains selected when you switch between views.
  - When **Auto context select is disabled** (unchecked), the **Model**, **Derivation**, and **Deployment** views do not keep the same object selected as you switch between views, so a different object can be retained in each view.
6. In the **Field Attributes** section, select the **Show Field Attributes** check box to display the **Field Attributes** tab in the User Defined Object editor.

By default, the **Field Attributes** tab is displayed only if field attributes are already defined for the object. You can override the default by selecting this option. For more information about working with legacy field attributes, see *About the Field Attributes, UDAs, and Extensions Pages* on page 81.
7. In the **User defaults** area, provide the following object names of Framework objects you select to be defaults with respect to assignment relationships.
  - Type the **Platform** name.
  - Type the **Application Engine** name.
  - Type the **Area** name.
  - Type the **View Engine** name.
  - Type your **Security Group** name, if any.
8. When you are done, click **OK**.



## Log on with Security Enabled

Some Galaxies have security associated with them. If you try to open a Galaxy with security, you need to log on to the Galaxy to open it.

---

**Note:** If you do not have logon rights, you cannot open a Galaxy.

---

For information about setting up security in the System Platform environment, see 01Standardize\_Security\_Configure Security.

### To log on to a Galaxy

1. When you open a Galaxy that has security enabled, the **Change User** dialog box appears.
2. Type your **user name** and **password**. If OS authentication security is enabled for the Galaxy, select the **Domain** that contains your user account from the dropdown list. If the list is unavailable, the selected Galaxy is on your local computer.

You can change your password after you type your user name and password. Click **Change Password**. Type the new password and then retype it.

3. Click **OK**. Your logon data is validated by the Galaxy Repository being accessed. Depending on operating system security, the IDE opens. If the GR does not recognize your user name or password, you are prompted to enter them.

## Change Users

You can change users in a Galaxy at any time. Any security restrictions associated with a user change when the user logs on or logs off from the Galaxy. For more information about users and security, see *Configuring Security* on page 481.

If the Galaxy has not been configured to enable security, you see a message. All users in an open-security environment are treated as the DefaultUser by the Galaxy. In an open-security environment, all users have full access to everything.

### To change users

1. On the **Galaxy** menu, click **Change User**.  
If security is enabled on the Galaxy, the **Change User** dialog box appears.
2. Enter your logon information and click **OK**.
  - If needed, click **Change Password** to change the password for the new user.
  - Type the new password and then retype it.
3. Click **OK**.



# CHAPTER 2

## Introduction to Objects

### About Templates and Instances

Before you start modeling your application using the Application Server, you must understand templates and object instances.

**Templates** are elements in Application Server that contain common configuration parameters for object instances that you use multiple times in your application.

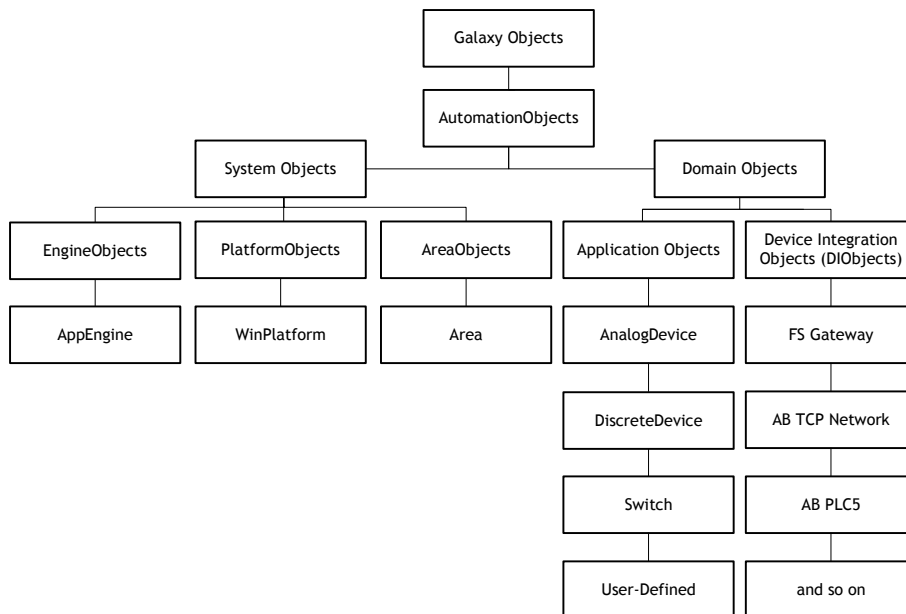
For example, you might create a template for valves. You configure the template with all the unique attributes for valves. You use that template to make object instances of valves. You can further configure and customize each object instance to represent a specific valve.

**Object instances** are the specific devices in your environment, such as diaphragm valves or very complex devices, like a reactor. You create an instance from a template and then customize the specific instance as needed.

Instances are deployed to the run-time environment. Templates exist in the development environment and cannot be deployed.

Creating templates and instances is very similar to object-oriented programming. For example, templates and instances have a parent/child relationship that involves inheriting attributes. There are differences, however, between object-oriented programming and creating templates and instances in Application Server.

Collectively, templates and instances are called objects. The following graphic shows the different kinds of objects and how they are organized.



If you are new to this kind of programming, the next section explains the basic concepts you need to know before you start. If you are familiar with object-oriented programming, the concepts in the next section may be familiar to you, but notice the important differences between object-oriented programming and Application Server.

## Instances

Instances are the run-time objects created from templates in Application Server. Instances are the specific things in your environment like processes, valves, conveyor belts, holding tanks, and sensors. Instances can get information from sensors on the real-world device or from application logic in Application Server. Instances exist during run time.

In your environment, you may have a few instances or several thousand. Many of these instances may be similar or identical, such as valves or holding tanks. Creating a new valve object from scratch when you have several thousand identical valves is time-consuming. That's where templates come in.

## Templates

Templates are high-level definitions of the devices in your environment. Templates are like a cookie cutter from which you can make many identical cookies.

You define a template for an object, like a valve, one time and then use that template when you need to define another instance of that item. Template names have a dollar sign (\$) as the first character of their name.

A template can specify application logic, alarms, security, and historical data for an object.

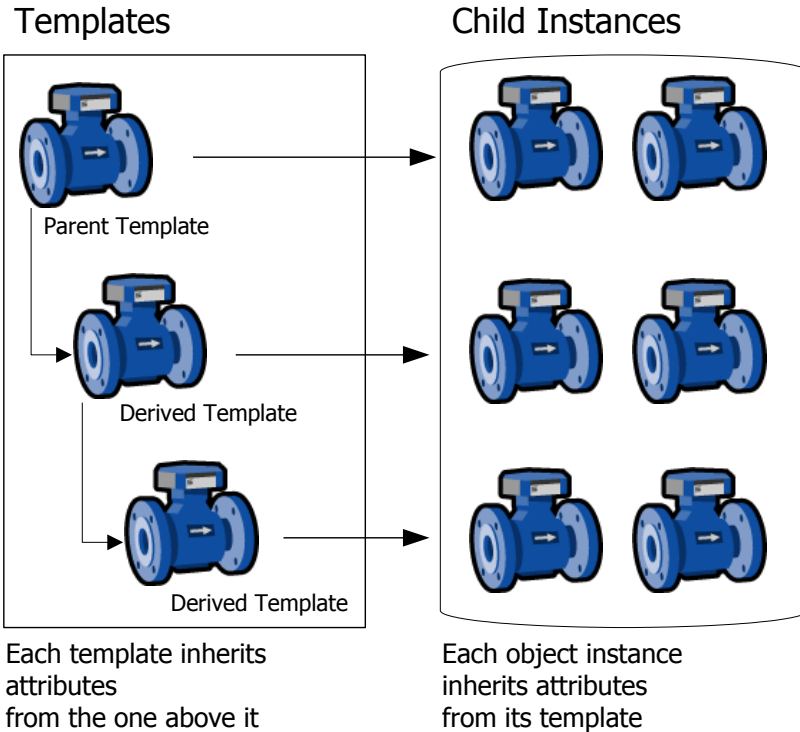
A template can also define an area of your environment. You can extend and customize a template by adding attributes, scripts, or features to meet the specific needs of your environment. Objects inherit attributes, scripts and features from their parents.

Application Server comes with predefined templates, called base templates. You cannot modify base templates. All templates you create are derived from base templates.

You can also nest templates, or contain them. Contained templates consist of nested object templates that represent complex devices consisting of smaller, simpler devices, including valves. A reactor is a good candidate for containment.

Templates only exist in the development environment.

Using the Diaphragm valve template, you can quickly create a Diaphragm valve instance when you need another Diaphragm valve in your application.



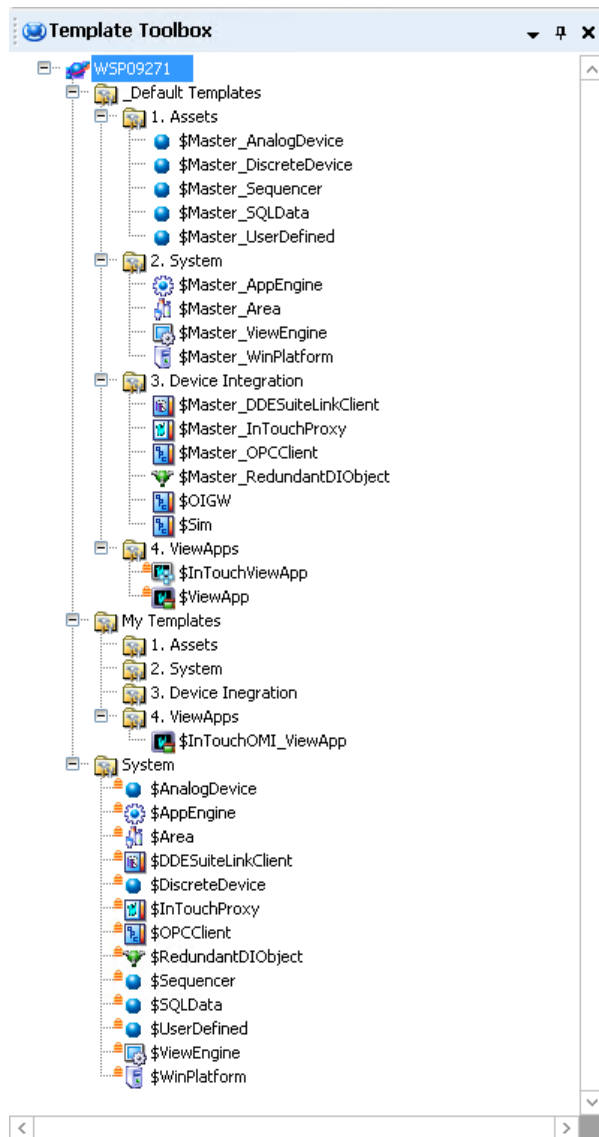
## Propagation

If you need to change something about all diaphragm valves, you can change the template for the Diaphragm valve and all diaphragm valves in your application inherit the changes, assuming the attributes are locked in the parent template. This makes it easy to maintain and update your application.

## About Base Templates

When you first open the IDE, you see a set of read-only base templates, which are indicated by an orange lock symbol. If you based your Galaxy on the Default.cab, you will also see a set of first-level derived templates, separated into subfolders, in the **Template Toolbox**.

You cannot modify base templates, but you can use them to create derived templates, which are copies of the base templates. We recommend that you use the first-level derived templates that are provided under the **\_Default Templates** folder to create additional templates. Most of these first-level derived templates have the prefix **Master\_**. You can modify derived templates and use them to create instances of them for your applications.



The Template Toolbox includes:

- **Asset templates**  
Use these templates to represent real devices in your Galaxy, such as pumps and valves.
- **Device Integration templates**  
Use these templates to create instances that communicate with external devices, such as PLCs.
- **System templates**  
Use these templates to define system instances, like other computers.

## Application Templates

These base templates let you easily create devices in your Galaxy. They contain the properties you need to set for each kind of device. For example, a DiscreteDevice device contains all the settings you need to specify for an on/off device. Of course, you can extend and customize any device by using attributes, scripts, and features.

## Device Integration Templates

These base templates provide communication with external devices. External devices run on the application engine.

For example:

- DInetwork object – Refers to the object that represents the network interface port to the device through the Data Access Server. The object provides diagnostics, and configuration for that specific card.
- DIdevice object – Refers to the object that represents the actual external device (such as a PLC or RTU), which is associated to the DInetwork Object.

## System Templates

These objects represent the parts of a Galaxy and not the domain they are monitoring/controlling. These base templates let you create more system level grouping and computers, such as areas you add objects to or another host AppEngine.

### WinPlatform Object

The WinPlatform platform object is a key base object because you need a platform to host the objects you are modeling. This object:

- Calculates various statistics for the node it is deployed to. These statistics are published in attributes.
- Monitors various statistics related to the node it is deployed to. These monitored attributes can be alarmed and historized.
- Start and stop engines, based on the engines startup type which are deployed to it.
- Monitor the running state of engines deployed to it. If the platform detects an engine failed, it can, optionally based on the value of the engine's restart attribute, restart the engine.

### AppEngine Object

The AppEngine object must have a Platform on which to run. This object:

- Hosts ApplicationObjects, Device Integration objects and areas.
- Contains the logic to set up and initialize objects when they are deployed.
- Contains the logic to remove objects from the engine when they are undeployed.
- Determines the scan time which all objects within that particular engine run.

### Area Object

All application objects belong to an area. Areas can contain sub-Areas. Areas provide a key organizational role in grouping alarm information and providing that information to those who use alarm/event clients to monitor their areas of responsibility.

The values of three Area object alarm attributes can be saved to the historian:

- Active alarm counter
- Unacknowledged alarm counter

- Disabled (or silenced) alarm counter

### ViewEngine Object

The ViewEngine object must have a Platform on which to run. This object:

- Hosts InTouchViewApp and ViewApp objects.
- Contains the logic to set up and initialize objects when they are deployed.
- Contains the logic to remove objects when they are undeployed.
- Determines the scan time which all objects within that particular engine run.
- Determines if the ViewApp objects that it hosts will be read/write or read-only.

---

**Note:** InTouch applications are not affected by the "ViewApps are Read Only" setting of the ViewEngine. This setting is used for the ViewApp object only.

---

### InTouchViewApp Object

The InTouchViewApp object must have a ViewEngine on which to run. This object:

- Manages the synchronization and delivery of files required by the associated InTouch® application.
- Provides run-time access to tags on the associated InTouch application.
- Starts WindowMaker for the associated InTouch application when edited.

---

**Note:** InTouch applications are not affected by the "ViewApps are Read Only" setting of the ViewEngine that hosts the InTouchViewApp object. This setting is used for the ViewApp object only.

---

### ViewApp Object

The ViewApp object must have a ViewEngine on which to run. This object:

- Provides visualization at run time for an AVEVA OMI application.
- Incorporates a Screen Profile and the content defined in an associated Layout.
- Provides run-time access to your production environment.
- The deployed ViewApp can be read/write or read-only, depending on whether the ViewEngine hosting the ViewApp object has its "ViewApp are Read Only" setting enabled (ViewAppsReadOnly attribute set to true).

## About Derived Templates

All templates you create within the IDE are derived templates.

When creating your Galaxy application, plan ahead and create derived templates for devices of a certain type so you can use the templates to create the individual instances.

A new derived template is an exact copy of its parent template with the possible exceptions of locking and security and modified attribute values. You can lock attributes to prevent them from being modified in child templates.

After you create a new derived template, you can customize it. For more information about customizing and extending templates, see *Create Derived Templates* on page 57.

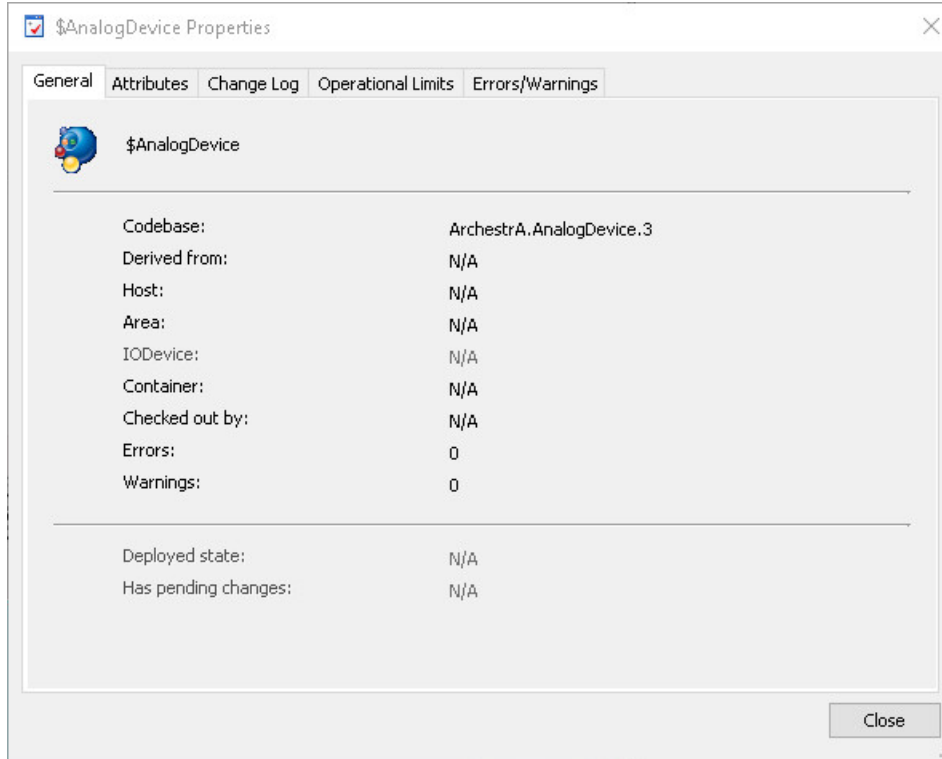
Every template has a set of attributes and default values. When you create an instance, attributes are inherited by the instance. In the instance, you can reconfigure many of the attributes inherited from the parent template if they are not locked on the parent template.

For more information about customizing instances, see *Edit Objects* on page 68.



## View Object Properties

You can view the properties of an object by right-clicking and clicking **Properties**. Object properties vary, depending on the type of selected object and whether it is a base template, a derived template, or an instance.



The **Properties** window contains the tabs that allow you to examine different object properties. Only tabs applicable to the object are displayed.

Tab Name	Displayed for Object Type:
General	All objects including GalaxyObject
Attributes	All objects except GalaxyObject
References	Instances only
Cross References	Instances only
Change Log	All objects including GalaxyObject
Operational Limits	All objects including GalaxyObject
Errors/Warnings	All objects except GalaxyObject

The **General** tab contains basic information for the object, such as the codebase (major version), what template it is derived from, and any current errors and warnings.

The **Attributes** tab lists the object's attributes and values. In addition to attributes that have been added to the object, attributes for Errors and InAlarm are also listed, as are attributes for CodeBase (major version of the object), MinorVersion ("dot" version), and ConfigVersion. The ConfigVersion will increment each time the object has been checked in after an edit.

The **References** tab lists I/O references for the object.

The **Cross References** tab lists any attributes that are cross referenced for the object.

The **Change Log** tab is a historical list of changes, including deploy/undeploy operations, that have been made to the object.

The **Operational Limits** tab lists prohibited actions and the reason for the prohibition. For example, Check In is not allowed for objects that are already checked in.

The **Errors/Warnings** tab lists any errors or warnings for the object.

For more information about specifying the properties of objects, see *Working with Objects* on page 51.

## CHAPTER 3

# Working with Objects

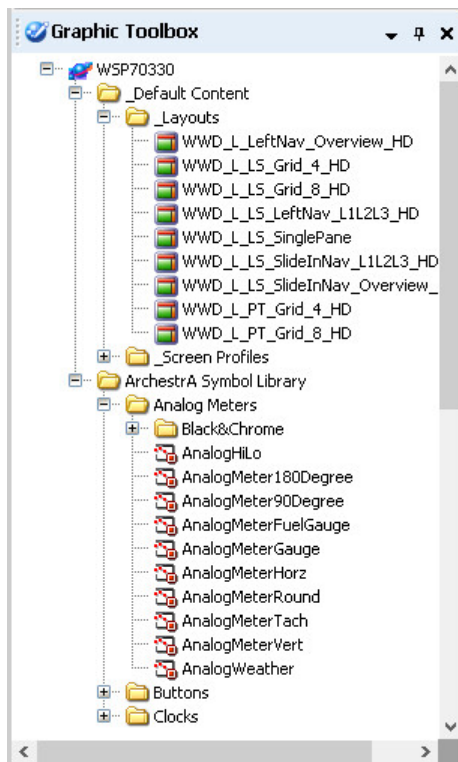
You can work with objects using the Application Server development environment. Both templates and instances are collectively referred to as objects. For more information about what templates and instances are, see *About Templates and Instances* on page 43.

## Manage Toolsets

Toolsets are high-level folders shown in the Template Toolbox and Graphic Toolbox. You can create a toolset to store the templates and graphics used in your application. You can also create toolsets within toolsets.

Use the Template Toolbox to view and organize object templates.

Use the Graphic Toolbox to view and organize Industrial Graphics, ViewApp Layouts, Screen Profiles, and controls and apps.



You can move content between their respective toolsets (folders). You can also show or hide toolsets to make the workspace less cluttered.

## Create Toolsets

When you create your own toolset, it must have a unique name. Toolset names are not case sensitive, so `Valves` is the same name as `valves`. A toolset name can be up to a maximum of 64 alphanumeric and special characters, including spaces, except `$`.

### To create a new toolset in the Toolbox

1. On the **Galaxy** menu, point to **New** and click either **Template Toolset** or **Graphic Toolset**.
2. Type a name for the new toolset.

A new toolset appears and is in focus. Now, you can drag templates into the new Template toolset, or you can drag graphics into the Graphics toolset.

## Create Child Toolsets

Toolsets can be created within existing toolsets. Nested toolsets help in further organizing templates and graphics. You can create a maximum of ten levels of toolset folders.

### To create a child toolset

1. Select the parent toolset.
2. On the **Galaxy** menu, point to **New** and click either **Template Toolset** or **Graphic Toolset**.
3. Type a name for the new toolset.

A new toolset appears beneath the parent toolset and is in focus.

4. Drag templates into the new child Template toolset, or you can drag graphics into the Graphics toolset.

## Deleting Toolsets

You can delete toolsets you no longer want or need. Before you start, make sure you move or delete all content from the toolset.

The toolset you want to delete must be empty, or it cannot be deleted.

### To delete a toolset

1. Select the toolset you want to delete.
2. On the **Edit** menu, click **Delete**.
3. Click **Yes** to delete the toolset.

## Using AVEVA Connect

AVEVA Connect is the common cloud repository allowing you to reuse graphics across various nodes.

Graphics can be downloaded and uploaded on demand. Graphics are stored in 'Stores' in AVEVA Connect; there are three types of stores – global, tenant and user specific. Within each store users can configure multiple drives. You must have an AVEVA Connect user account to manage graphics in the cloud. Each drive can be configured with different access levels for different users.

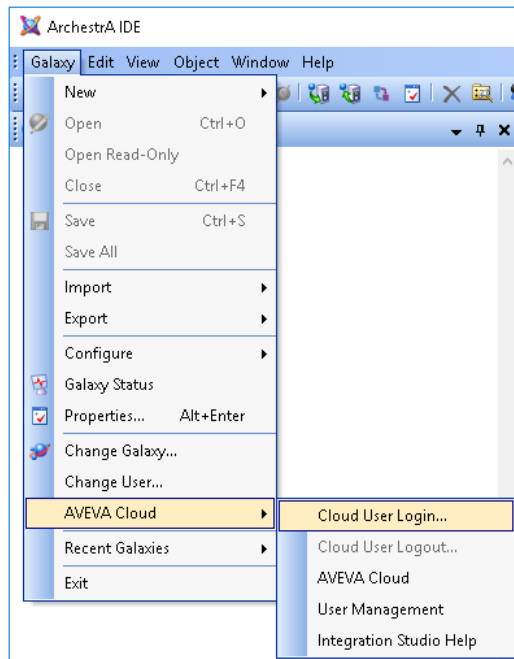
Users will have read-only or read-write access depending on the access granted by the administrator. Read-Only users can only view graphics in the cloud, they cannot upload or download graphics. Multiple users can access the graphics in the same drive, but only one user can edit or save a graphic at one time.

## Logging in to AVEVA Connect

You must have an AVEVA Connect account and authorization from your administration to access AVEVA Connect.

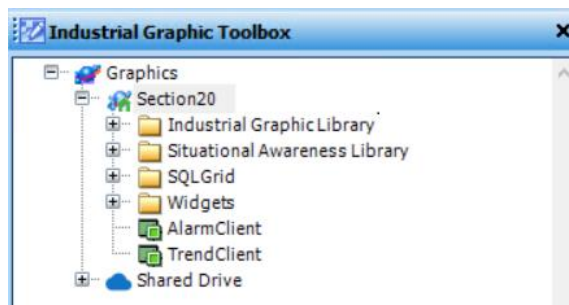
1. Open the IDE.

- From the **Galaxy** menu, select **AVEVA Cloud > Cloud User Login**.



In addition to **Cloud User Login**, the AVEVA Menu provides the following options:

- Cloud User Logout** - Enabled once you are logged in to AVEVA Connect
  - AVEVA Cloud** – Links to AVEVA Connect
  - User Management** – Links to AVEVA Connect Administrator Page
  - Integration Studio Help** – Links to the help page
- Enter your AVEVA Connect email address.
  - Enter your password. If the credentials are verified, you are signed in. Based on your authorization and privileges you will have access to AVEVA Connect.
  - Select the Store which you want to use.
  - AVEVA Connect appears as a separate repository within the Graphic Toolbox.

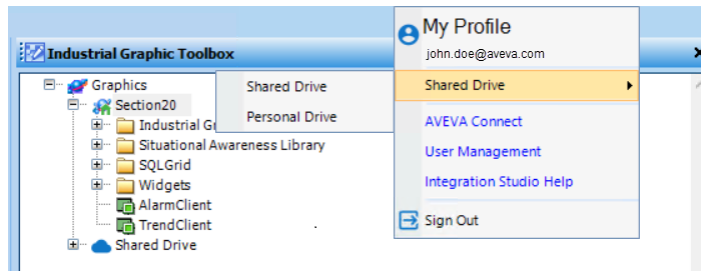


## Changing the AVEVA Connect Drive

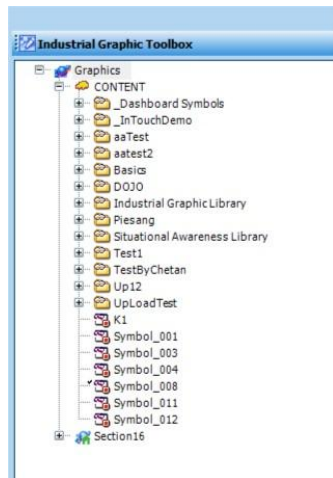
Once you are logged into a specific Store you can move between different drives and upload and download images.

- Click **AVEVA > DEFAULT DRIVE**.

2. Select a drive from the available options.



3. The graphic toolbox will refresh to display the selected drive.



## Uploading Graphics to AVEVA Connect

You can create folders on AVEVA Connect to organize your graphics, like the graphic toolbox.

1. Drag the graphic(s) from the local drive and drop it to the AVEVA Connect folder.
2. The **Upload Content** dialog appears and displays the progress of the operation. Click **View Details** to view the progress.
3. Click **Close**.
4. All embedded symbols are also uploaded, and a structure similar to the local repository is created on AVEVA Connect.

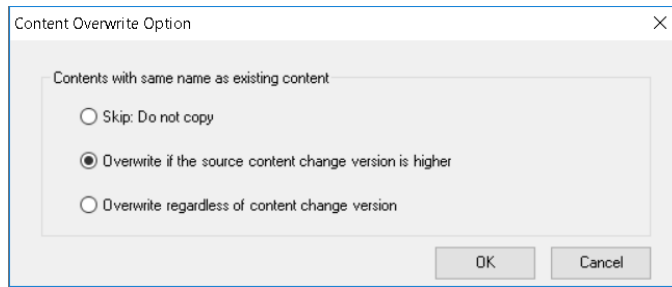
You can also upload folder(s) that contains multiple graphics.

## Downloading Graphics to the Local Repository

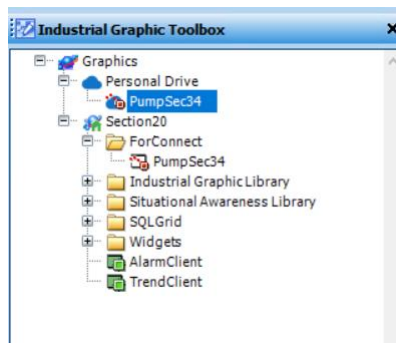
You can download graphics to the local repository and use it on your applications. You must have the appropriate permissions to

1. Drag the graphic(s) or folder from the AVEVA Drive and drop it to the local repository.

- The **Content Overwrite Option** dialog box appears, select the appropriate option.



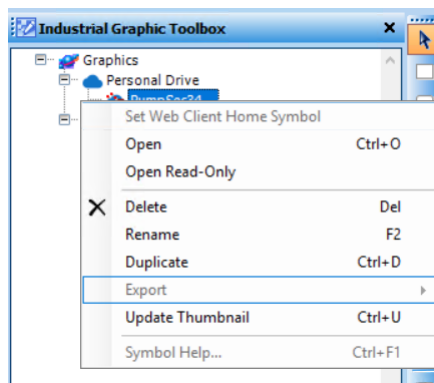
- The **Download Content** dialog box appears displaying the progress. Click **Close**. Any errors during downloading are also displayed.
- The graphic(s) are downloaded to the local repository and embedded graphics are also copied.



## Managing Graphics on AVEVA Connect

You can also create, rename, update, duplicate and delete graphics directly on AVEVA Connect.

- Right-click on AVEVA Connect or any of the folders to create new graphic or toolset.
- Double-click to edit graphics using the graphic editor.
- Right-Click and select the corresponding options to rename, duplicate or delete the graphic.
- The use is similar to managing local graphics. See the *Creating and Managing Industrial Graphics* guide for more information.



## Managing Graphics with Multiple Users

Multiple users can use the graphics available on AVEVA Connect at a time. When more than user is logged in, the graphic is locked automatically as soon as a user begins editing the graphic, and prevents other users from making any edits. A red tick mark appears against the graphic in the toolbox to inform the active user that they have the graphic checked out.

All other users can only view the graphic. A black tick mark will appear against the graphic name to signify that the graphic is being edited by another user.

After the first user saves and closes the graphic, another user can check out the graphic and edit it. Once the graphic is saved, the edited graphic can be viewed by other users.

## Manage Galaxy Style Libraries

You can import a Galaxy Style Library to load its Element Styles in a Galaxy. You can also modify Element Styles, and then export the Galaxy Style Library as custom user-defined Element Styles.

Optional Galaxy Style Library XML files are located in the ...\\Program Files ( x86)\\Archestra\\FrameWork\\Bin\\AdditionalElementStyles folder. The names of the XML files suggest the primary color schemes of the Element Styles within each optional Galaxy Style Library.

A Galaxy Style Library can be exported from the IDE and imported into other Galaxies.

---

**Note:** Only one Galaxy Style library can be loaded and used in a Galaxy at a time.

---

## Import a Galaxy Style Library

### To import a Galaxy Style Library

1. On the **Galaxy** menu, click **Import** and click **Galaxy Style Library**.
2. Browse for the library file. Galaxy Style Library files are XML files.
3. Select the file and click **Open**. The selected Galaxy Style Library is loaded in the IDE.

If you imported a Galaxy Style Library, it is now the active Galaxy Style Library for the entire Galaxy.

---

**Note:** If a Galaxy Style Library is imported to the IDE while a ViewApp is running, application graphics are refreshed with new Element Styles. The ViewApp refreshes automatically. A restart is not required.

---

## Export a Galaxy Style Library

You can create your own Galaxy Style Library by overriding the configuration of Element Styles and exporting the library. For information about setting Element Style overrides, see *Change the Visual Properties of an Element Style* on page 191.

### To export a Galaxy Style library

1. On the **Galaxy** menu, click **Export** and click **Galaxy Style Library**. The **Export Galaxy Style Library** dialog box appears.
2. Browse to a path and type a name for the exported library file.
3. Click **Save**. The file is saved with the specified name and a **.xml** file extension.



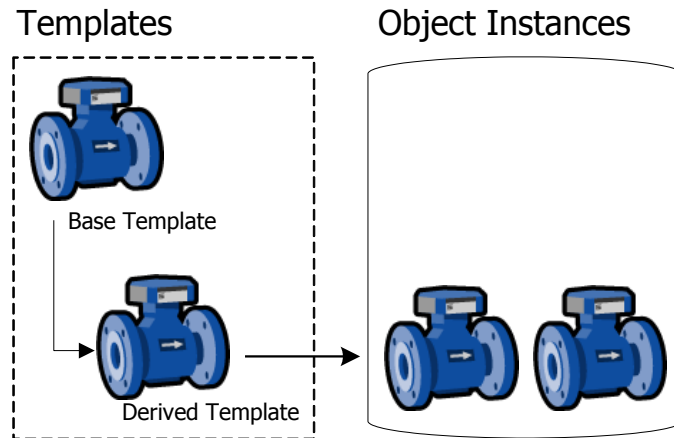
## Reset a Pre-Defined or User-Defined Galaxy Style Library

If you changed the values of Quality and Status values, Pre-Defined or User-Defined Galaxy Element Styles, or Format Styles, you can restore the default values by clicking the **Reset to Default** button.

## Create Derived Templates

All templates you create are derived templates. A derived template inherits attributes and behaviors from the parent template. You cannot change the attributes in a base template.

After you create a derived template, you can customize and modify its attributes. If you change locked attributes in the parent template, the changes propagate to the derived template.



After you create derived templates, you can customize them, and you can create instances of the templates. You can change and modify unlocked attributes in the instances, making adjustments to meet the needs of the specific object you are modeling.

For example, your plant processes can use several models of a pump made by a single vendor. Each model has unique characteristics that map to different attribute values of the DiscreteDevice base template.

### To derive a template from another template

1. Select the base template to use as the parent template in the **Template Toolbox** or **Derivation** views pane.
2. On the **Galaxy** menu, click **New** and click **Derived Template**. A derived template is created in the same toolset as its parent and placed in name edit mode. The default name is the same as the parent template followed by a numeric sequence.
3. Rename the derived template, if needed. Template names can be up to 32 alphanumeric or special characters, including the required \$ as the first character. The second character cannot be \$ and the name must include at least one letter. Template names cannot contain spaces.

---

**Note:** The following reserved names cannot be used as template names: Me, MyContainer, MyArea, MyHost, MyPlatform, MyEngine and System.

---

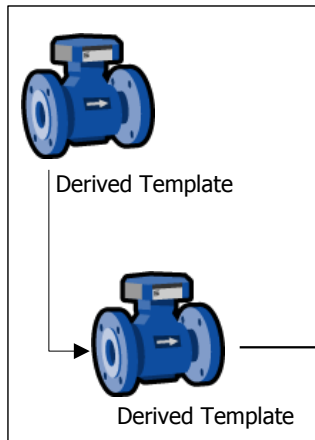
4. You are ready to customize your new template. For more information, see *Edit Objects* on page 68.

## Derive Templates from Another Derived Template

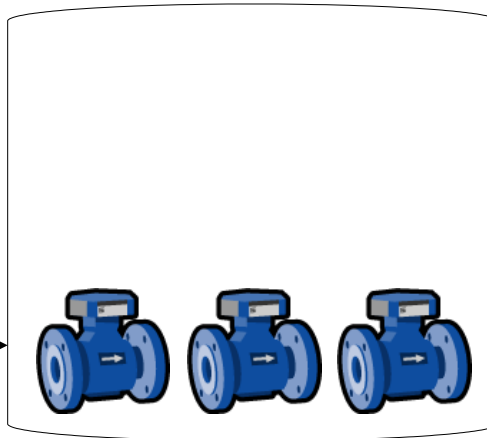
You can create derived templates from other derived templates. The child template inherits attributes from all parent templates. Any changed attributes in the immediate parent overrides attributes changes in grandparent levels.

If you change locked attributes in the parent template, the locked attributes propagate to the derived template.

## Templates



## Object Instances



A derived template is an exact copy of its parent with the exceptions of locking, security, and the unlocked attributes that have been edited. If you create a new derived template from an existing container template, the new derived template has the same contained templates.

A good practice is to create a hierarchy of derived templates until you reach logical endpoints. Then create instances from each unique derived template.

### To create a derived template from a derived template

1. In the **Template Toolbox** or **Derivation view**, select the derived template you want to use as the parent template.
2. On the **Galaxy** menu, click **New** and click **Derived Template**. A derived template is created in the same toolset as its parent. You can edit the name of the new derived template. The default name is the same as the parent template followed by a numeric sequence.

Template names can be up to 32 alphanumeric or special characters, including the required \$ as the first character. The second character cannot be \$ and the name must include at least one letter. Template names cannot contain spaces.

---

**Note:** The following reserved names cannot be used as template names: Me, MyContainer, MyArea, MyHost, MyPlatform, MyEngine and System.

---

3. You can create another derived template by repeating the previous steps, or you can customize your new derived template. For more information about customizing your template, see *Edit Objects* on page 68.

## Create Contained Templates

Containment is the relationship in which one object includes another. Containment relationships organize objects in a hierarchy. You can build objects that represent complex devices consisting of smaller, simpler devices.

In scripts, these objects can be referred to by the name that derives from the containment relationship. This name is called a hierarchical name.

An object can have three kinds of names if it is contained by another object. The three names include:

---

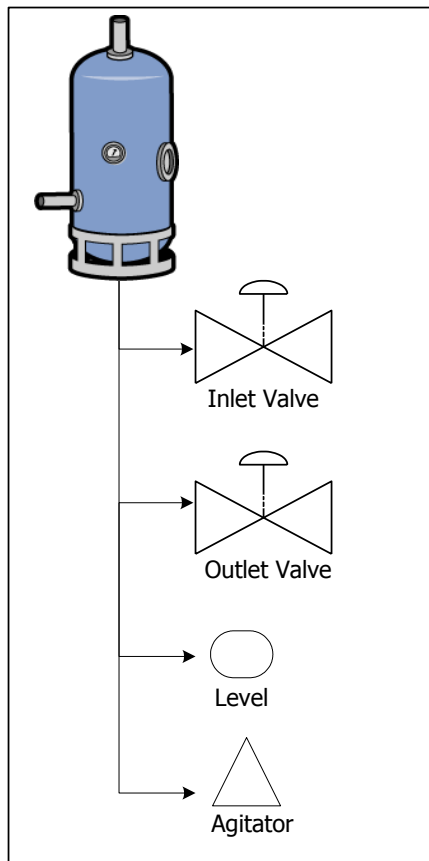
Name	Description
Tag Name	The unique name of the individual object. For example, <code>Valve1</code> .
Contained Name	The name of the object within the context of its container object. For example, the object whose Tag name is <code>Valve1</code> may also be referred to as <code>Tank1.Outlet</code> , if <code>Tank1</code> contains it and it has the contained name "Outlet".
Hierarchical Name	<p>Hierarchical names that are fully-qualified names of a contained object include the name of the objects that contain it.</p> <p>Because the object that contains it may also be contained, there are potentially multiple hierarchical names that refer to the same object.</p> <p>For example, if:</p> <ul style="list-style-type: none"><li>"Reactor1" contains Tank1 (also known within Reactor1 by its contained name "SurgeTank").</li><li>"Tank1" contains Valve1 (also known within Tank1 by its contained name "Outlet").</li></ul> <p>Valve1 could be referred to as:</p> <ul style="list-style-type: none"><li>"Valve1"</li><li>"Tank1.Outlet"</li><li>"Reactor1.SurgeTank.Outlet".</li></ul>

---

**Note:** Base templates cannot be contained by another template, either as the container or as the template being contained. You can only use containment with derived templates.

---

Higher level objects contain lower level objects. This allows you to more closely model complex plant equipment, like tank systems. You can nest templates to 10 levels.



---

**Note:** Objects can only contain objects like themselves. For example, ApplicationObjects can only be contained by other ApplicationObjects. Areas can only contain other Areas.

---

## ApplicationObject Containment

ApplicationObjects can be contained by other ApplicationObjects. This provides context for the contained object and a naming hierarchy that provides a powerful tool for referencing objects.

---

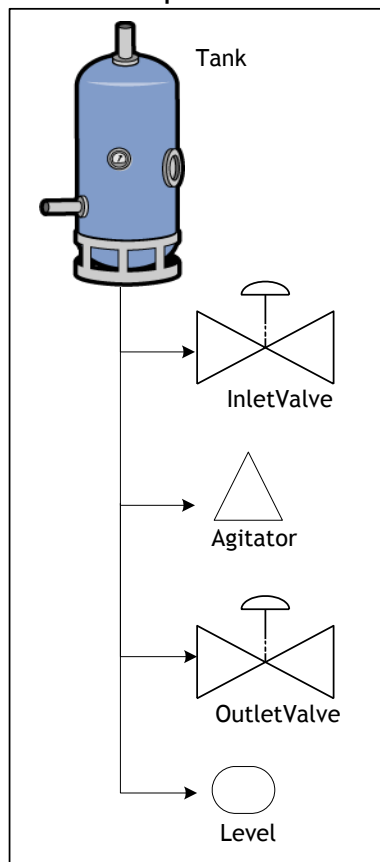
**Note:** Base templates cannot be contained by another template, either as the container or as the template being contained. You can only use containment with derived templates.

---

An example of a containment hierarchy is a tank that contains the following objects:

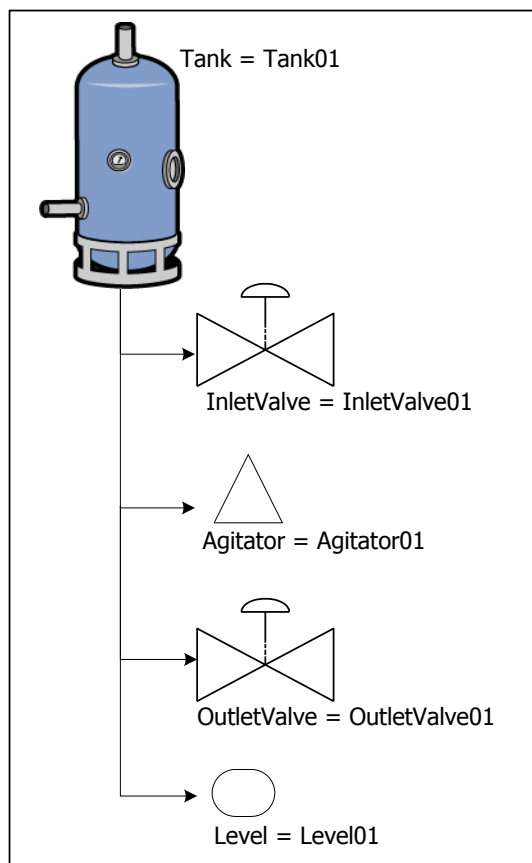
- Inlet Valve
- Agitator
- Outlet Valve
- Level

## Tank Template



To enable referencing and flexibility within scripting, these objects can be referenced in several different ways. Each object has a unique tag name, such as:

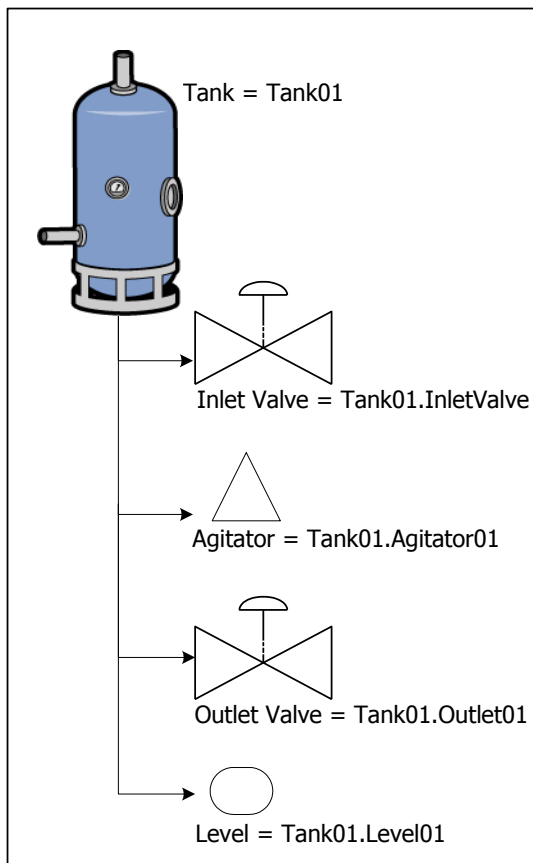
- Inlet Valve = InletValve01
- Agitator = Agitator01
- Outlet Valve = OutletValve01
- Level = Level01



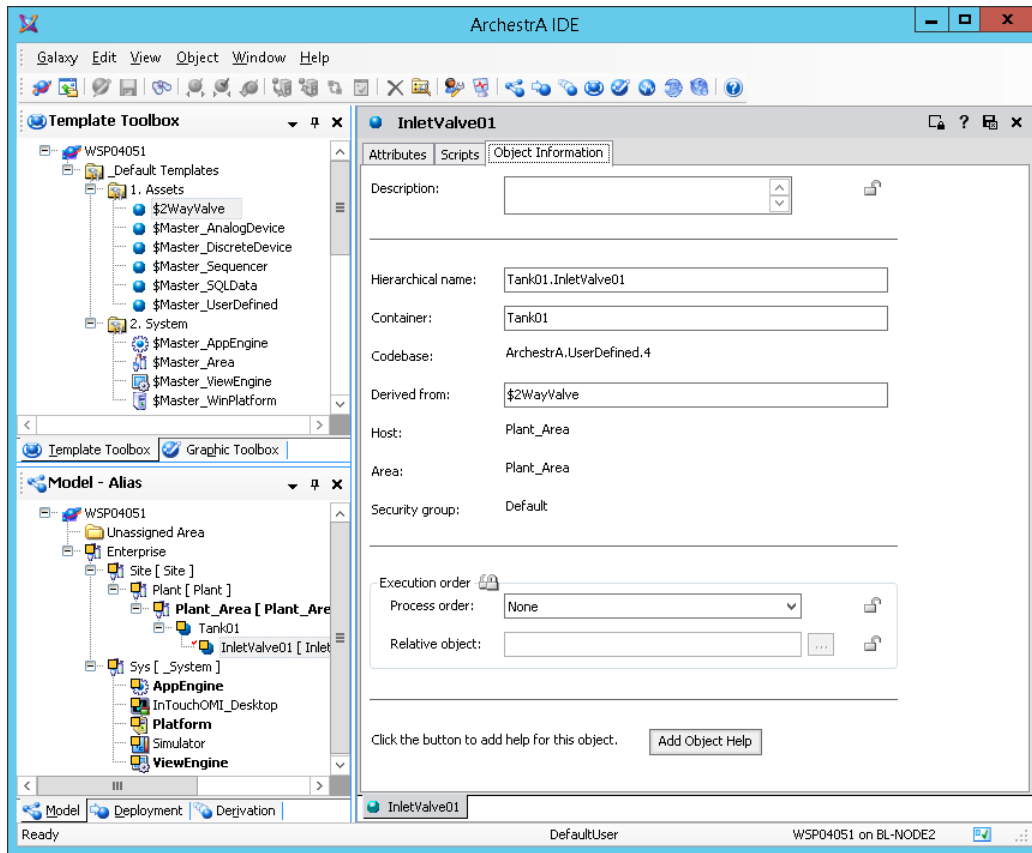
Within the context of each hierarchy, the contained names are unique, in that the names only refer to this tank system and the contained objects.

So if the tank is named `Tank01`, the contained names are:

- `Tank01.Inlet`
- `Tank01.Agitator`
- `Tank01.Outlet`
- `Tank01.Level`



This naming convention adds context to the instances contained by Tank01.



Additionally, you can use containment references in scripts such as:

- `Me.Outlet`: Allows a script running within the parent object to generically reference its child outlet instance.
- `MyContainer.Inlet`: Allows a script running in any of the children instances to reference another child instance named Inlet that belongs to the same parent.

## Using Contained Names

The contained name of a contained object only has to be unique in the context of its container.

An object can have three kinds of names, depending if it is contained by another object. The three names include:

Name	Description
Tag name	The unique name of the individual object. For example, <code>Valve1</code> .
Contained name	The name of the object within the context of its container object. For example, the object whose tag name is <code>Valve1</code> may also be referred to as <code>Tank1.Outlet</code> , if <code>Tank1</code> contains it and it has the contained name "Outlet".



Name	Description
Hierarchical name	<p>Hierarchical names that are fully-qualified names of a contained object include the name of the objects that contain it.</p> <p>Because the object that contains it may also be contained, there are potentially multiple hierarchical names that refer to the same object.</p> <p>For example, if:</p> <p>"Reactor1" contains Tank1 (also known within Reactor1 by its contained name "SurgeTank").</p> <p>"Tank1" contains Valve1 (also known within Tank1 by its contained name "Outlet").</p> <p>Valve1 could be referred to as:</p> <p>"Valve1"</p> <p>"Tank1.Outlet"</p> <p>"Reactor1.SurgeTank.Outlet".</p>

For example, an instance of a \$Tank is named Tank01. An instance of \$Valve called Valve01 is contained within the instance of \$Tank.

Change the contained name of Valve01 to InletValve. Now Valve01 can also be referred to by its hierarchical name Reactor1.InletValve. The name of the contained object can be changed, though, within the scope of the hierarchy.

Contained names can be up to 32 alphanumeric or special characters. The second character cannot be \$ and the name must include at least one letter. You cannot use spaces.

## Containment Examples

You can have a Tank object that contains two DiscreteDevice objects that represent its inlet and outlet valves.

**Note:** Base templates cannot be contained by another template, either as the container or as the template being contained. You can only use containment with derived templates.

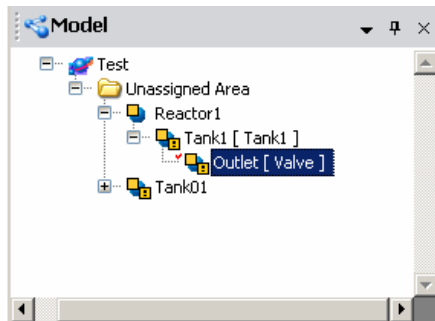
### To implement containment

1. Create the following instances: Tank1 from \$UserDefined and Valve from \$DiscreteDevice. Valve has only one name, Valve.
2. In the **Model** or **Deployment view**, drag Valve on to Tank.

**Note:** If Tank1 already contains an object with a contained name of Valve, the Galaxy generates a unique contained name for the newly contained object, such as Valve\_001.

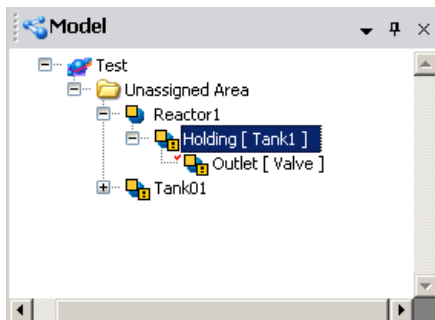
3. Change the contained name of Valve within Tank1 to Outlet. Valve can now be referred to by its tagname, Valve, as well as its hierarchical name, Tank1.Outlet.
4. Create an instance called Reactor1 from \$UserDefined.

5. In the **Model** or **Deployment** view, drag Tank1 onto Reactor1.



6. Change the contained name of Tank1 to Holding. Tank1 now has two names, Tank1 and Reactor1.Tank. Also, Valve1 has a three-part hierarchical name: Reactor1.Tank.Outlet.

For the three objects in this example (Reactor1 containing Tank1 containing Valve1), the following naming hierarchy exists:



### To implement template-level containment

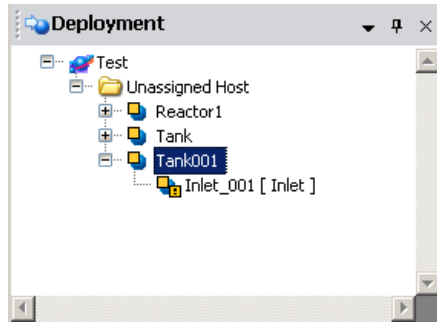
**Note:** Contained Templates do not have tagnames. When an instance hierarchy is created from a template and its contained children, unique tagnames will be created for the instances based on their contained names.

1. Create the following derived templates: \$Tank from \$UserDefined and \$Valve from \$DiscreteDevice.
2. Derive \$Inlet from \$Valve.
3. In the **Template Toolbox**, drag \$Inlet on to \$Tank. If \$Tank already contains a template named Inlet, the Galaxy generates a unique tagname for the new contained template, such as Inlet\_001.

The contained template now has a hierarchical name \$Tank.Inlet.

4. Create an instance (Tank001) of \$Tank.

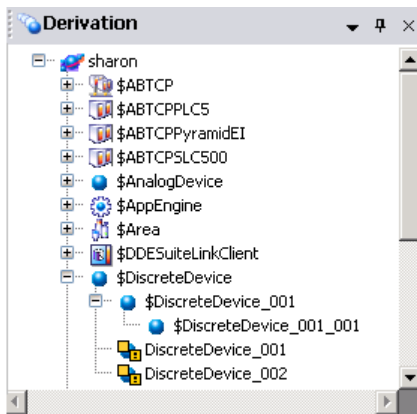
5. The **Model** and **Deployment views** show an instance `Tank001` that contains an instance called `Inlet`.



## View Containment Relationships

Containment relationships appear for templates in the **Template Toolbox**. For instances, the relationship appears in both the **Model** and **Deployment views**.

In the **Derivation** view, if a template contains other templates, you can expand it to show the containment under that template.



The **Derivation view** shows templates and instances with regard to containment in the following ways:

- Non-contained instances show their tagnames.
- Contained instances show their tagnames and hierarchical names.
- Non-contained templates show their template name.
- Contained templates show their hierarchical name.

## Rename Contained Objects

Before you rename a contained name of an object, make sure that the object is not checked out to another user or currently deployed.

The new contained name must comply with naming restrictions. Template names can be up to 32 alphanumeric or special characters, including the required \$ as the first character. The second character cannot be \$ and the name must include at least one letter. You cannot use spaces.

Contained names also cannot be the same contained name as an existing contained object within the same level of hierarchy in the containment relationship.

---

**WARNING! Be careful when renaming contained objects. References from other objects to the object being renamed are not automatically updated with the new name. You must update the references. Objects with broken references receive bad quality data at run-time.**

---

### To rename an object's contained name

1. Select the object in an Application view.
2. On the **Edit** menu, click **Rename Contained Name**.
3. Type a new contained name.

All IDEs connected to the Galaxy show the object's new contained name.

## Edit Objects

With the Object Editor, you define attributes specific to an object. When you open the Object Editor in non-ReadOnly mode, the object is checked out. No one else can edit an object while you are working with it. If someone else is already working on the object, you can open it to view but you cannot make changes.

The Object Editor contains pages that can be used to modify objects. To view a page in the editor, click its tab. Three pages – **Attributes**, **Scripts**, and **Object Information** – are common to all objects, while other pages are unique to certain object types. If you import a Galaxy or objects with field attributes, the **Field Attributes** page will be present for those objects; the Field Attributes page can also be enabled through a Galaxy user configuration option. See *Configure User Information* on page 38 for more information.

The **Attributes** page divides into two or more panes when you initially open it. The number of panes the Attributes page divides in depends on whether or not attributes or symbols have been defined for the object, whether the object is an instance or a template, and if it is an instance, whether or not it is derived from a template that includes an Object Wizard.



When editing an object, you may see attribute text boxes showing a --- (dash dash dash). The --- is a placeholder reference that does not cause the associated object to be placed in a warning configuration status when it is validated. You may also see attribute text boxes showing a ---.--- (dash dash dash dot dash dash dash). You need to provide a valid reference in the text box. The ---.--- placeholder causes the associated object to be placed in a warning configuration status when the associated object is validated.

---

**Important:** Edit the I/O auto assignment placeholder in the Object Editor ONLY if you do NOT want to use I/O auto assignment for the object. In most cases, I/O auto assignment is the preferred method.

---

When you save an object, the configuration data for the object is validated. If errors or warnings are identified during validation, a message appears. You can cancel the save or save the object configuration as it is.

- If you cancel, the Object Editor remains open so you can correct the problems.
- If you save the configuration as it is, the object is placed into a bad or warning state. The object's status is marked in the Galaxy database as Good, Warning or Error. Error means the object is undeployable.

### To edit an object in the Object Editor

1. Select the object.
2. On the **Galaxy** menu, click **Open**. A red check mark appears next to the object's icon indicating it is checked out and the Object Editor opens.

---

**Note:** If you are adding I/O attributes to an application object or system object, such as an area object, the preferred method of adding and editing I/O references is through the **IO Devices** view and **IO Device Mapping** view. Editing I/O references set for automatic assignment in the Object Editor disables automatic assignment. For more information about I/O auto assignment, see *Using I/O Auto Assignment* on page 148.

---

3. Make your changes. For more information about locking attributes, see *About the Attributes Page* on page 74. For more information about setting security, see *Set Object Security* on page 73.
4. When you finish configuring the object, click **Save** or **Close** on the **Galaxy** menu.
  - Save** keeps the editor open and saves all configuration changes to the Galaxy database.
  - Close** closes the editor.
  - To keep the object checked out, select the **Keep Checked Out** check box before closing.

## Object Editor User Assistance

Tooltips are available in the Object Editor. Point to any editor option and a tooltip appears, showing the attribute name. This name is used when referring to the attribute in scripts, for example.

Each object also includes documentation about usage, configuration, run-time behavior, and attributes. For help with configuring the object, select the object in the Template Toolbox or in an application view and press **Ctrl + F1**, or right-click on the object and select **Object Help** from the shortcut menu. If you want, you can customize the object help file. See *Customize Help* on page 81 for more information.

## Help File Structure

The header part of the Help file contains the following information:

- Tag name                      The object's name.
- Contained Name              The object's contained name. For more information, see *Create Contained Templates* on page 58.
- Description                    A short summary of what the object is for.
- Code Base                      The code version of the object.
- Derived From                  The immediate parent template for the object.

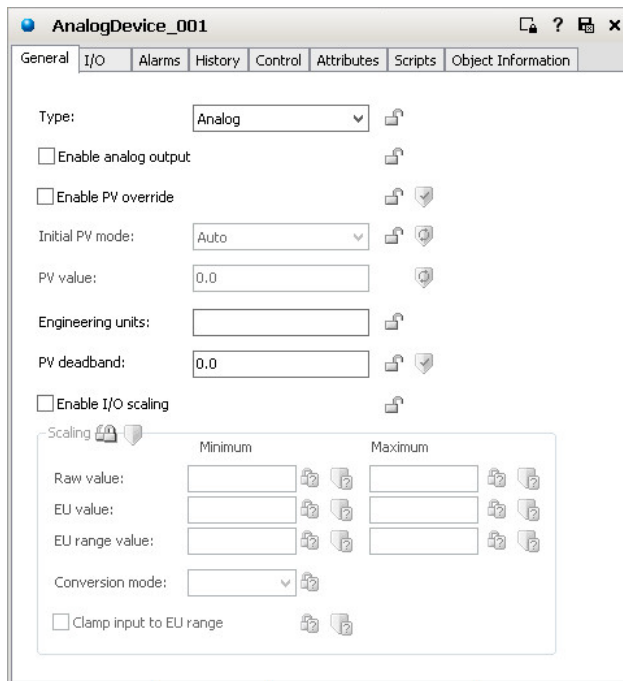
- **Object Version**            The configuration version of the object.
- **Process Order**            The run-time execution order within the host engine's scan (none, before, after) relative to the **Relative Object** element.
- **Relative Object**            The object that runs before or after in the **Process Order**.

The rest of the help file shows general information about the object.

## About the General Editor Layout

When you open the attributes for an instance or a template, you see the Object Editor. The Object Editor is where you configure the object's attributes and add scripts or associated graphics to the object. The Object Editor has several pages related to the type of object you select. If you are working with an instance, you see different pages than if you are working with a template.

For example, the following illustration shows an analog device template.



When you open the Object Editor, the object is automatically checked out so no other user can work on it. When you close the Object Editor, the object is checked in to the Galaxy database, if it was automatically checked out when the editor was opened.

- To keep the object checked out, click **Keep Checked Out** before closing.
- To save configuration changes you made, close the editor, and check the object back in. *About the Object Information Page* on page 80, click the **Close** icon.

After the object is checked in, other users can edit it.

## Lock and Unlock Template Attributes

When you create derived templates, you can lock or unlock some or all of the attributes and enabled features. Locking an attribute prevents it from being changed in derived templates or instances. You can only lock attributes in templates.

Locking an attribute in a template specifies that its value or setting is inherited by all derived objects, both templates and instances. Locking an attribute also makes the attribute act as a constant during run time.

**Note:** When you lock values and settings for an attribute, they are not visible when you use Object Wizards, and thus the locked values and settings will be inherited by all derived objects.

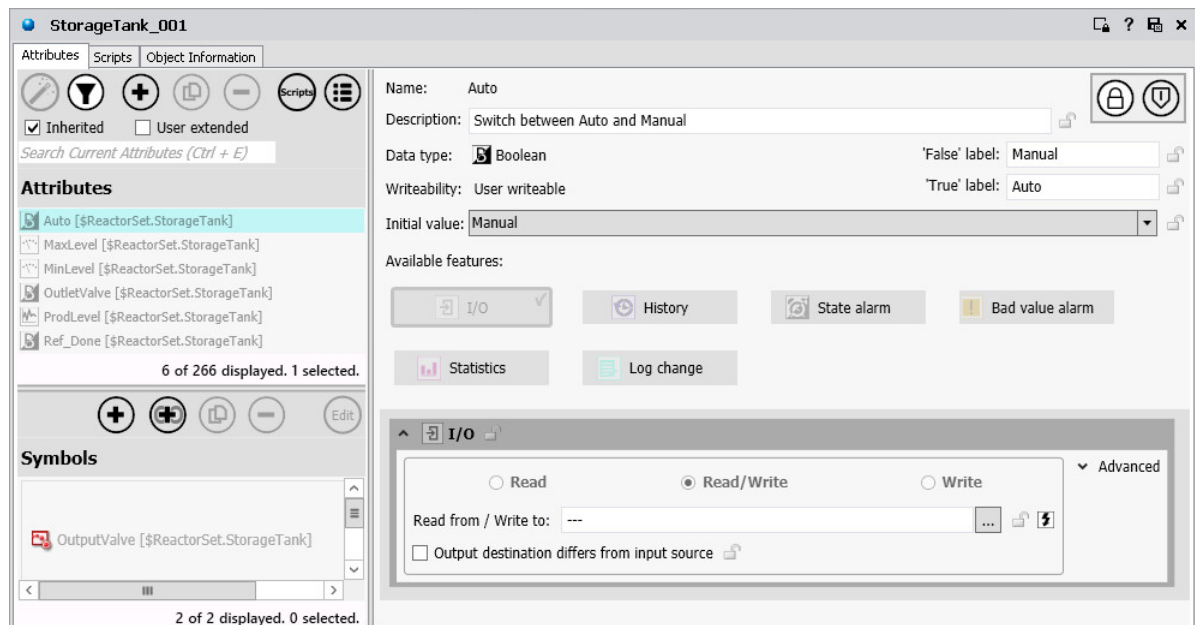


In the **Attributes** page, you must enable locking before an attribute or any of its features can be locked in a template. To enable locking, click the **Show/Hide Lock** icon to the right of the attribute name. When locking is enabled, lock symbols will appear next to values that can be locked. Lock symbols are not visible in the template or its derived instances unless locking is enabled.






- Attributes that are locked in a parent template are referred to as "locked in parent." This parent can be at any parent level above the selected object.
- Attributes that are locked in a template are referred to as "locked in me."

**Caution:** When using I/O auto assignment, do not lock the "Read from" field (input source) or "Write to" field (output destination). If you lock these fields in the parent template, they cannot be updated with the resolved reference when the object is deployed and the runtime value will be "---Auto---".

See *Group Locking/Security* on page 74 for information about locking or unlocking all of a feature's attributes with a single click.



Lock controls and status are shown with an icon. If the option is enabled, click the lock control to switch it between locked and unlocked. These icons mean:

Icon	Name	Description
	Locked (in me)	<p>The associated attribute is locked (in me) and enabled. Only templates can have this kind of lock. The attribute value is read/write.</p> <p>Derived templates and instances do not have a unique copy of this attribute. Child objects share the locked attribute of the parent.</p> <p>Changing the value of a locked attribute in the parent template updates the value of that attribute in all derived templates and instances.</p>
	Locked (in parent)	<p>The associated attribute is locked in the parent object and cannot be unlocked or modified by the child object. Both templates and instances can have these. The attribute is read-only.</p> <p>The templates and objects do not have a unique copy of this attribute, but instead use the attribute value in the parent where the attribute is locked.</p>
	Unlocked	<p>The associated attribute is unlocked and enabled. Both templates and instances can have this kind of lock. The attribute is read/write.</p> <p>The object has its own copy of the attribute value and the value is not shared by derived objects.</p>
	Indeterminate	Refers to a specified group of options. An indeterminate state indicates different lock states for individual options in the group.
	Undefined	The associated attribute doesn't exist. This indicates that another attribute be enabled before the associated attribute is created and before its lock status can be determined.

**Note:** Locking an attribute during configuration makes its value a constant. You cannot write to locked attributes during run time.

### To lock an attribute example

1. Create a derived template from the \$DiscreteDevice base template. Name the derived template \$Valve.
2. Open the \$Valve template and on the **State** page, lock one of the attributes by clicking the **Lock** icon.
3. Save \$Valve.
4. Create a derived template from \$Valve. Name it \$BigValve.
5. Create an instance from \$Valve named Valve1.

In the editor of \$Valve, the attribute lock icon shows the attribute is locked in me.

You cannot change the attribute value in \$BigValve and Valve1. The editor options for the attribute are disabled and the lock icon, if shown, indicates a lock in the parent. Also, the attribute lock icon in children derived from \$Valve is now locked and disabled.

If you change the attribute value in \$Valve, the change propagates to \$BigValve and Valve1 after you save the changes.



### To unlock an attribute example

1. Using the objects from the previous example, in the \$Valve template's editor, unlock the locked attribute.
2. Save \$Valve.

In the editor for \$Valve, the attribute lock icon shows it is unlocked.

The lock type for this attribute of \$BigValve now indicates locked in me. The lock type for this attribute of the Valve1 instance shows unlocked but the locking icon is unavailable.

## Set Object Security

Operators interact with objects through the individual attributes of those objects. Each attribute on the Object Editor that can be modified by operator's at run time and can have an associated security control, which is used to modify its run-time security classification.





In the **Attributes** page, security icons must be enabled before you can set security for an attribute or any of its features in a template. To enable security, click the **Show/Hide Security** icon to the right of the description field.

When security is enabled, security symbols will appear next to values for which security is configurable. Security symbols are not visible in the template or its derived instances unless enabled in the parent template.

If an attribute's security classification is configurable, click the security control to select one of seven possible states:


Security Icon	Description
Free Access	Lets you change this value without restriction even if you have no defined permissions on the object. Anyone can write to these attributes to perform safety or time critical tasks that can be hampered by an untimely logon request, such as halting a failing process.
Operate	Lets you work with Operate permissions to do certain normal day-to-day tasks. These include writing to attributes like Setpoint or Command for a Discrete Device object. This level of security requires you to have Operate permission for the security group for the object.
Secured Write	Requires you to authenticate using your user name and password each time you want to write to the attribute. You also need to have Operate permissions for the object.
Verified Write	Requires you to have Operate permissions to log on again and a second, different user to also log on as the verifier before writing to the attribute. The verifier needs to have Can Verify Writes permission for the object.
Tune	Allows end users with Tune Operational permissions to tune the attribute in the run-time environment. Examples of tuning are attributes that adjust alarm setpoints and PID sensitivity.

Security Icon	Description
 Configure	Allows end users with Configure Operational permissions to configure the attribute's value. Requires that the user first put the object off scan. Writing to these attributes is considered a significant configuration change. For example, a PLC register that defines a Discrete Device input.
 Read Only	Only allows users to read this attribute's value in the run-time environment. This attribute is never written to at run time, regardless of the user's permissions.

If an attribute's security is shown in gray, its security classification is locked in its parent object and cannot be changed or it requires the enabling of a group attribute.

## Group Locking/Security

The lock and security controls associated with option groups and features quickly set those conditions for all options in the group.

-  The group control typically reflects the setting for all options in the group or feature set. But, if at least one option in the group has a lock or security control that is different from the other options, the group control shows an indeterminate icon.

In addition to the undefined controls, the group controls for locking and security are the same as those for individual options.

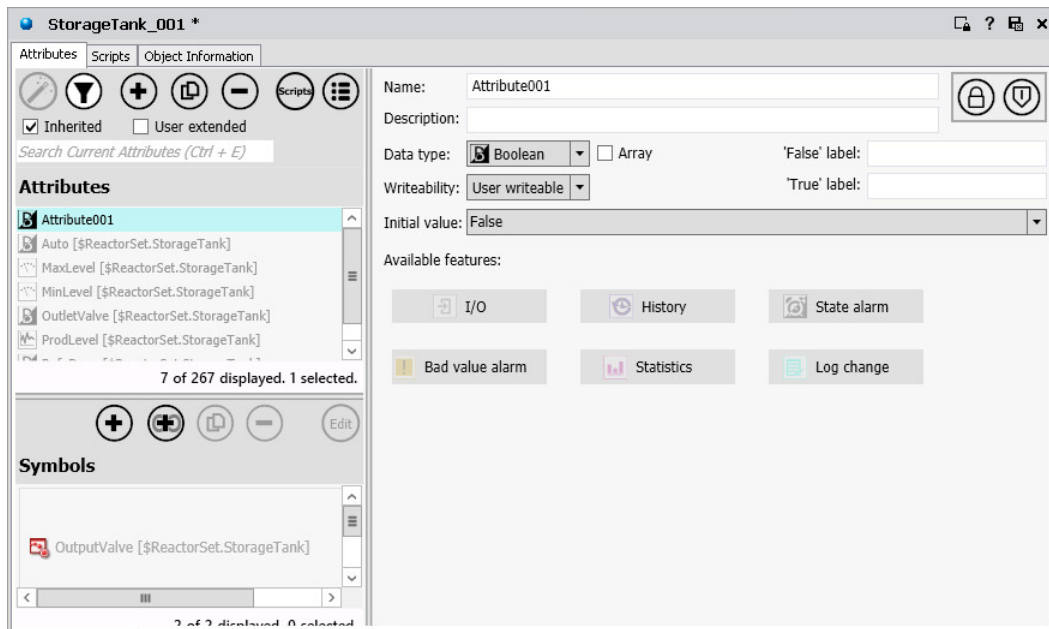
You can lock or unlock all of the attributes associated with a feature by selecting the lock icon next to the feature name, after you activate the feature. This will lock or unlock all of the attributes for the feature, unless an attribute was locked at a higher level. For example, if you locked an attribute in a template, and then created another derived template, the attribute that was locked in the original template cannot be unlocked (locked in parent).








If an attribute is locked in the template, you can change the value in that template, but not in the derived children. If you change the value in the parent template, the change propagates to all child objects. For more information, see *Configuring Objects* on page 121.

## About the Attributes Page

If the object you are editing does not have any attributes defined, the **Attributes** page will be empty. The **Attributes** page will also show a pane for symbols, and depending on the type of object (template or instance) and its contents, may show additional Object Wizard configuration panes.

You can activate various features, such as I/O, History, State Alarm and Statistics. See *Configuring Objects* on page 121 for more information about attributes and features. When you add an attribute to an object, information about the attribute is shown.



-  Opens the **Object Wizard** editor.
-  To filter attributes, click the **Filter** button. You can then select filtering criteria by checking source type, enabled feature type, writeability type, lock status, data type, visibility type (hidden or not hidden), and diagnostic type (configuration, run time, or both).
-  **Adds** a new attribute.
-  **Duplicates** the selected attributes.
-  **Removes** the selected attributes.
-  To display the **Scripts** pane used to associate scripts with an Object Wizard, click the **Scripts** button.
-  **Shows Details** about an attribute. The name, description, and icons representing activated features for the object's attributes will be shown.

When you add an attribute to an object, the Attributes page divides into three sections. The left side of the page lists attributes, the top right shows information about the currently selected attribute, and the bottom of the right side contains fields for configuring features.

You can search attributes by entering the characters you are trying to find in the Search Current Attributes text box. This will display attributes that contain the characters you enter.

---

**Note:** The search function finds attributes that contain the search term anywhere within the attribute name. For example, if you want to locate all attributes associated with Pump 125, you could enter "125" in the text box. All attributes with 125 anywhere in their names will be shown.

---

You can also search for attributes by typing the character you are searching for when the cursor is pointed at the attribute list. The first attribute that contains the character, anywhere within its name, will be highlighted.

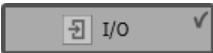


**Note:** The search begins at the attribute that is currently selected. Selection will move to the next attribute that contains the character entered.







- On the left side of the page, all attribute names for the object are listed. Pressing the **Show Details** button reveals additional information about the listed attributes. Check boxes are provided for **Inherited** and **User extended**.
  - **Inherited:** Checking this displays attributes that were added in the parent template used to create the object.
  - **User extended:** Checking this displays details about any features you added to the attribute, for example, for I/O.
- At the top of the right side, information for the current attribute is listed. Additional fields may be listed, depending on the data type of the attribute. However, all data types include the following properties:

Property	Description
Name	Attribute name (for example, Attribute001).
Description	User defined; a description of the attribute.
Data type	<p>Can be set to Boolean, Integer, Float, Double, String, Time, ElapsedTime, or InternationalizedString.</p> <p>The data type determines which features can be added to an attribute. For example, when configuring a Boolean data type, only two alarm features are available: <b>State Alarm</b> and <b>Bad Value Alarm</b>. When configuring an Integer data type, there are four available alarm features: <b>Limit Alarms</b>, <b>ROC Alarms</b>, <b>Deviation Alarms</b> and <b>Bad Value Alarms</b>.</p> <p>To create an array, check the <b>Array</b> option and specify the array's length in the <b># of elements</b> box. You can create an array for each data type except <b>InternationalizedString</b>.</p>

Property	Description
Writeability	<p>Can be set to Calculated, Calculated retentive, Object writeable, or User writeable. See the Object Viewer User's Guide for additional information about writeability categories.</p> <ul style="list-style-type: none"> <li>• <b>Calculated:</b> Permits only scripts within the same object to write to the attribute. Calculated attributes are not saved across restarts. If you select Calculated for an attribute, only scripts running on the same object can write to the attribute.</li> <li>• <b>Calculated Retentive:</b> Permits only scripts within the same object to write to the attribute. Calculated Retentive attributes are saved across engine restarts.</li> <li>• <b>Object Writeable:</b> Permits other objects to write to this attribute in addition to being set by scripts within this object. Object Writeable attributes are saved across restarts. This category is not user writeable.</li> <li>• <b>User Writeable:</b> Permits other users to write to this attribute in addition to being set by scripts and objects throughout the system. User Writeable attributes are saved across restarts. They can be locked at configuration time. If locked, they are read only.</li> </ul>
Initial value	<p>Specifies the initial value for the attribute when the object is deployed. Enter value data for each data type. In the case of a non-arrayed <b>Boolean</b>, select <b>True</b> or <b>False</b> in the list box. For an arrayed Boolean, select the appropriate checkbox.</p>

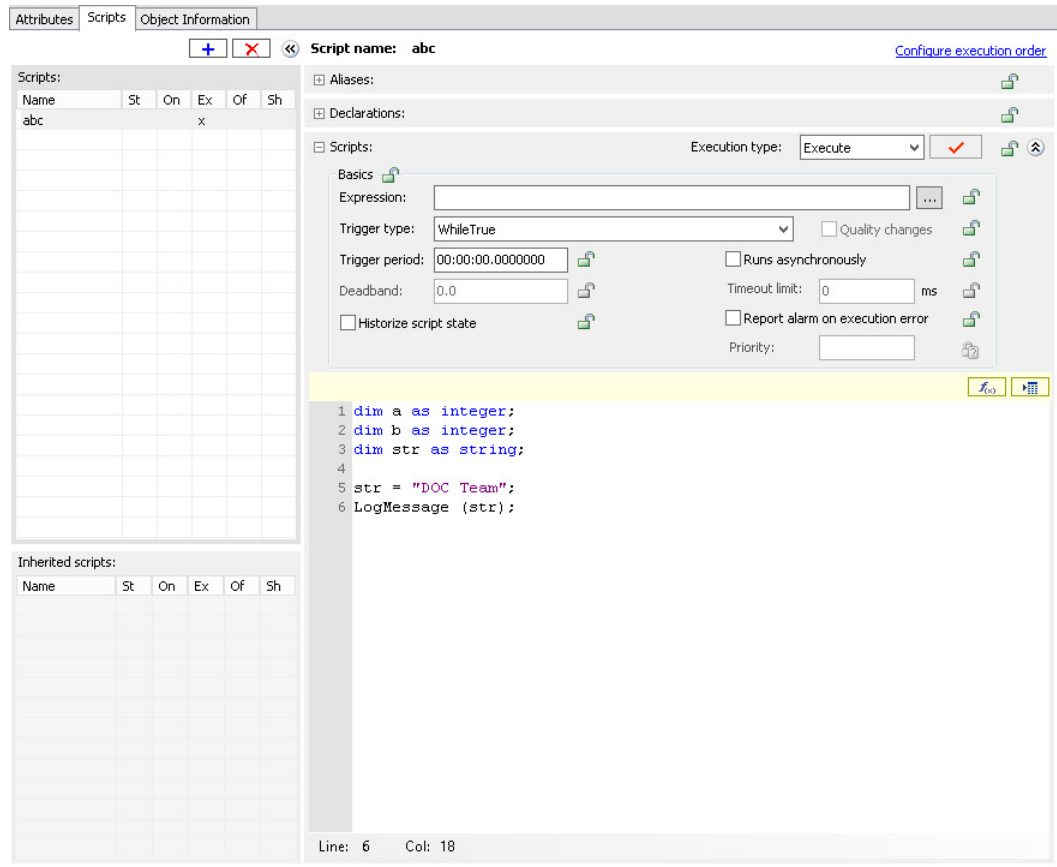
- At the bottom of the right side, fields for configuring activated features are displayed. Available features include:

Feature	Description
 I/O ✓	<p>The I/O feature is used to activate the attribute for I/O. You can select <b>Read</b> (input), <b>Read/Write</b> (input and output), or <b>Write</b> (output).</p> <p>When you select the I/O feature for application objects and system objects, such as areas, the "Read from / Write to attribute text box contains the placeholder reference &lt;IODevice&gt;.</p> <p>[HierarchicalName].[AttributeName].InputSource or &lt;IODevice&gt;.[HierarchicalName].[AttributeName]. OutputDest.</p> <p>You should not edit these placeholders in the Object Editor, but instead use the <b>IO Devices view</b> to assign the object to a scan group. This will allow the placeholder to automatically resolve to the correct I/O reference. See <i>Using I/O Auto Assignment</i> on page 148 for additional information.</p>
 History ✓	<p>The History feature enables historization to the HHistorian.</p> <p><b>Note:</b> The AppEngine hosting the object must be configured for historization.</p>
 State alarm ✓	<p>The State Alarm feature is available for Boolean data types. You can set alarm priorities, the alarm state, an alarm message, and time deadband.</p>


Feature	Description
 Limit alarms ✓	The Limit Alarms feature is available for integer, float, and double date types. With it, you can set value alarms (Hi, HiHi, Lo, LoLo), and the limits, priority, and messages to apply to each alarm limit.
 ROC alarms ✓	The Rate of Change (ROC) Alarm feature is available for integer, float, and double date types. With it, you can set alarms if changes within a time period exceed specified values, and the limits, priority, messages, and time period to apply to each ROC alarm.
 Deviation alarms ✓	The Deviation Alarm feature is available for integer, float, and double date types. With it, you can set deviation alarms (major and minor), and the tolerance, priority, messages, target value, deadband, and settling period.
 Bad value alarm ✓	The Bad Value Alarm feature adds an alarm if the value returned from the attribute is determined to be bad quality.
 Statistics ✓	The Statistics feature monitors statistics associated with the object.
 Log change ✓	The Log Change feature will cause the attribute to generate an event each time the attribute value changes.

## About the Scripts Page

The **Scripts** page is divided into several areas. To learn more about using scripts, see *Writing and Editing Scripts* on page 160.



The main areas of the Scripts page include:

- **Scripts** list: Shows all scripts currently associated with the object. The columns indicate which kind of trigger the script uses: Startup, On Scan, Execute, Off Scan and Shutdown. Click the **Add** button () to add a new script.
- **Inherited scripts name** list: Shows all scripts associated with the object's parent. The columns indicate which kind of trigger the script uses: Startup, On Scan, Execute, Off Scan and Shutdown.
- **Aliases** area: Lets you create and modify aliases that apply to the script you are working on. Aliases are logically descriptive names for typically long reference strings that you can use in the script to make the script more readable.
- **Declarations** area: Provides a place to add variable declaration statements, such as `DIM MyArray[1] as FLOAT;`. These declared variables live from the start to the shutdown of the object and can be used to hold values that persist from one execution of the script to the next. They apply only to the script in which they are declared.
- **Basics** area: Provides a location in which you set the expression, triggering conditions, and other settings that run the script in the run-time environment. See *Writing and Editing Scripts* on page 160 for descriptions of triggers and when they are executed. This area includes:

**Configure Execution Order:** Sets the execution order of multiple scripts (inherited and local) associated with this object.

**Historize Script State:** Select to send the state of the script to the Historian.

- **Script Creation** box: Shows the script you are writing.

## About the Object Information Page

The **Object Information** page is common to all object configuration editors.

The screenshot shows a window titled "StorageTank\_001 \*" with three tabs: "Attributes", "Scripts", and "Object Information". The "Object Information" tab is active and displays the following fields:

- Description:** Storage Tank
- Hierarchical name:** R31.StorageTank
- Container:** R31
- Codebase:** ArcestraA.UserDefined.4
- Derived from:** \$ReactorSet.StorageTank
- Host:** Demo\_Area
- Area:** Demo\_Area
- Security group:** Default
- Execution order:**
  - Process order:** None
  - Relative object:** (empty field with a browse button)

At the bottom, there is a text prompt: "Click the button to add help for this object." followed by an "Add Object Help" button.

This page includes the following fields:

- **Description:** A short summary of the object's purpose.
- **Hierarchical Name:** The fully qualified name of a contained object, including the container object's TagName.
- **Container:** The name of the other object that contains this object, if applicable.
- **Codebase:** The code version of the object.
- **Derived From:** The immediate parent template of the object, either a base or derived template.
- **Host:** Another object to which the object is assigned (for example, a WinPlatform hosts an AppEngine). An object's host determines where an object will run when it is deployed.
- **Area:** An object that represents a logical grouping to which this object belongs. An object's area mostly affects the way in which its alarms are reported to alarm clients.
- **Security group:** The security group the object is associated with.
- **Execution order:** If you want this object to run before or after another object within its engine's scan, select from the **Process order** list. Click the **Browse** button to specify the **Relative object** in the **Attribute Browser**. For more information about the **Attribute Browser**, see *Reference Objects Using the Galaxy Browser* on page 88.
- **Add Object Help:** Opens a copy of the HTML help page for the template this object is derived from. You can edit this information. This allows you to create Help about the object you are currently configuring for downstream users. This Help appears when you select an object in a view and then click **Object Help** on the **Help** menu. See *Customize Help* on page 81 for more information.



## Customize Help

Do not use Microsoft Word as an editor to create downstream object HTML help pages. Use an HTML or plain text editor instead.

If clicking **Add Object Help** opens Word on your computer, change the program associated with editing HTM files. Open the Windows Explorer's **Folder Options** dialog box and go to the **File Types** page to make this change. For more information about associating programs with files, see your Windows help.

## Locate the Help Folders

The path to each object's Help folder is unique. It depends on the path you selected when you installed the Galaxy Repository. The default path to an object's Help is:

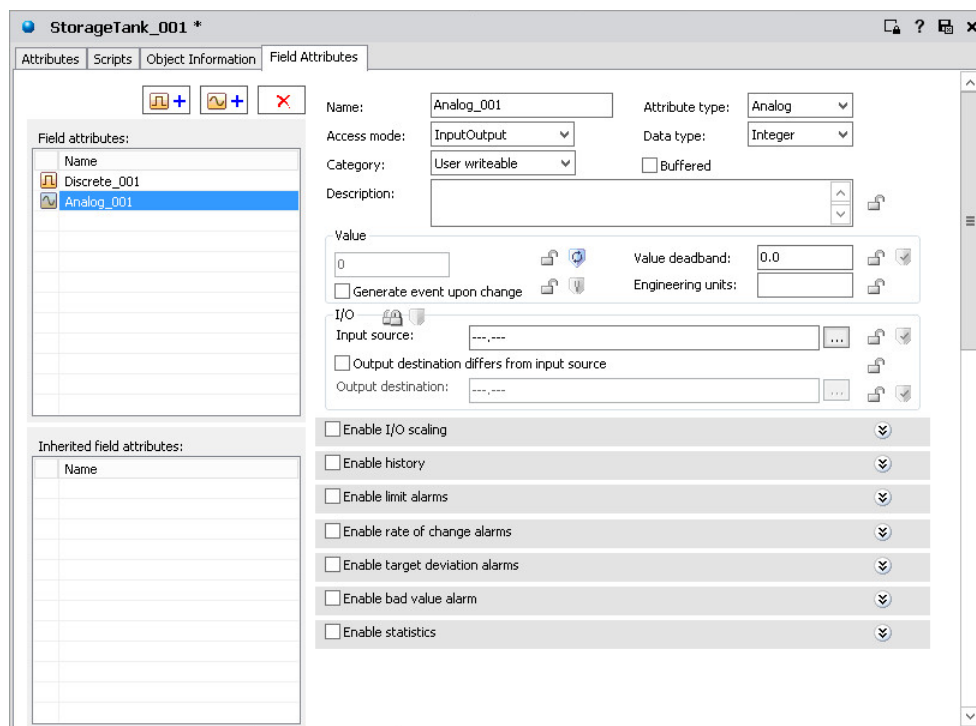
```
<Installation Path>\Program Files
(86)\ArchestrA\Framework\FileRepository\<YourGalaxyName>\Objects\<TheObjectID>\Help\1033
```

To add images to the Help file, place the images in the proper folder on the Galaxy Repository computer and use a relative path to those images in the HTML file.

For the previous example, place images in the \1033 folder or create an images folder under it.

## About the Field Attributes, UDAs, and Extensions Pages

Starting with Application Server 2014 R2, the functionality of Field Attributes, UDAs and attribute extensions has been combined into a single unit of functionality simply called an Attribute. The functions and capabilities of the **Field Attributes**, **UDAs**, and **Extensions** pages in the **Object Editor** have been combined and made available in the **Attributes** page.

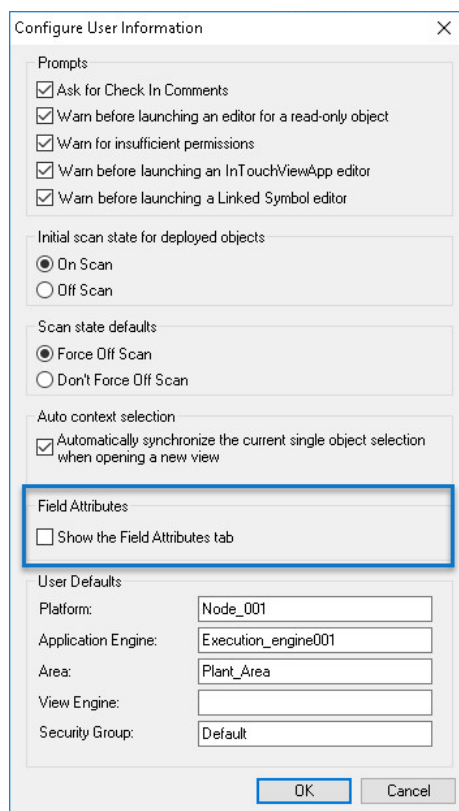


The **UDAs** and **Extensions** pages are no longer present in Application Server 2014 R2 and subsequent versions. By default, the **Field Attributes** page does not normally appear in Application Server 2014 R2 and later versions, but will under the circumstances detailed in the next section.

## View the Field Attributes Page

While the **Field Attributes** page is not displayed by default for Galaxies created with Application Server 2014 R2 and later versions, it will be displayed when you open a UserDefined Object in the Object Editor and any of the following conditions apply:

- The derived template or instance created from the \$UserDefined base template contains Field Attributes.
- You import UserDefined Objects created in a previous version of Application Server that contain Field Attributes.
- You have an existing Galaxy with Field Attributes already defined and migrate it to System Platform 2014 R2 or later.
- You select the **Show Field Attributes** option in the **Configure User Information** dialog. See *Configure User Information* on page 38 for more information.



## Determine Whether to Convert Field Attributes

Converting Field Attributes to Attributes provides the following advantages:

- It is easier and faster to work with Attributes, rather than Field Attributes.
- All of your Attributes will appear on the same Object Editor page if you convert Field Attributes.
- Attributes can provide performance advantages over Field Attributes during run time (the actual performance improvement can vary between systems).

You may not want to convert Field Attributes if:

- You have a large number of objects that use Field Attributes.
- You do not want to modify and test scripts that reference Field Attributes.

- The objects are working well using Field Attributes, and there is no active development on these objects. Additional effort will be required for converting, testing, and redeploying objects.

## Optimize Performance if You Use Field Attributes

If you choose to continue using Field Attributes, use the following guidelines to maximize performance:

- No more than 64 Field Attributes (maximum of 32 analog and 32 discrete) should be used for an individual object. If this limit is exceeded, the performance of object editing and check-in operations slows down significantly.
- Do not extend an existing Field Attribute by activating features on the **Attributes** page.  
For example, to add historization to an existing Field Attribute, select the **Enable history** option from the **Field Attributes** page.  
DO NOT use the Attributes page to add the History feature.
- Converting from Field Attributes to Attributes will greatly enhance performance, particularly if many Attributes are used.

## Convert Field Attributes to Attributes

You can choose which objects with Field Attributes to convert to Attributes, but only first-level derived objects can be selected. That is, you can only select templates and instances that are directly derived from the \$UserDefined base template. Objects derived from first-level templates or individual Field Attributes cannot be individually selected for conversion.

When you start the conversion process, you are presented with a list of all first-level templates and instances that contain Field Attributes. First-level templates without Field Attributes, but that have derived objects with Field Attributes, are also listed. From this list, you will select which objects to convert.

The conversion process is hierarchical and cascades to all child templates and instances of the selected objects. All Field Attributes contained in a template's hierarchy of derived objects are converted.

---

**Note:** Objects will be unavailable for selection (grayed out) if the object or any of its derived objects are deployed, checked out, or protected. See *Protecting Objects on Export* on page 102 for more information about protecting objects.

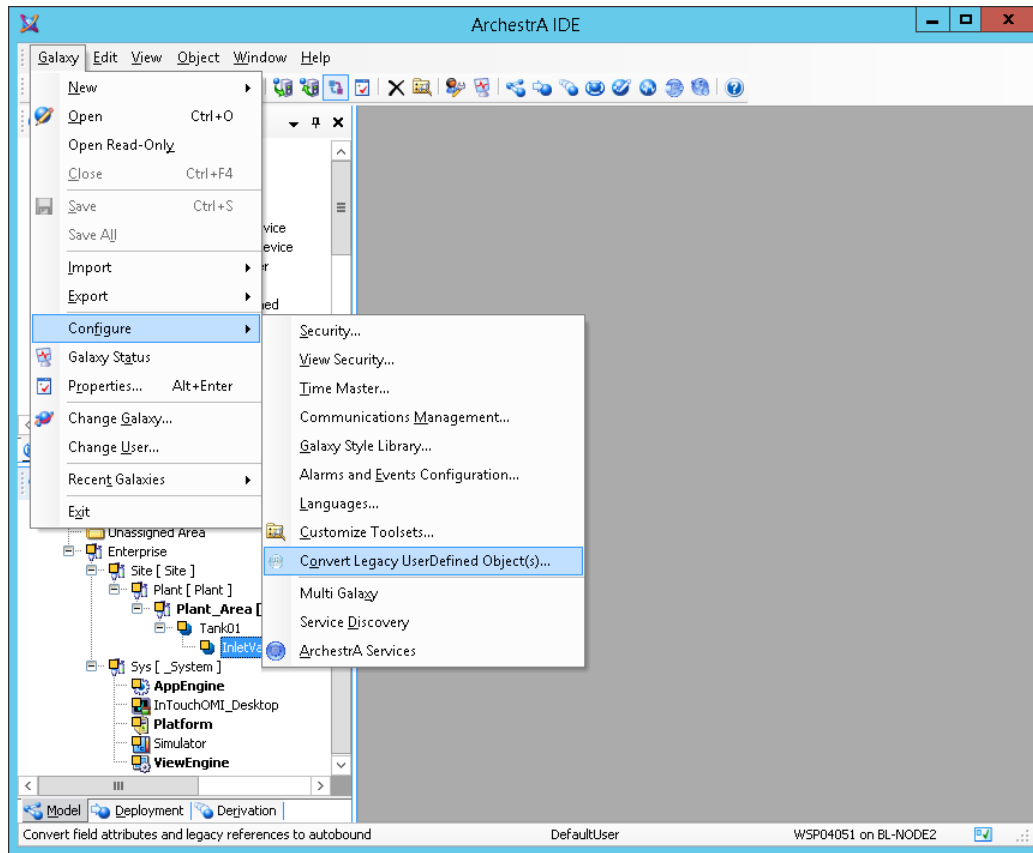
---

### To convert field attributes to attributes

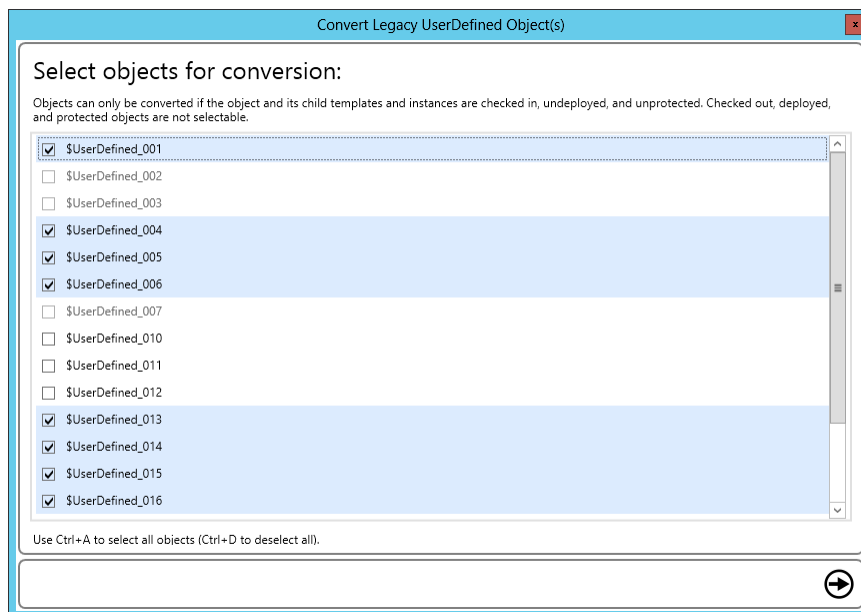
Before starting the conversion process, see *Special Considerations when Converting Field Attributes* on page 86 for a discussion of potential differences in behavior between converted Attributes and the original Field Attributes.

1. Back up all UserDefined Objects (UDOs) before starting Field Attribute conversion.
2. Ensure that all UDOs that you want to convert are checked in and not deployed. You will not be able to convert Field Attributes for objects that are deployed, checked out, or protected.

- From the Galaxy menu, select **Configure**, then **Convert Legacy UserDefined Object(s)...**



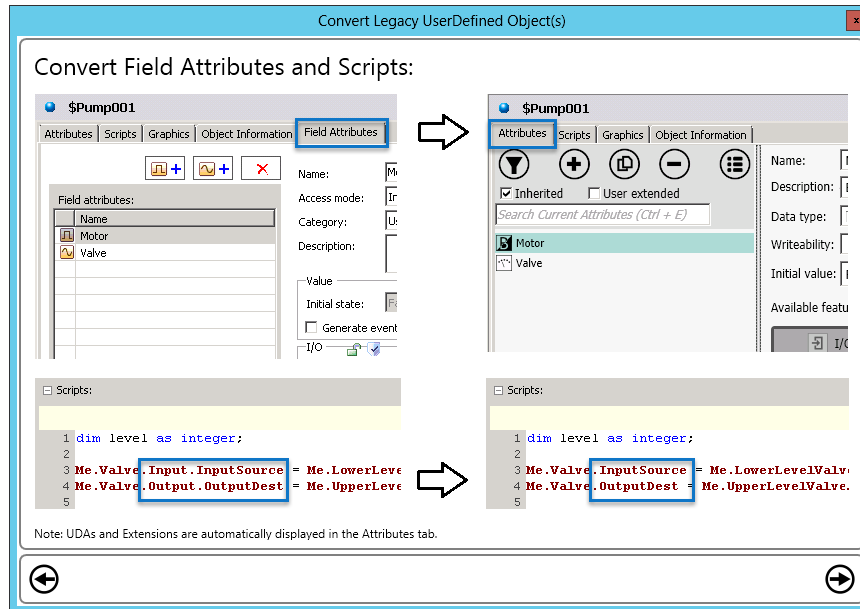
- When you select **Convert Legacy UserDefined Object(s)...** from the menu, a list of all first-level objects with Field Attributes is displayed. First-level templates that do not themselves include Field Attributes, but have child objects with Field Attributes, are also listed.




- Select the objects you wish to convert and click **next**.

**Note:** Objects will be grayed out and unavailable for selection if the object or any of its derived objects are deployed, checked out, or protected.

- The next screen is informational, and shows that Field Attributes will be consolidated onto the Attributes page. Click **next** to proceed.

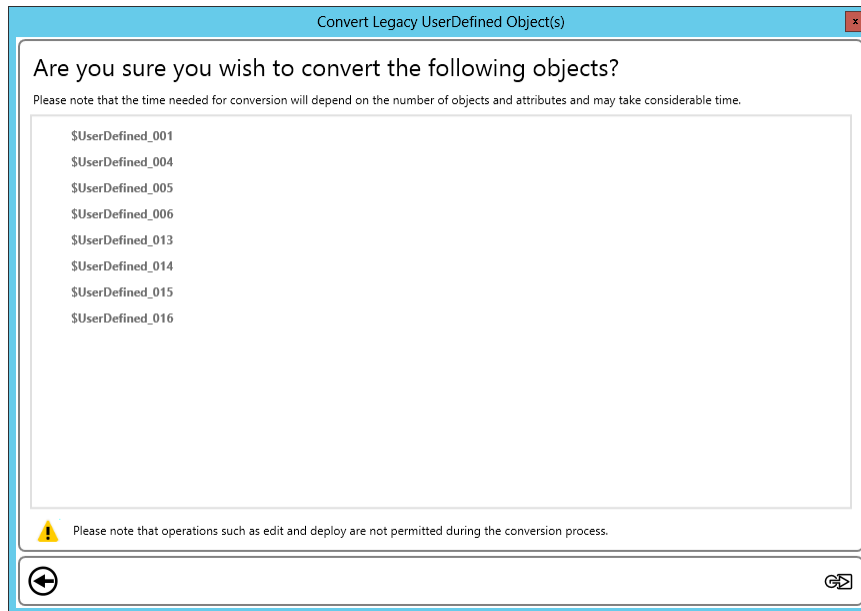


- The screen that follows is informational as well, and lists the limitations that apply to script conversion. These are:
  - Only field references that begin with "me." are converted.
  - Field references outside the selected objects are not converted.
  - Field references in scripts that use indirect functions or string manipulations are not converted.
  - Field references within Industrial Graphics are not converted.
- Click **next** to proceed.
- To begin converting Field Attributes for the selected first-level objects and their child objects, click the **Convert** button (  ).

**WARNING!** Once you start the conversion process, it cannot be cancelled. The time needed to convert Field Attributes depends on the number of objects with Field Attributes, the number of Field Attributes, and the depth and complexity of the object hierarchy. Field Attribute conversion requires at least as much time as Galaxy migration.

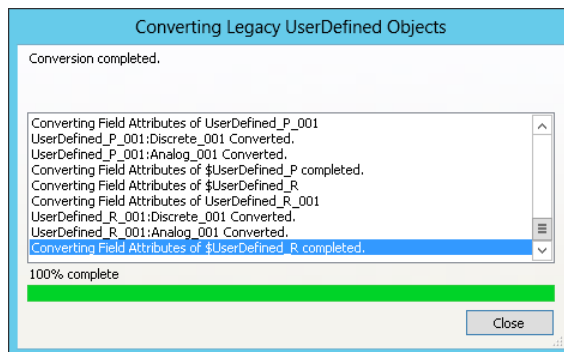
- Click **back** if you need to make any changes.

- To cancel, simply close the dialog.



**WARNING!** While the conversion process is active, you cannot deploy or edit objects.

- Each object is listed as its Field Attributes are converted. The status of the conversion process is also reported to the SMC Logger.



**Important:** In the event that Field Attribute conversion fails for an individual template, the template will remain checked out after the conversion process stops. Undo the checkout for the template, and then you can resume attribute conversion for any remaining objects. To find which template did not convert, check the SMC Logger.

- When all objects containing Field Attributes have been converted, click **Close**.

## Special Considerations when Converting Field Attributes

When you convert Field Attributes to Attributes, there may be some differences between the original Field Attribute and the converted Attribute.

- If the Field Attribute "Bad.Condition" was extended (for example, by adding historization), it will send alarm notifications after conversion, even if alarming was not enabled.
- Prior to conversion, the Field Attribute description is `Analog_001.Desc`. After conversion, this changes to `Analog_001.Description`.

- Prior to conversion the History Extension description points to `Analog_001.Desc`. After conversion, the History Feature points to `Analog_001.Hist.DescAttrName`.
- If a Field Attribute has "Generate event upon change" enabled, then the **Log change** feature (`LogDataChangeEvent`) will be added to the converted attribute.
- When the Field Attribute I/O option, "Output destination differs from input source," is selected and left unlocked, but **Output destination** is locked, the converted Attribute will have the following locking properties:
  - "Write to" will be locked. This is the same as the original Field Attribute property.
  - "Output destination differs from source" will be locked. This differs from the original Field Attribute property.

---

**Note:** Default names for Field Attributes begin with either "Discrete" or "Analog." The conversion process preserves this part of the Field Attribute name, even though this division does not apply to the converted Attributes.

---

## Limitations and Exclusions when Converting Field Attributes

Objects that are protected, checked out, or deployed will not be converted. If an object derived from a first-level template you want to convert is protected, checked out, or deployed, the first-level template will not be selectable for conversion. The entire hierarchy of the first-level object must be available for conversion (conversion works on an all or none basis).

The following Field Attribute features are not converted to Attribute features. You will have to be enter the information for them manually.

Field Attribute Feature	Equivalent Attribute Feature
Enable Deadband	Limit Alarms
Description	Description
Engineering Units	Eng units

## Update Scripts to Reference Converted Attributes

The conversion process applies certain changes to scripts that contain I/O references. In the conversion process, the Field Reference I/O namespace is truncated to match the Attribute namespace. These reference changes are as follows:

- `Me.Analog_001.Input.InputSource` converts to `Me.Analog_001.InputSource`
- `Me.Analog_001.Output.OutputDest` converts to `Me.Analog_001.OutputDest`
- `Me.Analog_001.Input.Value` converts to `Me.Analog_001.Value`

---

**Note:** Only Field Attributes that begin with "me." are converted. No other changes are made to scripts. Scripts within graphics are not converted.

---

While Field Attribute conversion updates I/O references within scripts, other attribute references are not updated. For example, while a Field Attribute description is truncated to "Desc," the description is not truncated for Attributes. Therefore, you will have to do some manual updating.

## Reference Objects Using the Galaxy Browser

Use the Galaxy Browser to browse for:

- Attributes of objects. You can quickly find an object attribute or attribute property and add a reference to it when you are configuring an object.
- Industrial Graphics.
- Graphic element properties.

The Galaxy Browser shows attributes, graphics, or attributes and elements, depending on what you are doing at the time you access the browser.

## Browse for Attributes

You use the Galaxy Browser to browse for:

- Attributes of objects, either instances or own relative references.
- Attributes of templates.

You can open the Galaxy Browser to browse for attributes from:

- Within an AutomationObject Editor. For example, from a script, from an attribute of type MxReference, or from a custom alarm message attribute field).
- Within an Industrial Graphic Editor. For example, to use in scripts, animation links and references, properties and custom properties.
- Within InTouch WindowMaker. For example, to use in a reference expression from an animation link or a script.

The Galaxy Browser shows objects in the left pane and the attributes associated with the current selection on the right pane. Only attributes that can be referenced at run time are shown. You can browse "Me." references for alarm messages only.

When you open the Galaxy Browser, the last browsed location for attributes is shown. The Galaxy Browser shows the object list based on the last used state (tag name or Hierarchical name). If the last used state of the browser was Tagname and the selected editor reference is a Hierarchical name, the browser opens in Tagname mode.

The status bar displays the attribute property name, and the it displays the graphic element attribute name and description.

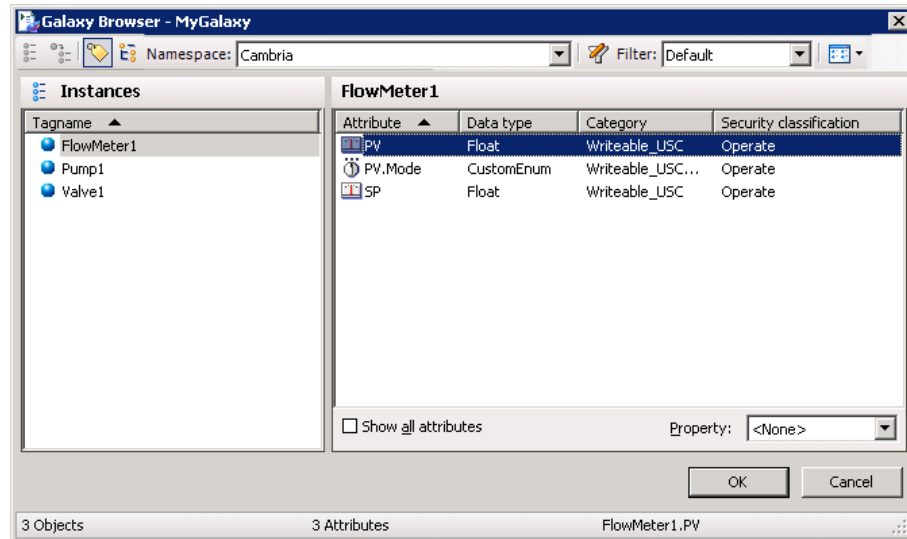
### To browse for attributes



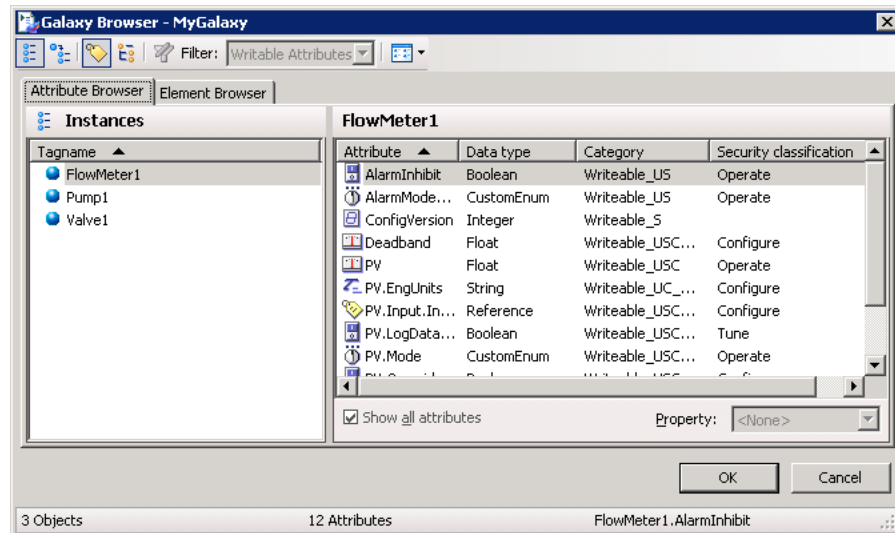
1. In any area on a page, click the **Browse** button, if available. The Galaxy Browser



opens.



If you are browsing for attributes to use with an Industrial Graphic, such as for an animation or script, the Galaxy Browser shows the attributes in an **Attribute Browser** tab.



- By default, the browser shows only those attributes that are frequently accessed. If you are viewing the attributes of an object for the first time, the right pane can be blank. Select the **Show all attributes** check box to show all of the object's attributes.



3. To filter the list of tag names, click the **Filter** button. For information about configuring a filter, see *Create a Filter for the Galaxy Browser* on page 94. To switch the content of the left pane between a **Tagname** and **Hierarchical Name** list of objects, click the **Show Tag name** or **Show Hierarchical name** buttons.
4. You do not have to explicitly make a selection in the **Property** list. If you only select an attribute (leaving **Property** set to <none>), the property of the attribute defaults to **Value**.

If the option in the Object Editor is already configured with an object reference, the **Attribute Browser** shows it or expands to the nearest matching object/attribute/property currently configured in the Galaxy.

If you selected text in the script editor, that text is used as the initial reference string and the browser finds the nearest attribute reference to the selected text

5. When you are done selecting the attribute/property, click **OK** to place the reference into the Object Editor and close the **Galaxy Browser**.
  - The fully-qualified reference string appears in the editor option.
  - If you are working in the script editor, the selected reference appears in the script at the current cursor position and replaces text that was selected.

## Browse with an ArcestrA OPC UA Client

You can use an instance of the ArcestrA OPC UA Client Service hosted by your Galaxy to browse a namespace in an OPC UA server.

You must prefix the item reference or attribute name with the Scope name/Service name defined in the OPC UA Client Service editor.

To access an OPC UA server item reference, use the following syntax:

```
<namespace>:<UAItemReference>
```

**Namespace** is the OPC UA client service scope name defined in the client service editor. For example "TestServer".

**UAItemReference** is the identifier of the item (the complete path of the OPCUA Server item, followed by the NodeId) in the OPC UA Server.

For example:

```
TestServer:PLC01.PLCObject1
```

For more information, see *Configuring and Deploying the OPC UA Service* on page 298

## View Attribute Details in the Galaxy Browser

When you view attributes in the Galaxy Browser, you see two areas. The objects shown in the left area include all of the logged in user's checked-out objects plus the checked-in versions of all other objects.

---

**Important:** The Galaxy Browser shows only the Primary AppEngine and its attributes of a redundant pair. Any Backup AppEngine is not shown.

---

The right area shows the attributes of the object selected in the left pane. Depending on the attribute selected, you can see these properties:

<none>                      Automatically defaults to the Value property of the selected attribute.

Category	Determines when and where the attribute's data exists (for example, configuration or run time), which users can write to it, and whether the attribute is lockable or unlockable.
Dimension1 (only for arrays)	Returns the dimension of the attribute if it is an array.
Locked	Determines whether the attribute is currently locked. Valid values are: Unlocked LockedInMe LockedInParent.
Quality	The quality of the attribute as defined in the OPC Draft 3.0 quality definition. OPC quality is stored and transported as a 16-bit value. OPC quality is stored for an attribute as a current quality, and it can be historized and sent to clients.
SecurityClassification	Determines which permissions a user has with respect to the attribute when using an AVEVA application in the run-time environment. Relevant only for attributes that can be written to by users in the run-time environment. If an attribute has no security, this column is blank.
Data Type	The data type of the attribute: <ul style="list-style-type: none"><li>• Integer</li><li>• Boolean</li><li>• Float</li><li>• Double</li><li>• String</li><li>• Internationalized String</li><li>• Time</li><li>• ElapsedTime</li><li>• ReferenceType</li><li>• CategorizedStatusType</li><li>• DataTypeEnum</li><li>• SecurityClassificationEnum</li><li>• DataQualityType</li><li>• CustomEnum</li><li>• CustomStruct</li></ul> For information about each of these, see the help for the object.

Value	<p>The primary value of the attribute.</p> <p>Sometimes, a list of numbers is included in the <b>Property</b> list. Those numbers map to single bits in an integer attribute's Value property. Valid bit field specifiers are:</p> <p>.00 (least significant bit)</p> <p>.01 .02 .03 .04 .05 .06 .07 .08 .09 .10 .11 .12 .13 .14 .15 .16 .17 .18 .19 .20 .21 .22 .23 .24 .25 .26 .27 .28 .29 .30</p> <p>.31 (most significant bit)</p>
-------	--

---

**Important:** Bit field specifiers are not allowed for integer arrays. Although bit field access is only supported in integers, they appear to be allowed for data types besides integer because they do not cause a warning during configuration. They cause errors in the run-time environment.

---

## Browse for Graphics

You use the Galaxy Browser to browse for graphics from:

- The Industrial Graphic Editor, when editing a graphic from the Galaxy, being either a graphic from an object (Template or Instance) or a graphic from the Graphic Toolbox.
- InTouch WindowMaker when editing a managed InTouch application hosted by the Galaxy.

The graphic currently being edited (or browsed from) does not appear in the list of graphics.

Within the same user session, the Galaxy Browser remembers the last browsed location for graphics and presents it whenever called as the starting location so that context is kept. Initial default location for graphic browsing is the Graphic Toolbox with the root selected (Galaxy node).

You can use the Galaxy Browser to browse graphics from:



The Graphic Toolbox. The browser shows the Graphics Toolbox toolset organization on the tree in the left pane and a list of the graphics contained in the currently selected node (Galaxy node or toolset node) in the right pane.



AutomationObject templates. The browser shows the Template Toolbox in the left pane and the right pane shows the graphics associated with the currently selected template in the right pane.



AutomationObject instances. The browser shows a flat list of existing instances of objects in the left pane. The right pane shows the graphics associated with the currently selected instance. You create or apply filters to reduce the scope of the instances shown in the left pane.



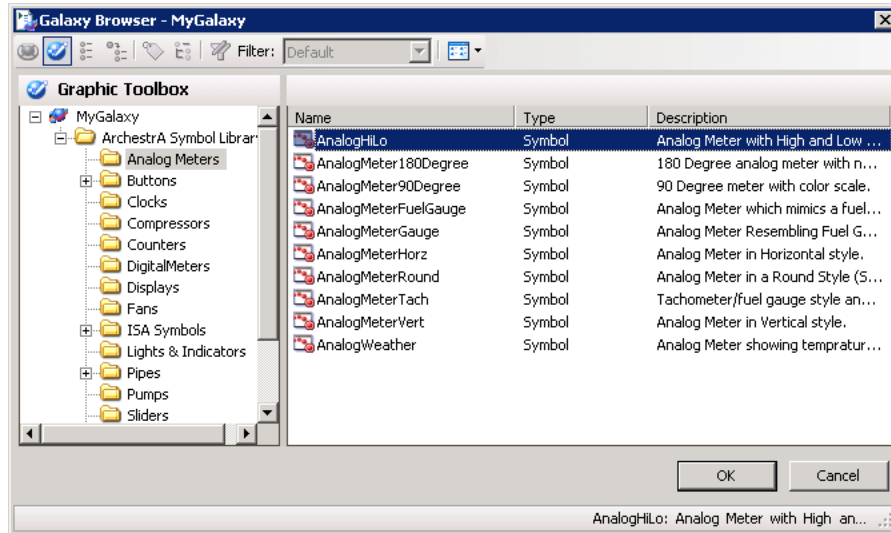
Relative references. This is possible only when you edit a graphic belonging to an object and browse for graphics from that specific object.

### To browse for graphics from the Graphic Editor



1. Click the **Embed Graphic** button in the Graphic Editor. The Galaxy Browser opens,

showing the location of the graphics on the left pane and the graphics associated with the current selection on the right pane.



2. Select a graphic from the list and then click **OK**.
3. Click in the canvas to place the graphic.

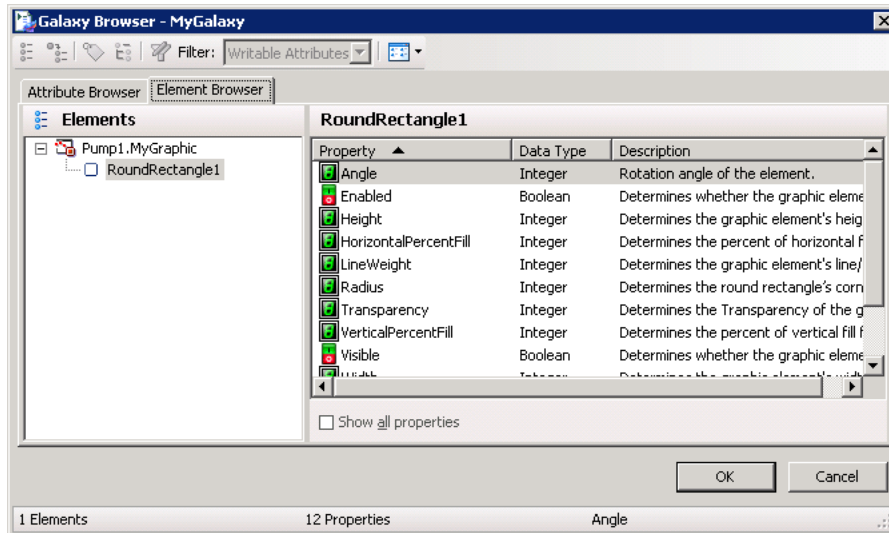
## Browse for Element Properties

If you are working on a graphic, you can create references to properties of other graphics. For example, you can reference another graphic's properties from an animation link or script. You can browse the properties of all elements on the canvas or custom properties.

## To browse for element properties



1. In any area on a page, click the **Browse** button, if available. The Galaxy Browser opens.



2. Click the **Element Browser** tab.
3. By default, the browser shows only those properties that are frequently accessed. If you are viewing the properties of an element for the first time, the right pane can be blank. Select the **Show all properties** check box to show all of the object's attributes.
4. Select the property and then click **OK**.
  - o The fully-qualified reference string appears in the option.
  - o If you are working in the script editor, the selected reference appears in the script at the current cursor position and replaces text that was selected.

For more information, see "Working with Element Styles" in the *Creating and Managing Industrial Graphics User Guide*.

## Create a Filter for the Galaxy Browser

You can create one or more filters that limit the list based on the object name or common attributes. You can also configure the columns you want to show for the list.

The Default filter provides an unfiltered list of objects and attributes. It cannot be edited.

## To create a filter



1. Click the **Filter** icon. The **Edit Filter** dialog box appears.

2. Click the **Plus** button and type a name for your new filter.
3. Click the **Filter** tab.
4. Configure the filter details.
5. Click the **Display** tab.

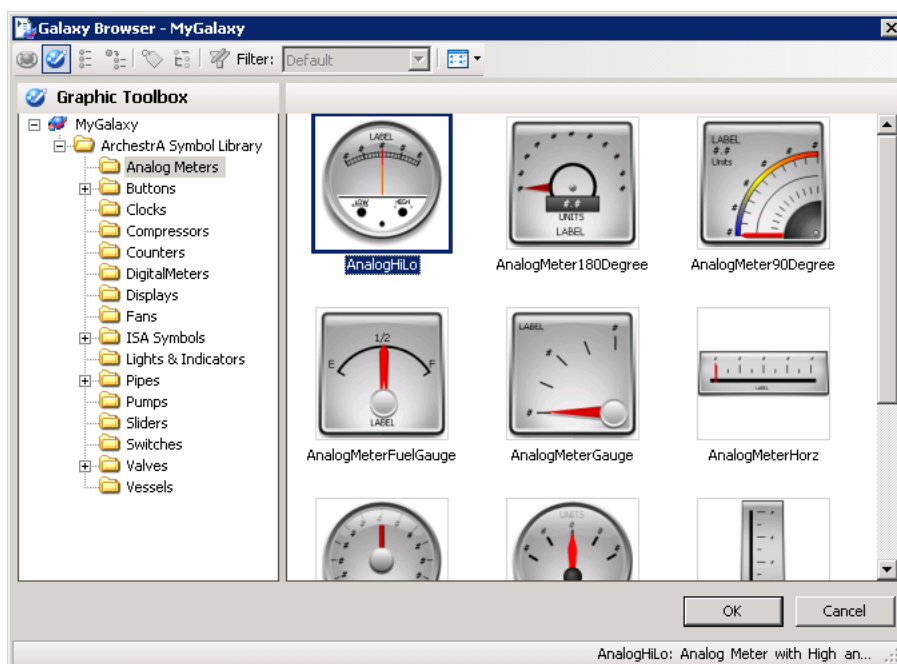
6. Configure the columns to show in the right pane of the **Galaxy Browser**. Use the up and down arrows to set the order for the columns.
7. Click **OK**. The new filter appears in the **Filter** list.

## Change How Information is Shown in the Galaxy Browser

You can change how information shown in right pane of the Galaxy Browser. You can view:

- A list of only the attribute or graphic names.
- A list of details for attributes or graphics.
- Named graphic icons.
- Thumbnails for graphics.

The following figure shows graphic thumbnails.





# CHAPTER 4

## Managing Objects

After you create several objects, like templates, you need to manage them. For example, you need to check objects in and out, you need to validate objects, and you may need to rename objects. You can also export and import objects, allowing you to reuse objects created in one Galaxy in another Galaxy.

### Check Out Objects

To make changes to an object, you must check out the object. Then, you can modify the object and save private versions of it before checking the object in for other users to use. You can select more than one object for checking out at the same time.

The Galaxy marks the objects as checked out to you and it updates the object's **Change Log** that you can view in the **Properties** dialog box. A check mark is shown next to an object's icon in the IDE. No one else can check out the object until you check it back in or until you perform an Undo **Checkout operation**. However, others can open the object for read-only viewing.

To find objects that have been checked out, use the **Find** dialog box.

#### To check objects out

1. In the **Template Toolbox** or **Application views**, select the objects you want to work with.
2. On the **Object** menu, click **Check Out**. Or, open the Object Editor. An object is automatically checked out to you when you open its editor.

The Object Editor opens. You are ready to make your changes.

### Check In Objects

After you finish making your changes, you check the object back into the Galaxy. When you check the object back in, a dialog box prompts you to enter comments about changes you made.

You can turn this dialog box off if you do not want to enter information about your changes. For more information, see *Customizing Your Workspace* on page 37.

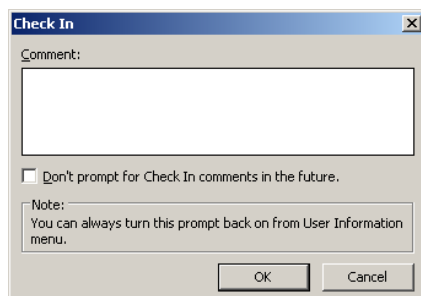
---

**Note:** If the object was automatically checked out when the editor was started and you close the editor without making any changes to the object's configuration, an undo-checkout is automatically performed.

---

#### To check in an object to the Galaxy database

1. In an **Application view** or the **Template Toolbox**, select the object after you are finished making your changes.
2. On the **Object** menu, click **Check In**. The **Check In** dialog box appears.



3. Type any comments you want and click **OK**.

## Validate Objects

Objects need to be validated before they can be deployed. An object validates its configuration either when you are configuring it, or typically when you save that configuration to the Galaxy database.

Validating an object's configuration includes:

- Checking allowable attribute value ranges.
- Compiling its scripts.
- Verifying its attribute references.
- Validating its extensions.
- Validating other configuration parameters that are unique to the object.

---

**Important:** Script validation on a template does not resolve references used in the script. For example, references to attributes that do not exist will not be discovered.

---

Typically, each option on the Object Editor that requires a string or numeric input has an allowable range of inputs. If you type an input outside the allowable range and then try to change the Object Editor page, close the Object Editor or save the object's configuration, a message appears about the input error, showing the allowable range.

### To open the Validation area

- On the **View** menu, click **Operations**. The **Validation** area opens.

## Validate Scripts and Other External Components

Some objects depend on external components to run, such as script function libraries and references to other objects' attributes. The status of these external components can change, perhaps disabling some capability of the object.

For example, an object refers to a value of an attribute of another object, which is subsequently deleted. This will result in the remaining object going to a Warning status.

Normally, the system will update the validation status of an object when the missing script function or object/attribute is later added to the system. But there are a few cases where the status of an object needs to be manually validated by the user.

For example:

- When importing scripts and script libraries, there are cases when the script will import before the associated library and validate incorrectly, and
- When graphics associated with an object are imported along with a graphic they embed, the containing graphic may be imported first and validated incorrectly.

In each of these situations, the object may incorrectly have a status of either Bad or Warning. In this case, you may want to manually validate the object to update its status, especially if the status is preventing the object from being deployed. For more information, see *Validate an Object Manually* on page 99.

Two kinds of indicators are shown in the object icons:

- Deployment status for instances only.
- Configuration status for templates and instances.

## Validate an Object Manually

After you check an object in, you can verify that an object's configuration is valid and update its status by manually validating it. You can use the **Template Toolbox**, the **Application views** or the **Find** dialog box to find objects that need to be validated.

To validate all objects in the Galaxy, validate the Galaxy object.

---

**Note:** For a large Galaxy this is potentially a time consuming operation, and should be used only when necessary.

---

You can select more than one object for validation.

If an object is being edited, validation may not be performed. Also, if validation is in process on an object, other operations you start on the object will fail.

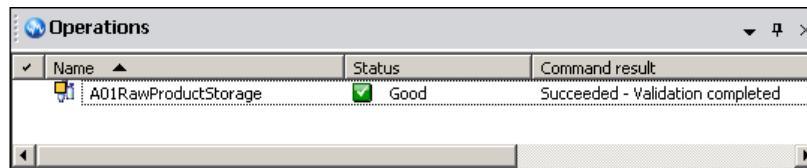
---

**Note:** You cannot cancel validation operations.

---

### To manually validate one or more objects

1. In the **Template Toolbox**, the **Application views** or the **Find** dialog box, select the objects you want to manually validate.
2. On the **Object** menu, click **Validate**. The **Operations** view in the IDE opens.



3. Continue using the IDE to perform other operations, if needed, while validation is going on, including work on other objects in the Galaxy. If you are validating a Galaxy, then you must leave the Galaxy alone until the validation process is complete.
4. When the validation process is complete, you see the results of the validation in the **Operations** View. If the validation failed on an object, you see a message. Correct the problem and validate again.

---

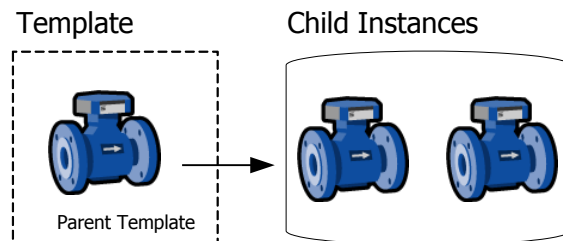
**Note:** You can also see the errors or warnings that led to an object having a status that is not Good by looking at the Error/Warnings tab within the object's Properties.

---

## Create an Instance

After you create templates, you can create instances. Creating instances makes a specific object from a template, with all the characteristics and attributes of the template.

After you have created an instance of an object, it can be deployed.



You can also customize an instance, if needed. For example, you can have a valve template. When you create an instance of that valve, you can specify the inputs and outputs for that specific valve on your factory floor.

### To create an instance

1. Select the template you want to use for the instance. For example, to create a valve instance, select a valve template.
2. On the **Galaxy** menu, click **New** and then click **Instance**. An instance is created.
3. Rename the instance. Select the instance. On the **Edit** menu, click **Rename**. Type the new name. Instance names can be up to 32 alphanumeric characters. You cannot use \$ as the first character. The name must include at least one letter. Instance names cannot include spaces.
4. To move the new instance in the **Deployment** view or **Model** view, drag the instance to the new location.

You are ready to configure the instance, if needed. For more information, see *Edit Objects* on page 68.

## Rename an Object

You can rename an object. Although you can change an object's containment relationship with another object, you cannot directly rename an object's hierarchical name. You can rename its tagname and contained name, and its hierarchical name will change automatically. See *Rename Contained Objects* on page 67 for more information about renaming contained objects.

Object names must be unique within each namespace, not within the Galaxy.

- Template names can be up to 32 alphanumeric characters, including the required \$ as the first character. The second character cannot be \$ and the name must include at least one letter. You cannot use spaces in an object name.
- Instance names can be up to 32 alphanumeric characters. You cannot use \$ as the first character. The name must include at least one letter. You cannot use spaces.

---

**Note:** You cannot use the following reserved names for objects: Me, MyContainer, MyArea, MyHost, MyPlatform, MyEngine and System.

---

An object can have three kinds of names if it is contained by another object. The three names include:

Name	Description
Tagname	The unique name of the individual object. For example, <code>Valve1</code> .
Contained name	The name of the object within the context of its container object. For example, the object whose Tagname is <code>Valve1</code> may also be referred to as <code>Tank1.Outlet</code> , if <code>Tank1</code> contains it and it has the contained name "Outlet".

Name	Description
Hierarchical Name	<p>Hierarchical names are fully-qualified names of contained objects that include the name of the objects that contain it.</p> <p>Because the object that contains it may also be contained, there are potentially multiple hierarchical names that refer to the same object.</p> <p>For example, if:</p> <p>"Reactor1" contains Tank1 (also known within Reactor1 by its contained name "SurgeTank").</p> <p>"Tank1" contains Valve1 (also known within Tank1 by its contained name "Outlet").</p> <p>Valve1 could be referred to as:</p> <p>"Valve1"</p> <p>"Tank1.Outlet"</p> <p>"Reactor1.SurgeTank.Outlet".</p>

When you rename an object, references from other objects to the object being renamed can be broken. Objects deployed with broken references receive bad quality data during run time.

**Note:** Some objects may refer to themselves or to parent/host objects up in the parent/child hierarchy. References that go up or down the hierarchy to refer to other objects are called relative references. Objects with relative referencing are updated automatically if you rename them.

After renaming, all IDEs connected to the Galaxy show the new object name.

### To rename an object's tagname

1. Select the object you want to rename.
2. On the **Edit** menu, click **Rename**.
3. Type the new name for the object.
4. When you are done, press **Enter**.

## Delete an Object

You can delete both templates and instances with the following exceptions. You cannot delete:

- Deployed instances
- Containers for other objects
- Objects checked out by other users
- Templates that have children (derived templates or instances)

**Note:** Make sure you correctly select the objects you want to delete. After you delete an object, you cannot undelete it. You must recreate it.

### To delete an object from the Galaxy

1. In the **Template Toolbox** or **Application views** area, select the object to delete. Select multiple objects by using **Shift+click** or **Ctrl+click**.
2. On the **Edit** menu, click **Delete**. When the message appears, confirm you want the object deleted and click **Yes**.

## Export Objects

You can export some or all of your Galaxy objects. When you export, you are exporting the objects' associated templates, configuration state, and containment state of those objects. The information is saved in an .aaPKG file.

After the Galaxy objects are exported, you can import them into the same or another Galaxy.

If your objects have scripts associated with them, you need to export the script library separately. For more information about exporting script libraries, see *Exporting Script Function Libraries* on page 104. For more information about scripts and script libraries, see the *Application Server Scripting Guide*.

If you are exporting an InTouchViewApp object and want to include graphics that are not embedded in the application, such as graphics called from a script, you need to embed the specific desired graphics in the InTouchViewApp.

---

**Note:** You can associate all Galaxy graphics with an InTouchViewApp for backup, restore, and deployment operations, but not for import and export operations.

---

Before you start, make sure all objects you want to export are checked in. If an object selected for export is checked out, the checked in version of that object is exported instead. This can lead to old versions of objects being exported.

Exporting an entire Galaxy is different than backing up the database. Unlike the case with backups, change logs for the objects are not exported. When you export objects, only the related security information for the specific object is exported.

### To export an object

1. In the **Template Toolbox** or **Application Views**, select one or more objects to export.
2. On the **Galaxy** menu, click **Export** and then click **Automation Object(s)**. The **Export Automation Object(s)** dialog box appears.  
To export all of the objects in the Galaxy, on the **Galaxy** menu, click **Export** and then click **All AutomationObjects**.
3. In the **Export** dialog box, browse to a path and type a name for the exported file.
4. Click **Save**. The file is saved with the specified name and an .aaPKG extension.
5. When the export is complete, click **Close**. Now you can import the .aaPKG file into another existing Galaxy.

## Protecting Objects on Export

Protect symbols and derived templates by flagging the objects as protected in the galaxy database. Protected symbol and template behavior is similar to that of a base template. This option is available through a specialized form of the **Export Object(s)** operation.

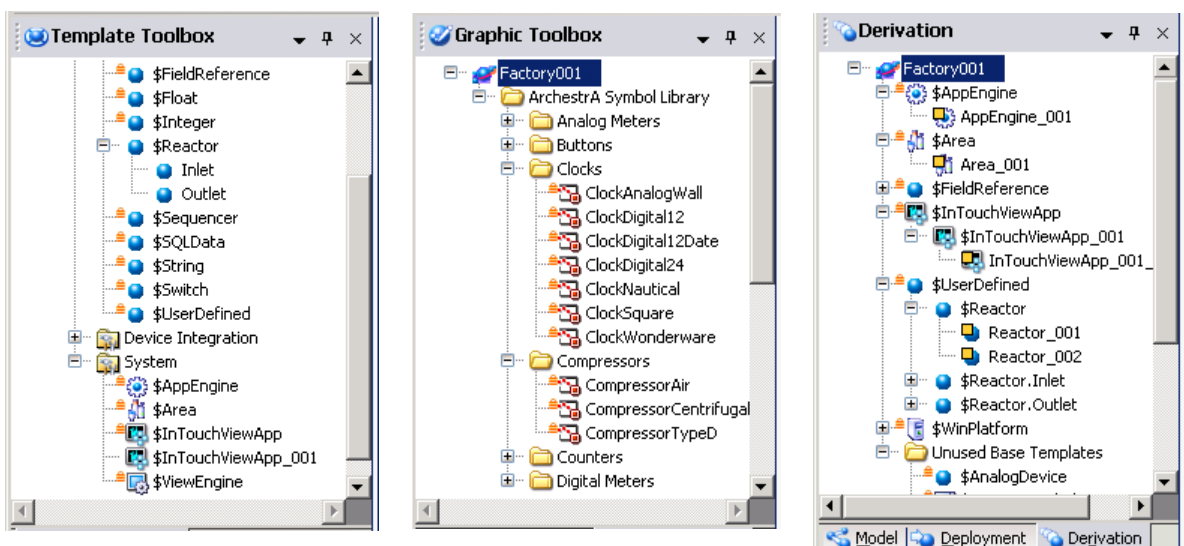
System Integrators and other system designers can use this functionality to protect objects designed in a master galaxy that are intended for use in production galaxies or galaxies on run-time nodes.

## About Protecting Objects on Export

Protecting an object on export does not change the object in the galaxy from which it was exported. Protection is effective only on import of protected objects. Specific behavior is described as follows:

Element or Function	Description
Base Templates	Are protected by default and cannot be checked out, edited or renamed.
Protected templates and symbols	Cannot be checked out, edited, or renamed, but can be deleted. A template derived from a protected template is unprotected.  Protected templates and symbols are marked with a lock icon.
Ancestor objects	Are protected in the exported .aaPKG file when a child object is protected.
Protection effective	Protection is effective on import of protected objects. A protected object retains its protected status even when exported using the standard <b>Export Object(s)</b> workflow.
Instances	Cannot be protected. Only templates and symbols can be protected.
Select both template and instance for export	The option to export objects as protected is disabled. Only templates and symbols can be protected.
Export workflow	The export workflow is the same as for all objects except that you can select the option to export <b>As Protected Object(s)</b> .
Graphics	Symbols and client controls directly or indirectly embedded in a protected graphic are also protected.  Protected symbols can be opened in the Graphic Editor as read-only.
Scripts	If the Execution type of a script is locked, the script and its attributes will not be visible in the <b>Object Editor</b> or <b>Properties</b> window. This also applies to all child objects.

Protected objects are marked in the **Template Toolbox**, **Graphic Toolbox**, or Application views (**Model**, **Deployment**, or **Derivation View**) with a gold-colored padlock icon.



## Exporting Objects as Protected

The export objects workflow is the same as for all objects with the exception of specifying the export of selected objects as protected.

### To protect objects on export

1. In the **Template Toolbox**, **Graphic Toolbox**, or in the **Application Views (Model, Deployment, or Derivation)**, select one or more templates or symbols to export.
2. On the **Galaxy** menu or context menu, click **Export** and then click **As Protected Object(s)**. The **Export Automation Object(s)** dialog box appears.
3. In the **Export Automation Object(s)** dialog box, browse to a path and type a name for the exported file.
4. Click **Save**. The file is saved with the specified name and an .aaPKG extension.
5. When the export is complete, click **Close**. Now you can import the .aaPKG file into another existing Galaxy.

## Exporting Objects with I/O Auto Assignment

Objects configured for I/O auto assignment can be exported the same way as other objects. See *Configuring Objects* on page 121 for additional information.

Application objects configured for I/O auto assignment are linked to DI objects and scan groups. Therefore, when you import these objects into a different Galaxy, the objects will look for DI object and scan group names that match the linkages that were made in their originating Galaxy. If a matching DI object and scan group are not found, all I/O auto assignment information is discarded, and the application objects are placed under the "Unassigned IO Device" folder in the **IO Devices** view.

- When exporting application objects, some I/O mapping assignments may be unavailable when importing the objects into a different Galaxy. Application objects that do not find a DI object and associated scan group that match their I/O references will lose their I/O mapping assignments and will have to be remapped.
- When exporting an application object with overrides in the I/O mapping table, overrides are preserved if a DI object and scan group that match its assignment from the original Galaxy are present in the new Galaxy.

---

**Note:** The I/O mapping references are permanently deleted if matching DI objects and scan groups are not found. The I/O references will not reconstitute, even if you later add the scan groups and DI objects to which the application objects were formerly mapped.

---

- When exporting DI objects, there are no special considerations related to I/O assignments. I/O references for application objects will not change when the DI objects are imported into another Galaxy.

---

**Note:** When importing a DI object into a different Galaxy, an existing DI object with the same name as the imported DI object may be overwritten. This can break device linkages to application objects previously configured within the Galaxy.

---

## Exporting Script Function Libraries

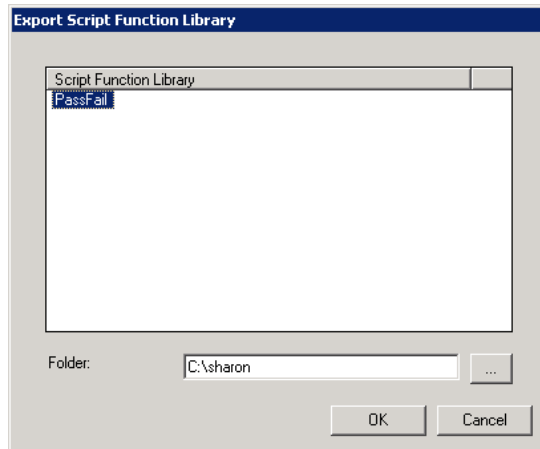
If you want to export objects that use scripts, the scripts are exported with the object.

Some scripts include functions that depend on external files called script function libraries. In this case, you must export the script function libraries separately.



## To export a script function library

1. On the **Galaxy** menu, click **Export** and click **Script Function Library**. The **Export Script Function Library** dialog box appears.



2. In the **Script Function Library** list, select the library or libraries you want to export. If needed, browse to folder where you keep your script libraries.
3. Click **OK**. The selected script library is exported. Each script is named with the name of the script and an **.aaSLIB** file name extension.
4. When the export is complete, click **Close**. Now you can import the **.aaSLIB** file into another existing Galaxy.

## Exporting Objects with Linked Content

When an object with linked content is exported, the linked content is exported along with the object. The exported object or objects can be a template, instance, or both. The linked content types that are exported include symbols, layouts, and external content items.

After import, relative references within linked symbols and layouts resolve to the current owning object.

## Importing Objects

You can reuse objects from another Galaxy in your Galaxy. This saves you a lot of time if the objects are already set up in another Galaxy.

Importing instances previously exported from a Galaxy retains previous associations, when possible, such as assignment, containment, derivation, and area.

You can import objects from exported **.aaPKG** files or from an **.aaPDF** file. An **.aaPDF** file contains the configuration data and implementation code for one or more base templates. It is created by a developer using the ArchestrA Object Toolkit.

You cannot have two objects with the same name or more than one copy of the same version of an object in the same Galaxy. When you import an object, you can choose how you want naming and version conflicts handled.

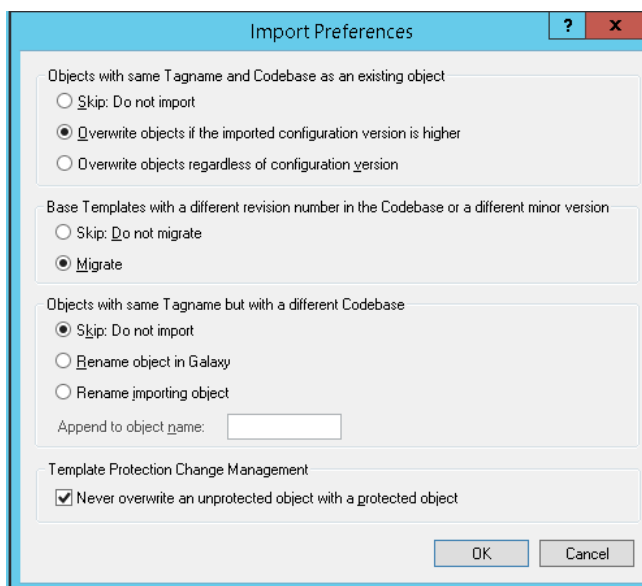
You should perform an "Upload Runtime Changes" before importing a new version base template if instances of the template are deployed. This saves changes made at Runtime to the Galaxy database.

Objects that were created in Application Server 2014 R2 or later cannot be imported into a Galaxy running an older version. For example, you cannot import an object created in Application Server 2017 into a Galaxy running Application Server 2012.

**Note:** The Application Server version is not the same as the Codebase version. Objects with newer Codebases can be imported.

## To import objects

1. On the **Galaxy** menu, click **Import** and click **Object(s)**. The **Import AutomationObject(s)** dialog box appears.
2. Browse for the file with either an **.aaPKG** or an **.aaPDF** extension. You can select more than one file. Click **Open**. The **Import Preferences** dialog box appears.



3. In the **Objects with same Tagname and Codebase as an existing object** area, select one of the following:
  - Skip: Do not import** leaves the existing object unchanged.
  - Overwrite objects if the imported configuration version is higher** replaces the existing object with the object being imported if the imported object has a newer configuration. This is the default.
  - Overwrite objects regardless of configuration version** replaces the existing object regardless of whether the existing object has an older configuration or the same configuration.
4. In the **Base Templates with a different revision number in the Codebase or a different minor version** area, select one of the following:
  - Skip: Do not migrate** - objects with an older codebase are not migrated when a newer codebase exists in the Galaxy.
  - Migrate** - objects with an older codebase are migrated when a newer the replacement object. For more information about migrating, see *After You Import* on page 119.
5. In the **Objects with same Tagname but with a different Codebase** area, select one of the following:
  - Skip: Do not import** leaves the existing object unchanged.
  - Rename object in Galaxy** imports an object with a matching tagname but a different codebase from the existing one. The existing object is not overwritten but is renamed.
  - Rename importing object** imports an object with a matching tagname but a different codebase from the existing one. The existing object is not overwritten. The imported object is renamed.

6. In the **Template Protection Change Management** area, click the checkbox to prohibit overwriting an unprotected object with a protected object (default). Clear the checkbox to allow overwrites.
7. Click **OK**. The import process starts.
8. When the import process is complete, you can start using the objects you imported.

## Importing Protected Objects

The import objects procedure is the same for all objects, protected or unprotected. The following describes import preferences and results specific to importing protected objects:

Import Preferences	Result for Protected Objects
<p><b>Objects with same Tagname and Codebase as an existing object:</b></p> <ul style="list-style-type: none"> <li>• <b>Skip: Do not import</b></li> <li>• <b>Overwrite objects if the imported configuration version is higher (default)</b></li> <li>• <b>Overwrite objects regardless of configuration version</b></li> </ul>	<p>Overwrite preferences are governed by the <b>Template Protection Change Management</b> option, "Never overwrite an unprotected object with a protected object" (default).</p>
<ul style="list-style-type: none"> <li>• <b>Overwrite objects regardless of configuration version</b></li> </ul>	<div data-bbox="646 747 1380 1415" data-label="Image"> </div>
<ul style="list-style-type: none"> <li>• <b>Overwrite objects regardless of configuration version</b></li> </ul>	<ul style="list-style-type: none"> <li>• Overwrites can occur if both the packaged and galaxy objects have the same protection status.</li> <li>• A protected object can overwrite an unprotected object only if specifically permitted by clearing the "Never overwrite ..." option.</li> <li>• When overwriting a protected parent object, protection is not cascaded to child objects. Unprotected child objects remain unprotected.</li> </ul>
<p><b>Base Templates with newer or older Codebases (or minor version updates):</b></p> <ul style="list-style-type: none"> <li>• <b>Skip: Do not migrate</b></li> <li>• <b>Migrate</b></li> </ul>	<p>Import and migration of base templates is unchanged by protected object import functionality.</p>

Import Preferences	Result for Protected Objects
<p><b>Objects with same Tagname but with a different Codebase:</b></p> <ul style="list-style-type: none"> <li>• <b>Skip: Do not import</b></li> <li>• <b>Rename object in Galaxy</b></li> <li>• <b>Rename importing object</b></li> </ul>	<p>Object renaming on import can occur only if the object to be renamed is not protected.</p> <ul style="list-style-type: none"> <li>• If the object in the Galaxy is protected, and the importing object is not, and <b>Rename importing object</b> is selected, the importing object is renamed to have the suffix "_new".</li> <li>• If the object in the Galaxy is unprotected, and the importing object is protected, and <b>Rename object in Galaxy</b> is selected, and the <b>Never overwrite an unprotected object with a protected object option</b> is not selected, the object in the Galaxy is renamed with the suffix "_old".</li> </ul>

## Importing Objects with I/O Auto Assignment

Objects configured for I/O auto assignments can be imported the same way as other objects. See *Importing Objects* on page 105 for additional information.

When importing objects that utilize I/O auto assignment, be aware of the following:

- If you are importing DI objects, there are no special considerations related to I/O auto assignment. I/O device mapping is not affected. An imported DI object will overwrite an existing DI object with the same name.
- If you are importing application objects, different outcomes can result, depending on the DI objects and scan groups that already exist in the Galaxy.
  - I/O device mapping is recreated, if possible, for each application object that is successfully imported and for which a DI object and scan group are found that match the existing I/O assignment. This includes all user-configured I/O attribute overrides.
  - Objects for which a DI object and scan group with matching names cannot be found are moved to the unassigned area. All I/O information for those objects is discarded. This includes default I/O auto assignment information as well as any custom configured I/O attribute overrides. You will need to assign these objects to a DI object and scan group in the new Galaxy and recreate all user-configured overrides. None of this information is preserved for application objects that do not find a matching DI object and scan group.

## Importing Objects with Linked Content

When you import an object with linked content, the object's links are restored and resolve to the current owning object. See *Exporting Objects with Linked Content* on page 105 for more information.

## Importing a SQLData Object

The SQLData object is used to store and retrieve data from a SQL Server database. Typical applications for this object include basic recipe management and database capture of attribute values.

The SQLData object can be configured to use one of the following authentication modes:

- Windows Integrated Security
- Windows Account
- SQL Server Authentication

When you import an object from another galaxy that is configured with SQL Server Authentication, the encrypted password does not follow the object, and the object will be in a warning state in the new galaxy. You must open the object in the Object Editor and enter a new password to remove the warning state.

## Importing Script Function Libraries

You can enhance an object's functionality by attaching a script to it. Some scripts include functions that depend on external files called script function libraries. Scripts are included in the object import operation, but you must import the script function libraries separately.

### To import a script function library

1. Select **Import** from the Galaxy menu.
2. Browse to the directory that contains the script function library you want to import.
3. Select the script function library to be imported, and then select **Open** to start importing the selected library. Acceptable file types and file name extensions for script function libraries are:
  - Script Library Files (.aaSLIB file extension)
  - .NET or .COM files (.dll, .tlb, .olb, or .exe file extensions)
  - InTouch Script Extension Files (.wdf file extensions)

An information window opens when the import successfully completes.

4. Validate any symbols that use the custom script functions contained in the imported library. See *Validate Objects* on page 98 for additional information.
5. Redeploy any InTouchViewApp instances containing symbols that use the script functions in the imported library.

---

**Note:** When you import a new version of an existing script library into a Galaxy, you must stop and restart the engine that hosts the script's owning object. The new script version does not replace the old version until after the hosting engine has restarted and you have redeployed the object.

---

If you import an object whose script references a script function library that is not resident in the Galaxy, the imported object is set to Bad state and cannot be deployed. To correct this, import the script function library and validate the object. For more information about scripts, see the *Application Server Scripting Guide*. For more information about validating scripts, see *Validate Objects* on page 98.

If you import a script function library that is a different version than the current library, a message is displayed to notify you that there are dependent objects. The message indicates how many objects will need to be redeployed if you continue with the import. If you continue with the import, the dependent objects are marked for redeployment.

Script function libraries that are COM libraries developed using Visual Studio 6 or earlier are not automatically deployed. To place this COM library on the target platform, you can either:

- Install it directly on the target platform and register it.
- Import it to the Galaxy, and then export it as an aaSLIB. Modify the aaSLIB xml to designate that the library is to be registered as a COM object. Reimport the aaSLIB so that it is automatically deployed and registered.

## Importing Client Controls

If you import a client control that is a different version than the current object, a message is displayed to notify you that there are dependent objects. The message indicates how many objects will need to be redeployed if you continue with the import. If you continue with the import, the dependent objects are marked for redeployment.

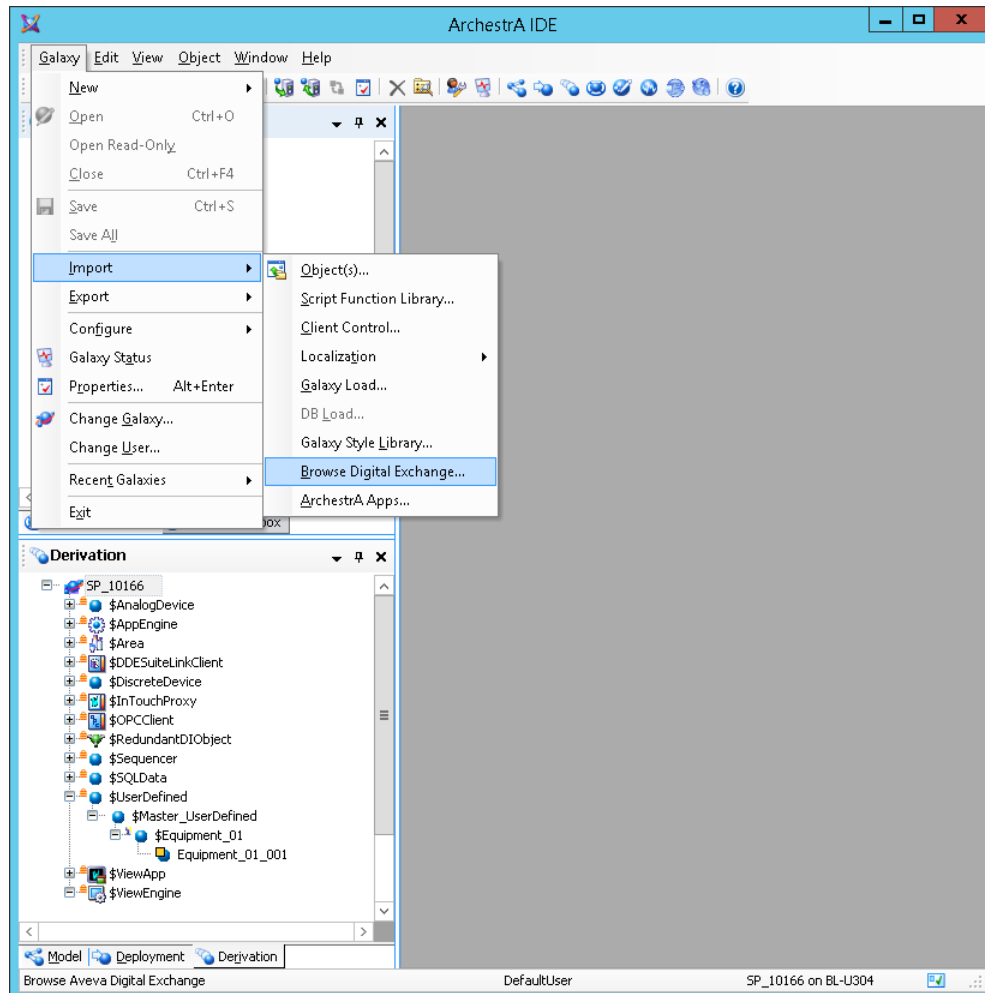
When the import process is complete, restart the IDE to apply the changes made by the import process. After the IDE has restarted, you can start using the objects you imported.

## Importing an App from the AVEVA Digital Exchange

A number of pre-built AVEVA OMI apps are available for download from the AVEVA Digital Exchange.

### To access the Digital Exchange

1. Select **Import** from the **Galaxy** menu.
2. Select **Digital Exchange**.



3. Select **Browse Digital Exchange**. Your browser will open and connect to the AVEVA Digital Exchange.
4. Log in, or sign up for new account.
5. Click on the link for the **System Platform Developers Community** and browse for apps. Select the app you want and follow the online instructions.

## Importing AVEVA Apps

You can import AVEVA apps developed with Windows Presentation Foundation (WPF), or Winforms/.NET controls through the IDE. Once you import the app or control, you can configure its properties and deploy it in run time.

An AVEVA App is collection of one or more controls primarily developed with Windows Presentation Foundation (WPF) for use in AVEVA OMI ViewApps. Other technologies such as Windows Forms (WinForms) and HTML5, can also be used. You can create your own AVEVA Apps, which can be imported via WPF interoperability. For more information about creating your own AVEVA Apps, see the AVEVA OMI SDK Help. The AVEVA OMI SDK is installed automatically when you install the IDE.

You can enhance a Galaxy's functionality by adding an embedded control from an AVEVA App to a pane in a layout for inclusion in a ViewApp.

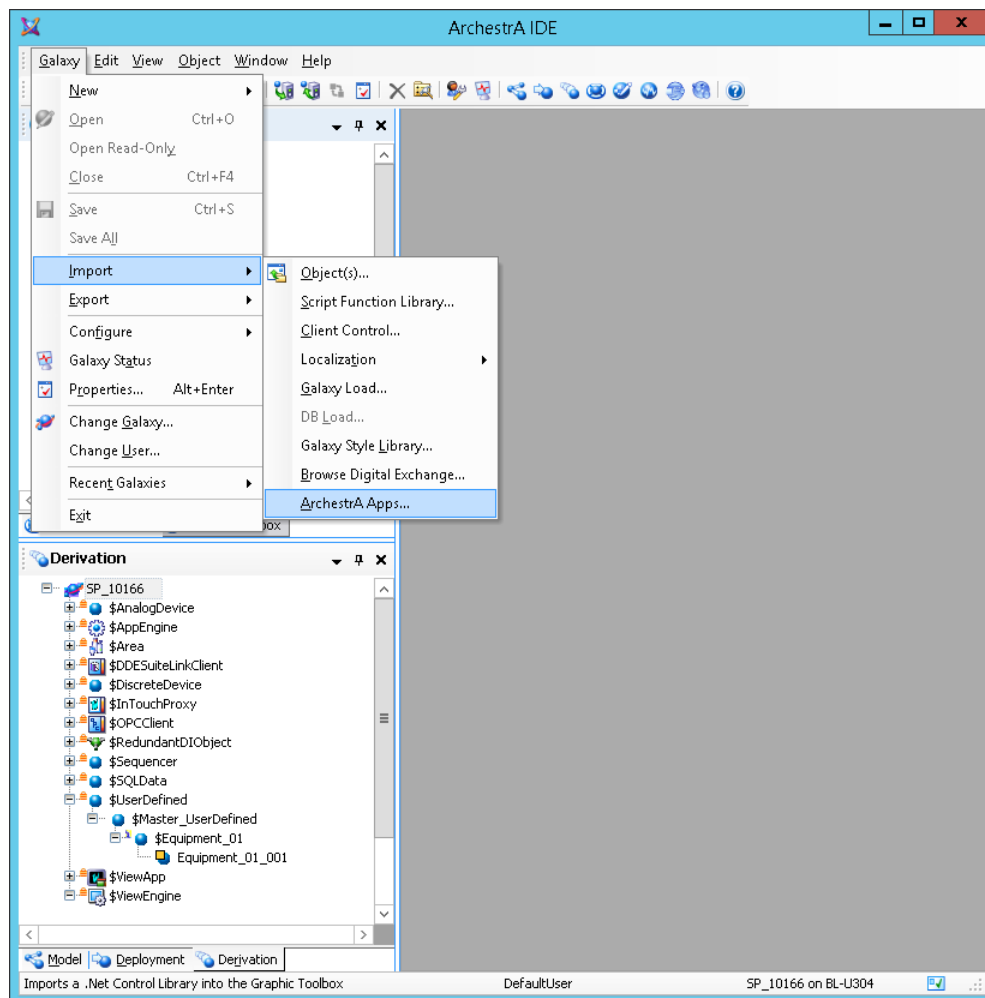
A control developed with Winform or .NET can be imported in much the same way, and like an AVEVA app, you can configure the control within a layout for inclusion in a ViewApp.

In order to qualify for import as an AVEVA App, the App should contain at least one WPF FrameworkElement Class control (System.Windows.FrameworkElement).

AVEVA Apps should be organized in a single folder which can optionally include subfolders. These subfolders typically contain locale files.

### To import an AVEVA App

1. Assemble the App, together with all required dependencies and related files, into a folder.
2. Select **Import** from the **Galaxy** menu.
3. Select **AVEVA App**.



4. Select the folder that contains the App to be imported.

The contained App, along with all dependent files, including dependent files contained in subfolders, are imported when you select **OK**.

If the App does not import successfully, see *Troubleshooting AVEVA or WPF Apps that Fail to Import* on page 116.

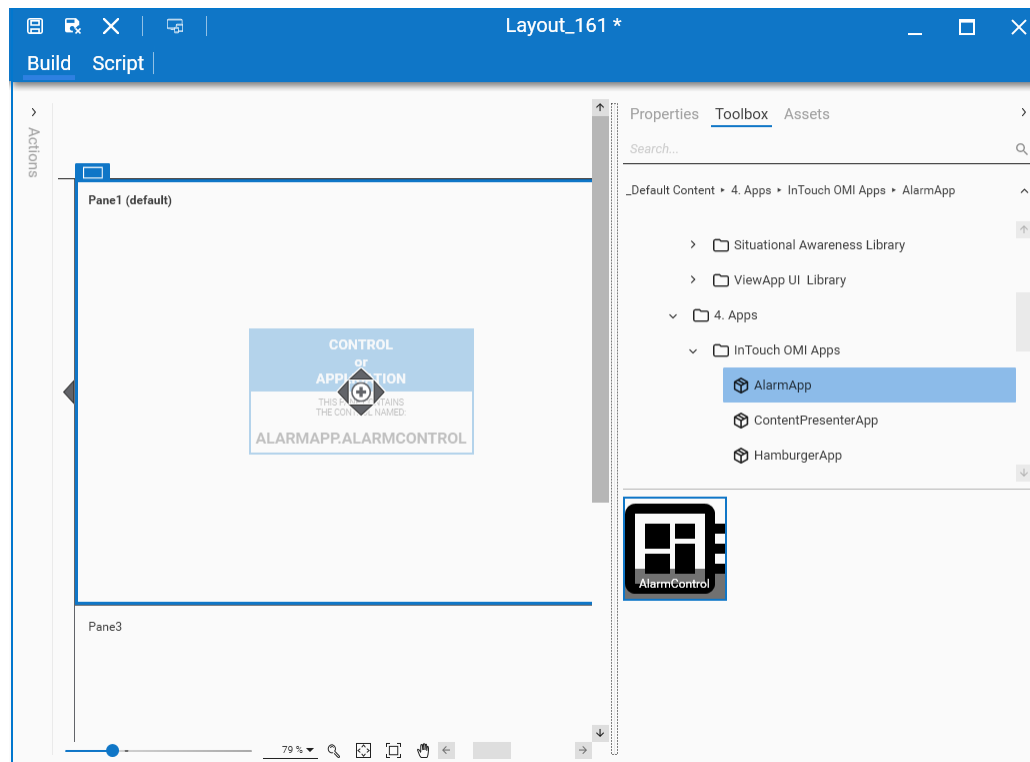
After the import process completes successfully, a new object appears in the Graphic Toolbox named DisplayModule\_00*n*, where *n* is 1 or the next available integer. The new object can be renamed, deleted, or exported as an .aaPKG file for use in another Galaxy. See *Export Objects* on page 102 for additional information.

## Add a Control from an AVEVA App to a ViewApp

Controls from the imported App appear in the **Toolbox** tab of the **Layout** or **ViewApp** editors.

### To assign a control from an App to a ViewApp

1. Open a layout in the **Layout** Editor and select the **Build** tab at the left of the editor.
2. Select the **Toolbox** tab on the right side of the editor and select a folder that contains the AVEVA Apps you want to add.
3. Select the AVEVA App to be added. The controls available in the App are shown.
4. Select a control and drag it onto a pane.
5. To view or override the control's public properties, select the **Properties** grid. Enter overrides as needed.
6. Save the layout. You can then use the layout that contains the embedded control in a ViewApp.



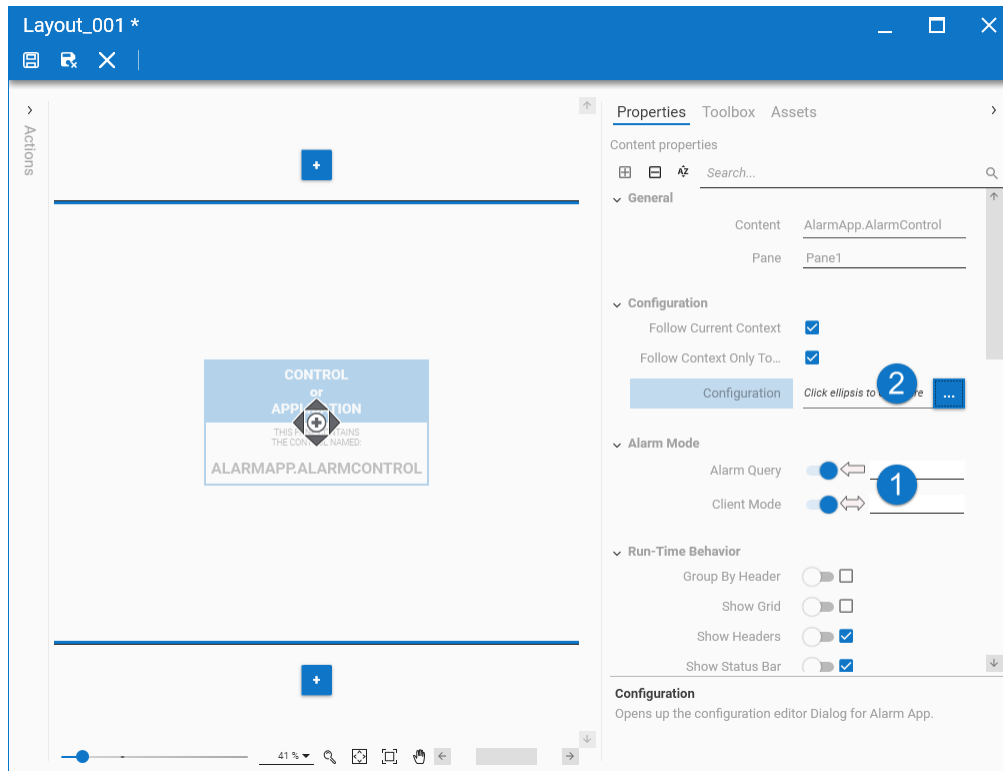
## Configurable Properties in AVEVA Apps

AVEVA apps can contain two types of configurable properties:

- **1** Dependency properties



- **2** CLR properties (.NET framework properties)



**Configurable Property Data Types**

- System.Boolean
- System.Byte
- System.Char
- System.Decimal
- System.Double
- System.Int16
- System.Int32
- System.Int64
- System.SByte
- System.Single
- System.String
- System.UInt16
- System.UInt32
- System.UInt64
- System.DateTime
- System.Drawing.Color
- System.Windows.Media.Color

- System.Windows.Media.Brush
- Any property that can be converted to or from string

CLR properties can only be set at configuration (design) time, while dependency properties can be bound to any attribute and can be set at run time. If a dependency property is changed at run time, the new value is propagated to the attribute. Dependency properties are data-bindable at design time, and you can set the binding direction to in (read), out (write), or both. To change the binding direction, click on the arrow to toggle the binding direction.

For dependency properties, you must specify the attribute to which you are binding the property, as well as the direction of the binding (in (read only), out (write only) or both). The attribute naming convention is the same you use, for example, when configuring an animation in the Graphic Editor. Note that the ViewApp Editor does not provide syntax validation for attribute names, but the Logger will list any configuration errors.

Binding is only supported for the following basic data types.

### **DataBindable Properties**

- System.Boolean
- System.Byte
- System.Char
- System.Decimal
- System.Double
- System.Int16
- System.Int32
- System.Int64
- System.SByte
- System.Single
- System.String
- System.UInt16
- System.UInt32
- System.UInt64

Any data type can be converted to and from the System.String type property.

## **AVEVA App Development Guidelines**

The following types of properties are available for configuring WPF-based displays:

- Public writable CLR properties on the display that are not hidden using the browsable attribute
- Dependency Properties
- Nested Properties which satisfy the criteria of configurable properties
- Enum and Flag properties
- Any property that can be converted to or from string

The following types of properties are NOT available for configuring WPF-displays:

- Any property defined by the System.Windows.UIElement class
- Any property defined by the System.Windows.FrameworkElement class

- The Tooltip property is an exception to this and can be configured
- The following properties defined by the System.Windows.Controls.Control class
  - IsTabStop
  - TabIndex
  - Template
- Any property defined by the System.Windows.Controls.ContentControl class

### Optional Filter File: AppManifest.xml

You can include an optional file called **AppManifest.xml** in your imported AVEVA App. See the *AVEVA OMI SDK Help*, located in the **AVEVA Documentation** folder, for additional information about AppManifest.xml. This file will allow you to filter properties when you are configuring the app. The basic file structure is:

- DLL name
  - Control name
    - Property

The following is a sample AppManifest.xml file, from the Hamburger App. The file specifies controls and properties that are exposed to System Platform IDE users as they configure a layout or ViewApp.

```
<?xml version="1.0"?>
<AppManifest AppVersion="1.0"?>
  <Filters>
    <!--List all the Controls that need to be exposed to the System Platform
    IDE user to utilize them in a View Application.
    Only the controls listed here will be available for the IDE users to place
    them on panes within a Layout or a ViewApp.-->
    <Control AssemblyName="AVEVA.Apps.HamburgerApp"
    ControlFullName="AVEVA.Apps.HamburgerApp.HamburgerButton">
      <!--List the public properties on this control that need to be exposed
      when user is configuring this control in a Layout or
      a ViewApp. The Property Editor within the Layout Editor and ViewApp
      Editor includes these properties when this control is
      selected and allows the user configure new default values.-->
      <Property Name="HamburgerColor" />
      <Property Name="SlideInPanePosition" />
      <Property Name="Background" />
    </Control>
  </Filters>
</AppManifest>
```

## AVEVA App Limitations and Restrictions

- Events and scripting are not supported
- Only basic datatype properties configuration
- Apps cannot be imported from a network share location
- Collections and arrays are not supported
- Upgrading Apps is not supported
- Nested properties can be viewed and configured, but they cannot be configured with data binding

- To host the .NET controls, use the WPF element **WindowsFormHost** to wrap the control into WPF.

```
<UserControl x:Class="AVEVA.Visualization.TrendApp.TrendControl"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
  mc:Ignorable="d"
  d:DesignHeight="300" d:DesignWidth="300">
  <Grid>
    <WindowsFormsHost x:Name="FormsHost" />
  </Grid>
</UserControl>
```

## Troubleshooting AVEVA or WPF Apps that Fail to Import

There are two major factors that may stop an AVEVA or WPF app from successfully importing.

**Expose controls and properties:** Make sure that you have included an AppManifest.xml file with the app, and that all controls and properties that are needed in your app are explicitly exposed in it. For additional information about the AppManifest.xml file, see *AVEVA App Development Guidelines* on page 114 and the **AppManifest.xml File** topic in the *AVEVA OMI SDK Online Help*, located in the **AVEVA Documentation** folder.

**Block business logic from executing during design time:** If you have business logic executing in the control constructor, or in static properties or static constructors, the import operation will not be successful. The control constructor is invoked when the control is instantiated. Therefore, do not include business logic within the control or static constructors, or in static properties.

- If you cannot remove business logic from a control or static constructor, or a static property, make sure the property is not invoked or that the property does not execute while in design mode.
- To block execution of logic within a constructor while in design time, add the following line to the constructor:

```
!DesignerProperties.GetIsInDesignMode
```

For additional information about app development, refer to the section on **App Development Best Practices** in the *AVEVA OMI SDK Online Help*, located in the **AVEVA Documentation** folder.

## Importing HTML5 Widgets

Widgets are small web components that can extend the functionality of a webpage or website. Custom-built websites can also incorporate widgets, by using open-source code or frameworks to provide certain functionality in whole or in part. A widget is a self-contained code block that slots into a website without changing any of its features. Widgets are most frequently used to provide on-screen user interface elements that ingrate with other platforms and data sources. A widget can be run on any web page on a website, with consistent placement and user interface, for example, social media, weather, RSS or podcast widgets.

By default, the following widgets are available under the Widgets folder in the Graphic Toolbox:

- Carousel
- Web Browser
- QR Code Scanner

You can import a widget for a standalone and managed InTouch application, or for use in an AVEVA OMI ViewApp. The file format is Custom Widget Package (.cwp), which includes HTML5, CSS, and Javascript files.

---

**Note:** Widgets can also be imported and exported from a galaxy as objects. In this case, import/export the widget as an object (aaPKG file type).

---

### Importing HTML Widgets

1. From the **Galaxy** menu, select **Import**.
2. Select **HTML5 Widget**.
3. Select a .cwp file.
4. Click **OK**.

The widget will appear in the Graphic Toolbox.

### After importing the widgets

1. Open a symbol or layout.
  - For symbols, embed the widget.
  - For layouts, navigate to the widget and drag it to a pane.
2. Set the widget properties as needed. To open object help for a widget, select the widget in the Graphic Toolbox and press **Ctrl + F1**.

Insert the widget on a window

The widget can now be viewed on WindowViewer and any web browser. Depending on the properties set in the design time you can manipulate the widget in runtime. Scripts using Custom Properties under 'Widget Properties' to modify widgets are not supported.

## Carousel Widget

A carousel widget allows you to cycle through elements—images or slides of text—like a carousel, without any input. This widget can be used to display dashboards, alerts or alarm information on large monitors on the plant floor.

### Properties

In addition to the standard graphics properties, you can also configure properties specific to the widget, under Widget Properties.

Name	Description
Autoplay	If the Autoplay property is set to true, the carousel widget will automatically start on load. If it is set to false, the user must select the next item to start the carousel. The default value is True.
GraphicNames	A comma separated list of graphics the carousel will display in runtime.
Interval	The amount of time delay (in milliseconds) between automatically cycling an item.
Keyboard	If the Keyboard property is set to true, the carousel will respond to keyboard inputs. The default value is True.

Loop	If the Loop property is set to true, the carousel will cycle through the graphics continuously, else it will stop after a single cycle. The default value is True.
Pause	If the Pause property is set to true, the carousel will pause the cycling of the graphics, when it detects the mouse hovering or a touch down event. The graphics will resume cycling when the mouse is moved away. The default value is True.

The carousel widget is based on the Bootstrap 4.0 Carousel component, for more information on bootstrap, go here: <https://getbootstrap.com/docs/4.0/components/carousel/>

## Web Browser Widget

Using the web browser widget, users can display a web site in WindowViewer and the Web Client.

If Web Client is running in HTTPS, then only HTTPS URL page can be loaded. If Web Client is running in HTTP, then HTTP and HTTPS can both be loaded. If the policy of the web site blocks cross domain access, then this widget will not work. The URL need not be in double quotes, but must be a valid URL.

### Properties

Name	Description
URL	The address of the website.

### Limitations

- If no protocol is specified, by default the https protocol will be used.
- If the Web Client is configured to use the HTTPS protocol, then only the HTTPS URL page will be loaded. If the HTTP URL is used, the web browser widget will display a message "Mixed Content: The page at 'https://localhost/intouchweb' was loaded over HTTPS, but requested an insecure frame 'http://\*\*\*\*\*'. this request has been blocked: the content must be served over HTTPS."
- The web browser widget will not function, if the web site policy blocks cross domain (cross origin) access. A link will be provided to open the web page in a separate tab.

## QR Code Scanner

The QRCode\_Scanner widget connects to a camera to scan for a QR code and returns the resulting string.

### Properties

Name	Description
QRCode	The resulting string of the scanned QR code. The default value is empty.

### Limitations

- The device must have a camera.
- Using the QR Code on a physical machine instead of a virtual machine is recommended.

- Access the web client using the secure URL (https://) when using the web client remotely.

### Usage

You can configure a script to read the QR code and display a graphic based on the scanned value.

In RunTime, the QR Code Scanner object will appear with two buttons - Start Camera;Stop Camera.

Click Start Camera and scan the QR Code. If the scanned code matches the script return value or graphic name, the graphic will be displayed.

## After You Import

Imported templates are listed in the proper toolset in the Template Toolset as defined in the object. Imported instances are shown in the Application views.

The following post-import rules apply:

- If a toolset does not exist, it is created.
- If the object belongs to a security group that does not exist, it is associated with the Default security group.
- If the object belongs to an area that does not exist, it is associated with the Unassigned Area.
- If the host to which the object is assigned does not exist, it is assigned to the Unassigned Host.
- If you selected **Migrate** from the **Import Preferences** dialog box, the migrated objects are marked with "software upgrade required" if they are deployed. These objects will be upgraded when the objects are redeployed.
- If you import a new version of an existing instance, the new version is marked as requiring deployment if the existing object is already deployed.





# CHAPTER 5

## Configuring Objects

Enhance and extend derived templates and instances by adding and configuring features, attributes, scripts, and symbols. These elements can be associated with an Object Wizard in a derived template to simplify the task of creating instances.

The **Object Editor** includes a separate page for adding scripts. All other elements (symbols, attributes, Object Wizards, and external content) are added and configured from the Object Editor's main page. For an overview of the **Attributes** page, see *About the Attributes Page* on page 74.

### Add Attributes to an Object

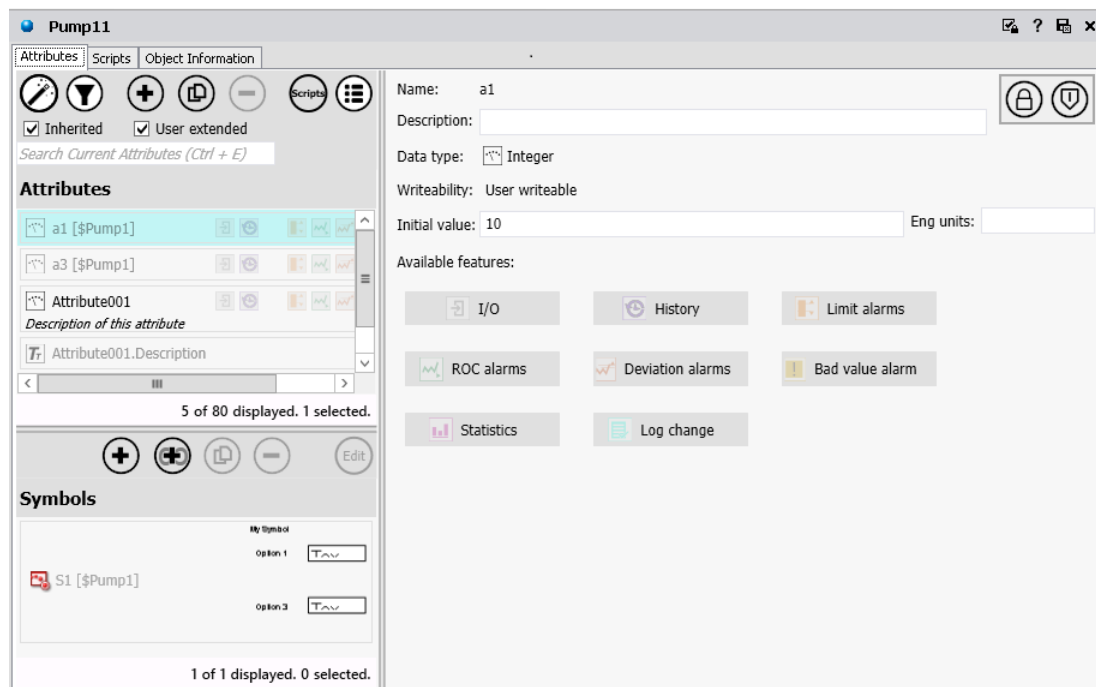
There are two ways to add attributes to an object. These are:

1. Select the **Add Attribute** button. The first attribute you add to an object has the default name "Attribute001" and the default data type "Boolean."

Subsequent attributes added with the **Add Attribute** button use the previous attribute's name and increments the number at the end of the name, or appends "1" at the end of the name if the previous name did not end in a number. It also retains the previous attribute's description, Data type, and Writeability.

2. Select the **Duplicate Attribute** button to duplicate an attribute that was previously configured. Selecting **Duplicate Attribute** retains any features that were configured for the selected attribute, in addition to name, description, Data type and Writeability.

**Note:** When you add an attribute to a derived template, the attribute, its data type, and writeability are automatically locked in derived objects.



If attribute parameters such as initial values and security classifications are locked in the template, they cannot be changed in child instances. If these parameters are unlocked in the template, the initial value and security are editable and lockable in derived templates. When unlocked in either the base or derived template, the value is editable in instances.

After you add an attribute to an instance, it appears in the **Attribute Browser** list for use with the scripting and attribute configuration functions. For more information about using the **Attribute Browser**, see *Reference Objects Using the Galaxy Browser* on page 88.

In the **Attributes** page, you can define the following initial information and parameters for the attribute:

- Add a new attribute to an object.
- Name the attribute and provide a description.
- Configure its data type.

For a Boolean data type, you can specify different text strings for the **'False' label** and **'True' label**. For example, if a Boolean attribute is associated with the status of a motor, you can specify the states as **"Stopped"** and **"Running"**. Text boxes appear for you to enter these strings when you select a Boolean data type.

These labels are also shown in the **Value** and **Limit** columns of the Alarm and Event database and InTouch AlarmView control.

- Specify the attribute writeability.
- Set initial values if the attribute is user writeable.
- Enable and set locks and security on the new attribute.
- Set whether the new attribute is an array and how many elements are in the array.

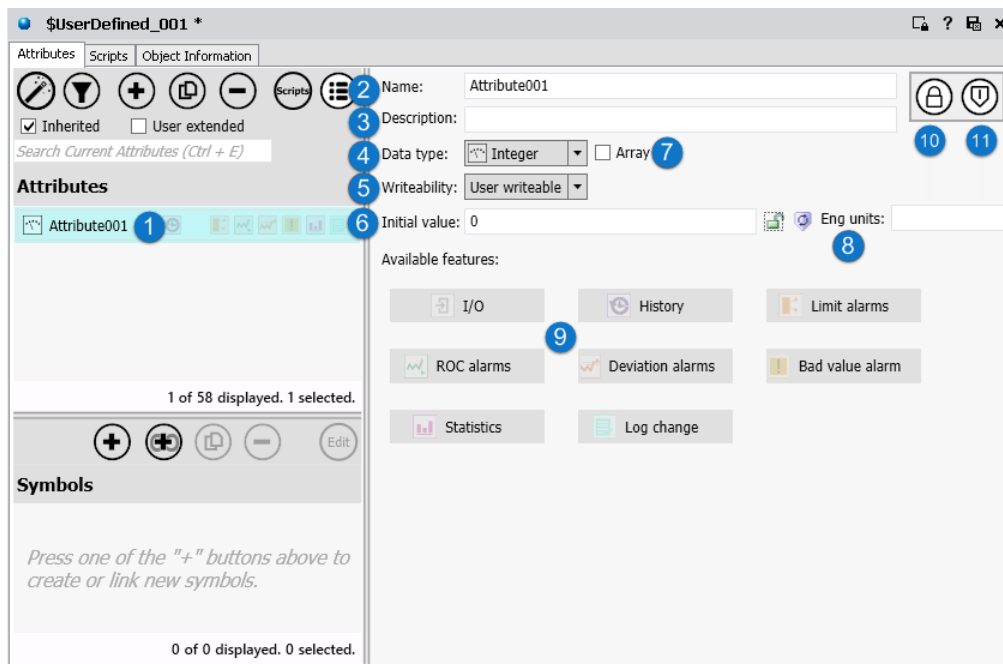
You can then add Features to the attribute. For more information, see *Adding Features to Attributes* on page 126.

## Configure an Attribute

### To configure attributes

1. Highlight the attribute you want to configure.
2. Edit the **Name** of the attribute.
3. Enter a **Description**.
4. Select a **Data type**:
  - Boolean: Enter False and True labels (if not False and True).
  - Integer: Enter Initial value and Engineering units.
  - Float: Enter Initial value and Engineering units.
  - Double: Enter Initial value and Engineering units.
  - String: Enter Initial value.
  - Time: Enter Initial value.
  - ElapsedTime: Enter Initial value.
  - InternationalizedString: Enter Initial value.
5. Select Writeability type:
  - Calculated
  - Calculated retentive

- Object writeable
  - User writeable
6. Enter the Initial value for the attribute.
  7. Select the checkbox if the attribute is part of an array.
  8. Enter the Engineering unit type (numeric data types only).
  9. Select Features and configure them. See *Adding Features to Attributes* on page 126 for additional information.
  10. Lock the Initial value, if you do not want users to be able to modify the value in derived objects. You can show or hide the lock icon by toggling the **Show/Hide Lock icons** button. See *Lock and Unlock Template Attributes* on page 70 for additional information.
  11. Configure the attribute's security state. You can show or hide the security icon by toggling the **Show/Hide Security icons** button. See *Set Object Security* on page 73 for additional information.

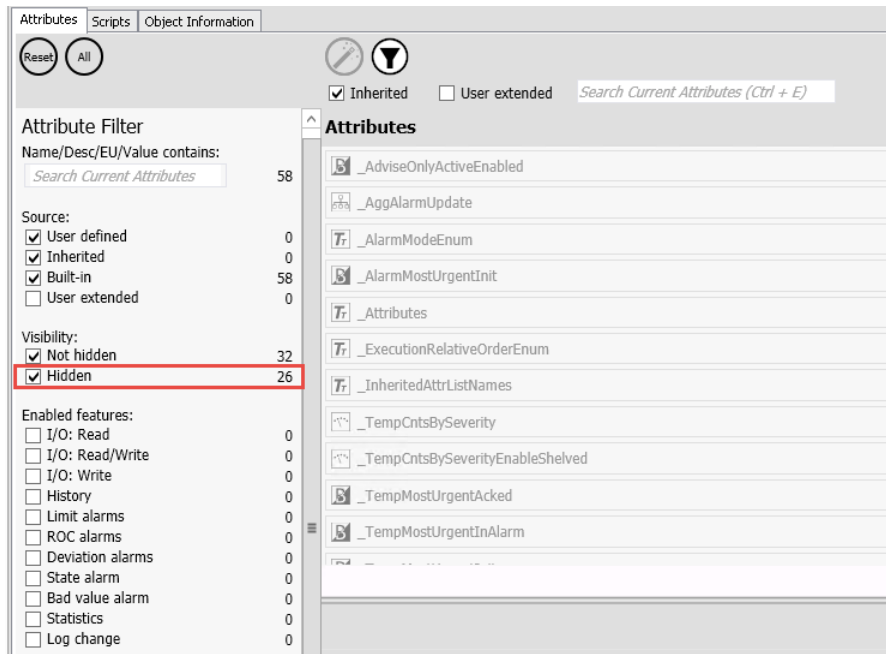


## Attribute Naming Conventions

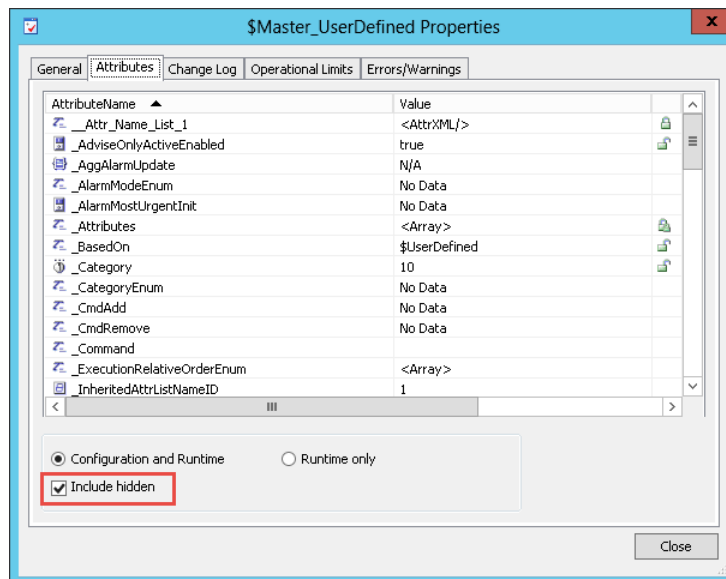
Attribute names can have up to 32 alphanumeric characters, including periods. Attribute names must include at least one letter.

- An attribute name that starts with an underscore (\_) as the first character of the name is a hidden attribute. By default, hidden attributes are not shown in the **Attribute Editor**, the **Properties>Attributes** dialog box, or the **Object Viewer**.

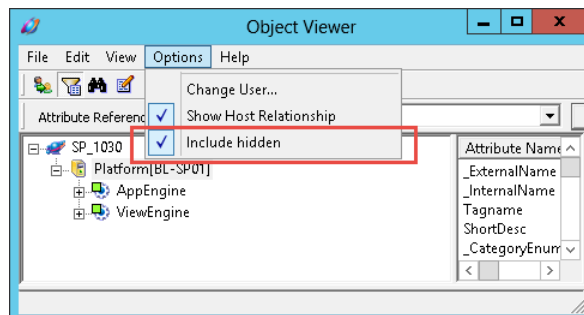
- To view hidden attributes in the **Attribute Editor**, select the **Hidden** filter option.



- To view hidden attributes in the **Properties** dialog box, select **Include Hidden**.



- To view hidden attributes in the **Object Viewer**, select **Include Hidden**.



---

**Note:** We recommend that you do not use hidden attributes in scripts.

---

- Using the word "quality" as an attribute name is not supported. "Quality" is used by InTouch HMI in a set of dotfields to show the reliability of data values assigned to an I/O tag. An attribute named "Quality" cannot be accessed through FS Gateway or InTouch due to a naming conflict.

---

**Important:** After creating an attribute, it is available for adding Features or additional attributes. If you extend an attribute you have added to an object or another attribute, and then delete or rename that attribute, all associated Features and additional attributes added to the object are lost.

---

### To create and associate a attribute with an object

1. On the **Attributes** page of the Object Editor, click the **Add (+)** button. An attribute is added to the **Attributes** list.
2. Type the new attribute name and add an optional description.
3. In the **Data type** list, select the **Data type** for the new attribute. The available Features and basic options change depending on your selection in the **Data type** list.
4. Set the remaining parameters as needed.

---

**Note:** For detailed information about each item on the **Attributes** page, see *About the Attributes Page* on page 74.

---

5. Lock the values, if needed. The lock is available only when you are working with a template. If you are working with an instance, it shows the lock status for the value in the parent object.
6. Set any security you need. For more information about setting security, see *Set Object Security* on page 73.
7. Save and close the Object Editor when you are done.

## Attributes and Scripting

Follow these guidelines and best practices when using attributes in scripts:

- If you use **Calculated** and **Calculated retentive** attributes as counters, they must be manually initialized. For example, if you use `me.Attribute=me.Attribute+ 1` as a counter in a script, you must also initialize the attribute with something like `me.Attribute=1` or `me.Attribute=<some attribute value>`.
- **Calculated** attributes can be initialized in scripts with **Execution type** triggers of On Scan and Execute, but not initialized in Startup scripts.
- If you use an attribute **array** in a **startup script**, value and quality may become bad when the array is initialized in the startup script. To prevent this, set the dimension after initializing the array as shown below. This will ensure that the value is set and the attribute quality is good.

```
Me.ArrayAttribute[]="0,0,0,0,0,0,0,0,0,0,0";
Me.ArrayAttribute.Dimension1 = 10;
```

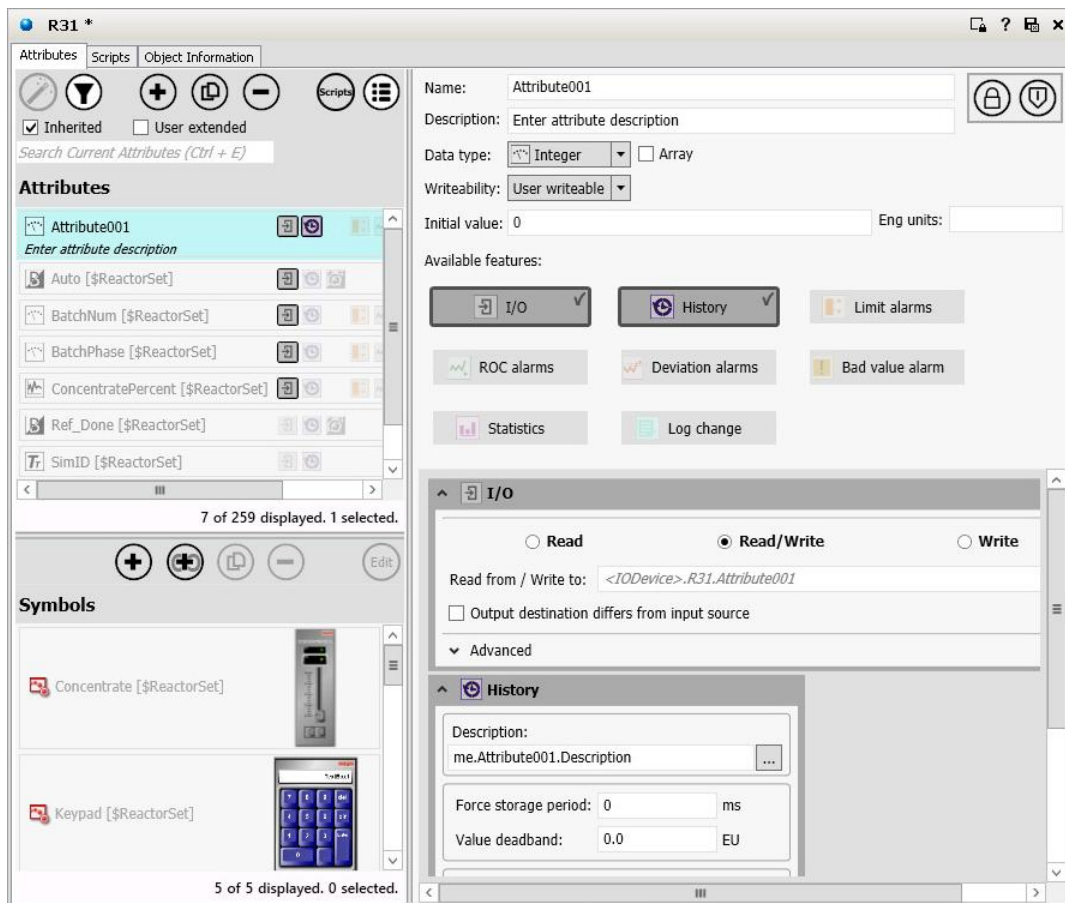
- You must initialize **Calculated retentive** attributes in Startup scripts and you can initialize these attributes in On Scan and Execute scripts. A **Calculated retentive** attribute retains the attribute's current value after a computer restart, redundancy-related failover, or similar situation in which valid checkpoint data is present.

Your Startup script should contain a statement testing the Boolean value of the `Engine.StartingFromCheckpoint` attribute on the object's AppEngine. If the value is TRUE, do not initialize the attribute. If the value is FALSE, initialize the attribute. For more information about `Engine.StartingFromCheckpoint`, see the help for the AppEngine object, available from the AppEngine object editor.

- We recommend that you **do not use hidden attributes** (attribute names that begin with an underscore) in scripts.

## Adding Features to Attributes

The **Attributes** page allows you to configure an existing attribute for I/O, alarms, and history functionality not embedded in the original object.



## About Features Inheritance

You can add Features to attributes that are in either derived templates or in instances. You cannot add Features to attributes in Base templates. The following parent-child object characteristics also apply to Features added to objects:

- If you add a Feature to an attribute in a derived template that has objects derived from it, all child objects inherit the Feature.
- You cannot add a Feature to attributes on derived objects that duplicate parent object Features in name and type.
- You cannot add a Feature with the same name as an existing Feature.
- Renaming a Feature on an attribute in the template to which it was originally added renames all other objects derived from the template. This change happens when the template is checked in.
- You can check in a template with an attribute configured with a new Feature with the same name as an existing Feature on an attribute in a derived object. The template definition of the Feature overrides the Feature in the derived object.

- If you remove a Feature on an attribute from a template, that Feature is removed from any child object. You see the change when you check in the template.

### To create and associate a Feature with an object

1. On the **Attributes** page, select an attribute from the **Attributes List**. The available Features dynamically change to allowed Feature rules for the selected attribute type.
2. Click the button for the Feature you want to apply to the selected attribute. The associated parameters for each kind of Feature become available. For detailed information about each item on the **Attributes** page, see *About the Attributes Page* on page 74.
3. Select the parameters for the Feature you have added. Available Features, which vary depending on the attribute's Data Type and Writeability, are the following:
  - **I/O**: For information about adding the I/O Feature, see *Using the I/O Feature* on page 127.
  - **History**: For information about adding the History Feature, see *Using the History Feature* on page 136.
  - **Limit alarms**: For information about adding a Limit alarm Feature, see *Using the Limit Alarms Feature* on page 139.
  - **ROC alarms**: For information about adding a Rate of Change (ROC) alarm Feature, see *Using the ROC Alarms Feature* on page 140.
  - **Deviation alarms**: For information about adding a Deviation alarm Feature, see *Using the Deviation Alarms Feature* on page 141.
  - **State alarm**: For information about adding a State alarm, see *Using the State Alarm Feature* on page 143.
  - **Bad Value alarm**: For information about adding a Bad Value alarm, see *Using the Bad Value Alarms Feature* on page 146.
  - **Statistics**: For information about adding a Statistics Feature, see *Using the Statistics Feature* on page 147.
  - **Log change**: For information about adding a Log Change Feature, see *Using the Log Change Feature* on page 148.
4. Lock the values, if needed. The lock symbol is available only when you are working with a template. If you are working with an instance, it shows the lock condition of the value in the parent object.
5. Set any security for the attribute. For more information about setting security, see *Set Object Security* on page 73.
6. Save and close the Object Editor to include the new Features in the configured object.

## Using the I/O Feature

The I/O Feature allows you to configure all aspects of data input and output for an attribute.

You can configure I/O type and you can specify input sources and output destinations. The I/O types you can specify are:

- Read (Input): See *Configure I/O as Read-only* on page 129.
- Read/Write (InputOutput): See *Configure I/O as Read/Write* on page 130.
- Write (Output): See *Configure I/O as Write-only* on page 132.

You can also configure advanced properties, described in the following table. The attribute's data type and I/O type determine what Advanced I/O properties are available.

I/O Feature Advanced Property	Description
Buffered	<p>Enable buffered data to propagate to data subscribers the entire subset of values accumulated within a single scan cycle.</p> <p>Buffering data ensures that if a given attribute changes its value several times during a single scan cycle, there is no folding of data, which occurs when an accumulation of values of multiple data changes within a single scan are overwritten and only the latest value is stored.</p>
Deadband	<p>Specify the minimum amount by which a value must vary in order for the attribute to register a change, for example, by triggering an alarm, historizing an alarm or event, or triggering a script.</p> <p>The Deadband property is not available for string data types.</p>
Reflect input to output	<p>Enable to propagate an input value to an output destination. Enabling automatically disables the <b>Output destination differs from input source</b> option.</p> <p>Typically, set this option when you want to read an input from one source, manipulate its value in a script, and send the manipulated value to a different destination address, all during a single scan of an object. For more information, see <i>Using Read/Write I/O in Scripts</i> on page 132.</p> <p>Available only when the I/O type is Read/Write. You must set separate <b>Read from</b> and <b>Write to</b> properties.</p>
Output every scan	<p>Available only when <b>Reflect input to output</b> is selected. Write to the specified output destination occurs even when there has been no state or data change since the previous scan.</p> <p>The timestamp is not updated if there has been no state or data change since the previous scan.</p>
Enable I/O scaling	<p>Enables scaling between the raw value and the Engineering Units (EU) value. Scaling is the process of taking raw data from a device and presenting it as an appropriate value for your application.</p> <p>Application Server supports two types of data scaling: linear and square root, described in this table.</p> <p>Available only for integer, float, and double data types.</p>
Maximum	<p>Available only when I/O scaling is enabled.</p> <p>Raw: The raw input maximum to be used in the scaling equation.</p> <p>EU value: The maximum value in engineering units to be used in the scaling equation.</p> <p>Extended EU range: The highest value allowed for the attribute before it is clamped, if clamping is enabled, or set to NaN. This value must be greater than or equal to the specified maximum EU value.</p>



I/O Feature Advanced Property	Description
Minimum	<p>Available only when I/O scaling is enabled.</p> <p>Raw: The raw input minimum to be used in the scaling equation.</p> <p>EU value: The minimum value in engineering units to be used in the scaling equation.</p> <p>Extended EU range: The lowest value allowed for the attribute before it is clamped, if clamping is enabled, or set to NaN. This value must be less than or equal to the specified minimum EU value.</p>
Conversion mode	<p>Available only when I/O scaling is enabled.</p> <p>Select Linear or Square Root conversion mode from the list.</p> <p>Linear: Typically converts directly from one value scale and type (raw) to another value scale and type (EU).</p> $\text{ScaledValue} = ((\text{RawValue} - \text{RawMin}) / (\text{RawMax} - \text{RawMin})) * (\text{EngUnitsMax} - \text{EngUnitsMin}) + \text{EngUnitsMin}$ <p>Square Root: Provides more precise scaling, typically used when the raw value is too large to be usefully scaled to an EU.</p> $\text{ScaledValue} = \text{sqrt}((\text{RawValue} - \text{RawMin}) / (\text{RawMax} - \text{RawMin}))$ <p>If <math>\text{RawValue} &lt; \text{RawMin}</math>, then <math>\text{ScaledValue} = (\text{sqrt}(\text{RawMin})) * (\text{EngUnitsMax} - \text{EngUnitsMin}) + \text{EngUnitsMin}</math></p>
Clamp input to EU range	<p>Available only when I/O scaling is enabled.</p> <p>If enabled, the scaling calculation result is clamped at either the maximum EU range value or the minimum EU range value, and the attribute quality is set to Uncertain.</p> <p>If not enabled, and the attribute value exceeds the EU range, then the value will continue to scale out of range, and the quality is set to Bad.</p>

## Configure I/O as Read-only

Select **Read** in the I/O parameters area. Define an input **source** by using I/O auto-assignment, or by typing in the reference string, or by clicking the **Attribute Browser** button at the right.

- Use I/O auto-assignment to prepare the input source for automatic assignment to a configured Device Integration (DI) object or other data source. For information about using I/O auto-assignment, see *Using I/O Auto Assignment* on page 148.

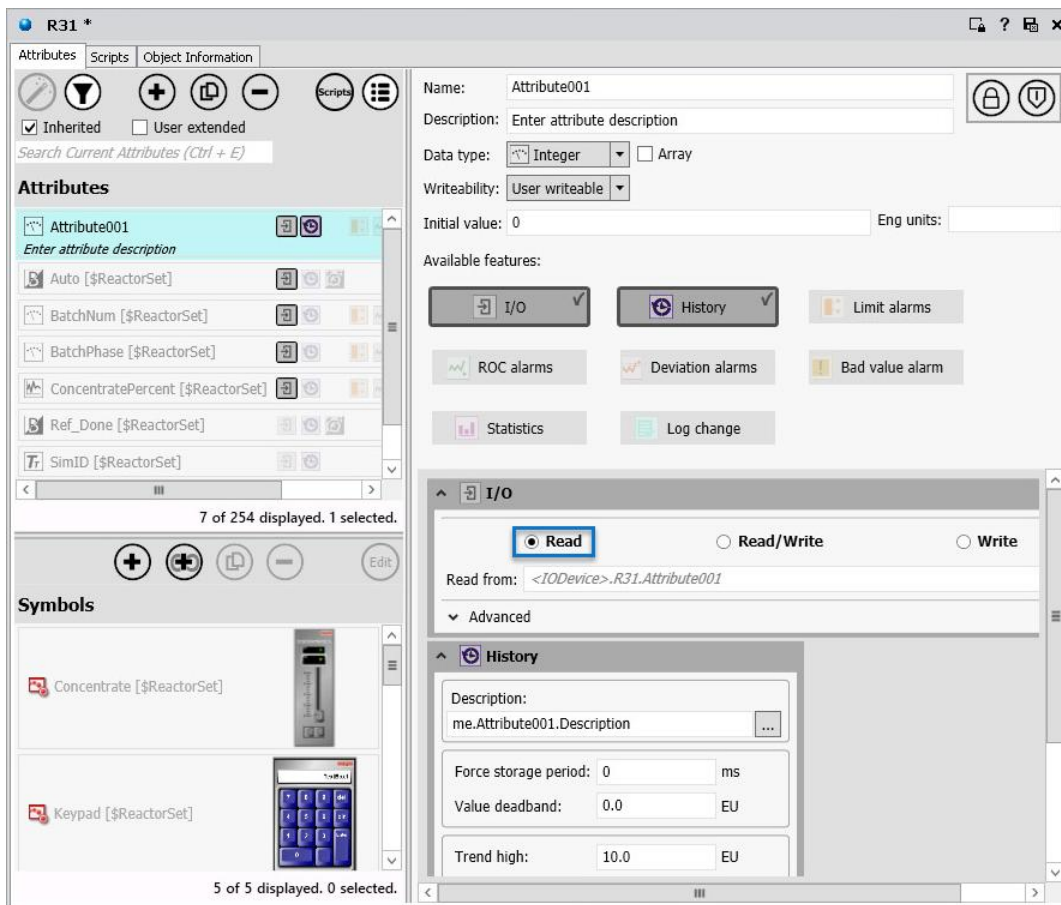
---

**Important:** If the InputSource attribute is locked in the parent template, the attribute cannot be updated with the resolved reference when the object is deployed, and the run-time value will be "---Auto---".

---

- Use the **Attribute Browser** to select an attribute and automatically insert the correct reference string for that attribute. For more information about using the Attribute Browser, see *Reference Objects Using the Galaxy Browser* on page 88.

- If you are configuring an I/O attribute for use with the Telemetry Server, see *Configure I/O for Use with AVEVA Telemetry Server Communication Drivers* on page 135 for additional information.



You can add multiple Read (input) I/O Features to an object. However, you cannot add a Read/Write (InputOutput) I/O Feature to an attribute that already has either a Read or a Write (output) I/O Feature. Arrays are not supported.

---

**Note:** Lockable attributes can be configured with a Read I/O Feature, but they only function correctly during run time if the configured attribute is unlocked.

---

If the data types of the attribute and its I/O source attributes are the same, they are set to equal values according to the object’s execution rate. If the two attributes are different data types, coercion rules are applied.

If coercion fails or the input value is out of the attribute’s range, quality for the configured attribute is set to Bad. Otherwise, the configured attribute quality matches the **source** attribute. When the object is Off Scan, quality is always Bad and user sets are accepted.

Attributes configured with a Read I/O Feature are not protected by their security classification. The only enforced security specifies if an IDE user can edit or add features to the object. A Read I/O Feature can be added to a template or instance. If added to a template, the existence of the Read I/O Feature is automatically locked in derived objects.

### Configure I/O as Read/Write

Select **Read/Write** in the I/O parameters area. Define an input **source** by using I/O auto-assignment, or by typing in the reference string, or by clicking the **Attribute Browser** button at the right.

- Use I/O auto-assignment to prepare the input source for automatic assignment to a configured Device Integration object or other data source. For information about using I/O auto-assignment, see *Using I/O Auto Assignment* on page 148.

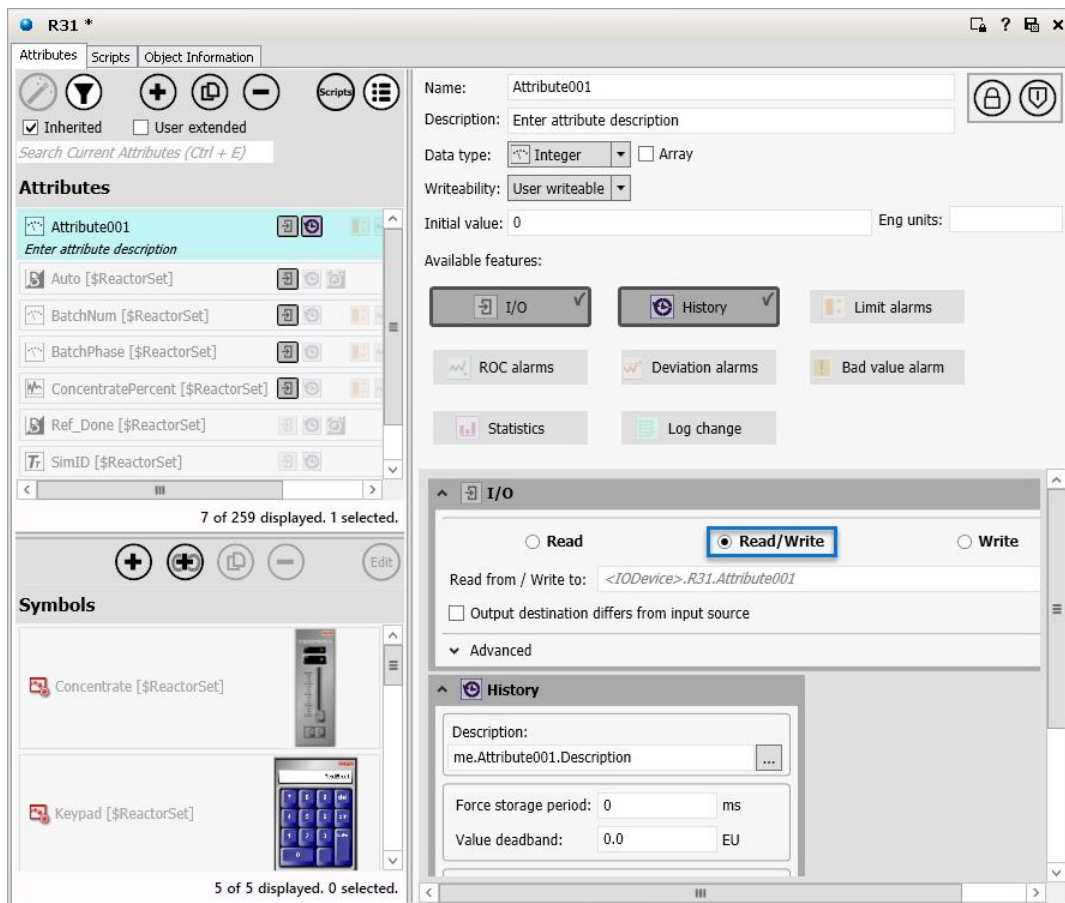
If InputSource or OutputDest attributes, or both, are locked in the parent template, the attributes cannot be updated with the resolved reference when the object is deployed, and the run-time value will be "---Auto---".

- Use the **Attribute Browser** to select an attribute and automatically insert the correct reference string for that attribute. For more information about using the Attribute Browser, see *Reference Objects Using the Galaxy Browser* on page 88.
- If you are configuring an I/O attribute for use with the Telemetry Server, see *Configure I/O for Use with AVEVA Telemetry Server Communication Drivers* on page 135 for additional information.

If the output destination and the input source are not the same, click **Output destination differs from input source**. Enter a **Destination** attribute by using I/O auto-assignment, by typing in the reference string, or by using the **Attribute Browser** to search for the reference string in an object.

**Important:** If you clear the **Output destination differs from input source** check box, the **Write to** text box automatically shows "---". In the run-time environment, "---" is the same reference as the **Read from** value entered during configuration time. During run time, you can change the **Read from** reference. During configuration, do not lock the **Write to** parameter if you clear the **Output destination differs from input source** check box.

A Read/Write I/O Feature allows an attribute in a template or an instance to be configured so that its value is both read from and written to an external reference. The Read/Write I/O Feature monitors the value/quality of an input and sends outputs on state change.



The **Write to** (output) destination can be the same or different from the **Read from** (source). The references are always to another acceptable attribute type in the Galaxy.

You can add multiple Read/Write I/O Features to an object. However, you cannot add a Read/Write I/O Feature to an attribute that already has a Read or Write I/O Feature.

---

**Note:** You can add a Read/Write I/O Feature to lockable attributes, but they only function correctly during run time if the configured attribute is unlocked.

---

## When Objects Are On Scan

When an object is On Scan, the value and quality of the attribute configured with a Read/Write I/O Feature mirrors the quality of the externally referenced attribute during a successful read. The data quality of the attribute is set to Bad when reads fail. Reads can fail because of communication errors or datatype conversion failures.

While the object is On Scan, the data can change quality. If an external set (for example, from a user) to an attribute changes either the value or quality, then a write of the attribute's value to the destination occurs during the next execute phase. The quality must be Good or Uncertain for a write to occur. For writes to occur because of a quality change, the quality change must be a transition from Bad or Initializing to Good or Uncertain.

The attribute called WriteValue is publicly exposed and plays an important role in driving outputs. When the object is Off Scan, quality is always Bad and user sets are accepted.

## Using Read/Write I/O in Scripts

Two common types of scripts can be written on an attribute configured with a Read/Write I/O Feature: One can look at the input side and one can look at the output side.

The input side script uses the current value coming from the input source location and performs logic or calculations on it. This script refers directly to the attribute in its expressions. For example, if the attribute is "me.attribute1", the script refers directly to "me.attribute1" for data change conditions and for expressions within the script.

The output side script can modify output or validate a new requested output value. This script refers to the "WriteValue" attribute configured on the attribute: "me.attribute1.WriteValue".

To validate a new requested value to the attribute1, for example, a data change condition expression is written on "me.attribute1.WriteValue". In addition, if the script wants to do clamping or validation, it can manipulate the "me.attribute1.WriteValue" directly to clamp the output value. For example:

```
If (me.attribute1.WriteValue > 100.0 ) then  
  
    Me.attribute1.WriteValue = 100.0;  
  
Endif;
```

The data change expression for this script is "me.attribute1.WriteValue" because this value changes when a new value is about to be written to the field.

The script can intercept this value just before output and manipulate it. To prevent WriteValue from being written out, its data quality can be set to Bad with the SetBad() function.

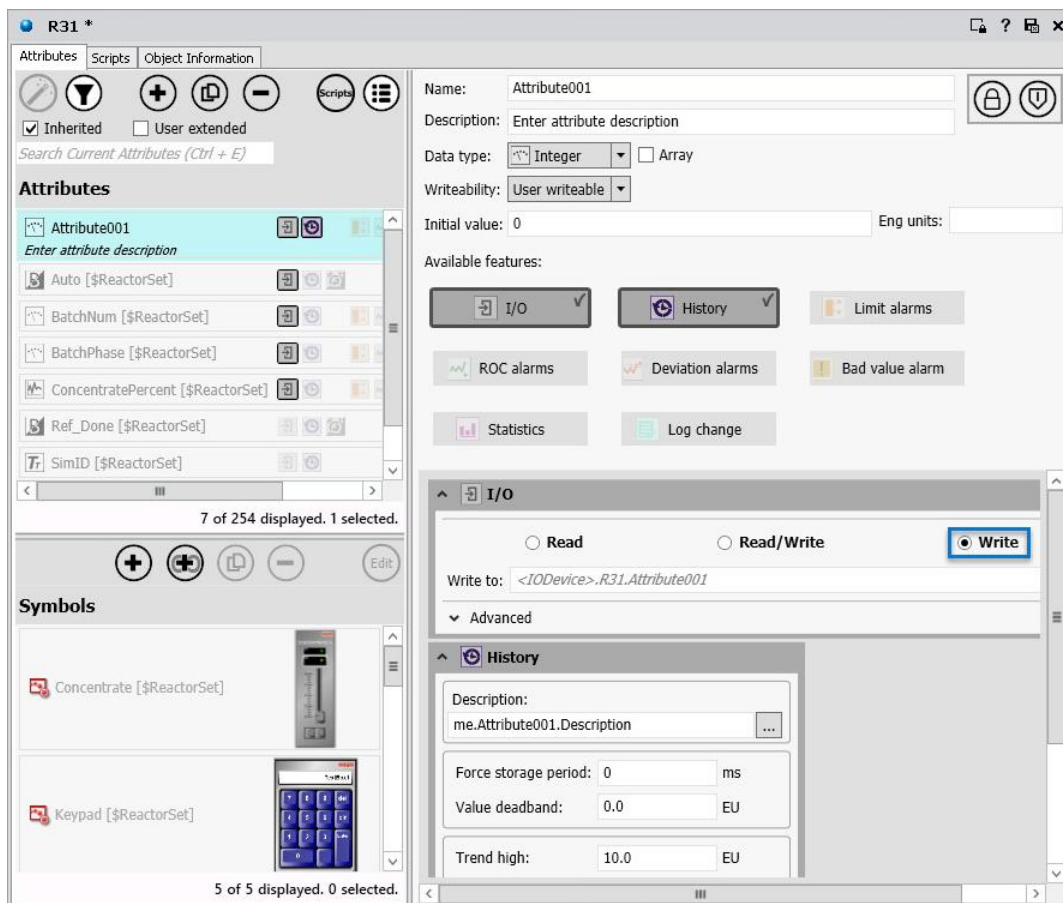
For more information, see *Working with Outputs* on page 134.

## Configure I/O as Write-only

Select **Write** in the I/O parameters area. Define a **Write to** (output) **destination** by using I/O auto-assignment, or by typing in the reference string, or by clicking the **Attribute Browser** button at the right.

- Use I/O auto-assignment to prepare the output destination for automatic assignment to a configured Device Integration (DI) object or other data source. For information about using I/O auto-assignment, see *Using I/O Auto Assignment* on page 148.
- If the OutputDest attribute is locked in the parent template, the attribute cannot be updated with the resolved reference when the object is deployed, and the run-time value will be "---Auto---".
- Use the **Attribute Browser** to select an attribute and automatically insert the correct reference string for that attribute. For more information about using the Attribute Browser, see *Reference Objects Using the Galaxy Browser* on page 88.

Select the **Output Every Scan** check box if you want the attribute to write to the **Destination** attribute every scan period of the object. Otherwise, the write executes only when the value is modified or when quality changes from Bad or Initializing to Good or Uncertain.



Writeable and Calculated attributes can be configured with a Write I/O Feature. Arrays are not supported.

A Write I/O Feature can be added to a derived template or to an instance. If added to a template, the existence of the Write I/O Feature is automatically locked in derived objects. The output **Destination** attribute is separately lockable in templates.

If the data types of the configured and destination attributes are the same and only when the quality of the extended attribute is good, the two attributes are set to equal values according to the configured object's execution rate. If the two attributes are different data types, coercion rules are applied. If coercion fails, the extended attribute is placed into a configuration error and type mismatch state.

An attribute that is enhanced with a Write I/O Feature has the following characteristics:

- A value can be output only when quality is Good or Uncertain. The quality is not output, only the value is output, because quality is not output on sets.
- When the quality changes from Bad or Initializing to Good or Uncertain, the value is output, even if the value is not modified.
- When the quality changes from Good to Uncertain, with no value modification, the value is not output.
- When the object goes Off Scan, no output is done.
- When the extended object is Off Scan, quality is always Good and user sets are accepted.

## Working with Outputs

The following information applies to the functionality of Read/Write and Write Features as well as to the output function of the Field-Reference, Switch, and Analog-Device objects.

If a single set request is made to a destination attribute during a single scan cycle, that value is sent to the destination. During a single scan cycle, though, more than one set request to the same destination is possible. In that case, folding occurs and the last value is sent to the destination.

During a single scan cycle, only the last value requested during a scan cycle is sent to its destination when the object executes. Its status is marked as Pending as it waits for write confirmation from the destination object. All other set requests during that scan cycle are marked as successfully completed.

If one or more new sets are requested during the next scan cycle, then the second scan cycle's value is determined as described in the preceding paragraphs. It is then sent to the destination when the object executes again and the value sent to the destination during the previous scan cycle is marked with successful completion status even if write confirmation is not received.

Within a single scan cycle, data is folded and only the last set requested is sent to the destination, unless the **Buffered** attribute is enabled. For example, an {11,24,35,35,22,36,40} sequence of set requests results in a value of 40 being sent to the destination object. All other values result in successful completion status.

The exception to this behavior is when you enable the **Buffered** attribute. For more information, see I/O Feature advanced properties information in *Using the I/O Feature* on page 127.

Boolean data types are an exception to folding behavior when the **Buffered** attribute is not enabled. Boolean data types are used in User sets from InTouch or FS Gateway. This allows an unknown user input rate (for example, repeated button pushes) with a consistent object scan rate for outputs, and creates reproducible results.

In this case, a combination of folding as described plus maintenance of a queue of one element deep better meets the expectation of users. To begin with, the first value set after the object is deployed (the default True or False) is always written to its destination.

Subsequently, the following occurs during a single scan cycle: A two-tiered caching scheme of a Value to be Sent and a Next Value to be Sent is implemented. The Value to be Sent is based on data change as compared to the last value sent to the destination object. The Next Value to be Sent is based on data change as compared to the Value to be Sent value.

When the first data change occurs, the new value is cached in the Value to be Sent queue. Folding occurs if the same value is requested again. If another value change occurs, this second value is cached in the Next Value to be Sent queue. Again, folding occurs if the same value is requested again.

The Value to be Sent value is sent during the next scan cycle, and the Next Value to be Sent value is sent during the following scan cycle.

---

**Note:** In the case of Boolean data types used in Supervisory sets (sets between ApplicationObjects and ArcestrA) or a mixture of Supervisory and User sets during a single scan cycle, the behavior is the same as the other data types.

---

For Boolean data types and User sets, the following examples apply:

Previous Scan Cycle Value Sent	Scan Cycle Set Requests	Value to be Sent	Next Value to be Sent
0	1,0,0,1,1	1	none
1	1,0,0,1,1	0	1
0	1,1,0,0	1	0
1	1,1,0,0	0	none

When the same attribute is extended with an Input extension and an Output extension, writes to the Output extension's **Destination** occur every scan regardless of whether the extended attribute has changed.

This behavior occurs even when the **Output Every Scan** check box is cleared, which may add more network traffic. The behavior does not apply to an Input extension.

## Quality of Read, Read/Write, and Write I/O

When the object is On Scan, the value and quality of an attribute configured with a Read I/O Feature mirrors the quality of the externally referenced attribute in the case of successful reads. The data quality of the configured attribute is set to Bad when reads fail because of communication errors or datatype conversion failures.

While the extended object is On Scan, it behaves as follows: If an external set (for example, from a user) to the configured attribute causes either the value or quality to change, then a write of the configured attribute's value to the destination occurs during the next execute phase.

The quality must be Good or Uncertain for a write to occur. For writes to occur because of a quality change, the quality change must be a transition from Bad or Initializing to Good or Uncertain. The attribute called WriteValue is publicly exposed.

When the configured object is Off Scan, quality is always Bad and user sets are accepted.

## Configure I/O for Use with AVEVA Telemetry Server Communication Drivers

If you are using AVEVA Telemetry Server Communication Drivers to send and receive data from Application Server, you will need to configure I/O attributes, in accordance with the applicable communications protocol. The Telemetry Server communicates with outstations via the DNP3, IEC-60870, and/or Modbus protocols. Use the reference syntax that corresponds with the protocol you are using. Since some outstations may send data to the Telemetry Server only once per day, you will also need to enable buffering to ensure that data is not lost.

---

**Note:** If multiple attributes with I/O enabled are configured for buffered data from the same data point in the Telemetry Server, only one attribute will receive the buffered data at run time.

---



---

**Note:** The IEC-60870 protocol is not supported in AVEVA Telemetry Server version 1.0.

---

### Dynamic References

You can configure Telemetry Server data points using dynamic references. When you create and deploy an attribute with a dynamic reference in Application Server, the point is automatically generated on the Telemetry Server, and eliminates the need to manually create the same reference in two places.

## Additional Information

For more information about enabling buffered data and creating Telemetry Server references, see the following sections:

- *Working with Buffered Data* on page 439
- *Telemetry Server Data References* on page 464
- *Telemetry Server Dynamic Tags* on page 465

## Using the History Feature

Any attribute that exists at run time and is not already historized can be configured with a history Feature.

The screenshot displays the configuration window for an attribute named 'Attribute001'. The 'Available features' section includes buttons for I/O, History (checked), Limit alarms, ROC alarms, Deviation alarms, Bad value alarm, Statistics, and Log change. The 'History' feature is expanded, showing the following configuration options:

- Description: me.Attribute001.Description
- Force storage period: 0 ms
- Value deadband: 0.0 EU
- Trend high: 10.0 EU
- Trend low: 0.0 EU
- Interpolation type: SystemDefault
- Rollover value: 0.0
- Enable swinging door
- Rate deadband: 0.0 %

A history Feature can be added to a template or an instance attribute. If added to a template attribute, the existence of the history Feature is automatically locked in derived objects.

You can configure Writeable and Calculated attributes of the following data types with a history Feature:

- Float, Double (stored as a Float)
- Integer
- Boolean
- String stored as Unicode, 512 character limit
- Custom Enumeration stored as an Integer
- ElapsedTime stored as seconds

You can configure the following attributes for a history Feature:

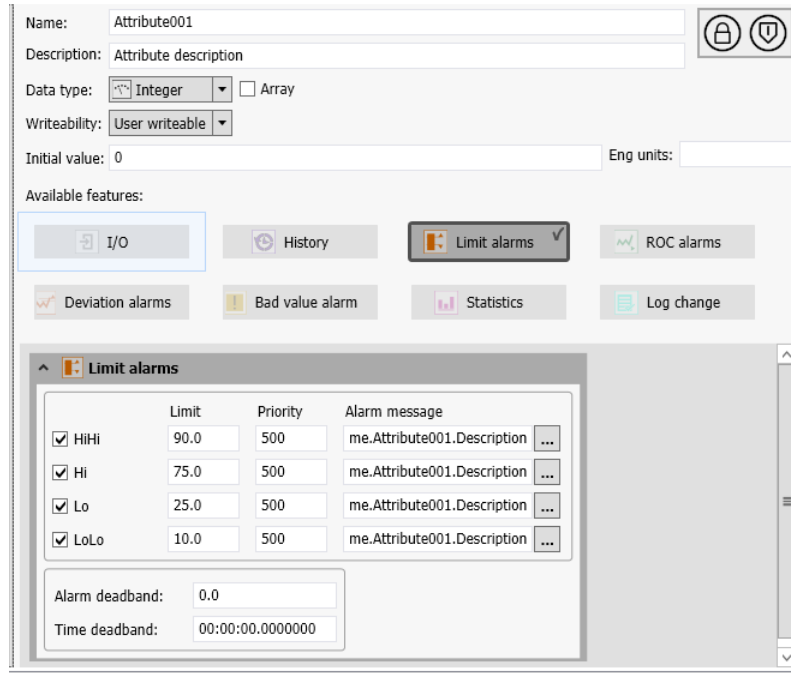


History Feature Attributes	Description
Description	<p>The attribute containing the description string, if any, associated with the attribute being configured.</p> <p>Enter the attribute manually, or use the browse button to open the Galaxy Browser to find a specific attribute.</p> <p>Example: For the description associated with Attribute001 that has been added to object Reactor31, you would enter or browse to "Reactor31.Attribute001.description".</p> <p>The description also can be a simple string without reference to an attribute.</p>
Force Storage Period	<p>Period after which the value, in milliseconds, must be historized even if the value has not changed.</p> <p>A value of 0 disables the parameter.</p> <p>Example: A setting of 3600000 indicates the value must be stored once per hour (measured from the time the object was last put onscan).</p> <p>If the force storage period value is less than the scan period of the host object, forced storage will occur every scan period (effectively equivalent to setting a value deadband of 0).</p>
Value Deadband	<p>The threshold value, measured in engineering units, that the absolute value of the difference between the new and last-stored values must differ before storing the new value to history.</p> <p>A value of zero (0) is valid and means that any level of change results in the new value being stored.</p> <p>A change in Quality always causes a new record to be stored, regardless of whether the Value has changed.</p>
Trend High	<p>The default top of a trend scale.</p> <p>This value must be greater than or equal to the low value for the trend.</p> <p>If this value is changed at run time, the maximum engineering unit change is not reflected in the HHistorian until you redeploy the object.</p> <p>This attribute applies only to numeric data types.</p>
Trend Low	<p>The default bottom of a trend scale.</p> <p>This value must be less than or equal to the high value for the trend.</p> <p>If this value is changed at run time, the minimum engineering unit change is not reflected in the HHistorian until you redeploy the object.</p> <p>This attribute only applies to numeric data types.</p>
Enable swinging door	<p>Enable and provide a valid swinging door rate deadband and the force storage period becomes the deadband override period.</p> <p>Boolean or string data types cannot be configured with a swinging door deadband.</p>

<b>History Feature Attributes</b>	<b>Description</b>
Rate deadband	<p>Available only if swinging door is enabled.</p> <p>The percentage rate of change deadband based on the change in the slope of incoming data values to the Historian.</p> <p>Example: A swinging door rate deadband of 10 percent means that data is saved to the Historian if the percentage change in slope of consecutive data values exceeds 10 percent.</p> <p>Default is 0.0, which indicates a swinging door rate deadband is not applied. Any percentage greater than 0.0 can be assigned to the rate deadband.</p>
Interpolation type	<p>The method used by the Historian to interpolate analog historical data. The interpolation type determines which analog value is selected during a Historian data retrieval cycle.</p> <p>Select an Interpolation type:</p> <p><b>System Default:</b> The Historian system-wide interpolation setting is used. The system-wide setting must be either stairstep or linear interpolated.</p> <p><b>Stairstep:</b> The last known value is returned with the given cycle time. If no valid value can be found, a NULL is returned to the Historian.</p> <p><b>Linear:</b> Historian calculates a new value at the given cycle time by interpolating between the last known value prior to the cycle time and the first value after the cycle time.</p>
Rollover value	<p>A positive integer value that represents a tag's reset limit when the Historian operates in counter retrieval mode.</p> <p>In counter retrieval mode the Historian uses a tag's rollover value to calculate and return the delta change between consecutive retrieval cycles.</p> <p>The default value is 0.0.</p> <p>Boolean and string data types cannot be configured with a rollover value.</p>

## Using the Limit Alarms Feature

Select the **Limit alarms** Feature to add and configure a Limit Alarm on an attribute of Integer, Float, or Double data type. You can add a Limit alarm Feature to a template or instance. If added to a template attribute, the Limit alarm Feature is automatically locked in derived objects. Limit alarm Features cannot be added to attribute arrays.



You can enable up to four categories of Limit alarms. When enabled, alarms will be triggered when the value reaches the configured limit.

- HiHi
- Hi
- Lo
- LoLo

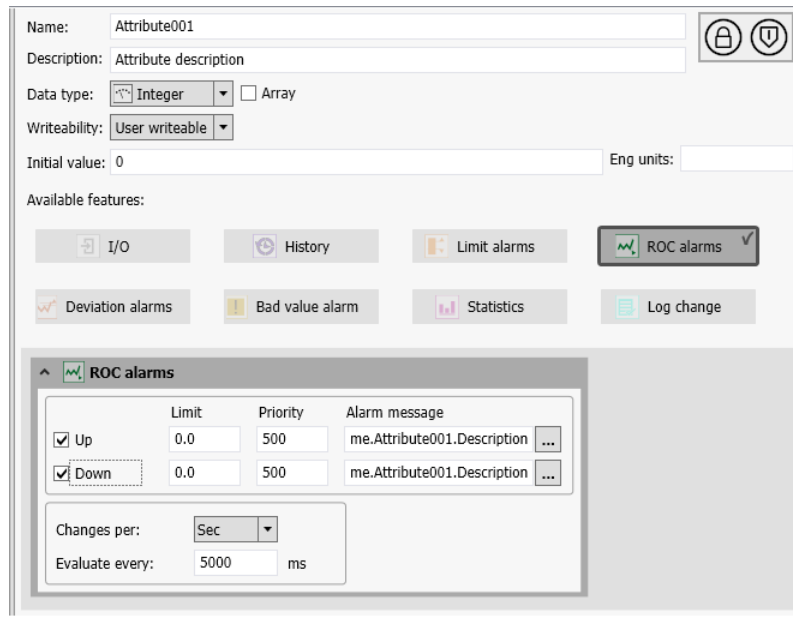
For each Limit alarm, specify the following parameters:

Limit Alarm Feature Parameters	Description
Limit	The limit that the attribute value must exceed to trigger an alarm. Enter the limit value for each enabled Limit alarm category.
Priority	A numeric value for the urgency of the alarm. Valid values are 1 through 999, with 1 being the most urgent.  For more information about pre-configuring alarm priorities, see Mapping Alarm Severity to Priority.
Alarm message	Browse and select an existing attribute or type a text string as an alarm message. This text string appears in the InTouch alarm view.

Limit Alarm Feature Parameters	Description
Alarm deadband	The amount, in engineering units, that the attribute value must exceed the configured HiHi, Hi, Lo, or LoLo limit before an alarm is triggered.
Time deadband	The time, in seconds, that must elapse after the attribute value exceeds an alarm limit before the alarm is triggered.

## Using the ROC Alarms Feature

Select the **ROC alarms** Feature to add and configure a Rate of Change (ROC) alarm on an attribute of Integer, Float, or Double data type. You can add a ROC alarm Feature to a template or instance. If added to a template attribute, the ROC alarm Feature is automatically locked in derived objects. ROC alarm Features cannot be added to attribute arrays.



You can enable up to two categories of ROC alarms. When enabled, alarms will be triggered when the value reaches the configured limit.

- Up: Rate of increase
- Down: Rate of decrease.

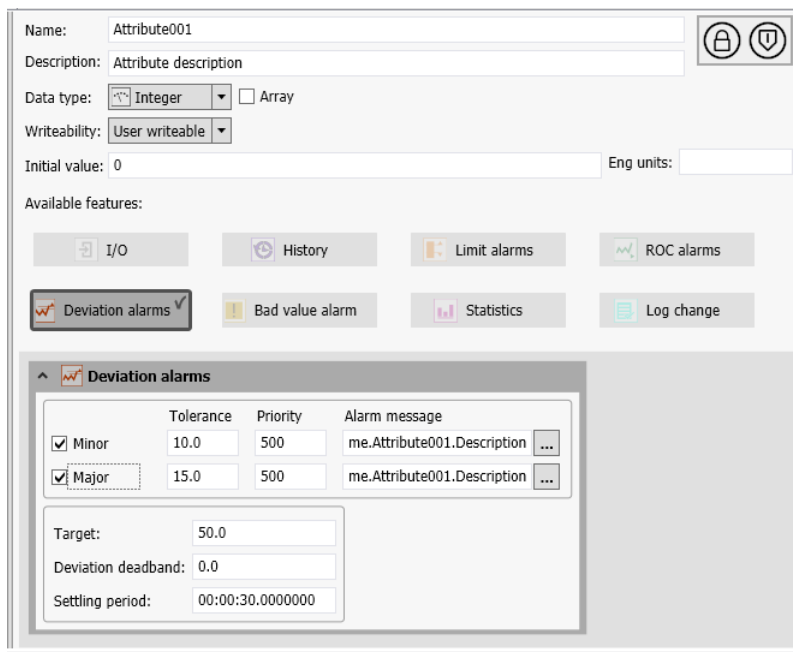
For each ROC alarm, specify the following parameters:

ROC Alarms Feature Parameters	Description
Limit	The limit that the attribute rate of change value must exceed to trigger an alarm. Enter the limit value for each enabled alarm category.  The down limit is a positive value, but indicates a negative rate of change, or a descending rate Hi limit.

ROC Alarms Feature Parameters	Description
Priority	A numeric value for the urgency of the alarm. Valid values are 1 through 999, with 1 being the most urgent.  For more information about pre-configuring alarm priorities, see Mapping Alarm Severity to Priority.
Alarm message	Browse and select an existing attribute or type a text string as an alarm message. This text string appears in the InTouch alarm view.
Changes per	Select a unit of time for the rate of change calculation – seconds, minutes, hours, or days.
Evaluate every	The time interval, in milliseconds, at which the rate of change calculation and detection is performed.

## Using the Deviation Alarms Feature

Select the **Deviation alarms** Feature to add and configure a Deviation alarm on an attribute of Integer, Float, or Double data type. You can add a Deviation alarm Feature to a template or instance. If added to a template attribute, the Deviation alarm Feature is automatically locked in derived objects. Deviation alarm Features cannot be added to attribute arrays.



You can enable up to two categories of Deviation alarms. When enabled, an alarm will be triggered when the attribute level deviates from a target value by a configured amount.

- Major: The major alarm deviation tolerance.
- Minor: The minor alarm deviation tolerance.

For each Deviation alarm, specify the following parameters:

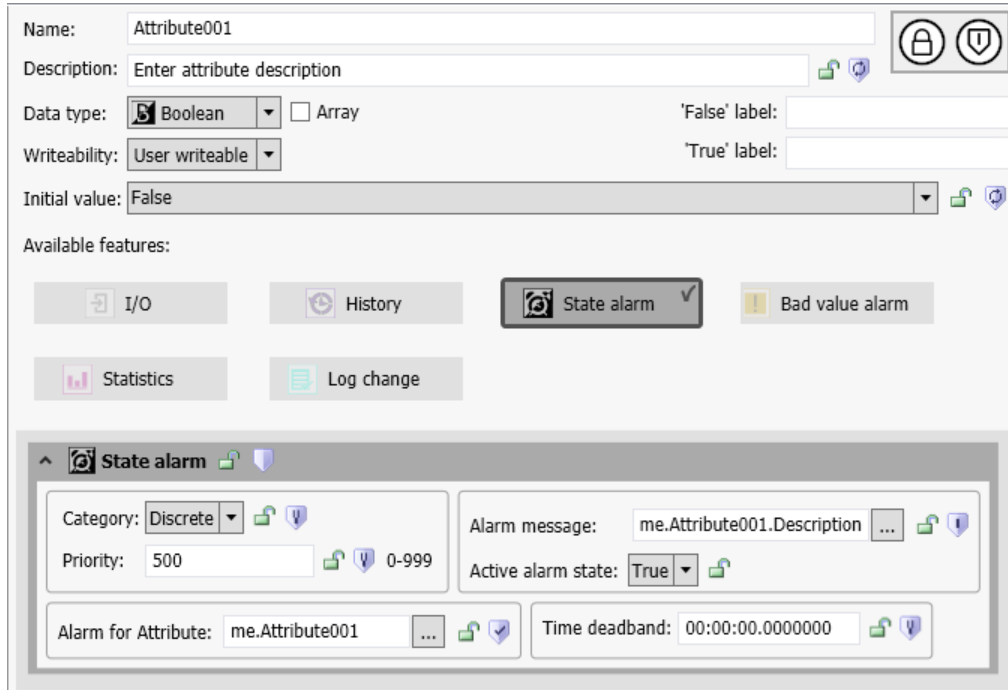
---

<b>Deviation Alarms Feature Parameters</b>	<b>Description</b>
Tolerance	The major or minor deviation tolerance that the attribute value must exceed to trigger an alarm.
Priority	A numeric value for the urgency of the alarm. Valid values are 1 through 999, with 1 being the most urgent.  For more information about pre-configuring alarm priorities, see Mapping Alarm Severity to Priority.
Alarm message	Browse and select an existing attribute or type a text string as an alarm message. This text string appears in the InTouch alarm view.
Target	The setpoint value for deviation calculation.
Deviation deadband	The amount the attribute value must drop (measured in engineering units) below any of the deviation limits before the respective condition is set to FALSE.
Settling period	The time, in seconds, allowed for the attribute value to return within allowable range after a Target change is made before the deviation alarm is triggered.  A deviation alarm cannot occur again until the alarm clears.  The value you specify will be converted to the following format: days hh:mm:ss.ffffff.

---

## Using the State Alarm Feature

Select the **State alarm** Feature to add and configure a State alarm on an attribute of Boolean data type. You can add a State alarm Feature to a template or instance. If added to a template attribute, the State alarm Feature is automatically locked in derived objects. State alarm Features cannot be added to attribute arrays.



You can set an alarm message and Priority for a state alarm. The time deadband is used to filter out rapid, transitory value spikes, and the active alarm state determines the trigger value (True or False) that will activate the alarm. You can also link the State alarm to an external attribute to capture alarm conditions detected in the PLC, rather than in the Application Server object.

For each State alarm, specify the following parameters:

### State Alarms Feature Parameters

Parameters	Description
Category	The alarm category. Select a category from the list box. See <i>State Alarms</i> on page 362 for additional information.
Priority	A numeric value for the urgency of the alarm. Valid values are 1 through 999, with 1 being the most urgent.  For more information about pre-configuring alarm priorities, see <i>Configuring Alarm Historization, Mapping, and Shelving Priority Ranges</i> on page 391.
Alarm message	Browse and select an existing attribute or type a text string as an alarm message. This text string appears in the InTouch alarm view.
Active alarm state	The state of the active alarm, True or False, when the alarm is triggered.

---

<b>State Alarms Feature Parameters</b>	<b>Description</b>
Alarm for Attribute	<p>This lets you link to an external attribute, such as a PV attribute, to provide historical state alarm data from the PLC. This allows the PLC to pass alarm conditions it detects directly to Application Server. You can use the Attribute Browser to select which attribute to link.</p> <p>The Alarm for Attribute parameter is for use only with Historian Clients and is not compatible with InTouch HMI clients.</p> <p>For more information, see <i>How to Configure an External Alarm Source</i> on page 144.</p>
Time deadband	<p>The time, in seconds, that must elapse after the attribute value exceeds an alarm limit before the alarm is triggered.</p>

---

## How to Configure an External Alarm Source

You can only configure an external alarm source for use with Historian Clients, not InTouch clients.

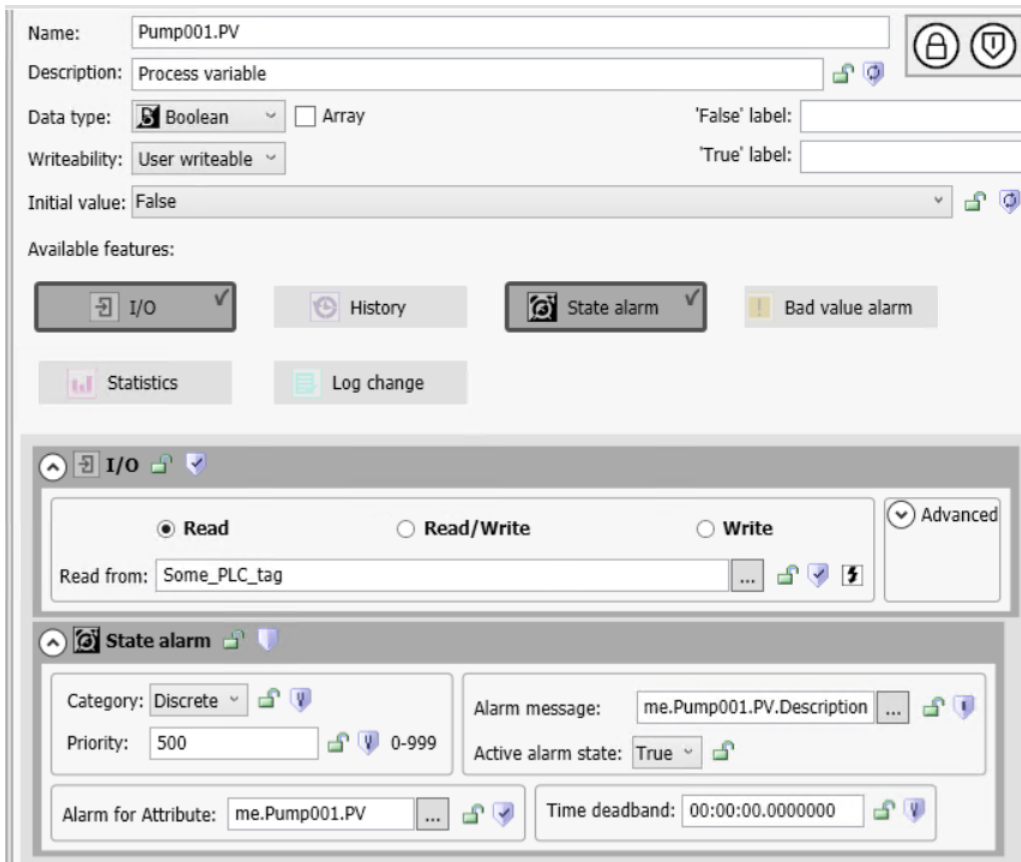
To get the alarm history records for an externally detected alarm (from within a PLC), create a reference to the `source_processvariable` property, as follows:

1. Create a Boolean user-defined attribute.
2. Enable I/O and State Alarm features for the Boolean.
3. Bind the IO feature for the Boolean to a tag in the PLC.
4. Set the I/O to read the PLC alarm bit where the alarm is being detected in the PLC. Note that there is nothing in the Boolean attribute definition to indicate it represents an alarm associated with another attribute (like a “.PV” attribute).
5. To establish the alarm association, associate the Boolean (created in step 1) with another attribute. At runtime, this association will set the “`source_processvariable`” in the alarm history event. This allows clients of History Event and Alarms records to correlate externally detected alarms with the attributes against which they are being reported.

See the sample configuration, below, for additional details.



### Sample External Alarm Source Configuration



**Attribute Name:** Pump001.PV

**Type:** Boolean

**IO:** Enabled

Points to a tag within the PLC.

**State Alarm:** Enabled

**Alarm For Attribute:** me.Pump001.PV

**Attribute Name:** Pump.PV.Hi

**Type:** Boolean:

**IO:** Enabled:

Points to an alarm tag in the PLC.

This tag will be true if there is a high alarm generated by Pump.PV.

**State Alarm:** Enabled -

**Alarm For Attribute:** me.Pump001.PV

**Attribute Name:** Pump.PV.HiHi

**Type:** Boolean:

**IO: Enabled:**

Points to an alarm tag in the PLC.

This tag will be true if a high-high alarm is generated by Pump.PV.

**State Alarm: Enabled**

**Alarm For Attribute:** me.Pump001.PV

## Using the Bad Value Alarms Feature

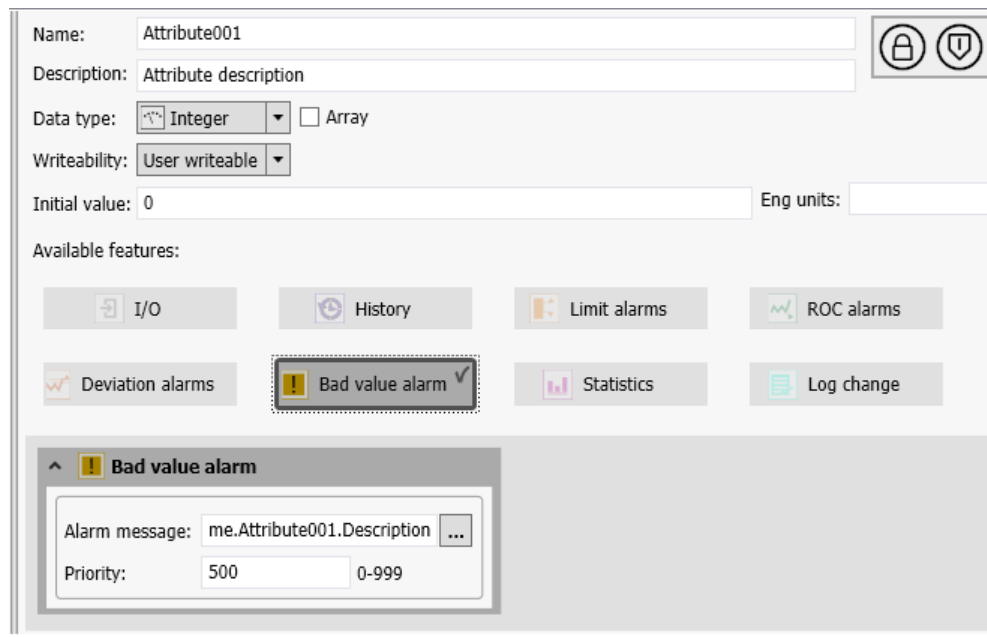
Select the **Bad value alarms** Feature to add and configure a Bad Value alarm on an attribute of Boolean, Integer, Float, or Double data type. If enabled, alarms will be triggered when the attribute has bad data value or data quality.

A bad value results from bad data quality. Bad quality can occur when:

- The PLC indicates that quality is bad (for example, if communications with a physical device is lost).
- There is a communications error (for example, an OPC or I/O server failure).
- There is a configuration error (for example, a broken I/O reference).
- An out of range value is recorded.
- A script sets data quality to bad.

You can configure graphic elements to highlight when a bad data quality state occurs, and its underlying cause. See *Use Quality and Status Styles to Provide Run-Time Feedback* on page 169 for more information.

You can add a Bad Value alarm Feature to a template or instance. If added to a template attribute, the Bad Value alarm Feature is automatically locked in derived objects. Bad Value alarm Features cannot be added to attribute arrays.



In the **Alarm message** box, you can browse and select an existing attribute or you can type a text string as an alarm message. This text string appears in the InTouch alarm view.

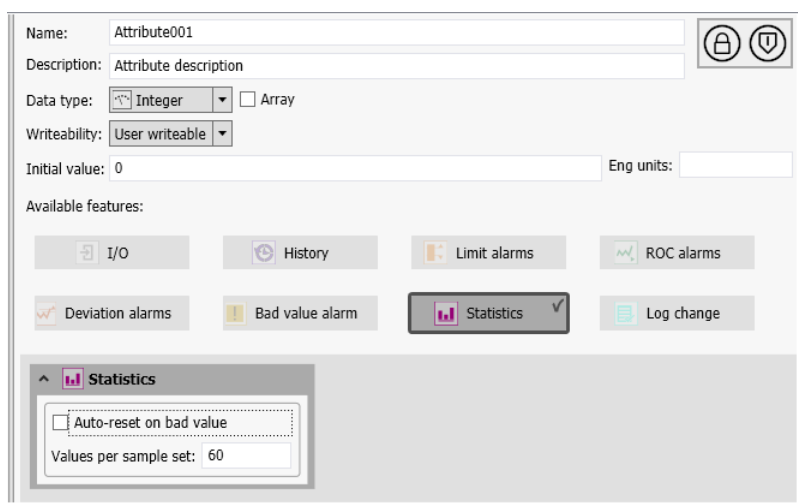
Specify a **Priority** for this alarm, a numeric value for the urgency of the alarm. Valid values are 1 through 999, with 1 being the most urgent.

For more information about pre-configuring alarm priorities, see *Configuring Alarm Historization, Mapping, and Shelving Priority Ranges* on page 391.

## Using the Statistics Feature

Select the **Statistics** Feature to add and configure calculation of statistics on an attribute of Boolean, Integer, Float, or Double data type. If enabled, the following statistics are calculated:

- Time since last transition: If no change in the input value, the elapsed time value increments to show the total elapsed time since the last transition. A change in input value resets the elapsed time since last transition.
- Average: Defined as the sum of the sample size divided by the number of sample size attributes in the data set.
- Standard deviation: Defined as the amount of variation or "scatter" from the average.



You can configure the following parameters for the Statistics Feature:

Statistics Feature Parameters	Description
Auto reset on bad value	If enabled, the input automatically triggers a reset command when the quality of an input goes from BAD or UNCERTAIN to GOOD.
Values per sample set	The maximum number of samples to collect in calculating the mean and standard deviation. The minimum number of samples is 2. A large sample set will impact run-time performance. This attribute is unavailable for a Boolean data type.

## Using the Log Change Feature

Select the **Log Change** Feature to log system events for an attribute of Boolean, Integer, Float, or Double data type.

When enabled, the Log Change feature logs data change events as well as user events.

## Using I/O Auto Assignment

Instead of configuring I/O references manually, or writing scripts to set them at run time, you can use I/O auto assignment. Manual configuration of I/O references can be time consuming. Scripting these references eliminates the issues of manual configuration, but can significantly increase the time needed for deploying objects. With I/O auto assignment, you do not need to check out individual objects to configure I/O references, and you do not experience the run-time penalties associated with scripting.

**Note:** I/O auto assignment is the default setting for application and other system objects, such as area objects. Device Integration objects must be manually configured.

When you add input or output attributes to an area or application object in the **Attributes** page of the Object Editor, the default setting prepares these attributes for I/O automatic assignment. The auto assignment reference appears in the I/O section of the **Attributes** page if you have enabled the I/O attribute feature. The default string for an input reference is:

```
<IODevice>.[ObjectName].[AttributeName].InputSource
```

where *[ObjectName]* is the hierarchical name of the application or system object, and *[AttributeName]* is the name of the attribute.

The default string for an output reference is:

```
<IODevice>.[ObjectName].[AttributeName].OutputDest
```

The string *<IODevice>* is a placeholder that indicates the I/O reference will be built automatically in a process called auto binding. The reference resolves automatically as an auto-bound reference when you link the object to a scan group and DI object, without having to manually enter or script the reference.

The following is an example of an I/O reference string before the object has been assigned to a scan group and DI object:

```
<IODevice>.Mixer.Tank.Inlet.InputSource
```

Once you assign the object to a scan group, the reference resolves to include the assigned Device Integration object and scan group. For example, assigning the object to the scan group "Fast" under DI object "OPC001" will change the reference to:

```
OPC001.Fast.Mixer.Tank.Inlet.InputSource
```

**Important:** Do not lock InputSource or OutputDest attributes when using I/O auto assignment. If either InputSource or OutputDest attributes, or both, are locked in the parent template, the attributes cannot be updated with the resolved auto-bound reference when the object is deployed, and the run-time value will be "---Auto---".

Name: Attribute001

Description: Attribute description

Data type: Integer  Array

Writeability: User writeable

Initial value: 0 Eng units:

Available features:

I/O  History  Limit alarms  ROC alarms

Deviation alarms  Bad value alarm  Statistics  Log change

**I/O**

Read  Read/Write  Write

Read from / Write to: <IODevice>.R31.Attribute001

Output destination differs from input source

**Advanced**

Buffered Deadband: 0

Reflect input to output

Enable I/O scaling

I/O auto assignment is configured in the **IO Devices view**. Use this view to associate application and system objects with DI objects and scan groups.

Scan groups are contained by DI objects and help categorize devices that are associated with them on the basis of how often their I/O points are scanned. For basic information about DI objects, refer to *Device Integration Templates* on page 47. For basic information about scan groups, refer to the OPCClient Object help file.

**Note:** A DI object will not appear in the **IO Devices view** unless at least one custom scan group has been defined for it.

Assign objects that need to be closely monitored or that have a faster rate of change to a scan group with a faster scan rate. Objects with slower rates of change can be assigned to a scan group with a slower scan rate. The scan rate is the rate at which the AVEVA run time will perform scans. This may be different than the scan rate a PLC uses to monitor the hardware attached to it.

**Note:** Fast scan groups consume more network and computing resources than slow scan groups. In a large Galaxy, load balancing may require you to use multiple scan groups with similar scan rates.

The I/O references for the objects' attributes set to use I/O automatic assignment are dynamically generated when you assign an object to a scan group. The auto-bound reference replaces the placeholder string and is displayed in the IDE. It is generally in the following format:

```
<DIObject>.<ScanGroup>.<Object>.<Attribute>
```

<DIObject> and <ScanGroup> correspond to the name of the DI object and scan group to which you assign the object. For example, if the DI object name is `DI01` and the scan group name is `FastSG`, then the auto-bound reference will start with `DI01.FastSG`.

<Object> may consist of multiple elements if there are contained objects. The following is an example of an object string:

```
Tank3.Valve1.Attr2
```

Here, `Tank3.Valve1` corresponds to <object>. The complete auto-bound I/O reference would be:

```
DI01.FastSG.Tank3.Valve1.Attr2
```

Although you assign objects, references are generated for each I/O attribute associated with the object, and not for the object itself. Therefore, you may have to change the I/O references for a subset of an object's attributes. Once objects are associated with DI objects and scan groups, you can validate the auto-bound assignment for each of the object's attributes in the **IO Device Mapping view**, and if necessary, you can enter override values in this view.

## Assigning Areas and Objects to Scan Groups

The **IO Devices view** displays all application and system objects, as well as DI objects that have custom scan groups associated with them. Templates are not shown in this view, nor are DI objects that do not have custom scan groups defined for them (DI objects with only the default scan group are not displayed in the IO Devices view).

---

**Note:** You cannot assign application and system objects to a default scan group; these assignments can only be made for custom scan groups.

---

In the **IO Devices view**, objects and areas are initially unassigned and placed in a flat view under the "Unassigned IO Devices" folder. You then select one or more of these unassigned objects or areas and drag and drop them onto a scan group. Deployed application and system objects cannot be assigned to a scan group. Undeploy them first.

---

**Note:** DI objects can remain deployed during the assignment process.

---

Once an object is attached to a scan group, the I/O references for the objects' attributes set to use I/O automatic assignment are automatically generated for you by concatenating the DI object, scan group, object (including contained objects), and attribute names.

You can assign multiple objects for auto-binding at once. Hold down the **shift** or **ctrl** key to select multiple objects and drag the selected items to a scan group. Object assignment is based on hierarchy and area inheritance, as follows:

- If all the objects you select are at the same hierarchical level, all contained (subordinate) objects that have not been assigned, except sub-areas, are also selected and will move to the same scan group.
  - To assign a sub-area, it must be specifically selected.
  - If some contained objects were previously assigned to a different scan group, these will retain their original assignments. To change an existing assignment, you must specifically select the assigned object and move it to a different scan group.
- If you select objects at different hierarchical levels for assignment to a scan group – for example, if you select an area, an object within the same area, and an object from another area – then only the selected items move to the scan group.
- If you select an area that contains another area, only the area selected will move. You must specifically select an area to assign it, regardless of its hierarchical level.

When an area or containing object is assigned to a scan group, the objects it contains are assigned as follows:

- Objects within the area that share the area's device linkage will be reassigned to the new device along with the area. This remains true even in the case where both the area and the contained objects are not currently linked to a device. Unassigned objects within the area will use the area's assignment.
- Objects within the area that are already assigned to a scan group that is different from that of the hosting area will retain their original assignments.
- If necessary, you can reassign contained objects to a different scan group after the initial assignment, or override assignments in the **IO Device Mapping view**.

---

**Note:** You cannot change scan group or DI object assignments for deployed objects in the **IO Devices view**. You must undeploy the object first.

---

Deleting a scan group will delete the I/O assignments of any objects that were previously assigned to it, including override values.

---

**Caution:** Creating or deleting scan groups within a derived DI object template may overwrite I/O assignments when these changes are propagated to instances of the DI object template. If this happens, all I/O auto assignment information for objects linked to the updated DI objects, including override values, will be lost. The objects will move to the "Unassigned I/O Devices" folder.

---

### To assign an object or area to a scan group

1. In the **IO Devices view** or **Model view**, select application objects and areas for assignment.
2. Drag and drop the objects and areas to a scan group in the **IO Devices view**.
3. The object or area and all contained objects will be moved to the selected scan group.

---

**Note:** If the area or object has contained objects that were previously assigned to a different scan group, these will retain their original assignments. Contained areas will not move and must be overtly selected.

---

4. The I/O references for each attribute set for I/O auto assignment will be dynamically configured and displayed as auto-bound in the **IO Device Mapping view**.

## Renaming Application, System, and DI Objects

You can rename application, system and DI objects in the **IO Devices**, **Model** and **Deployment** views. If an area or object has already been assigned to a scan group, its I/O references are updated automatically.

## Renaming Scan Groups

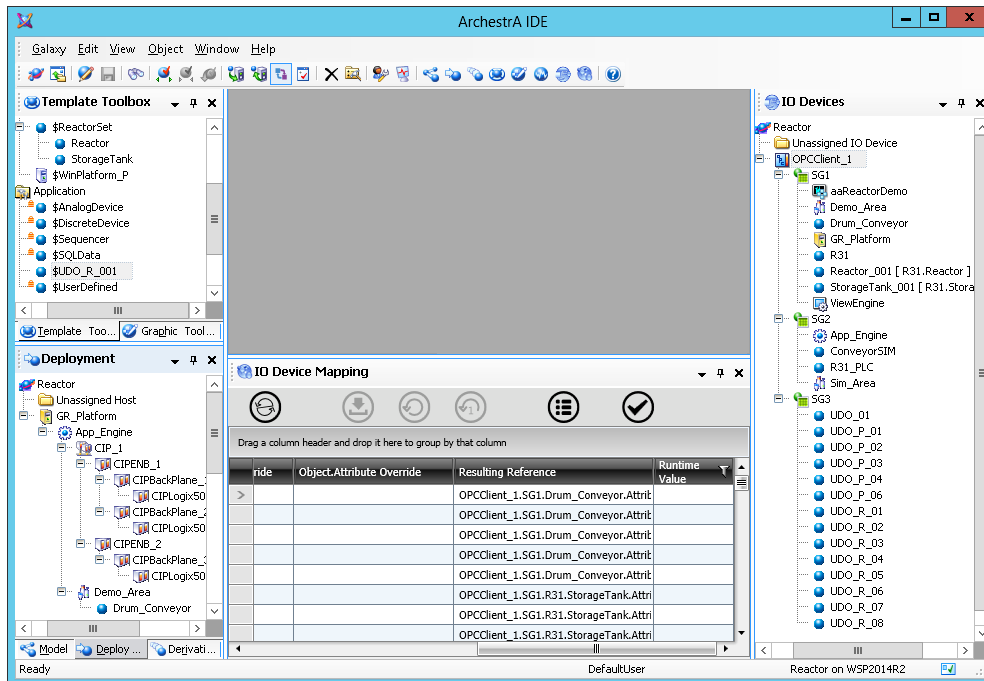
Changes to scan groups (renaming, adding, or deleting them) can only be done by editing the DI object that contains the scan group in the Object Editor. Use the DI object's **Scan Group** page to make any necessary changes.

If scan groups are renamed at the template level, the changes will propagate to the DI object instances and I/O assignments will be preserved.

## Validating and Editing I/O Assignments

The **IO Device Mapping view** is a table that displays I/O auto assignment references for application and system objects that are linked to DI object scan groups. Only auto-bound references are displayed in the **IO Device Mapping view**. Application and system objects not yet assigned to a scan group, and manually configured references are not shown. To be visible in the **IO Device Mapping view**, the object must be selected in the **IO Devices view**.

When you initially open the **IO Device Mapping view** after starting the IDE, the table will scroll so the far right column is in view.



- Selecting a DI object in the **IO Devices view** lists I/O auto assignment attributes for all auto-bound objects linked to all scan groups under it.
- Selecting individual scan groups restricts the scope of the information displayed in the **I/O Device Mapping view** to the auto-bound objects that have been linked to the selected scan groups.
- Selecting individual application or system object further restricts the scope of attributes displayed to only those associated with the selected object.

---

**Note:** You can select multiple items in the **IO Devices view**. Selected items can be at different hierarchical levels. Selecting a subordinate object will exclude non-selected objects within the device hierarchy, even though the parent object is selected.

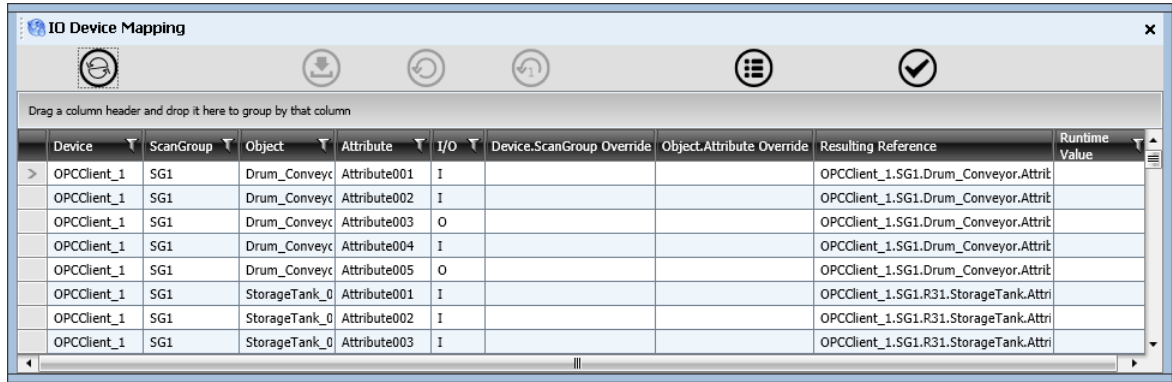
---

In the **IO Device Mapping view**, you can view and validate I/O references for each automatically generated attribute, and you can override the automatically generated I/O references. As is the case in the **IO Devices view**, you do not have to check out objects to change their I/O assignments.



## Using the IO Mapping View

The IO Mapping view is divided into columns, each of which displays information for an I/O attribute that has been auto assigned.

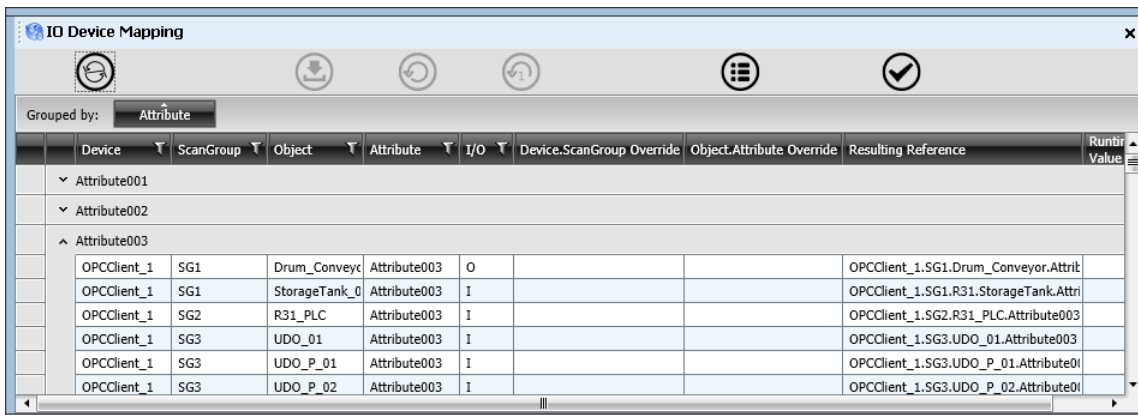


Column Heading	Definition
Device	Device Integration Object (DIO) name
ScanGroup	Scan group name
Object	Application object or area object name
Attribute	Attribute name
I/O	Input or Output
Device.ScanGroup Override	Override setting: enter an override value to replace the DIO and scan group that was automatically assigned
Object.Attribute Override	Override setting: enter an override value to replace the object and attribute names
Resulting Reference	Concatenated I/O reference string (for example, "DIO1.SG2.Mixer3.Attr5" or "DIO1.SG2.Conveyor.40001")
Runtime Value	Data returned from run time when the <b>Validate References</b> button has been clicked

You can change how I/O attributes are displayed by sorting, grouping, or filtering the attributes.

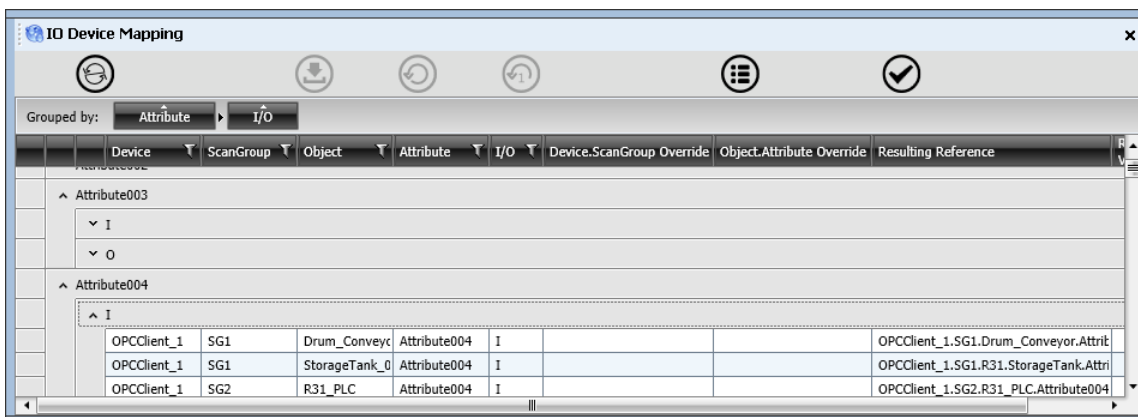
Attributes are initially sorted by Device name (DI object). To sort attributes using a different column, click on the column heading once to sort in ascending order. Click a second time on the same column heading to sort in descending order. A third click on the column heading turns off sorting for that column.

To group objects by I/O assignment criteria, drag a column heading to the top of the table in the **IO Device Mapping view**. For example, if you drag the heading "Attribute" to the top of the table, I/O assignments will be grouped by attribute name. In the figure below, I/O assignments are grouped by attribute name. Note each attribute grouping has been expanded.



**Note:** Grouping is not available for the **Device.ScanGroup Override**, **Object.Attribute Override**, or **Resulting Reference** columns.

You can use more than one column heading to group I/O assignments. In the figure below, assignments are grouped first by attribute name, and then by I/O type.



Columns can also be filtered to display only certain entries by clicking the filter symbol and then selecting the filtering criteria. When a filter is applied, the color of the filter symbol changes to orange.

**Note:** Filtering is not available for the **Device.ScanGroup Override**, **Object.Attribute Override**, or **Resulting Reference** columns.

The table contains additional information that you can reveal by clicking the **Show Details** button.

**Details Column Heading**

**Definition**

Attribute Data Type	Attribute type as defined by the attribute definition set in the Object Editor
Target Data Type	Attribute type of the target, as determined by the run time

Details Column Heading	Definition
Runtime Quality	Data quality from run time

## Validating I/O References

Use the **IO Devices** pane and the **IO Device Mapping** pane to populate the **Resulting References** column in the **IO Device Mapping** grid. You can then validate auto-bound I/O attributes that have been assigned in **IO Device Mapping**. To begin I/O reference validation, press the **Validate References** button, available in the **IO Device Mapping** pane.

Validating I/O references is separate and distinct from the **Validate** function available in the **Template Toolbox** and in the application views. That **Validate** function validates the object's references configuration against the the Galaxy Repository. It is a configuration validation.

In contrast, validating I/O references is a run-time validation, and performs a one-time "get" from the run-time subsystem to retrieve the value and quality of the resulting references. A reference string in the **Resulting References** column is validated if a valid value and good quality are returned for that string. Objects should be deployed for an I/O reference validation to be meaningful.

**Note:** As a best practice, it is more efficient to validate each reference while performing I/O auto-assignment than it is to perform a single validation after configuring and deploying the entire application.

An estimate of the time needed for validation is displayed. The time is updated as validation progresses. When validation completes, the total number of unresolved I/O references is displayed along with the total number of auto-bound references checked, for example, "2 out of 5000 reference(s) failed to validate." To view references that did not validate during the validation period, click the "Show Advanced Columns" button. This will reveal the **Runtime Quality** column.

You can sort the **Runtime Quality** column to aggregate unresolved I/O references at the top of the list of references. See *Using the IO Mapping View* on page 153 for additional information on sorting. You can then move the associated objects to a different scan group or apply overrides. Recheck the references after completing any required reassignments and overrides.

## Overriding I/O Auto Assignments

The **IO Device Mapping view** contains two override columns. To change an I/O assignment, enter the new parameter in one or both of the two override columns.

The information in other columns cannot be edited directly. You can also reassign an object to a different scan group in the **IO Devices view**, in which case the new assignment will be reflected in the **IO Device Mapping view**.

**Note:** If an object is already deployed, changing any of its I/O assignments in the IO Mapping view will change the object's status from "Deployed" to "Deployed with pending configuration changes."

To override the DI object and scan group assignment that was made in the **IO Devices view**, enter the value you want in the column **Device.ScanGroup Override**. Naming restrictions for the Device.ScanGroup Override value are as follows:

- The value must contain one DI object name and one scan group name contained by the DI object, separated by a period (.).
- The value cannot start or end with a period.
- The value cannot have any leading spaces.

- The I/O reference cannot be longer than 329 characters.

To override the object and attribute name, enter the value you want in the column **Object.Attribute Override**. Naming restrictions for the Object.Attribute Override value are as follows:

- The value cannot have any leading spaces.
- The I/O reference cannot be longer than 329 characters.

You can change attribute information using the Object Editor (after checking out the object), instead of overriding values in the **IO Device Mapping view**. However, if you enter the specific I/O reference for an attribute in the Object Editor (manual configuration), it will no longer appear in the **IO Device Mapping view**. Only attributes configured for I/O automatic assignment appear in this view.

## Overriding Large Numbers of Attributes

If you have a large number of overrides to enter, information can be copied and pasted between the **IO Device Mapping view** and Microsoft Excel. However, you must be careful when moving data to and from the **IO Device Mapping view**. Observe the following conditions:

- Only data in the two override columns is editable. All other columns are read only.
- Even though you can only change the override columns, always copy all columns. This will help you verify that data is being copied correctly when you paste back into the **IO Device Mapping view**.
- When you copy data from the **IO Device Mapping view**, column headers are automatically included to help you while editing in Excel. When you copy your edited data back to the **IO Device Mapping view**, do not copy this header row, only copy the data.
- If you need to sort attributes, sort them in the **IO Device Mapping view** before you copy them into Excel.
- Do not change the sort order of the attributes in the **IO Device Mapping view** until you have pasted the edits made in Excel and applied them.
- Do not sort the attributes in Excel. Attributes can only be sorted in the **IO Device Mapping view** and the sort order cannot change until your edits have been applied.

---

**WARNING! The IO Device Mapping view does not verify data that is copied to it. Therefore, if the order of attributes is different in Excel than it is in the IO Mapping view, the I/O references will be broken. Once changes are applied, there is no way to undo them.**

---

### To validate and edit I/O assignments

1. Click the **Validate References** button to check that the I/O assignments in the IO Device Mapping table are valid. This action retrieves data and quality information from the run-time data acquisition subsystem.

---

**Important:** I/O assignments can only be validated if the Platform, AppEngine, and associated DI object have been successfully configured and deployed.

---

2. Check the data returned in the **Runtime Value** column. If the assignment is not valid or if the quality is bad, no data will be returned (the Value column will be empty), and "Not Connected" will be displayed in the **Runtime Quality** column (click the **Show Details** button to reveal this column).
3. To isolate invalid assignments, select the filter icon at the top of the **Runtime Quality** column filter for "Not Connected" messages.
4. Edit the I/O assignment of objects as needed.
  - To override a scan group and DI object assignment, enter the new value in the **Device.ScanGroup Override** column. The override value for this column cannot contain more than one period (.) to separate the DI object name from the scan group name.

- To override an object and attribute name, enter the new value in the **Object.Attribute Override** column.
5. Click **Apply Pending Overrides** to save your changes.
    - To discard the last unsaved override setting, click **Undo Last Pending Override**.
    - To discard the all unsaved override settings, click **Undo All Pending Overrides**.

## Uploading Run-Time Configuration Changes for Auto Assigned Objects

If changes are made during run time to an auto assigned I/O reference, you can upload these changes as overrides to the original assignments. Invalid changes are ignored, and an error message is sent to the SMC Logger.

- Run-time changes to scan groups or DI objects are saved as **Device.ScanGroup** overrides.
- Run-time changes to objects or attributes are saved as **Object.Attribute** overrides.

When you look at the object in the **IO Devices view**, you will see that the original I/O auto assignment setting is retained and the object remains auto assigned to its original DI object and scan group.

The changes uploaded from run time will appear in the **Resulting References** column of the **I/O Device Mapping view**. The same information from run time will appear when you examine the object's I/O attributes in the **Object Editor**. These changes are preserved across subsequent redeployments.

### To upload run-time configuration changes

1. From the **Model** or **Derivation** view, select the object(s) with run-time changes you want to preserve.
2. Click **Upload Runtime Changes** from the **Object** menu. The **Upload Runtime Changes** dialog box will appear and provide status of the upload process.

## I/O Auto Assignment Workflow Example

This section describes the basic steps for implementing I/O auto assignment, from start to finish. The basic steps are:

1. Use the **Attributes** page of the **Object Editor** to prepare system objects and application objects for I/O auto assignment.
2. Use the **Object Editor** to configure DI objects and scan groups.
3. Use the **IO Devices** view to assign system and application objects to scan groups (objects must be undeployed).
4. Use the **IO Device Mapping** view to validate I/O references. You can enter overrides in this view, if necessary.

### To prepare objects for I/O auto assignment

1. Create a derived template from the \$UserDefined base template and open it in the Object Editor.
2. Add attributes as needed. For each attribute that requires input or output, click the I/O button.

The default I/O mode is **Read/Write**, and the default string indicating that the attribute is primed for I/O auto assignment appears in the **Read from / Write to** field.

3. Change the I/O mode as needed.

---

**Important:** Do not change the default string if you want to use auto assignment for the attribute. The default string indicates that auto assignment is not yet complete and the object is not linked to a PLC.

---

4. Save and check in the template, then close the Object Editor.
5. In the **Template Toolbox**, create instances from the UDO derived template. The instances appear in the **Model view**, under Unassigned Area.

### To configure DI objects and scan groups

1. In the **Template Toolbox**, create a Device Integration (DI) object from the \$OPCCClient, \$DDESuiteLinkClient, or \$Redundant DIObject template. The DI object appears under "Unassigned Area" in the **Model view** and under "Unassigned IO Devices" in the **IO Devices view**.
2. Add one more scan groups to the DI object you created in the previous step. The naming convention for the scan groups must conform to the naming convention of the PLC with which you are connecting.

The scan groups will show as belonging to the DI object in the **IO Devices view**.

### To assign system and application (user created) objects to scan groups - Method 1

1. Create a new area and set it as the default. Do not deploy the area yet.
2. In the **IO Devices view**, drag and drop the area to a scan group.
3. Create new application objects. As you create new objects, they will appear in the **IO Devices view** under the default area.

The IDE will complete the references for each I/O attribute for you.

4. If you need to change the scan group assignment for an object, drag and drop it to the new scan group.

### To assign system and application (user created) objects to scan groups - Method 2

1. Create a new area and assign it to a scan group.
2. Create new application objects. These will appear under the Unassigned folder in the **Model view**.
3. Assign the application objects to the area assigned to the scan group in step 1. All application objects assigned to the area will inherit the area's linkage to the DI object and scan group.
4. If you need to change the scan group assignment for an object in the area, drag and drop it to the new scan group.

### To validate I/O references (and override auto assigned values, if necessary)

1. Open the **IO Device Mapping view**. This view opens automatically if you select an object, scan group, or DI object in the **IO Devices view**.

Each row in the IO Device Mapping table contains the I/O reference for one attribute.

2. Enter an override value for the attribute in one of the override columns.

---

**Note:** The **DI Object.Scangroup Override** column can only contain the DI object name and scan group name, separated by a period (.).

---

3. Deploy the platform and associated DI objects.
4. Click the **Validate References** button to check that communication is established with the PLC.
5. Deploy the objects.

## Using I/O Auto Assignment with OPC Clients

Using I/O auto assignment with the OPC communication protocol may require that you add overrides to allow communication between the IDE and certain PLCs, such as Allen-Bradley Logix and Siemens S7 controllers.

If you are using auto assignment to communicate with a device connected to an OPC Server on a different node (for example, a DAServer in a remote Galaxy), you will use overrides to build valid I/O references with all required parameters to fulfill the OPC hierarchical path. If bulk edits are required, you can copy and paste between the **IO Device Mapping view** and Microsoft Excel.

I/O auto assignment uses the syntax: `<DI Object.ScanGroup>. <Object.Attribute>`. It is the first part of the syntax that limits how the OPC client builds the I/O auto assignment reference, since additional parameters in the device hierarchy, such as Ethernet ports and backplanes are not included. While these parameters may be logically viewed as part of the `DI Object.ScanGroup` reference, you must add the parameters to the **Object.Attribute override** column (not the **DI Object.ScanGroup override** column) to create the fully qualified I/O reference.

Using DDESuiteLink instead of OPC will accommodate the automatically generated I/O reference to these PLCs, and in most cases, overrides will not be needed.

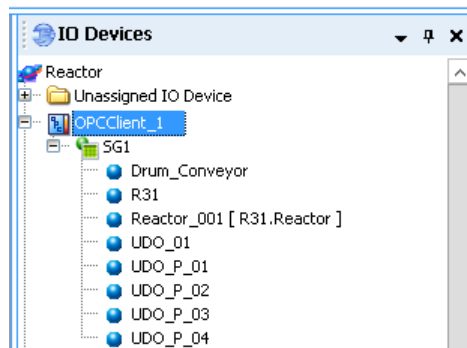
**Caution:** Do not change the order of references, either in Excel or in the IO Device Mapping view by sorting, grouping, or filtering the reference list while editing references.

## Override Scenario 1 — Siemens S7 Controllers with an OPC UA Client

The following scenario illustrates using I/O auto assignment with a Siemens S7 PLC. The Siemens S7 PLC in this example scenario includes a TCP/IP port that is in addition to the OPC DI object and scan group names captured by I/O auto assignment in the **IO Devices view**.

### To configure the Siemens S7 PLC with the OPC Client

1. Select the OPCClient object or scan group that contains the PLC in the **IO Devices view**.



I/O attributes for each object associated with the OPCClient or scan group (depending on which is selected in the **IO Devices view**) are shown in the **IO Device Mapping view**.

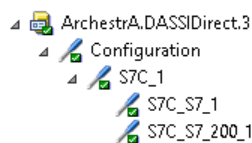
2. In the **IO Device Mapping view**, the resulting reference will be in the form:  
`OPCClientName.ScanGroupName.ObjectName.AttributeName`

Example (**Resulting Reference** column):

```
OPCClient_01.SG1.UDO_01.Attribute_001
```

3. Do not edit the **Device.ScanGroup Override** column (leave it blank). This leaves the DI Object and scan group assignment as defined in the **IO Devices view**.
4. Enter the fully qualified field reference in the **Object.Attribute Override** column. For example:

```
S7C_1.S7C_200_1.UDO_01.Attribute001
```



5. Apply the override. The resulting reference will be displayed. For example:

```
OPCClient_01.SG1.S7C_1.S7C_200_1.UDO_01.Attribute001
```

## Override Scenario 2 — Allen-Bradley CIP Controllers with an OPC UA Client

The following scenario illustrates using I/O auto assignment with a Allen Bradley Logix PLC. The Allen Bradley Logix PLC in this example scenario includes a device hierarchy with a port object, ethernet port, backplane, and controller. These are in addition to the OPC DI object and scan group names captured by I/O auto assignment in the **IO Devices view**.

### To configure the Allen-Bradley Logix PLC with the OPC Client

1. Select the OPCClient object or scan group that contains the PLC in the **IO Devices view**.

I/O attributes for each object associated with the OPCClient or scan group (depending on which is selected in the **IO Devices view**) are shown in the **IO Device Mapping view**.

2. In the **IO Device Mapping view**, the resulting reference will be in the form:

OPCClientName.ScanGroupName.ObjectName.AttributeName

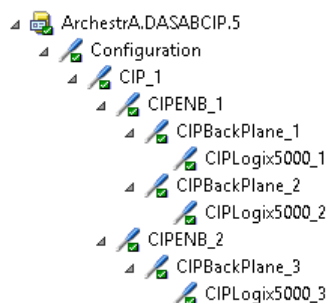
Example (**Resulting Reference** column):

OPCClient\_01.SG1.UDO\_01.Attribute\_001

3. Do not edit the **Device.ScanGroup Override** column (leave it blank). This leaves the DI Object and scan group assignment as defined in the **IO Devices view**.

4. Enter the fully qualified field reference in the **Object.Attribute Override** column. For example:

CIP\_1.CIPENB\_1.CIPBackPlane\_1.CIPLogix5000\_1.UDO\_01. Attribute001



5. Apply the override. The resulting reference will be displayed. For example:

OPCClient\_01.SG1.CIP\_1.CIPENB\_1.CIPBackPlane\_1.CIPLogix5000\_1.UDO\_01.Attribute001

## Writing and Editing Scripts

You can write scripts that run commands and logical operations based on specified criteria being met to extend and customize your objects. For example, a script can be configured to start running when a key is pressed, a window is opened, or the value of an attribute changes.

With scripts, you can create a variety of customized and automated system functions. A script adds behavior that runs when the object that contains the script is deployed and the object is either:

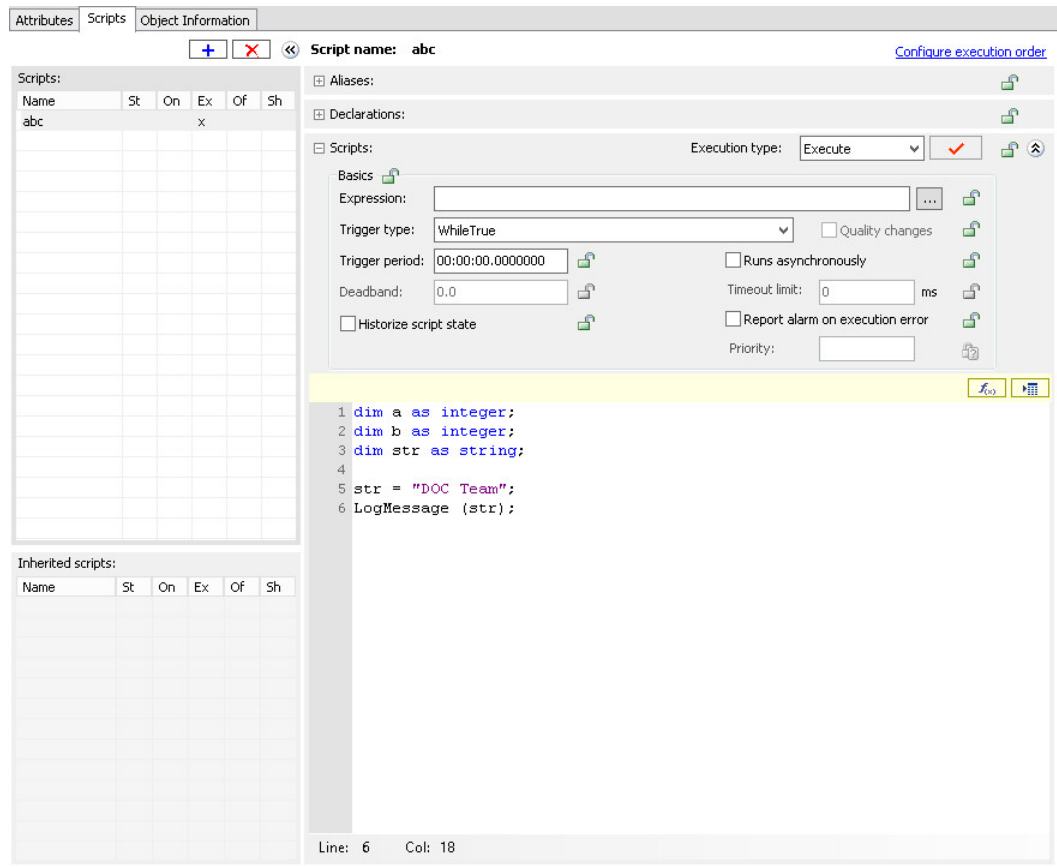
- On scan in the run-time environment or
- Changes scan or start/shutdown state

A script typically runs based on attributes of the object that contains it, but can be started by another script based on changing values of attributes of more than one object.

When a script condition is true, the script runs at least once immediately. The maximum length of a script trigger period is 49-days. A script never runs if the trigger period exceeds 49-days.



For specific information about writing scripts, including the scripting language, syntax, commands, and using .NET, see the *Application Server Scripting Guide*. For more information about the scripts page, see *About the Scripts Page* on page 79.



**Important:** You cannot pass attributes as parameters for system objects. Instead, use a local variable as an intermediary or explicitly convert the attribute to a string using an appropriate function call when calling the system object.

## About Scripts

The following characteristics apply to the scripting environment:

- Script text has no length limitations.
- Selecting a script function from the **Script Function Library** dialog box adds it and its syntax to the script text where you can edit it.
- You can save a script with syntax errors, but the object cannot be deployed until you correct the script syntax errors.
- You can validate your scripts before using them. This helps you avoid syntactically correct but semantically incorrect combinations such as two statements declaring the same variable. Variables can be declared only one time in a single block.
- You can change the name of a script at any time by renaming it in the Object Editor.
- In the run-time environment, a script execution error stops the script's current execution. Script execution is retried on the next AppEngine scan.

## Script Execution

The existence and execution order of scripts associated with an object are inherently locked at each stage of development in the template, derived template, and instance. For example, a set of scripts associated with a template are treated as an ordered block in the **Configure Execution Order** dialog box when configuring execution order in a next-generation derived template.

New scripts in the derived template can be ran in any order before and after the template's block of scripts. The derived template's execution order is treated as a block in any downstream derived templates or instances. Scripts cannot trigger any faster than the scan period of the AppEngine the script is associated with or faster than the scan period of the AppEngine that hosts the object that the script is associated with.

Scripts run in one of two modes:

- Synchronous scripting mode is the default for running scripts in the run-time environment. This mode runs scripts in order while an object is running on scan.
- Asynchronous scripting mode is a group of scripts running on the same, lower priority execution thread. These scripts only support Execute triggering and run independently from each other. Set the maximum number of independent threads in the AppEngine configuration editor. To use either scripting mode, you must select **Execute** as the **Execution Type** in the **Scripts** area on the **Scripts** page.

### To create and associate a script with an object

1. Add a script. On the **Scripts** page of the Object Editor, click the **Add ( + )** button. A script is added to the **Scripts Name** list.
2. Type a name for the script and press Enter. Script names can be up to 32 alphanumeric characters, including periods. At least one character must be a letter.

---

**Note:** For detailed information about each item on the **Scripts** page, see *About the Scripts Page* on page 79.

---

3. Select a trigger that starts the script in the run-time environment.

**Execution Type** triggers include: Startup, On Scan, Execute, Off Scan and Shutdown.

- If you select **Startup**, **On Scan**, **Off Scan**, or **Shutdown**, the **Basics** group is unavailable. The script is triggered when the object starts, goes on scan, goes off scan, or shuts down.

If you select **Execute**, the **Basics** group is available.

If you selected **Execute** as the script trigger, select a **Trigger Type**. Depending on the type selected, you are required to enter an **Expression** and/or **Trigger Period** and **Deadband** values. When the combination of **Expression**, **Trigger Type**, **Trigger Period** and/or **Deadband** is satisfied in the run-time environment, the script starts running. See the following table for more information.

The **Trigger Period** format is as follows:

Hours:Minutes:Seconds:Milliseconds

For example, 3 hours, 5 minutes, and 10.5 seconds is:

03:05:10.5000000

Expressions are limited to one language statement in length and calling only synchronous mode script functions. Avoid using script functions with side effects in expressions because subtle behaviors can occur.

Trigger Type	Description
Periodic	When the object containing the script is going On Scan, a Periodic script evaluates its expression at the next scheduled scan period of the AppEngine. The script then runs periodically at the trigger interval specified in the <b>Trigger Period</b> box. A time interval of zero (0) starts the script during every scan. This trigger does not require an expression.
While True	When the object containing the script is going On Scan, a While True script evaluates its expression at the next scheduled scan period of the AppEngine. The script runs if true and then periodically thereafter at the trigger interval.  The script continues running as long as the <b>Expression</b> value evaluates to true. A Trigger Period is required. Zero (0) evaluates the expression at the AppEngine scan period and non-zero means the expression is evaluated at the specified time interval.
On True	When the object containing the script is going On Scan, an On True script evaluates its expression at the next scheduled scan period. The script starts at the transition between the expression going from false to true.
On False	When the object containing the script is going On Scan, an On False script evaluates its expression at the next scheduled scan period. The script starts at the transition between the expression going from true to false.
Data Change	Scripts run when the value or quality of the expression changes. The expression must evaluate to a single, non-arrayed value of the following types: integer, real, time, elapsedtime, string, double, Boolean, custom enumeration and quality. To allow execution based on quality, select the <b>Quality changes</b> check box.  A deadband can be specified for all data types. Deadband units for time and elapsedtime types are milliseconds. Deadband is always ignored for strings because any change (even from "ABC" to "abc") is considered a change. Only major changes in quality (from Good/Uncertain to Bad/Initializing or vice versa) are considered changes.  After the object is put on scan, Data Change-triggered scripts start running at the AppEngine's next scan period and then at each subsequent scan period in which the value or quality changes.
While False	When the object containing the script is going On Scan, a While False script evaluates its expression at the next scheduled scan period, runs if false, and then periodically thereafter at the trigger interval.  The script runs as long as the <b>Expression</b> value evaluates to be false. A Trigger Period is required. Zero (0) evaluates the expression at the AppEngine scan period and non-zero means the expression is evaluated at the specified time interval.

4. Select one or more of the following:

- Set the **Runs Asynchronously** and associated **Timeout Limit** parameters, as needed. The Timeout Limit is set in milliseconds. Note that if a script exceeds the Timeout Limit, it continues to execute all sequential statements, but generates a warning in the logger when it completes. If the script contains a loop, it will exit the loop and complete execution of the script, when the timeout limit is reached.
  - Select **Report Alarm on Execution Error** and set a **Priority** for the alarm if you want the alarming function to alert you if a script execution failure occurs.
  - Select **Historize Script State** to store the state of the script in your application's Historian.
5. In the **Declarations** area, type variable declarations about the script you are writing.
  6. Set aliases for the reference strings in the **Aliases** area. This can simplify the script code and allows script code to be created and locked at a template level using alias names. When an individual instance of that template is created, you can link external attributes to the alias names.  
  
In the **Aliases** area, click the **Add** button to add a new alias. An alias is added to the list. The name is shown in edit mode. Double-click the **Reference** entry, and enter a reference string for the alias. You can also click the **Browse** button at the end of the **Reference** block to open the Attribute Browser for easy selection of an object's attributes.
  7. Write the script in the **Script Creation** box. Use the **Display Script Function Browser** and **Display Attribute Browser** buttons to help you insert script functions and object attribute references in your script. For help with the specific commands and syntax, see "QuickScript .NET Functions" in the *Application Server Scripting Guide*.  
  
Click the **Validate Script** button to validate if your script contains any syntax errors.
  8. Order the scripts. If you have more than one script associated with a single object, click **Configure Execution Order**. Ordering does not apply to asynchronous scripts. If a script is added to an instance derived from a template that contains scripts, the new script automatically defaults to running after the derived scripts.
  9. When you are done creating your script and setting its execution triggering parameters, save and close the Object Editor.

## Script Locking and Change Propagation

When you lock a script in a template, the following rules apply:

- The name of a script and its existence is implicitly always locked. This means:
  - You cannot delete the script in derived objects.
  - You cannot change the name of the script in derived objects.
  - If you rename the script in the template, the name changes in all derived objects.
  - If you delete a script from the template, the script is deleted from all derived objects.
  - If you add a script to the template, the script is added to all derived objects.
  - You can add scripts to derived objects. Adding scripts to derived objects does not affect parent object scripts.
- You can lock or unlock the script text in a template. There is script text for **Declarations, Execute, Startup, Shutdown, On Scan** and **Off Scan**.
- You cannot separately lock a script in the script editor. A single group lock is used to lock or unlock all scripts at once. Once a script is locked, derived templates and instances cannot modify any of the script text.
- You can lock alias names and alias references separately.

- When you create aliases for a script, an AliasReference attribute is created for each script with aliases. To lock alias references, you must lock the AliasReference attribute.
- Changes to alias names only propagate to derived objects when aliases are locked in the **Scripts** page.
- Changes to alias references only propagate to derived objects when the AliasReference attribute is locked in the **Attributes** page.
- If the AliasReference attribute is not locked, the alias references will be editable in derived objects, but changes made in the parent template will not propagate to derived objects.
- The script description, runs asynchronous flag, expression, trigger type, trigger period, deadband and execution error alarm are individually lockable and can be locked separately from the script text. A group lock is provided for this group of attributes.
- When you add a script to a template, all properties of the script are editable.
- When you add a script to an instance, all properties of the script are editable, except the lock properties (locks do not serve any purpose in instance since you cannot derive child objects from them).

---

**Important:** An expression typically uses attribute references. If you lock the expression and the associated script in a template, use aliases in both the expression and the script. This allows you to specify the attributes that the aliases point to on a per-instance basis while the script code is locked.

---

The following rules apply to the derivation behavior of locked script attributes:

- If an attribute is locked in a template, then all templates and instances derived from the template share the copy of the value of the locked attribute. A change to the value is only allowed in the template that locked it. The change propagates to all derived templates and instances.  
  
For scripts, locking an attribute of the script, such as its script text or execution type, in a template means all derived templates and instances point to that locked attribute. Future changes to that locked attribute's value, such as modifying the script text, propagate and appear in all derived templates and instances.  
  
If instances are deployed, they are marked pending update status. After they are redeployed, the change to the locked attribute in the template exists in the deployed instance.
- If an attribute is not locked in a template, then all templates and instances derived from that template receive their own copy of the value of that unlocked attribute. A change to that unlocked value is allowed in derived templates and instances because they own their own copy. Any change to the unlocked attribute value in the template does not propagate to any derived template or instance.  
  
An unlocked attribute in a script (such as expression or script order) in a template means that all derived templates and instances have their own copy and the value of that unlocked attribute can change. Future changes to that locked attribute's value (for example, modified expression) in the template does not propagate to any derived template or instance. If instances are deployed, their status does not change to pending update. Redeploying them does not cause the value to change in the deployed instance.

## Creating and Working with Content

The **Content** pane of the **Attributes** page lets you:

- Add, modify, rename or delete object-owned symbols. Object-owned means that the symbol exists only for that particular object. Note that object-owned symbols are propagated to child objects.
- Link the object to content (symbols, layouts, External Content) in the Graphic Toolbox. This is shared content, that can be linked to multiple objects. Links to content are propagated to child objects.

- View content inherited from parent templates. However, you cannot add, modify, or delete inherited content. To edit an inherited content item, you must check out and edit the derived template that owns the content.

You must have the derived template or object instance checked out in order to add, modify, or delete content associated with the object; otherwise, the content is read-only and you can only view it.

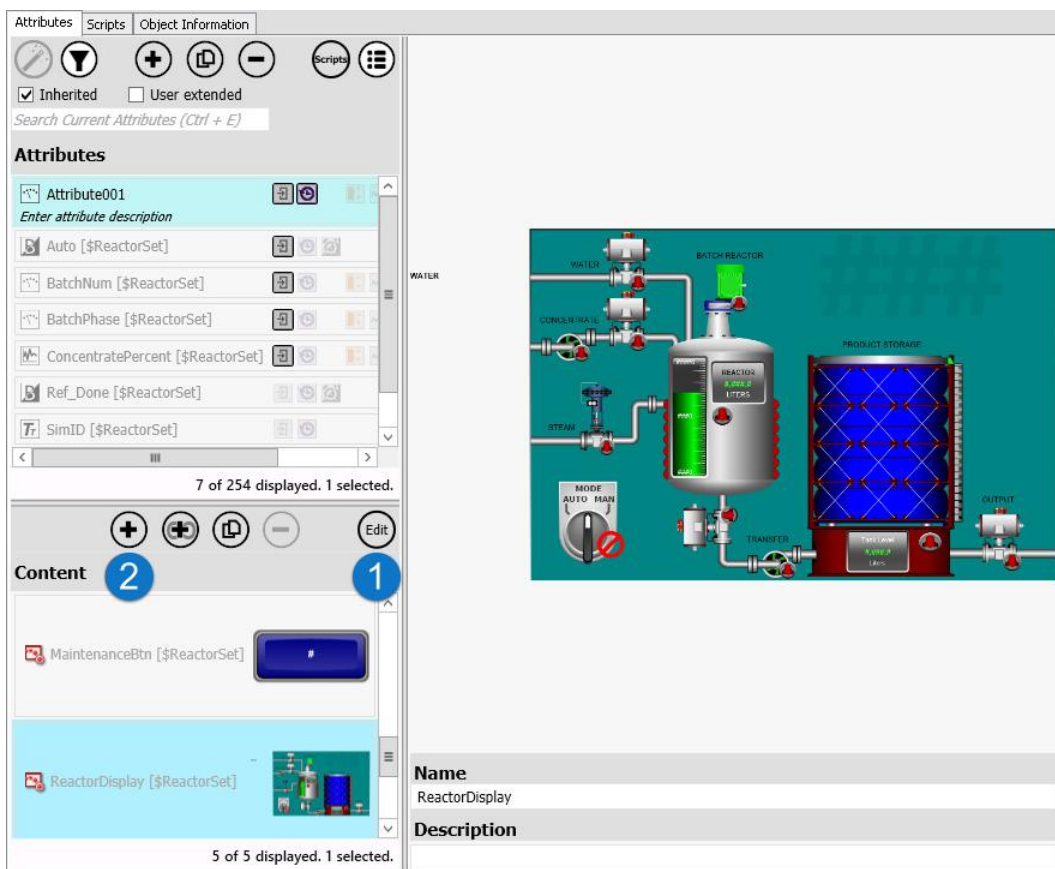
You can only add content to derived templates or instances, not to a base template.

**To add or link to content:**

Linked content can be a symbol, an external content item, or a layout.

- To **link** to a content item in the Graphic Toolbox to the object, select the **Link Content** button. This opens the **Galaxy Browser**. Then, select the symbol, layout, or External Content item to be linked to the object. Linked content can be shared by multiple objects. Relative references within the linked content will resolve correctly.
- To create an **object-owned symbol**, select the **Add Content** button (2).
- To edit an owned-symbol, select the symbol and click the **Edit** (1) button.

**Note:** If you edit a linked symbol, layout or other shared content, the changes will appear in all objects that link to that content.

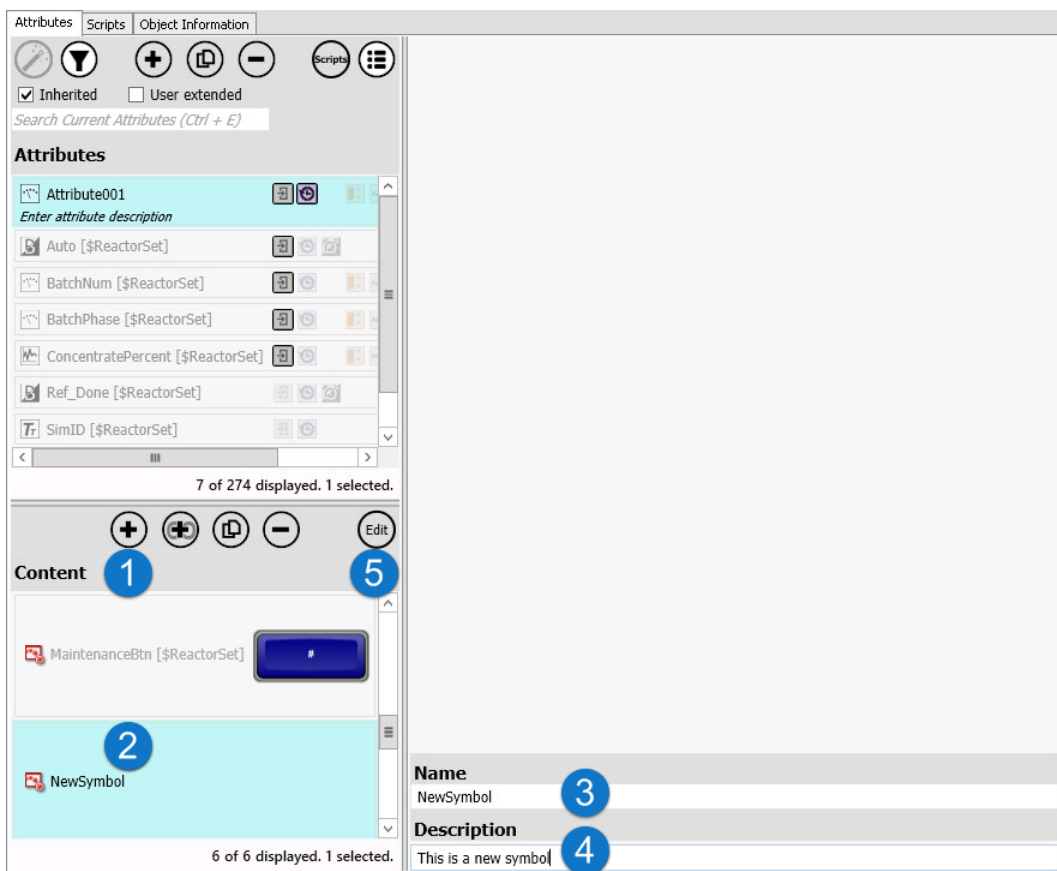


## Adding Graphics Items

### To add a symbol to the object

This procedure adds an owned symbol to the object. To link a symbol or external content item, see *Link to Shared Content in the Graphic Toolbox* on page 236.

1. From the **Content** pane in the **Attributes** page of the Object Editor, click the **Add (+)** button. A new symbol name and symbol icon are added to the pane.
2. Select the new symbol in the **Symbols** pane.
3. In the **Details** pane, edit the new local symbol name.
4. In the **Description** box, type a description for the graphic symbol being added.
5. Click **Edit**. The Graphic Editor opens. For instructions on using the Graphic Editor, see "About Creating and Managing Industrial Graphics" in the *Creating and Managing Industrial Graphics User Guide*.



## Errors and Warnings in an Owned Graphic

When you save an owned symbol (graphic) that you have edited in the **Graphics Editor**, the owning object is also saved. If there are any errors and warnings, they will be shown in the **Save Confirmation** dialog. However, the errors and warnings include not just the symbol you are saving, but because the owning object is also being saved, the dialog lists all errors and warning related to the owning object and other symbols owned by the object.

To determine if the warning or error is related to the symbol you are saving, or if it is related to the owning object (or other symbol owned by the object), check the prefix in the warning/error message. The first (left-most) entry in each error/warning message lists the symbol or object name.

## Modifying Graphics

All modifications made to graphic symbols referenced by other objects are visible in the referenced objects.

### To modify a graphic symbol

1. On the **Graphics** page of the Object Editor, click the **Name** of the graphic symbol to be modified.
2. Click **Open**. The graphics tool box opens, showing the selected graphic symbol.
3. Make changes. See "About Creating and Managing Industrial Graphics" in the *Creating and Managing Industrial Graphics User Guide*.

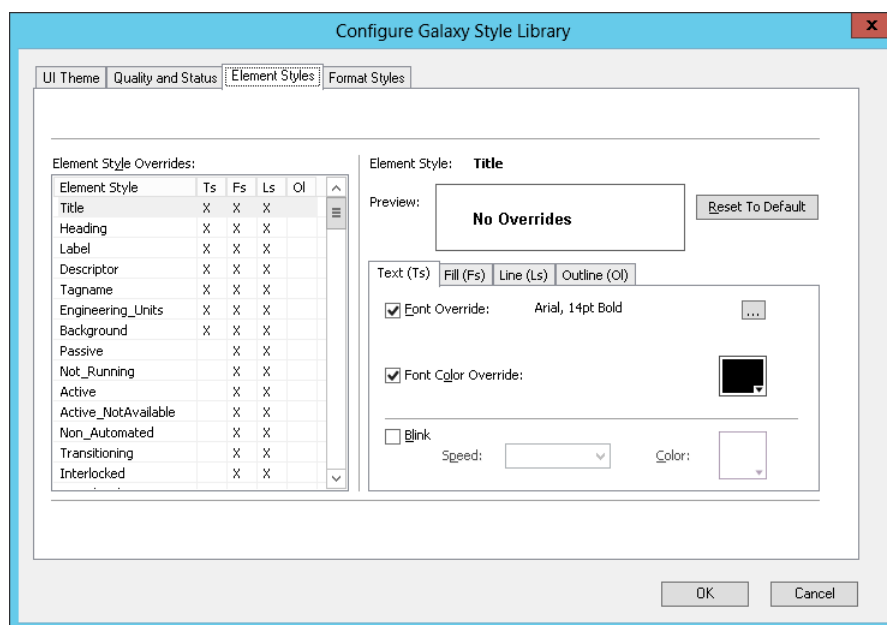
## Set Graphics Properties with a Galaxy Style Library

To view and manage the settings for the Galaxy Style Library, select **Configure** on the the **Galaxy** menu. A Galaxy Style Library lets you set the appearance and behaviors of different graphics properties within a ViewApp at run time.

- *Use Element Styles to Modify Graphics* on page 168 define graphics properties such as font size, font color, fill color, line weight, and line color.
- *Use Quality and Status Styles to Provide Run-Time Feedback* on page 169 provide visual cues about different quality and status states.
- *Use UI Themes to Set Appearance of WPF Controls* on page 169 define graphics properties for WPF and navigation controls used in ViewApps.

## Use Element Styles to Modify Graphics

Element Styles define one or more visual properties such as fill, line, text, blink, outline, and status properties of graphic elements. You can apply an Element Style to a graphic element to set preconfigured visual properties defined in that Element Style. Element Styles establish consistent visual standards for symbols.

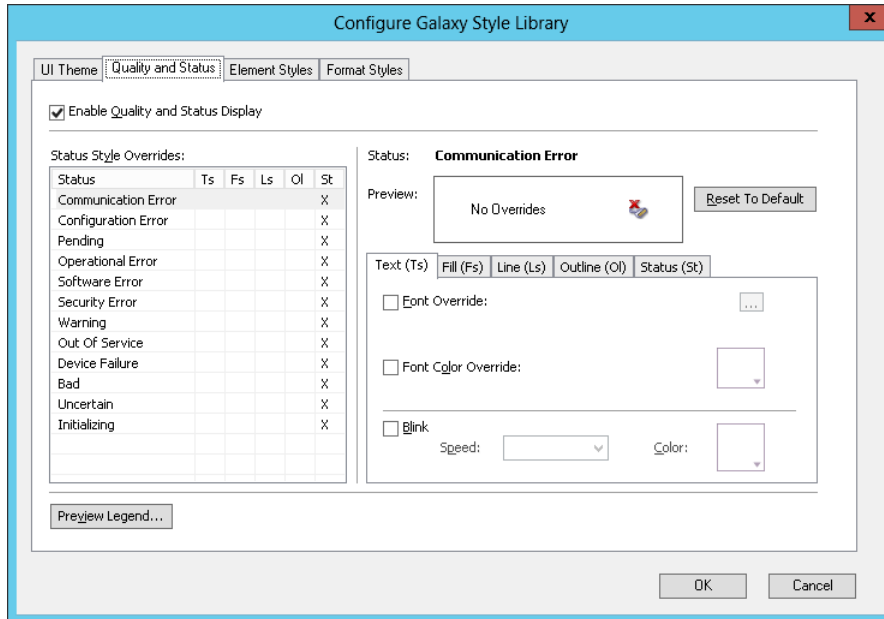


For more information about Element Styles, see *Understanding Element Styles* on page 188.



## Use Quality and Status Styles to Provide Run-Time Feedback

In the same way that Element Styles can be used to preconfigure consistent visual properties of graphic elements, Quality and Status styles can be preconfigured to provide visual elements specific to quality and status states at run time.



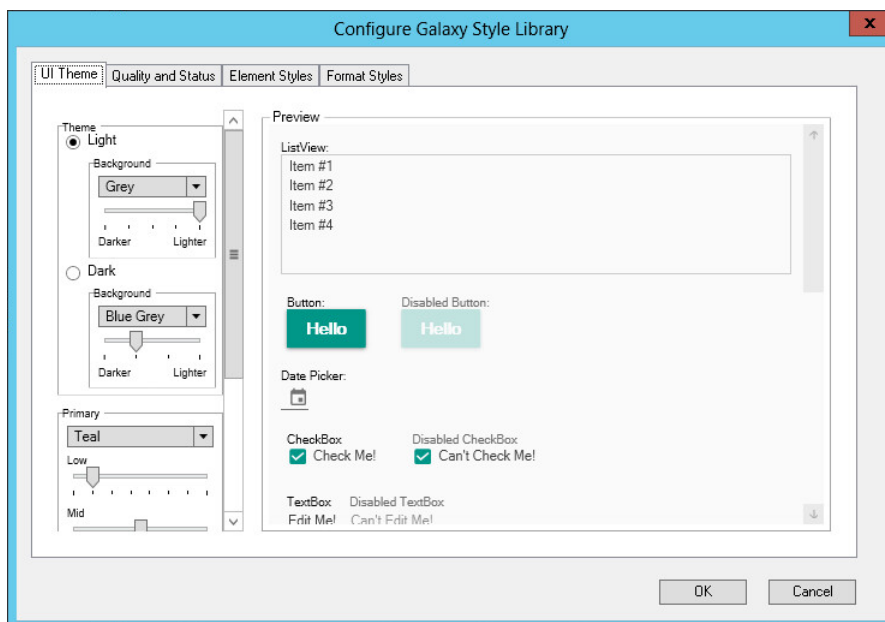
You can configure text, fill, line, outline, blink, and associated display icons for the following Status styles:

- Communication Error
- Configuration Error
- Pending
- Operational Error
- Software Error
- Security Error
- Warning
- Out of Service
- Device Failure
- Bad
- Uncertain
- Installing

## Use UI Themes to Set Appearance of WPF Controls

UI Themes define the color settings that are applied to WPF controls used in ViewApps. In order to set the colors for a WPF control, the control must comply with the Material Design Spec. See the MSDN WPF Community Projects website for additional information:

<https://blogs.msdn.microsoft.com/wpf/2015/10/29/wpf-community-projects/>



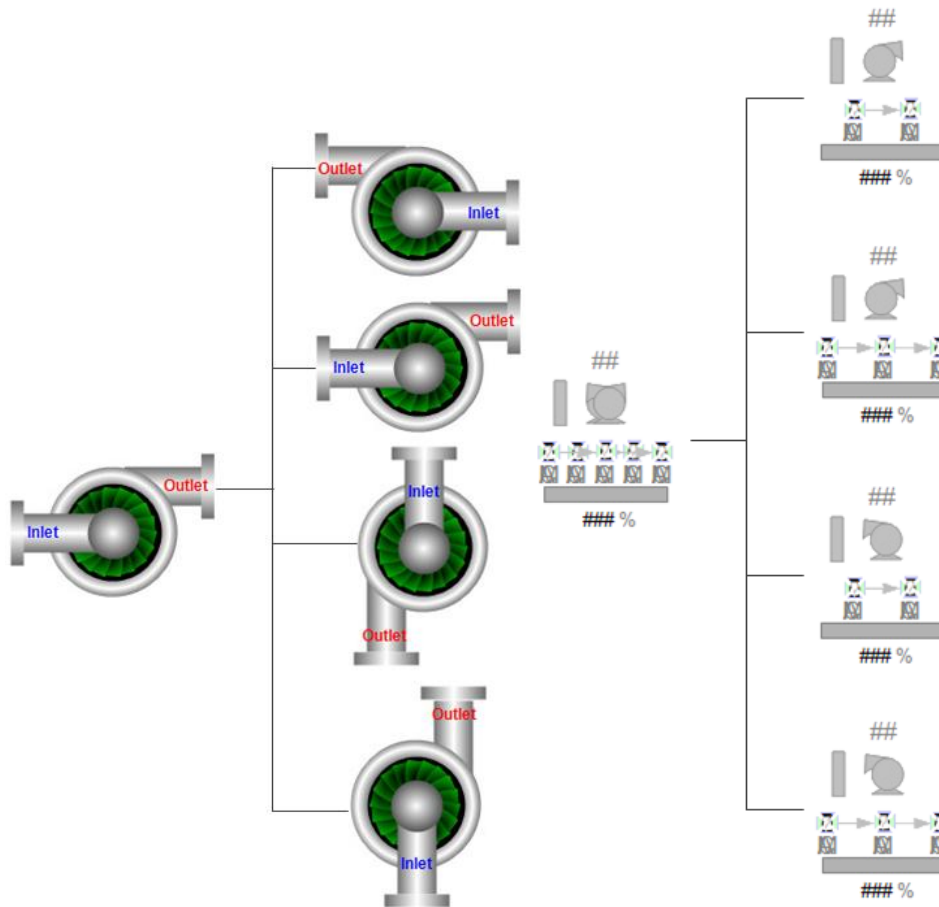
The UI Theme includes the following configurable parameters:

- **Theme** (light or dark): default – **light**
- **Background** color
  - Default **light** background color: grey, lightest setting
  - Default **dark** background color: blue-grey, second darkest setting
- **Primary** color – default **teal**
  - **Low**. This setting is only used by WPF controls that explicitly include the Primary Low property. The **low** setting is always darker than the **mid** setting. Default is the second darkest setting (second from left).
  - **Mid**. This setting is used by all WPF controls that use the Material Design Spec. Default is the middle setting.
  - **High**. This setting is only used by WPF controls that explicitly include the Primary High property. The **high** setting is always lighter than the **mid** setting. Default is the second lightest setting (second from right)
- **Accent** color – default **purple** (darkest setting)

## Using Symbol Wizards

The Industrial Graphic Editor includes the Symbol Wizard Editor, which can be used to create symbols containing multiple visual or functional configurations called Symbol Wizards. A Symbol Wizard can be embedded into managed InTouch applications like standard Industrial Graphics.

A Symbol Wizard’s configuration is selected to meet the requirements of an application. Incorporating multiple configurations in a single Symbol Wizard reduces the number of symbols needed to develop an AVEVA application.



The Symbol Wizard Editor can create Symbol Wizards from traditional Industrial Graphics and Situational Awareness Library symbols. Both types of symbols are located in the Graphic Toolbox in separate toolkit folders. Symbol Wizards are saved in the IDE's Graphic Toolbox and are not associated with any specific AVEVA object template or object instance. Except for the ability to select a specific symbol configuration, Symbol Wizards behave like standard Industrial Graphics.

Situational Awareness Library symbols provide an additional benefit of including defined properties and their associated attributes to more easily create configurations. Some Situational Awareness Library symbols include a Type property to assign a specific function for a symbol configuration. For example, a meter symbol can be configured to represent a thermometer, a pressure meter, or flow meter by simply changing the attribute assigned to the Type property.

Typically, the process of creating and embedding a Symbol Wizard in an application requires the involvement of a Designer and a Consumer. A Designer creates Symbol Wizards using the Symbol Wizard Editor. A Consumer selects a configuration of a Symbol Wizard and embeds the instance of the symbol into managed InTouch applications.

## Creating Symbol Wizards

A Designer uses the Symbol Wizard Editor to define the various required symbol configurations based on a set of rules and symbol layers. A Designer defines a set of layers, which are used to group a set of graphic elements, custom properties, and named scripts. Graphic elements and other symbol properties can be assigned to no layers or multiple layers. Graphic elements that are not assigned to any layer always appear in all symbol configurations.

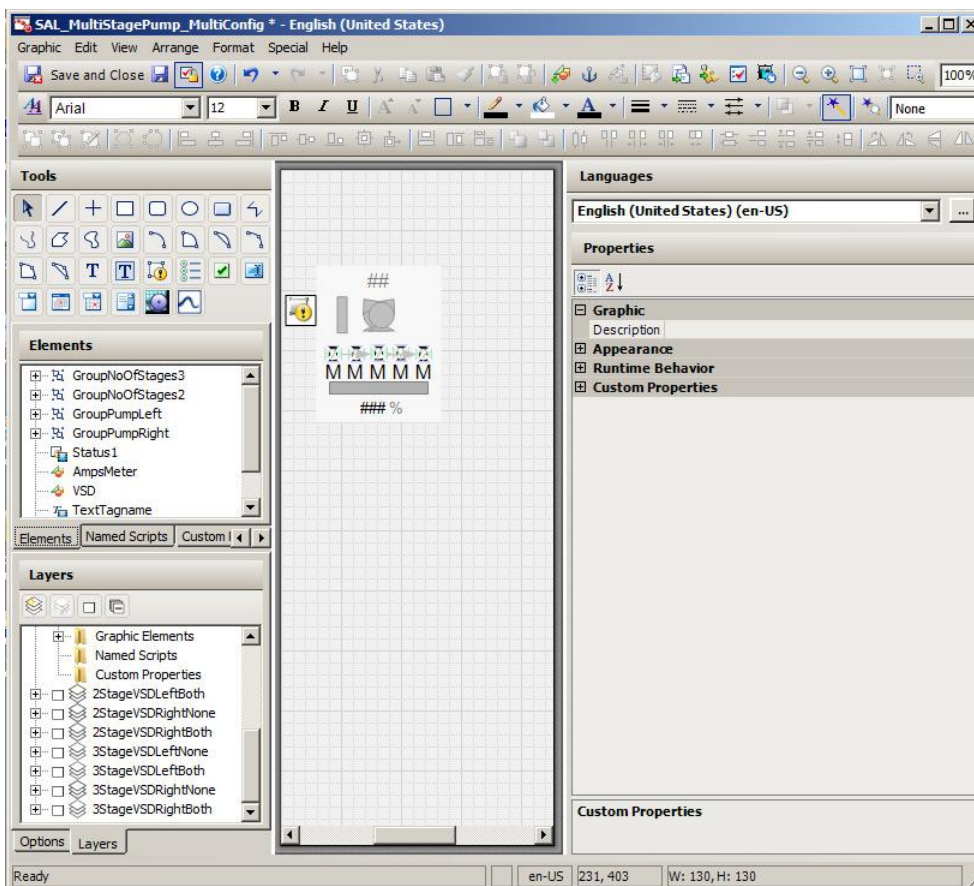
A Designer can create a rule for each layer that defines the conditions when the layer is included in a symbol configuration. Rules are assigned with Choice Groups, Choices, and Options. A Designer selects a configuration to be the symbol default that appears when the symbol is embedded in a managed InTouch application.

After creating all symbol configurations, the Designer verifies how each configuration of a symbol using the Symbol Wizard Preview. Designers set values in the **Wizard Options** view to verify that each configuration appears as designed based on the layer rules set for the symbol.

When a Symbol Wizard is finished, the Designer saves it to the Galaxy library so that it is available for use in managed InTouch applications. For more information about creating Symbol Wizards, see "Working with Symbol Wizards" in the *Creating and Managing Industrial Graphics User Guide*.

## Embedding Symbol Wizards into an Application

Symbol Wizards are stored in a Galaxy library just like standard Industrial Graphics. When a Consumer selects a Symbol Wizard and embeds it into a managed InTouch application, the Symbol Wizard's default configuration is selected. The Consumer can change the symbol's configuration by changing the options from the Symbol Wizard's **Wizard Options** section of the **Properties** view. Depending on the selected configuration, there can be additional configuration-related properties that can be selected by the consumer.



After selecting a symbol configuration and changing any properties, the Consumer saves the Symbol Wizard so that it can be embedded into a window from WindowMaker.

While the InTouch application is running, the Symbol Wizard appears as the configuration selected by the Consumer. A Symbol Wizard configuration cannot be changed during run-time.

For more information about how to embed Symbol Wizards into a managed InTouch applications, see "Working with Symbol Wizards" in the *Creating and Managing Industrial Graphics User Guide*.

## Renaming Graphics

You can rename a graphic symbol.

### To rename a graphic symbol

1. On the **Graphics** page of the Object Editor, click the **Name** of the graphic symbol to be renamed.
2. Type the new name, and press **Enter**. The new name is saved.

## Deleting Graphics

---

**Caution:** Deleting a graphic symbol with embedded references breaks the links to their related objects. "Symbol Not Found" appears when you open objects whose embedded graphic symbols have been deleted.

---

### To delete a graphic symbol

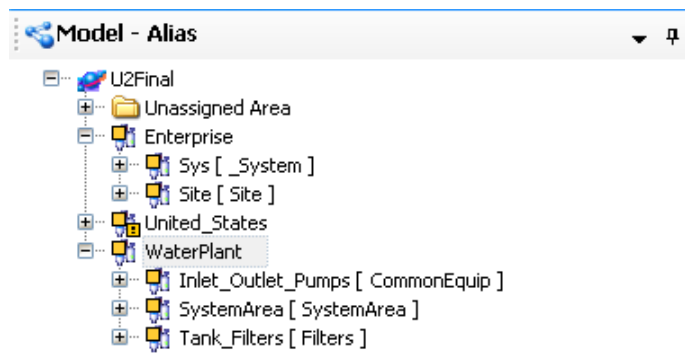
1. On the **Graphics** page of the Object Editor, click the **Name** of the graphic symbol to be deleted.
2. Click the **Delete** button. The **Delete** dialog box appears. The left pane lists the graphic symbol selected for deletion. The right pane shows all embedded references to the selected graphic.
3. Click **Yes**.

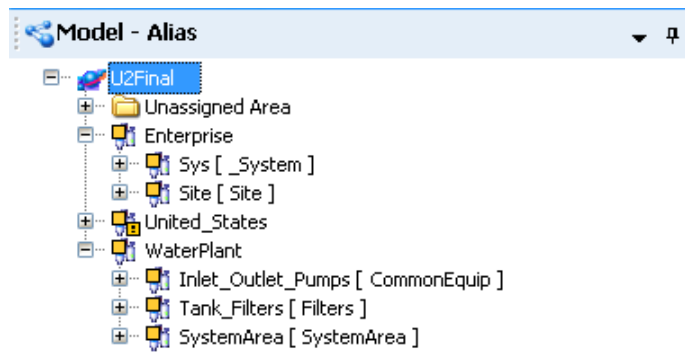
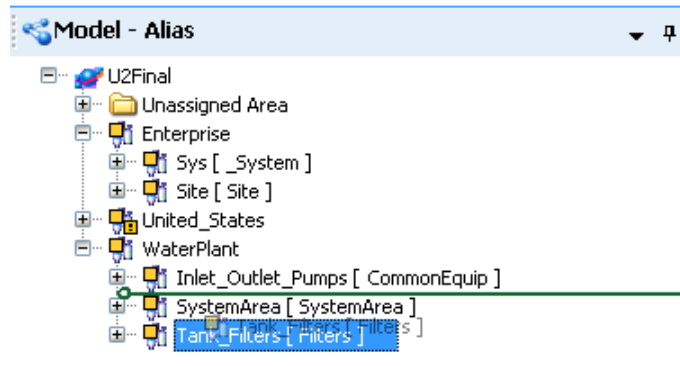
## Rearrange Objects to Improve AVEVA OMI ViewApp Navigation

By default, a running ViewApp shows objects as navigation items that appear in alphabetic order by their assigned Galaxy names in the navigation model. An alphabetic order of navigation items may not be appropriate for operators. Objects can be rearranged to more closely resemble the functional organization of different components in a production process.

Objects can be rearranged from the **Model** view of the System Platform IDE. After objects are moved, the **Assets** listings of objects in the ViewApp and Layout editors update to show the new arrangement.

The following figures show the sequence of drag and drop steps to rearrange derived objects from the **Model** view of the System Platform IDE. In this example, the Tank\_Filters area is moved from its alphabetic location in the hierarchy to a new location immediately beneath the Inlet\_Outlet\_Pumps area. As an object is moved to a different position, the indentation of a horizontal line indicates the level within the hierarchy the object will be placed.





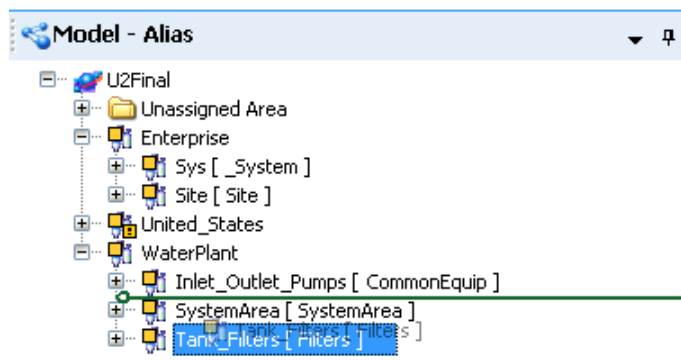
## Rearrange Objects in the Model View

You drag and drop an object with your mouse to move an object to a different location within the ViewApp navigation model. You can rearrange objects from the **Model** view of the System Platform IDE.

### To rearrange objects from the System Platform IDE

1. Open the System Platform IDE and show the **Model** view.
2. Make sure that all objects shown in the **Model** view are undeployed.
3. Select the object to be moved within the **Model** view.
4. Press and hold the left mouse key and move the object to a different location within the hierarchy of objects in the **Model** view.

A horizontal line shows the hierarchical level within the navigation model that the object will be placed.



The horizontal line does not appear when you attempt to make an invalid move.

5. Release the mouse key to drop the selected object.

## More Information About Rearranging Objects in the Model View

Rearranged objects maintain their location within a list of objects for the following Galaxy operations:

- Export and import of a Galaxy
- Back up and restore a Galaxy
- GRLoad and GRDump operations

Also, objects can be programmatically moved to another location within a list of objects using the GRAccess Toolkit.

## Rearranged Objects in the Export and Import of a Galaxy

When rearranged objects are exported to an aaPKG file, object rearranged objects are retained when the aaPKG file is imported into another Galaxy. If the imported Galaxy includes rearranged objects, but the existing Galaxy does not, the object order of the imported Galaxy takes precedence.

## Rearranged Objects in Backup and Restore of a Galaxy

When rearranged objects are backed up to a CAB file, rearranged objects are retained when the CAB file is restored to another Galaxy.

## Rearranged Objects in a GRLoad and GRDump

When one or more objects are selected and exported through GRDump, then GRLoad can be used to import the contents of GRDump into another Galaxy. Objects retain their rearranged locations in the GRLoad import operation in the new Galaxy.

## Rearrange an Object Programmatically using GRAccess Toolkit

You can use the GRAccess object model to write programs that automate the configuration of local and remote Galaxies. You can move an object in the Model view hierarchy using the SortOrder property that programmatically moves an object after a specified predecessor object in the list of objects shown in the **Model** view.

### Class

IgObject

### Syntax

[C#]

```
string SortOrder { set; get; };
```

[Visual C++]

[propget]

```
HRESULT SortOrder(  
[out, retval] BSTR* thePredecessorName  
);
```

[propput]

```
HRESULT SortOrder(  
[in] BSTR newPredecessorName  
);
```

### Parameters

*thePredecessorName*

The current predecessor of this object.

*newPredecessorName*

The new predecessor of this object. The object does not need to be checked out.

### Remarks

SortOrder sets or returns the preceding object name in the navigation hierarchy. When the sort order is set, the tagname of the asset is displayed in the ViewApp navigation hierarchy just below the predecessor.

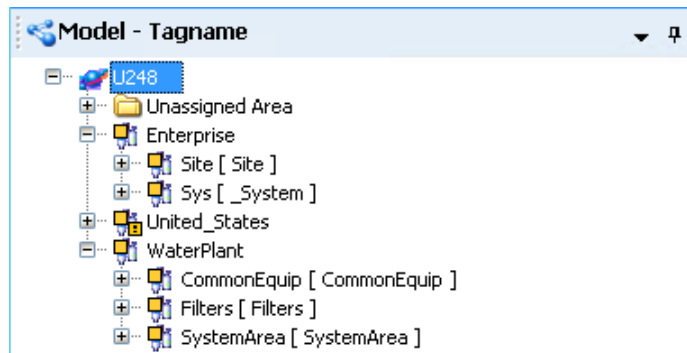
## Assign an Alias Name to an Object

A running ViewApp can show objects as navigation items that appear by their assigned Galaxy names in the navigation model. Galaxy names may not be intuitive and easily recognizable to operators. Objects can be assigned more understandable alias names.

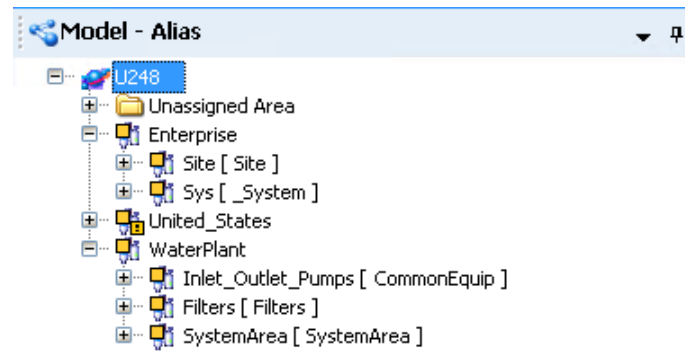
An alias name is just a string and not associated with any object attribute. An alias does not provide another way to address items in the navigation model of a ViewApp. Scripts and animations must still reference an object based on the assigned Galaxy tagnames.

Alias names are assigned from the **Model** view of the System Platform IDE. You can toggle the **Model** view to show the objects by their tagname or alias names. In the example below, the CommonEquip area object has been assigned a more specific Inlet\_Outlet\_Pumps alias name.

### Objects by Tagname



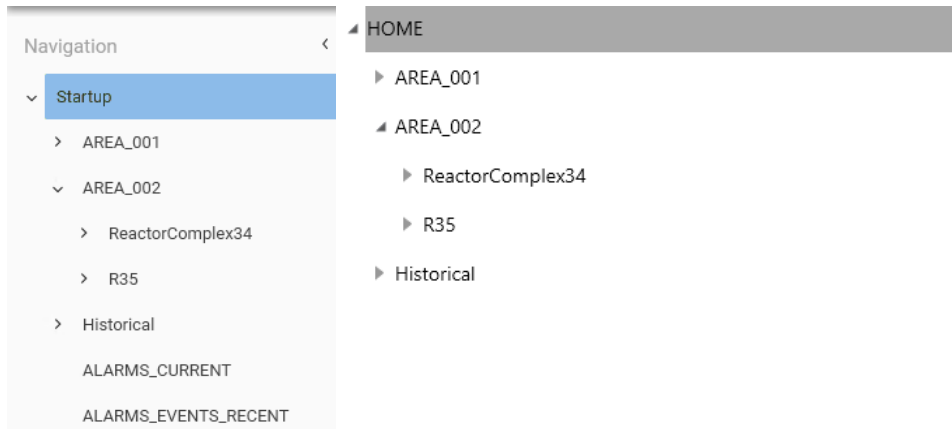
### Objects by Alias Names



Alias names can be shown from the ViewApp and Layout editors but cannot be modified. Also, if an alias was assigned to an object and the ViewApp or Layout editors were open, the editors must be closed and reopened to show the new alias name.

An alias name appears in the navigation model of the ViewApp during run time or in preview mode.





Alias names are associated with navigation attributes that have the word Title in their names. For example, the CurrentTitle attribute returns the alias name of the current navigation item in a running ViewApp. You can write scripts using alias names to change the current focus of a ViewApp. For more information about navigation attributes that reference alias names, see [11Reference\\_Attributes\\_Navigation1](#).

## Steps to Assign or Remove an Alias Name

You can assign an alias name to a derived instance of an object that appears in the **Model** view of the System Platform IDE. The System Platform IDE includes a **Show Alias** option that enables you to toggle the **Model** view to show the original tagname and alias names of objects.

An alias name must adhere to the following rules:

- An alias name can be up to a maximum of 128 characters in length.
- An alias name can consist of alphanumeric characters and special keyboard characters ( !, @, #, \$, %, ^, &, \*, (, ), -, \_, =, +).
- An alias name can start with any alphanumeric or special character.
- An alias name can contain blank spaces.
- Multiple objects can be assigned the same alias name in a Galaxy.

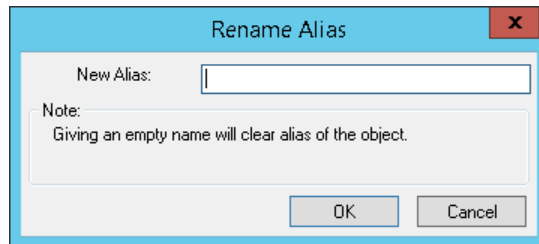
### To assign an alias name to an object


1. Open the **Model** view of the System Platform IDE to show the list of derived instances.
2. Ensure the derived instance you want to assign an alias name is undeployed.
3. Select the object instance from the **Model** view.
4. Right-click on the object to show the shortcut menu and select the **Rename Alias** option.

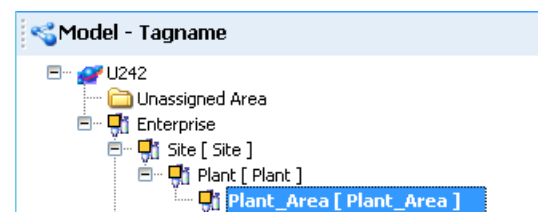
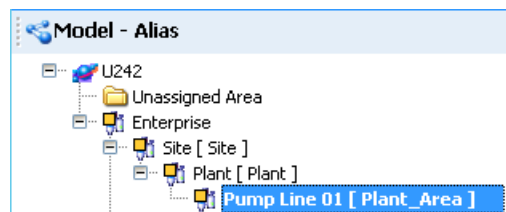
You can also assign an alias name by other methods:

- **Keyboard shortcut:** Ctrl + Shift + F2
- **Menu bar option:** Select **Edit** from the menu bar and select the **Rename Alias** option.

The **Rename Alias** dialog box appears with a field to assign a new Alias name.



5. Enter an alias name that meets the format rules and select **OK**.  
The object should appear in the **Model** view with its assigned alias name.
6. Verify the alias name has been assigned to the object.
  - a. Right-click on the object with an assigned alias name and select the **Show Alias** option from the shortcut menu.  
You can also verify an alias name by other methods:
    - **Keyboard shortcut:** Ctrl + Shift + F3
    - **Menu bar option:** Select the **AliasTag name** button  on the menu bar.
  - b. Toggle the **Show Alias** option to show the alias name and the tagname of the object. The title bar of the **Model** view indicates whether you are looking at the alias or tagnames of the objects.



## More Information About Alias Names

Objects retain their aliases when objects are exported, backed up, and loaded as part of a Galaxy operation. Also, object aliases can be programmatically set and returned using the GRAccess Toolkit.

## Alias Retention in Galaxy Operations

Alias names assigned to Galaxy objects are retained for the following Galaxy operations:

- Export and import of a Galaxy
- Back up, restore and migrate Galaxy
- Migrate a Galaxy
- GRLoad and GRDump operations

## Alias Names in the Export and Import of a Galaxy

When objects assigned alias names are exported to an aaPKG file, object Alias names are retained when the aaPKG file is imported into another Galaxy with the following exception:

AliasName is a reserved keyword in System Platform 2020 R2 objects. If an import package contains previous versions of objects with name "AliasName", the import process skips the attribute and logs a warning message in the SMC log. Attributes with name AliasName are not imported.

## Alias Names in Backup and Restore of a Galaxy

When objects assigned alias names are backed up to a CAB file, object alias names are retained when the CAB file is restored to another Galaxy.

## Migrate a Galaxy

When objects assigned alias names are migrated, object Alias names are retained when migrated to another Galaxy with the following exception:

AliasName is a reserved keyword in System Platform 2020 R2 objects. If a migration package contains previous versions of objects with name "AliasName", the migration process skips the attribute and logs a warning message in the SMC log. Attributes with name AliasName are not migrated.

## Alias Names in a GRLoad and GRDump

When one or more objects are selected and exported through GRDump, then GRLoad can be used to import the contents of GRDump into another Galaxy. Objects retain their alias names in the GRLoad import operation in the new Galaxy.

## Assign an Alias Name Programmatically using GRAccess Toolkit

You can use the GRAccess object model to write programs that automate the configuration of local and remote Galaxies. You can rename object aliases using the Alias property that programmatically gets or sets the alias name of an object.

### Class

IgObject

### Syntax

[C#]

```
string AliasName { set; get; };
```

[Visual C++]

[propget]

```
HRESULT Alias(  
[out, retval] BSTR* theAlias  
);
```

[propput]

```
HRESULT Alias(  
[in] BSTR newAlias  
);
```

### Parameters

*theAlias*

The current alias of this object.

*newAlias*

The new alias of this object, or blank if the object does not have an alias. The object does not need to be checked out. The alias rename operation is immediate.

# Working with the Industrial Graphic Editor

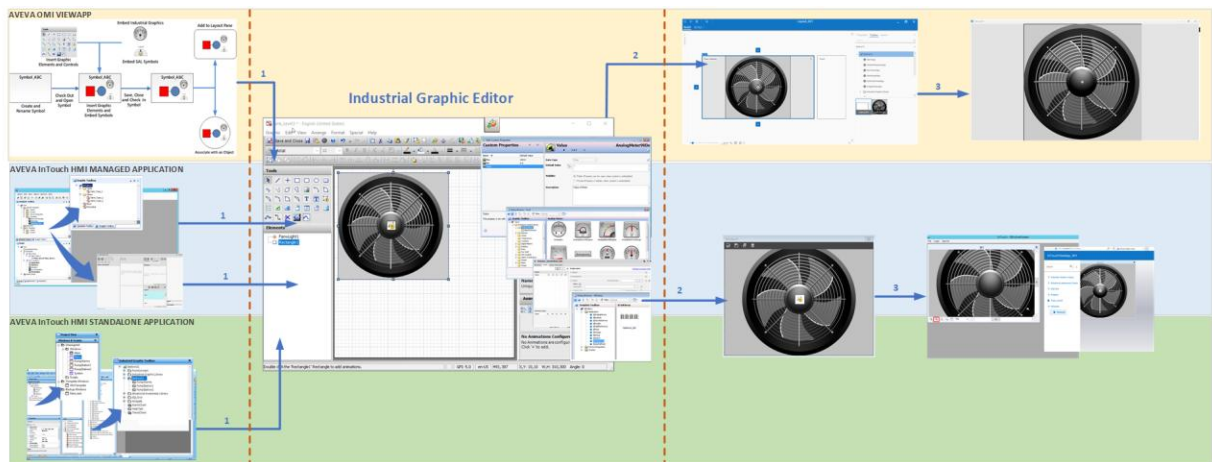
This section describes how to use Industrial Graphics and the Industrial Graphic Editor to build a graphical interface for monitoring and interacting with Application Objects at run time. It also describes some of the basic steps for using Industrial Graphics with the AVEVA Operations Management Interface(OMI). Topics covered in this section include:

- *Creating Symbols in the IDE on page 184*
- *Adding Industrial Graphics to AVEVA OMI Panes on page 199*
- *Working with Element Styles on page 188*
- *Working with the SignedWrite() Function for Secured and Verified Writes on page 212*
- *Working with the Show/Hide Content Script Functions on page 217*

For detailed information about Industrial Graphics and the Industrial Graphic Editor, see the *AVEVA Industrial Graphic User Guide* or *Creating and Managing Industrial Graphics*.

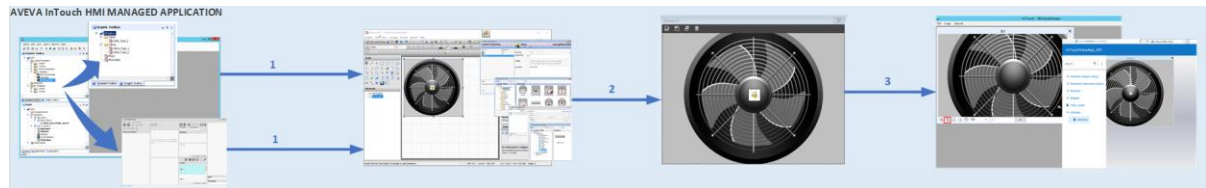
## Industrial Graphic Editor Workflows for Use with HMI/SCADA Applications

The Industrial Graphic Editor is the standard graphic editor used across various HMI/SCADA applications. It can be used to create graphics that can be embedded into HMI/SCADA applications to represent assets on the plant floor. The overall workflow is described in the following sections.

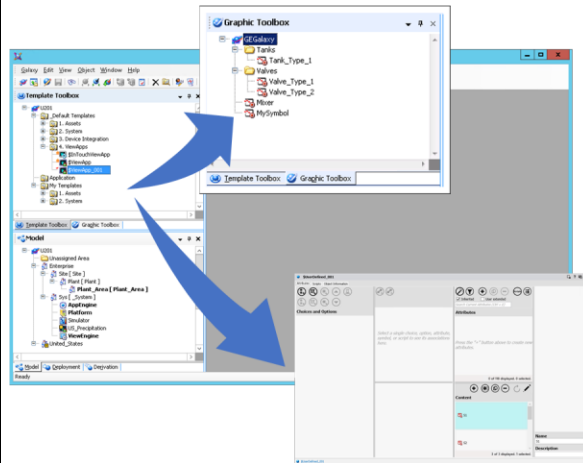


## InTouch Managed Applications

The workflow for InTouch HMI managed applications starts with the IDE.



### Step 1



### In the IDE:

From the Graphic Toolbox, create a new symbol, or select an existing graphic.

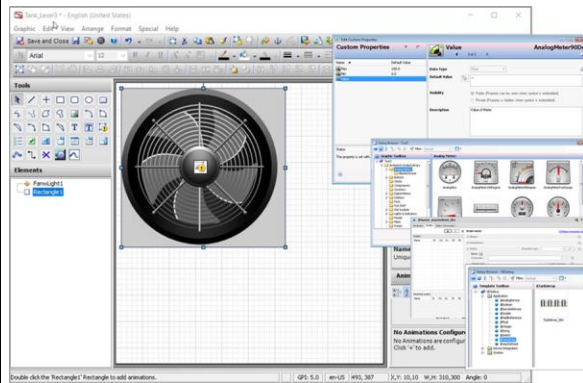
- Open the graphic to launch the Industrial Graphic Editor.

- or -

Open a \$UserDefined object in the Template Toolbox.

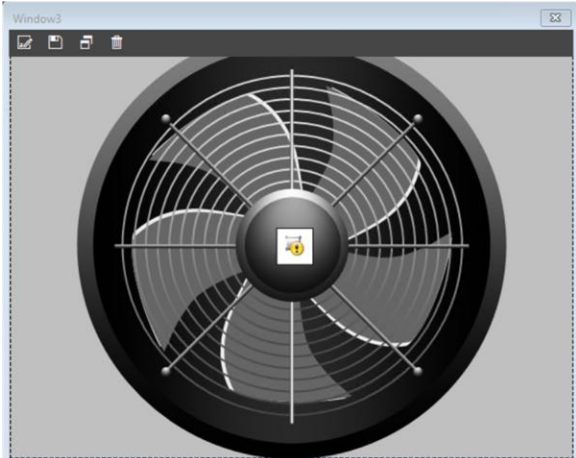
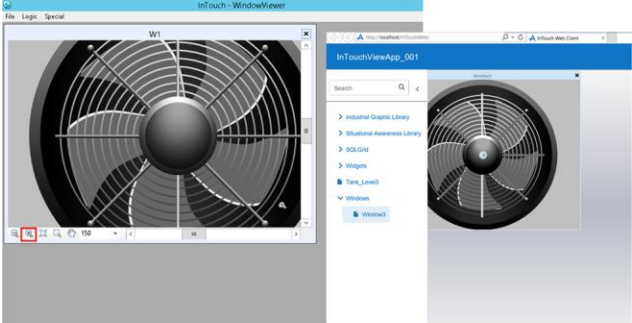
1. From the Content pane, click the **+** button to add a graphic to the object.
2. Click the **Edit Content** button to launch the Industrial Graphic Editor.

### Step 2



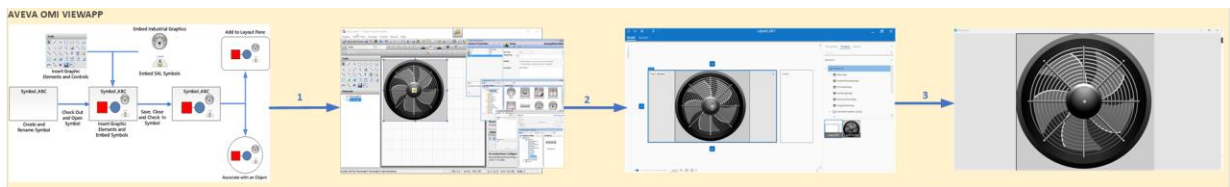
Edit the graphic in the Industrial Graphic Editor. Modify graphical elements as needed.

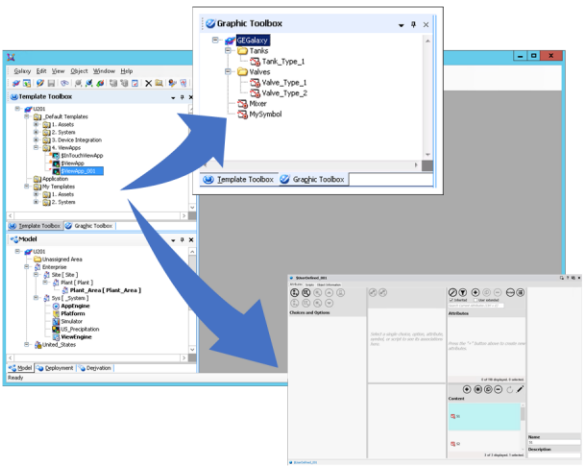
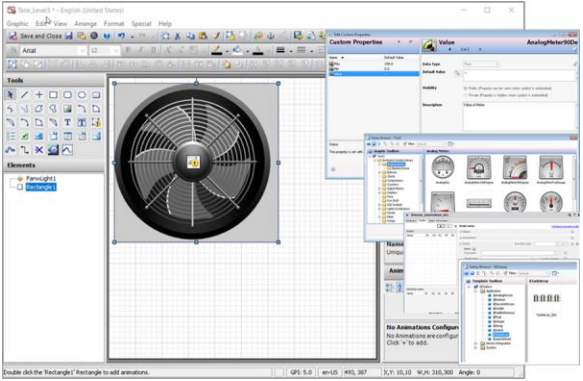
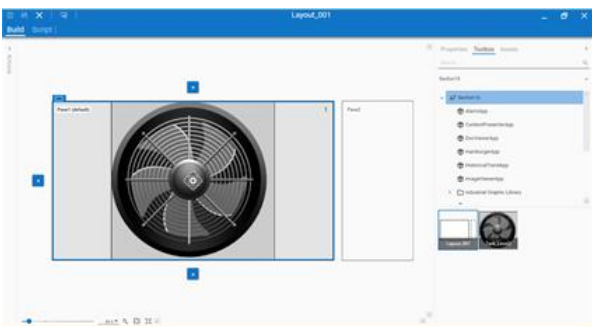
- Insert graphic elements and embed symbols.
- Add/modify scripts, references, custom properties, and animations to enable the graphic to represent the asset, and to respond to commands and data changes at run time.

<p><b>Step 3</b></p> 	<p>Embed the graphic in a Window.</p>
<p><b>Step 4</b></p> 	<p>To view the graphic in runtime, launch WindowViewer or the Web Client.</p>

## AVEVA OMI ViewApps

Use the Industrial Graphic Editor to create symbols that represent assets in an AVEVA ViewApp.



<p><b>Step 1</b></p> 	<p><b>In the IDE</b></p> <p>From the Graphic Toolbox, create a new symbol, or select an existing graphic.</p> <ul style="list-style-type: none"> <li>• Open the graphic to launch the Industrial Graphic Editor.</li> <li>- or -</li> </ul> <p>Open a \$UserDefined object in the Template Toolbox.</p> <ol style="list-style-type: none"> <li>1. From the Content pane, click the + button to add a graphic to the object.</li> <li>2. Click the <b>Edit Content</b> button to launch the Industrial Graphic Editor.</li> </ol>
<p><b>Step 2</b></p> 	<p>Edit the graphic in the Industrial Graphic Editor. Modify graphical elements as needed.</p> <ul style="list-style-type: none"> <li>• Insert graphic elements and embed symbols.</li> <li>• Add/modify scripts, references, custom properties, and animations to enable the graphic to represent the asset, and to respond to commands and data changes at run time.</li> </ul>
<p><b>Step 3</b></p> 	<p>Drag and drop the graphic onto a ViewApp layout pane in the Layout Editor.</p>

<p><b>Step 4</b></p> 	<p>To view the graphic in run time, deploy the ViewApp.</p>
--	---

## Creating Symbols in the IDE

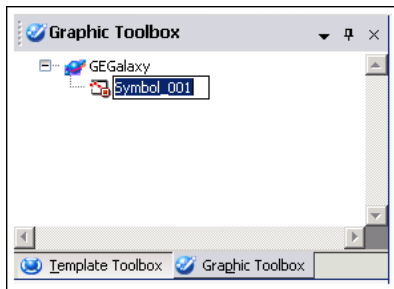
### Creating Symbols in the Graphic Toolbox

You can create a new symbol in the Graphic Toolbox. It is then listed in the Graphic Toolbox with a proposed default name. You can:

- Rename the symbol.
- Move the symbol.
- Edit the symbol with the Industrial Graphic Editor.

#### To create a new symbol from the IDE

1. On the **Galaxy** menu, point to **New**, and then click **Symbol**. The Graphic Toolbox appears and a new symbol is listed.




---

**Note:** You can also press CTRL + SHIFT + S to create a new Industrial graphic or right-click and then select **New** and **Symbol** from the shortcut menu.

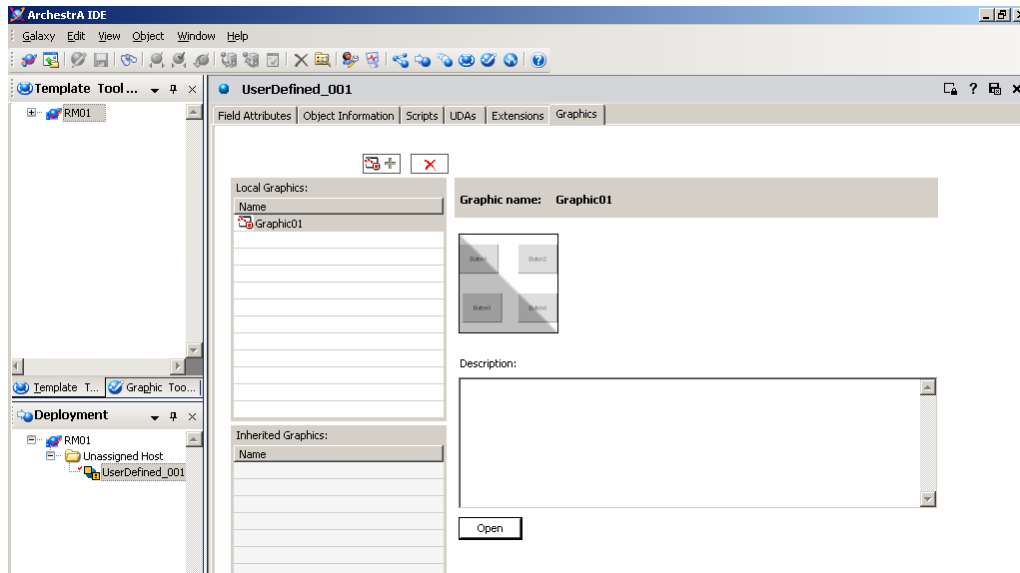
---

2. Rename the symbol.

Names must be unique within the entire Graphic Toolbox hierarchy. Valid characters for symbol names include alphanumeric characters, #, and \_ (underscore). Symbol names cannot contain spaces and the symbol name cannot begin with the \$character.



3. Double-click the symbol name. The Industrial Graphic Editor appears.



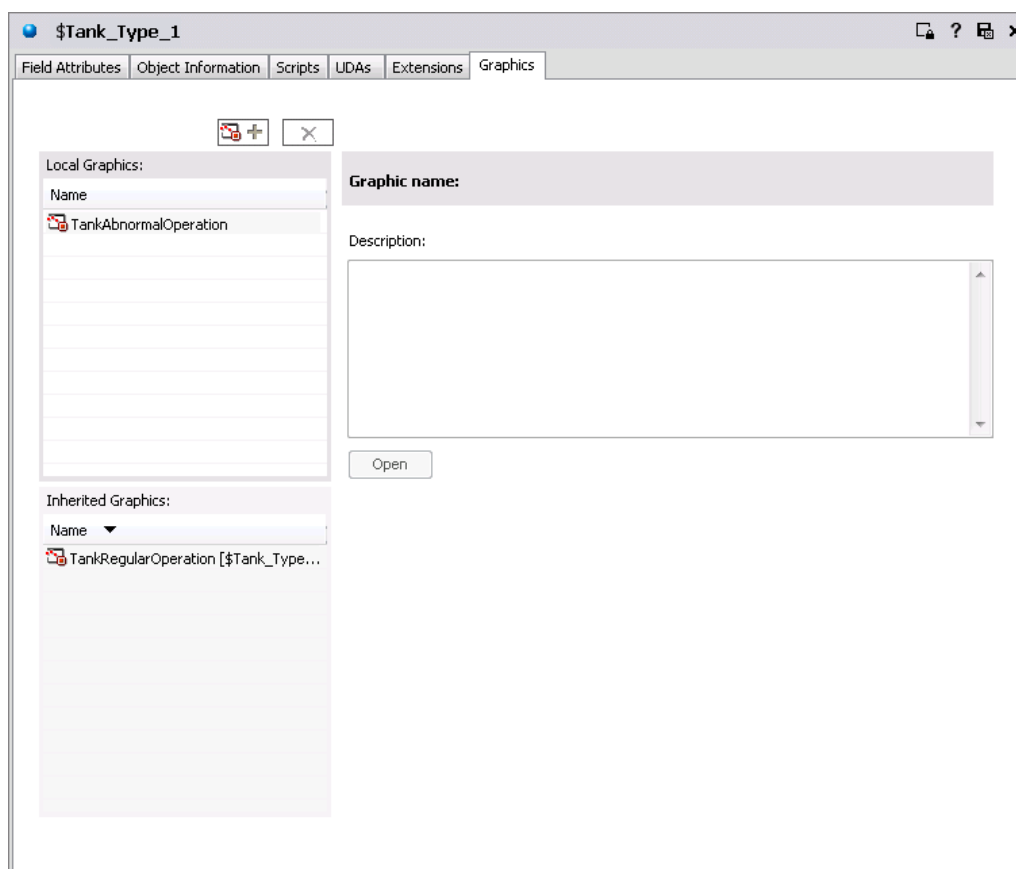
4. Draw the symbol.

## Creating Symbols in AutomationObject Templates

You can create a symbol from the **Graphics** tab in an AutomationObject template. Creating a symbol this way automatically associates the new symbol with the AutomationObject.

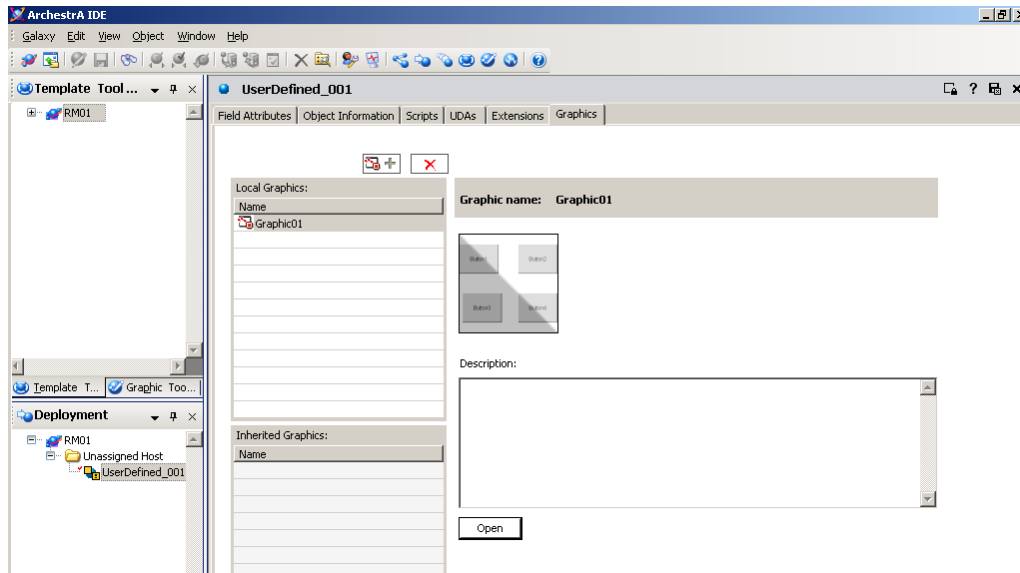
## To create a new symbol for an AutomationObject template

1. Open the AutomationObject template. Click the **Graphics** tab. Any local and inherited symbols are listed.



2. Click the **New Symbol** icon. Give the new symbol a name. Names must be unique. Valid characters for symbol names include alphanumeric characters, \$, #, and \_ (underscore). Symbol names cannot include spaces and the symbol name cannot begin with the \$ character.
3. If needed, type the description of the symbol in the **Description** box.

4. Click the symbol name and click **Open**. The Industrial Graphic Editor appears.



5. Draw the symbol.

## Creating Symbols in AutomationObject Instances

You can create a symbol from the **Graphics** tab in an AutomationObject instance. Creating a symbol this way automatically associates the new symbol with the AutomationObject instance.

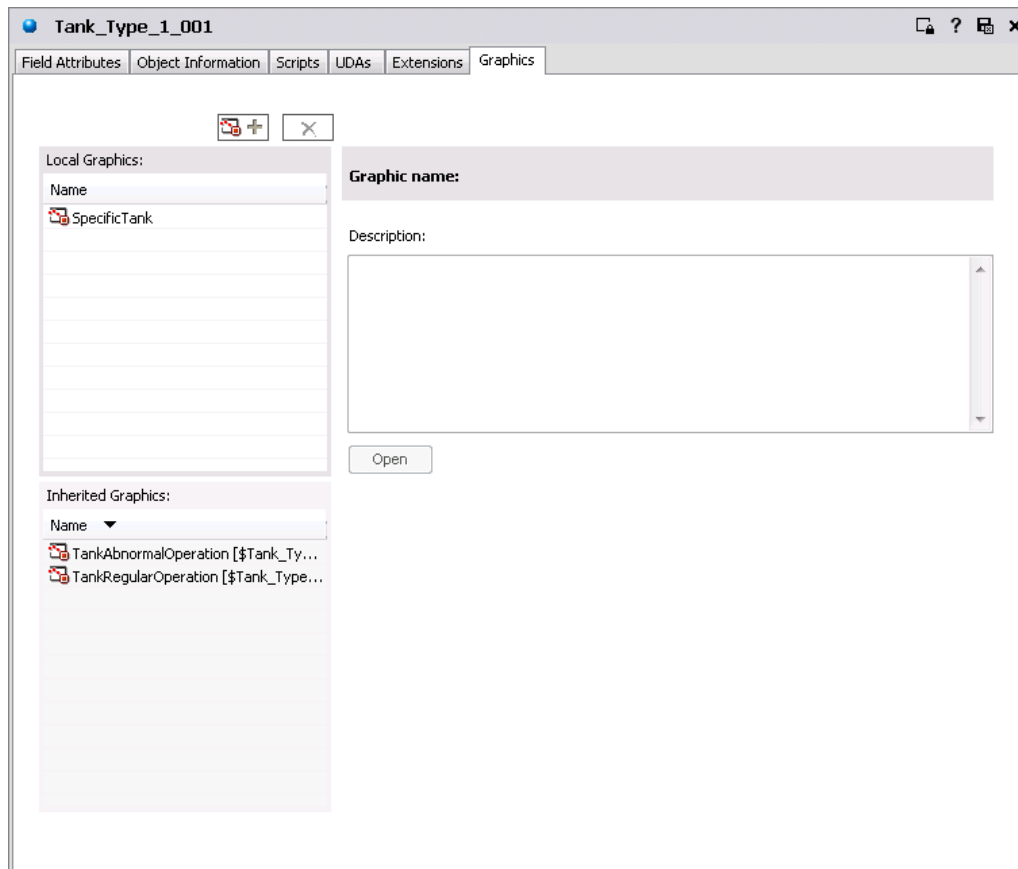
---

**Note:** AutomationObjects can also inherit symbols from their parent template. You can only view an inherited graphic in read-only mode. Inherited graphics cannot be removed or edited.

---

## To create a new symbol for an AutomationObject instance

1. Open the AutomationObject instance. Click the **Graphics** tab. Any local and inherited symbols are listed.



2. Click the **New Symbol** icon. Give the new symbol a name. Names must be unique. Valid characters for symbol names include alphanumeric characters, \$, #, and \_ (underscore). Symbol names cannot include spaces and the symbol name cannot begin with the \$ character.
3. If needed, type the description of the symbol in the **Description** box.
4. Select the symbol name, and then click **Open**. The Industrial Graphic Editor appears.
5. Draw the symbol.

### Symbol Change Propagation

If the symbol is hosted by the Graphic Toolbox and edited: All symbols hosted by AutomationObject templates and instances that contain embedded instances of this symbol are also updated.

If the symbol is hosted by an AutomationObject and edited: All symbols hosted by derived AutomationObjects are also updated.

## Working with Element Styles

### Understanding Element Styles

An Element Style defines a set of visual properties that determine the appearance of text, lines, graphic outlines, and interior fill shown in Industrial Graphics. An Element Style that is applied to a symbol sets pre-configured visual property values that take precedence over a symbol's native visual properties.

Element Styles provide the means for developers to establish consistent visual standards in their ArchestrA applications. An Element Style can define the same visual properties of text, lines, fill, and outlines for all symbols or graphics that belong to an application.

Likewise, Element Styles can show the current status of an object represented by a symbol. For example, an Element Style animation can be applied to a symbol when an object transitions to an alarm state.



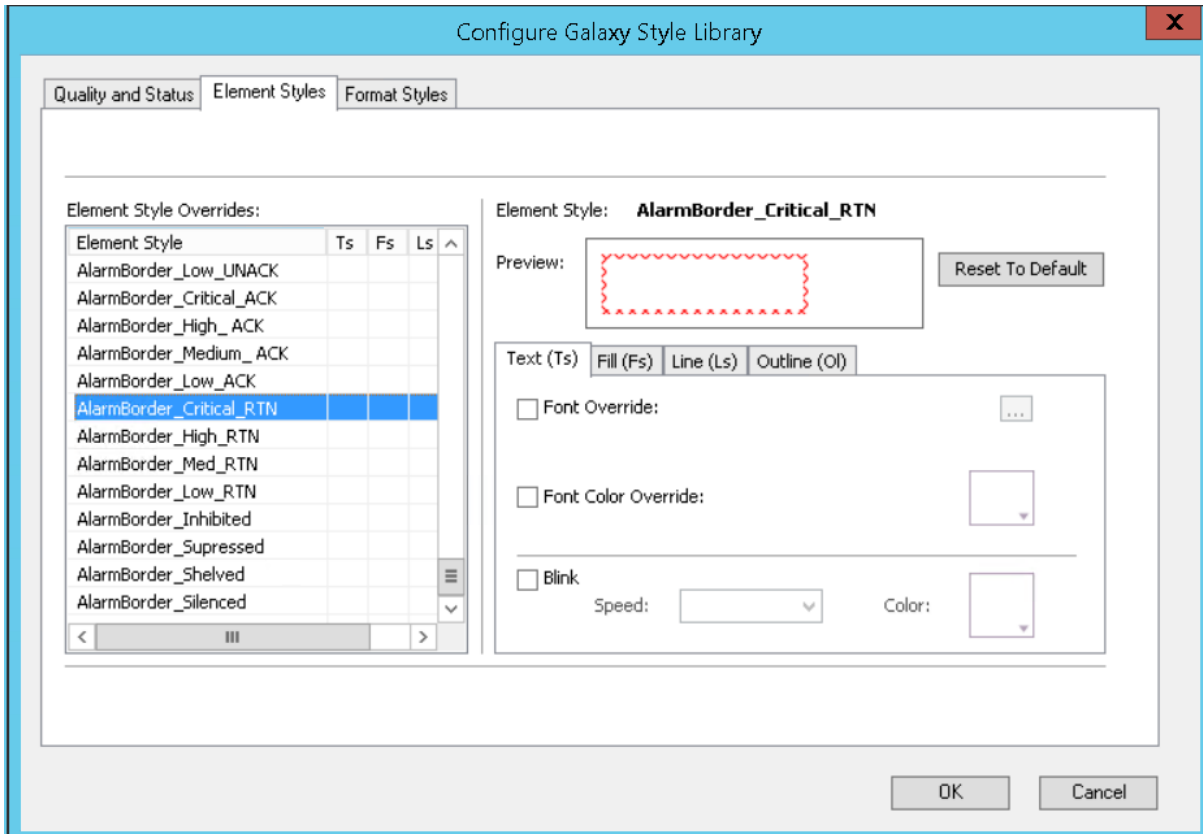
### Element Styles in Managed InTouch Applications

WindowViewer can run only one application at a time. If a platform is deployed on a local node, the configured styles of the Galaxy take precedence over any configured styles in any other standalone applications.

## Galaxy Style Library

The Galaxy Style Library includes a set of predefined Element Styles.

The predefined values of the Element Styles in this library can be changed. However, existing Element Styles cannot be renamed or deleted. Also, new Element Styles cannot be added to the library.



## Visual Properties Defined by Element Styles

The following table lists the visual properties of graphic elements defined in an Element Style.

Graphic Element	Element Properties
Text	<ul style="list-style-type: none"> <li>• Font family</li> <li>• Font size</li> <li>• Font style</li> <li>• Font color</li> <li>• Blink On/Off</li> </ul>
Fill	<ul style="list-style-type: none"> <li>• Fill color</li> <li>• Fill gradient</li> <li>• Fill pattern</li> <li>• Fill texture</li> <li>• Blink On/Off</li> </ul>
Line	<ul style="list-style-type: none"> <li>• Line pattern</li> <li>• Line weight</li> <li>• Line color</li> <li>• Blink On/Off</li> </ul>
Outline	<ul style="list-style-type: none"> <li>• Outline Show/Hide</li> <li>• Outline Pattern</li> <li>• Outline Weight</li> <li>• Outline Color</li> <li>• Blink On/Off</li> </ul>

An Element Style may not define every visual property. If a property value is not defined in an applied Element Style, the element's native style is used and can be changed. However, if an element's property value is defined in an applied Element Style, the element's native properties are disabled and cannot be changed.

## Element Styles in Animations

You can configure an element or a group of elements with Boolean or truth table animations that determine whether Element Styles are applied based on evaluated conditions or expressions. See *Configuring an Animation Using Element Styles* on page 197.

## Property Style Order of Precedence

To understand the behavior of an element's properties when an Element Style is applied, you should understand the order of precedence for the levels at which property styles are applied.

## Updating Element Styles at Application Run Time

You can update the Elements Styles applied to symbols or graphics included in a running application.

- Updating Element Styles from the IDE

When an application is deployed and updates were made to the applied Element Styles from the System Platform IDE, those updates will be propagated to the graphic elements in a running application without requiring WindowViewer to be closed and re-opened.

- Importing an updated Graphic Style Library

Importing an updated Graphic Style Library that includes different applied Element Styles will propagate those changes to graphic elements in a running application without requiring WindowViewer to be closed and re-opened.

## Managing Element Styles

At the Galaxy level, you can import and export Galaxy Style Libraries containing custom Element Styles to applications running on other Galaxies. This section describes the tasks to create a set of custom Element Styles that can be used in other Galaxies.

### Importing and Exporting Galaxy Style Libraries

You can import a Galaxy Style Library to load its Element Styles in a Galaxy. You can also modify Element Styles, and then export the Galaxy Style Library as custom user-defined Element Styles.

Optional Galaxy Style Library XML files are located in the ...\\Program Files ( x86)\\ArchestrA\\FrameWork\\Bin\\AdditionalElementStyles folder. The names of the XML files suggest the primary color schemes of the Element Styles within each optional Galaxy Style Library. For more information about importing and exporting Galaxy Style Libraries, see "Manage Objects" in the *Application Server User Guide*.

### Change the Visual Properties of an Element Style

You can modify the visual properties of any Element Style in the currently loaded Galaxy Styles Library. You modify properties by setting overrides on the **Element Styles** tab in the **Configure Galaxy Style Library** dialog box.

In the **Configure Galaxy Style Library** dialog box, you can:

- Modify the appearance of text by setting overrides for the text font, text size, text style, text color, and blinking.
- Modify the appearance of graphic fill by setting overrides for fill color and blinking.
- Override the appearance of the line pattern, weight, color, and blinking.
- Override the appearance of the outline line pattern, weight, color, and blinking.
- Preview the appearance of an Element Style.
- Reset Element Style visual properties to their default values.

#### To show the current Element Styles of a Galaxy

1. On the **Galaxy** menu, click **Configure** and click **Galaxy Style Library**. The **Configure Galaxy Style Library** dialog box appears.
2. Click the **Element Styles** tab.

The **Element Styles** tab includes the following fields:

- The **Element Style Overrides** grid lists the Element Styles included in the library. An X within grid cells indicates style properties that have been overridden.
- The **Preview** field shows the appearance of an element when the current Element Style is applied.
- The **Reset to Default** button returns all modified Element Styles to their default values.

- The property tabs include related fields to set values for each property defined in the selected Element Style.

## Overriding the Element Style Text Properties

You can modify an Element Style's text visual properties by setting alternative values for text font, text color, text style, and blink rate.

### To change the appearance of text in an Element Style

1. On the **Galaxy** menu, point to **Configure**, and then click **Galaxy Style Library**. The **Configure Galaxy Style Library** dialog box appears.
2. Select an Element Style from the **Element Style Overrides** list.
3. Click the **Text (Ts)** tab.
4. To change the font, select **Font Override**, click the browse button, and select a font from the **Font** dialog box.
5. To override the font color:
  - a. Select **Font Color Override**.
  - b. Click the color box.
  - c. Select a color from the **Select Font Color** dialog box.
6. To override the text blink behavior:
  - a. Select **Blink**.
  - b. Select a blinking speed from the **Speed** list.
  - c. Click the color box to show the **Select Blink Color** dialog box.
  - d. Select the color, gradient, pattern, and texture for the blink style.
7. Click **OK**.

## Overriding the Element Style Fill Properties

You can modify an Element Style's fill visual properties by setting alternative values for fill color and blink rate.

### To override the fill appearance of an Element Style

1. On the **Galaxy** menu, point to **Configure**, and then click **Galaxy Style Library**. The **Configure Galaxy Style Library** dialog box appears.
2. Select an Element Style from the **Element Style Overrides** list.
3. Click the **Fill** tab.
4. To override the fill style:
  - a. Select **Fill Color Override**.
  - b. Click the color box.
  - c. Select a style from the **Select Fill Color** dialog box.
5. To override the fill blink behavior:
  - a. Select **Blink**.
  - b. Select a blink speed from the **Speed** list.
  - c. Click the color box.
  - d. Select a style from the **Select Fill Color** dialog box.



6. Click **OK**.

### Overriding the Element Style Line Properties

You can modify an Element Style's line visual properties by setting alternative values for line color, line pattern, and line weight

#### To override the line appearance of an Element Style

1. On the **Galaxy** menu, point to **Configure**, and then click **Galaxy Style Library**. The **Configure Galaxy Style Library** dialog box appears.
2. Select an Element Style from the **Element Style Overrides** list.
3. Click the **Line** tab.
4. To override the line pattern, select **Line Pattern Override** and select a line pattern from the adjacent list.
5. To override the line weight, select **Line Weight Override** and enter a new line weight in the adjacent box.
6. To override line color properties:
  - a. Select **Line Color Override**.
  - b. Click the color box.
  - c. Select a color style from the **Select Line Color** dialog box.
7. To override the line blink behavior:
  - a. Select **Blink**.
  - b. Select a blinking speed from the **Speed** list.
  - c. Click the color box.
  - d. Select a style from the **Select Blink Color** dialog box.
8. Click **OK**.

### Overriding the Element Style Outline Properties

You can modify an Element Style's outline visual properties by setting alternative values for text font, text color, text style, and blink rate.

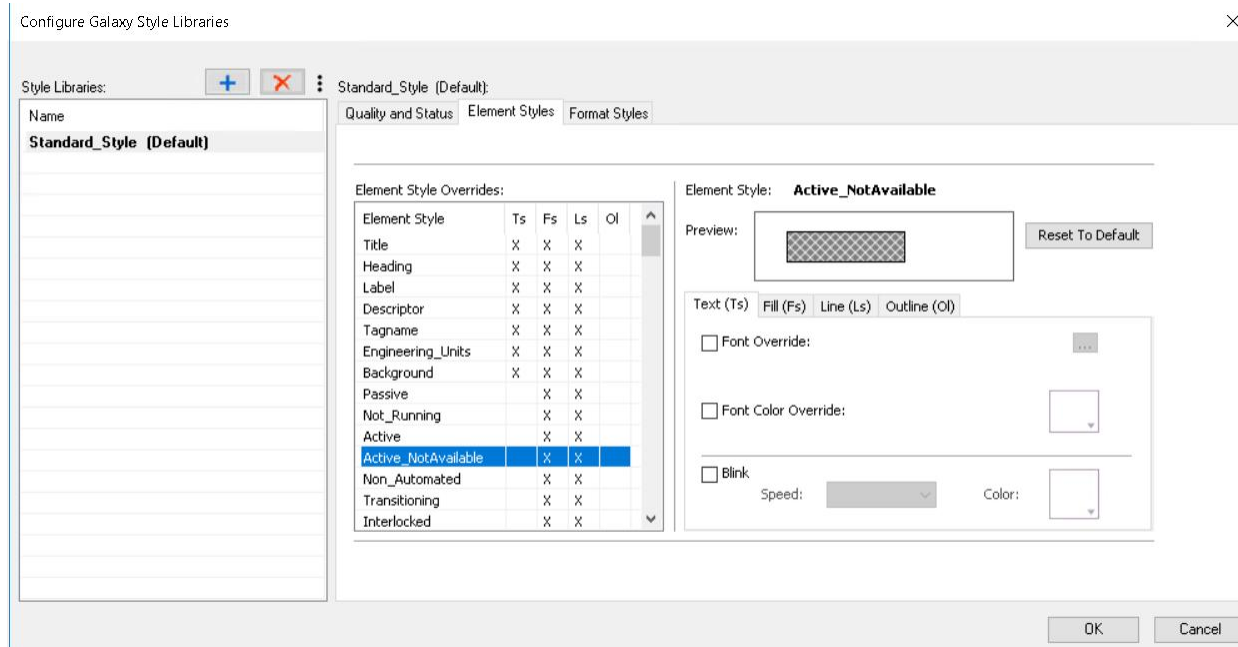
#### To override the outline appearance of an Element Style

1. On the **Galaxy** menu, point to **Configure**, and then click **Galaxy Style Library**. The **Configure Galaxy Style Library** dialog box appears.
2. Select an Element Style from the **Element Style Overrides** list.
3. Click the **Outline** tab.
4. Select **Show Outline**.
5. To set the line pattern, select **Line Pattern** and select a line pattern from the adjacent list.
6. To set the line weight, select **Line Weight** and type a line weight in the adjacent box.
7. To set the line style:
  - a. Click the color box next to **Line Color**.
  - b. Select a style from the **Select Line Color** dialog box.
8. To set the line blink behavior:
  - a. Select **Blink**.

- b. Select a blinking speed from the **Speed** list.
- c. Click the color box.
- d. Select a blink style from the **Select Blink Color** dialog box.

### Previewing an Element Style

The **Preview** area shows the appearance of an Element Style’s current assigned property values.



### To preview an Element Style

1. On the **Galaxy** menu, point to **Configure**, and then click **Galaxy Style Library**. The **Configure Galaxy Style Libraries** dialog box appears.
2. Select the **Element Styles** tab.
3. Select an Element Style from the **Element Style Overrides** list.

The **Preview** field updates to show the appearance of the selected Element Style.

### Resetting an Element Style to Default Values

You can reset an Element Style to its original default property values.

---

**Note:** Resetting an Element Style resets visual properties to their original default values, not to any previous override settings.

---

### To reset an Element Style to default values

1. On the **Galaxy** menu, point to **Configure**, and then click **Galaxy Style Library**. The **Configure Galaxy Style Library** dialog box appears.
2. Select the **Element Styles** tab.
3. Select one or more Element Styles from the **Element Style Overrides** list.
4. Click **Reset to Default**. The selected Element Style properties are reset to their default values.

## Changing the Visual Properties of User-Defined Element Styles

The Galaxy Style Library includes a set of 25 user-defined Element Styles. User-defined Element Styles appear towards the bottom of the list of the **Element Style Overrides** field and are named User\_Defined\_01 to User\_Defined\_25.

All visual properties of user-defined Element Styles are initially set to default values. Visual properties can be individually configured for each user-defined Element Style by setting overrides for text, fill, line, and outline as other predefined Element Styles.

## Applying Element Styles to Elements

You can apply Element Styles to one or more graphic elements. Unlike setting Element Style overrides that change the appearance of an Element Style's properties, applying an Element Style to a graphic element overrides the element's native properties.

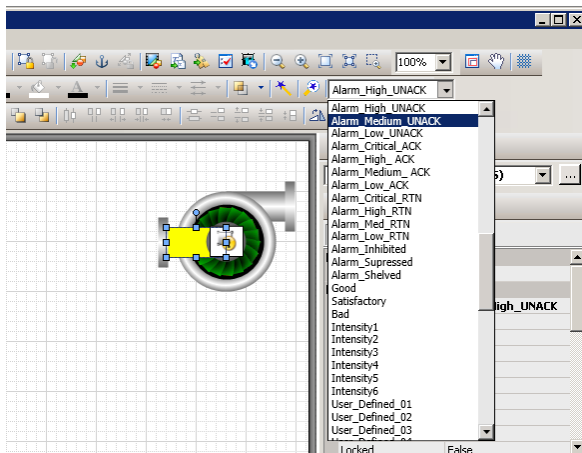
Applying Element Styles to elements can help standardize the appearance of those elements in the Galaxy and show the current state of an object represented by a symbol or graphic. For more information, see *Change the Visual Properties of an Element Style* on page 191.

## Using the Element Style List

The Industrial Graphic Editor menu bar contains an **Element Style** list to select an Element Style and apply it to a selected element of a symbol or graphic.

### To apply an Element Style to a graphic element

1. Open the symbol or graphic in the Industrial Graphic Editor.
2. Select one or more elements from the graphic or symbol.
3. Select an Element Style from the **Element Styles** list to apply to the selected elements.



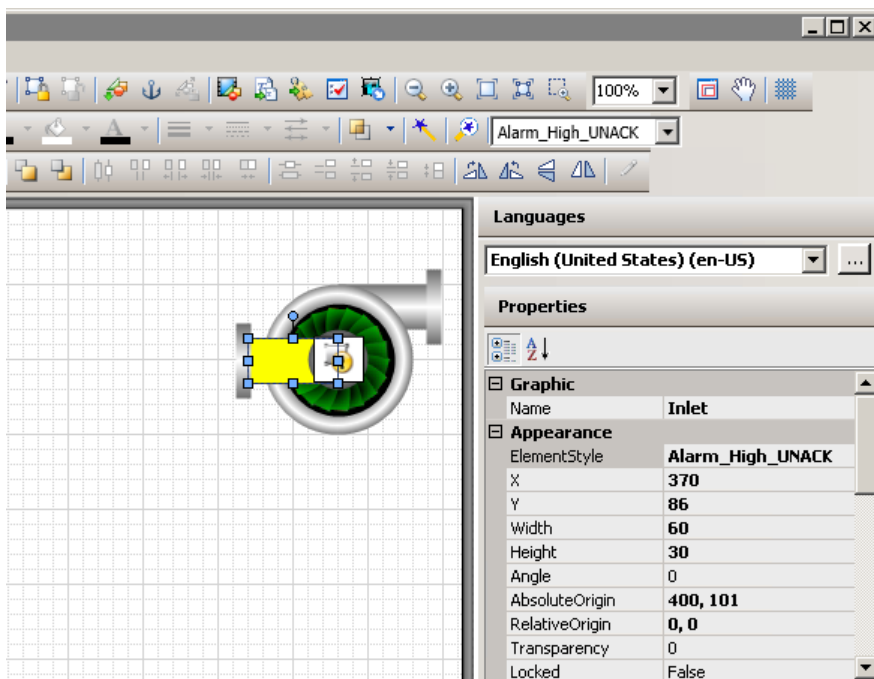
## Using the Properties Grid

The Industrial Graphic Editor **Properties** view contains an **Element Style** Appearance item to select an Element Style and apply it to a selected element of a symbol or graphic.

### To apply an Element Style from the Properties Editor

1. Open the symbol or graphic in the Industrial Graphic Editor.
2. Select one or more elements from the graphic or symbol.

3. In the **Appearance** category of the **Properties** Editor, select an Element Style from the **Element Style** list.



## Using Format Painter

You can use the Industrial Graphic Editor's Format Painter to copy an Element Style from one graphic element to another.

### To apply an Element Style using Format Painter

1. Open a symbol or graphic in the Industrial Graphic Editor.
2. Select the element with the Element Style you want to copy.
3. On the **Edit** menu, click **Format Painter**. The pointer appears as the Format Painter cursor.
4. Select the elements you want to apply the Element Style to. The Element Style is applied to the clicked element.

## Clearing an Element Style

When an Element Style is applied to an element, you cannot edit the element's styles that are controlled by the applied Element Style. However, you can clear the application of the Element Style so that all of the styles can be edited.

### To clear an Element Style

1. Select the element.
2. Select **None** in the Element Style list.

## Selecting an Element Style as a Default for a Canvas

You can select an Element Style at the canvas level. The selected Element Style is applied to any graphic element or groups that you create on the canvas.

## Applying Element Styles to Groups of Elements

You can apply an Element Style on a group of elements in the same way that you apply an Element Style to an element. However, the group's run-time behavior must be set to **TreatAsIcon**.

### Setting a Group's Run-time Behavior to TreatAsIcon

To apply an Element Style to a graphic element group, the group's **TreatAsIcon** property must be set to **True**. Otherwise, the Element Style lists are disabled when an element group is selected.

#### To set a group's TreatAsIcon property to true

1. Select the element group to which the Element Style will be applied.
2. On the **Properties** menu, click **Run-time Behavior** and click **TreatAsIcon**.
3. Select **True** from the drop-down list.

### Understanding Element Style Behavior with a Group of Elements

- The Element Style applied to a group has higher precedence than the property styles applied to individual graphic elements in the group.
- If the Element Style applied to a group of elements has undefined property styles, then the element continues to use its Element Style or element-level settings for undefined property styles.
- If the Element Style that is applied to a group of elements has defined property styles, then those property styles override the property styles defined at the element level for elements in the group.
- An Element Style cannot be applied to a nested element group.
- If you add an element to a group that has a group-level Element Style applied, the group Element Style is applied to it.

## Configuring an Animation Using Element Styles

You can configure an element or a group of elements with a:

- Boolean animation that applies Element Styles based on a binary True/False condition.
- Truth table animation that applies Element Styles based on a range of possible values.

The truth table animation that applies Element Styles:

- Associates expressions of any data type supported by Application Server or InTouch to an Element Style.
- Defines as many conditions as required and applies a separate Element Style for each condition
- Defines the conditions to apply an Element Style by specifying a comparison operator (=, >, >=, <, <=) and a breakpoint, which itself can be a value, an attribute reference, or an expression.
- Arranges conditions in the order that Element Styles are processed.

### Configuring a Boolean Animation Using Element Styles

You can configure an element or a group of elements with a Boolean animation that uses only two Element Styles.

#### To configure an element or a group of elements with an Element Style that uses Boolean animation

1. Open the symbol or graphic in the IDE Industrial Graphic Editor.

2. Select the element or element group.
3. On the **Special** menu, click **Edit Animations**. The **Edit Animations** dialog box appears.
4. Click the **Add** icon and select **Element Style**. The Element Style animation is added to the Animation list and the **Element Style** state selection panel appears.
5. Click the **Boolean** button. The **Boolean Element Style** configuration panel appears.
6. In the **Boolean** text box, enter a Boolean numeric value, attribute reference, or an expression.
7. Clear **ElementStyle** in the **True, 1, On** area or **False, 0, Off** area if you do not want a different Element Style for the true or false condition than the default Element Style that is shown in the **Element Style** list.
8. In the **True, 1, On** area, select the Element Style in the list to use when the expression is true.
9. In the **False, 0, Off** area, select the Element Style in the list to use when the expression is false.
10. Click **OK**.

## Configuring a Truth Table Animation with Element Styles

You can configure an element or a group of elements with a Truth Table animation that selects multiple Element Styles based on a set of evaluated values or expressions.

### To configure an element or a group of elements with an Element Style that uses Truth Table animation

1. Open the symbol or graphic in the IDE Industrial Graphic Editor.
2. Select the element or group.
3. On the **Special** menu, click **Edit Animations**. The **Edit Animations** dialog box appears.
4. Click the **Add** icon and select **Element Style**. The Element Style animation is added to the Animation list and the **Element Style** state selection panel appears.
5. Click the **Truth Table** button. The **Truth Table Element Style** configuration panel appears. The Element Style that is applied to the element is shown in the **Element Style** list at the bottom of the panel.
6. In the **Expression Or Reference** area:
  - Select the data type of the expression from the list.
  - Type a value, attribute reference or expression in the text box.
7. If the data type of the expression is string or internationalized string, you can specify to ignore the case by selecting **Ignore Case**.
8. In the **Truth Table**, select the **Element Style** check box and select the Element Style for one of the conditions to be defined in the truth table.
9. In the **Operator** column, select a comparison operator.
10. In the **Value or Expression** column, type a value, attribute reference, or expression.
11. To add other conditions:
  - a. Click the **Add** icon. An additional condition is added to the truth table.
  - b. Select the **Element Style** check box, select the Element Style for the condition, select an operator, and enter the condition value or expression.
12. After adding all truth table conditions, click **OK**.

Truth Table animation is typically used to set Element Styles to the different states of an object. For example, you can set Truth Table conditions to show different Element Styles that represent the following alarm conditions:

- When the attribute TankLevel\_001.PV is 0 then no Element Style is applied.
- When the attribute TankLevel\_001.PV is less than 20, then the Element Style is Alarm\_Minor\_Dev.
- When the attribute TankLevel\_001.PV is greater than the attribute Standards.TankMax then the Element Style is Alarm\_Major\_Dev.

### Deleting a Condition from an Animation Truth Table

You can delete a condition from an animation Truth Table to remove the associated Element Style from the animation.

#### To delete a condition from a Truth Table animation that uses Element Styles

1. Open the **Edit Animations** dialog box, **Truth Table Element Style** panel.
2. Select the condition you want to delete.
3. Click the **Remove** icon. The condition is removed.

### Changing the Processing Order of Element Styles in a Truth Table Animation

You can change the processing order of Element Styles by moving the conditions up or down in the Truth Table list. The Element Style at the top of the Truth Table list is processed first. The remaining Element Styles are processed in order based on their position from the top of the list.

#### To change the processing order of Element Style conditions

1. Open the **Edit Animations** dialog box, **Truth Table Element Style** panel.
2. Select the condition you want to move up or down the condition list in order for it to be processed sooner or later.
3. Click the:
  - **Arrow up** icon to move the condition up in the truth table.
  - **Arrow down** icon to move the condition down in the truth table.

## Adding Industrial Graphics to AVEVA OMI Panes

### Create Fixed Size Symbols

By default, a symbol attempts to scale to the size of its hosting pane and maintain the symbol's horizontal and vertical aspect ratio. A symbol's size created using the Industrial Graphic Editor is determined by a bounding rectangle around all graphic elements and embedded symbols that compose the symbol. The size and position of the graphic elements and embedded symbols determines the size of the bounding rectangle around the symbol.

When a symbol is placed in a layout pane, If the symbol's bounding rectangle exceeds the size of its hosting pane, the bounding rectangle becomes smaller by reducing the original size of its contents. If the symbol's bounding rectangle is smaller than the pane size, the bounding rectangle increases and enlarges the original size of its contents.

Symbols that have a defined fixed size render in run time identically to symbols without a fixed size. A symbol's size adjusts to the size of the pane while maintaining its aspect ratio as appropriate. Most symbols that have a configured fix size are typically placed in a specific target pane of a known size that enables them to be the rendered exactly as designed in Industrial Graphic Editor without any required resizing during configuration or run time.

## Create a Symbol to Fit a Pane of a Known Size

This topic describes the general sequence of steps to create a fixed size symbol designed to fit a ViewApp pane with a known size. This workflow is typical of creating content that will fit exactly within a pane while editing a ViewApp with the ViewApp Editor.

### Important:

The size of a symbol in run time is affected by the resolution of the screen that will show the running ViewApp and the specified width and height of the layout pane in which the symbol is placed. Also, the layout **Display Mode** property includes a set of options that determines how content is resized during run time to fit within the size of a layout pane.

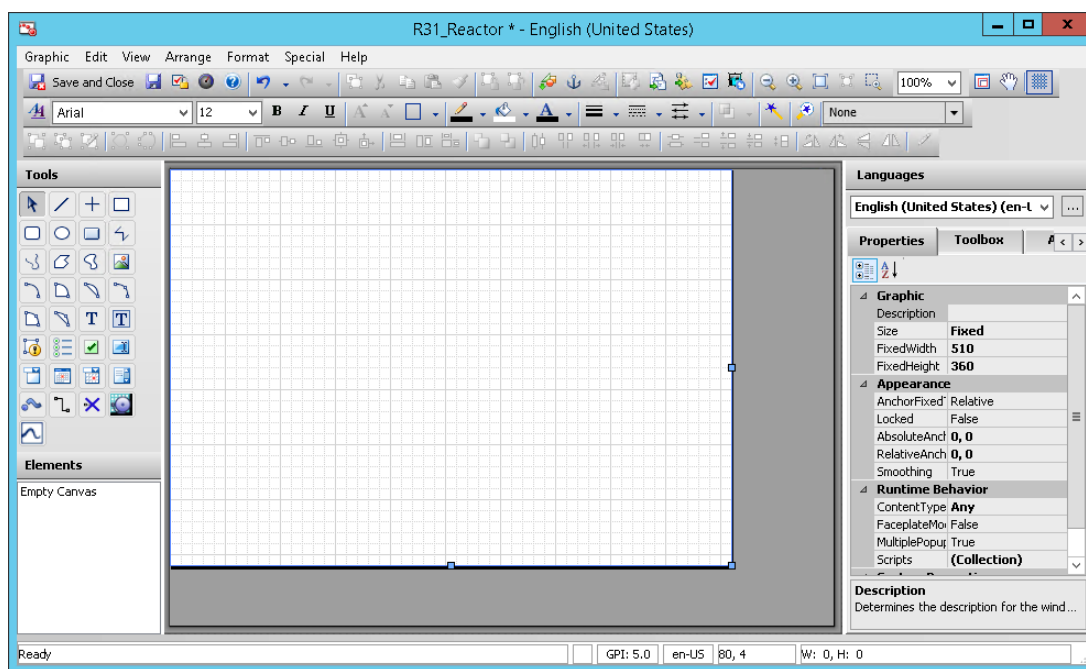
The following procedure describes the steps to create a fixed size symbol that will fit within a pane of a known size. You should create a fixed size symbol whose width and height matches the resolution of the screen profile. You should set the the dimensions of the hosting pane to the same size as the symbol.

### To create a fixed size symbol that fits within a pane of a known size

1. Open the ViewApp in the ViewApp Editor that will contain the fixed size symbol.
2. Right-click on the pane that will contain the fixed size symbol and select **Create Graphic**.  
A dialog box appears requesting a name for the graphic to be created.
3. Assign a name to the graphic.

The Industrial Graphic Editor opens and the canvas shows the dimensions of the pane that the request to create a graphic was submitted. The **Size** property is set to **Fixed** and the **FixedWidth** and **FixedHeight** properties show the dimensions of the pane in pixels.

The white portion of the Industrial Graphic Editor's canvas represents the size of the pane where the request to create a graphic was submitted. Portions of the canvas to the right and bottom of the white area are gray. The gray portion of the canvas is the area outside the vertical and horizontal dimensions of the pane.



4. Place graphic elements and symbols to create a symbol.



5. Save and close the symbol from the Industrial Graphic Editor to return to the ViewApp Editor.  
The symbol you created appears in the pane where the request to create a graphic was submitted. The symbol should fit into the pane at the same size as it was designed in the Industrial Graphic Editor.
6. Save the ViewApp and deploy it.
7. Open the deployed ViewApp  
In run time, the ViewApp will show the symbol you created at the same size as it was originally designed in the Graphic and ViewApp editors.  
The rendered symbol shows:
  - The symbol fits in the pane at the same size as it was designed in the Industrial Graphic Editor.
  - The symbol's size does not change in run time.
  - Any blank spaces around the symbol are maintained and appear in the symbol during run time.

## Create a Fixed Sized Symbol for a Known Pane Size

This topic describes the general procedure to create a fixed size symbol that will fit a pane of a known size even if the pane does not exist when the fixed size symbol is created.

This workflow is typically the case where two different employees work on a ViewApp to update it to reflect changes in a production environment. The first employee is usually a user who creates a fixed size symbol. A second employee will drag and drop the symbol in a pane while editing the ViewApp with the ViewApp Editor.

### To create a fixed size graphic that will fit a pane with a known size

1. The first employee creates a new symbol and opens it in the Industrial Graphic Editor.  
The entire canvas area of the Industrial Graphic Editor is enabled to place graphic elements and symbols.
2. Configure the symbol to a known pane size.
  - a. Select the **Graphic** property **Size** and set it to **Fixed**.  
The **FixedWidth** and **FixedHeight** properties appear after the **Size** property is set to **Fixed**.
  - b. Update the values of **Fixed Width** and **Fixed Height** properties to the pixel dimensions of a pane with a known size.  
You can also move the grab handles on the fixed size boundary to set the dimensions of the symbol.
3. Place the graphic elements and other symbols within the fixed size area of the symbol.
4. If you want white space around the symbol, adjust the width and height of the symbol to show a white margin around the graphic elements and symbols that compose your symbol.
5. Save and close the symbol.
6. The second employee creates a layout to be used in the ViewApp with a pane that is the same size as the fixed size symbol.
7. Open the ViewApp with the ViewApp Editor.
8. Assign a Screen Profile and the layout containing the pane to the ViewApp.
9. Select a screen that contains the layout with the pane of a known size.
10. Select the fixed size symbol created by the first employee and drag and drop it in the pane.

The configured symbol appears in the selected pane of the ViewApp when it is edited by the ViewApp Editor. The symbol preview shows:

- The symbol fits in the targeted pane with the same size as it was designed in the Industrial Graphic Editor.
- If necessary, the symbol can scale larger or smaller to fit the pane.
- Any white space placed around the symbol while it was created is maintained when the symbol is placed in a pane.

11. Save the ViewApp and deploy it

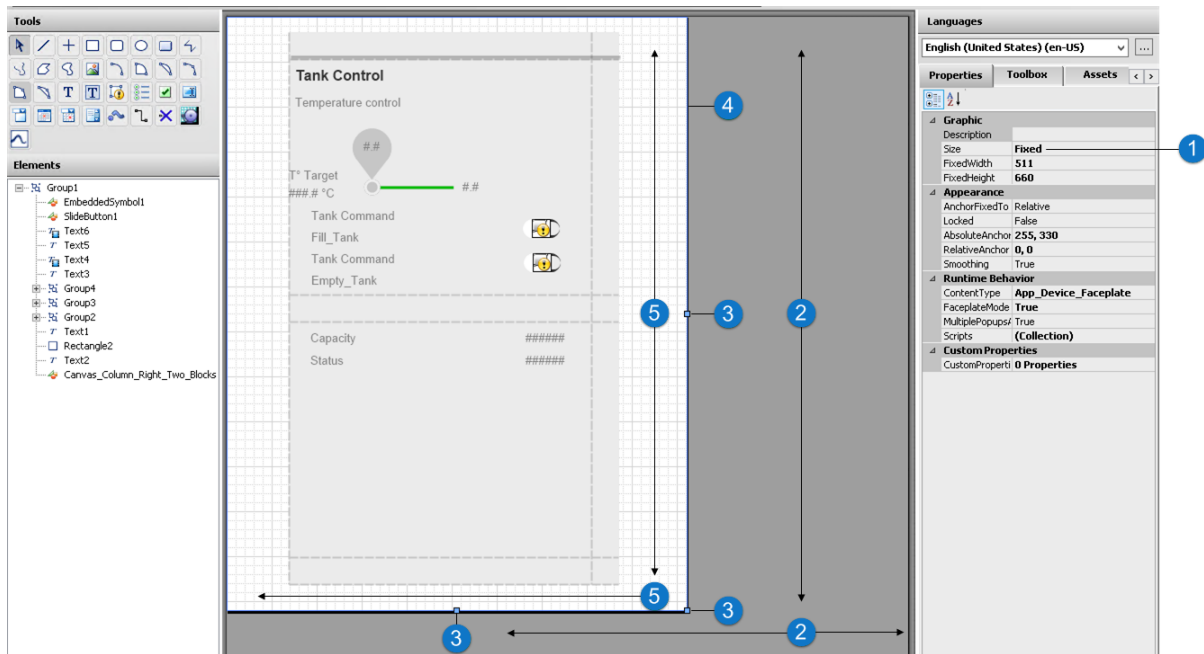
12. Open the running ViewApp.

The running ViewApp shows the fixed size symbol in the target pane at the same size as it was created in the Industrial Graphic Editor and configured in the ViewApp Editor. The rendered fixed size symbol graphic shows:

- The symbol fits in the pane with the same size as it was designed in the Industrial Graphic Editor
- Any white space placed around the fixed size symbol is maintained.

## Use the Industrial Graphic Editor to Create Fixed Size Symbols

After placing a symbol or graphic elements on the canvas of the Industrial Graphic Editor, you can set several properties to specify a fixed size for the symbol. The following screen shot shows how to adjust the fixed size of a symbol.



1	<p>The <b>Size</b> property is set to <b>Auto</b> by default, which automatically sizes symbols. Set <b>Size</b> to <b>Fixed</b> to create fixed size symbols. When set to <b>Fixed</b>, the <b>FixedWidth</b> and <b>FixedHeight</b> properties appear, which show the dimensions of the fixed area of a symbol in pixels.</p> <p>The symbol's fixed size width and height values can be entered directly in the <b>FixedWidth</b> and <b>FixedHeight</b> property fields.</p> <ul style="list-style-type: none"> <li>• <b>FixedWidth</b> <ul style="list-style-type: none"> <li>○ Default value = 500</li> <li>○ Maximum value = 7680</li> <li>○ Minimum value = 0</li> </ul> </li> <li>• <b>FixedHeight</b> <ul style="list-style-type: none"> <li>○ Default value = 500</li> <li>○ Maximum value = 4320</li> <li>○ Minimum value = 0</li> </ul> </li> </ul> <p>You can also change the width and height of the symbol's fixed size area by moving the boundary with its grab handles.</p>
2	<p>The gray area represents that part of the Industrial Graphic Editor canvas that is outside the boundary of the fixed size area of the symbol.</p>
3	<p>Grab handles on the fixed size boundary can be moved to change the size of the fixed area of a symbol. The <b>FixedWidth</b> and <b>FixedHeight</b> properties update to show the current width and height when you move the boundary.</p> <p>Grab handles are placed at the following locations on the fixed size boundary:</p> <ul style="list-style-type: none"> <li>• Top right corner</li> <li>• Center right side</li> <li>• Bottom right corner</li> <li>• Bottom center</li> </ul>
4	<p>The thick black line shows the boundary between the Industrial Graphic Editor canvas within the fixed size area of the symbol set by the symbol's <b>FixedWidth</b> and <b>FixedHeight</b> properties and the canvas area outside the fixed dimension area.</p>
5	<p>The white area of the Industrial Graphic Editor canvas is within the fixed size boundary of the symbol.</p>

You can place graphic elements from the Industrial Graphic Editor's **Tools** section or embed symbols as you would normally do when the **Size** property is set to **Auto**. As long as the graphics elements or embedded symbols are placed inside the fixed size boundary, the symbol will fit the size of a pane perfectly. Graphics within the symbol maintain same size, scale, spacing, and white spaces around the graphics on the canvas specified during design time.

## More Information About Fixed Size Symbols During Configuration

Symbols created for a layout pane obtain size dimension from the pane's size by default. The values assigned to the **FixedHeight** and **FixedWidth** properties automatically match the height and width of the pane. Symbols that are designed with its elements placed within the dimension boundaries will fit in the pane.

When the dimensions of a symbol are changed while editing in the ViewApp Editor, the following behaviors are observed:

### Fixed Size Symbol Size is Greater Than Pane Size

- When the symbol size is larger than hosted pane size, elements placed outside the pane are included in the configuration view. The symbol's size is reduced to fit it entirely within the dimensions of the pane.
- When the **FixedWidth** and **FixedHeight** properties of a symbol match the pane size, but there are graphic elements beyond the symbol's fixed width and height boundaries, the symbol's size is reduced to fit the graphic elements outside the fixed size area within the dimensions of the pane.

#### **Fixed Sized Symbol Size is Smaller Than Pane Size**

When a symbol's fixed size is smaller than the hosted pane size, the symbol's size is increased to fit the pane size. The size of all of a symbol's graphic elements are increased and appear larger.

#### **Fixed Size Symbol Embedded in Another Symbol**

- The embedded fixed size symbol maintains its size configured at design time.
- There will not be any clipping of any white space provided around primitives or embedded symbols present in the symbol.
- There will not be any scaling of graphics. An exception to this is if the embedded symbol was resized at design time, the graphic will look scaled.

### **Fixed Size Symbol Behavior During Run Time**

The following list summarizes the key behaviors of fixed size symbols during run time.

- Symbols that have the **Size** property set to **Fixed**, render all graphic elements placed within the fixed size boundary at their configured fixed size at run time.
  - Symbols do not change size or aspect ratio within the pane that is hosting the graphic symbol.
  - Symbols configured to fit exactly within a hosting pane, render to fit exactly within the size of a pane. The graphic elements in the symbol render at the same size and placement as designed in the Industrial Graphic Editor.
  - The white space around primitives or embedded symbols present in the symbol maintains its size and placement within the symbol.
- Fixed size symbols change size or aspect ratio to fit the hosting pane during run time in the following cases:
  - When the symbol's size is larger or smaller than the hosting pane.
  - When symbol size is the same as the hosting pane but there are graphics that are placed outside the fixed size boundary configured for the symbol. This includes graphics that are placed on negative coordinates as well.
  - When symbol size is the same as the hosting pane, but graphics that are placed on the boundary of the symbol marked by the size configured for the graphic.
- Embedded symbols follow the same behavior as standard fixed size symbols.

## **Using Custom Properties with Application Server**

This section provides information specific to Application Server for using custom properties in Industrial Graphics. You can use custom properties to extend the functionality of graphics at run time, such as showing historical data, and to change references dynamically through binding and scripting.

Topics covered in this section include:

- *Examples of Using Custom Properties* on page 205
- *Using Custom Properties to Show Historical Summary Data* on page 205
- *Using Binding in Custom Properties* on page 209

- *Changing the Expression or Reference of a Custom Property at Run Time* on page 211

For additional background information, see *Using Custom Properties in the Industrial Graphic Editor User Guide*.

## Examples of Using Custom Properties

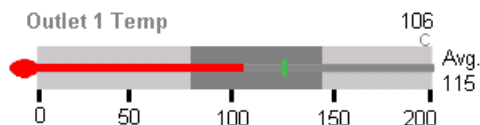
Possible uses for using custom properties are:

- A "TankLevel" custom property of type Writable Attribute can be given a value of "me.pv".
- A "MaxFillLevel" Custom Property of type Expression can be given a value of "Me.MaxCapacity - 200".

## Using Custom Properties to Show Historical Summary Data

You can add a custom property to reference historical data collected over a specified period during run time. The Historian can transform this data to create a set of analog or state statistics that can be shown by Industrial graphic animation during run time.

For example, consider an example of a temperature meter graphic that shows an optimal temperature range and you would like to know what the average temperature has been over the last 15 minutes. You can add a Value Display animation that shows the average temperature derived from analog temperature data saved in the Historian and referenced by a custom property.



## Analog Statistical Summary Data

A custom property analog reference can subscribe to statistics from analog data collected over a defined summary period and saved to the Historian. The following table lists the analog historical statistical data that can be specified for a custom property.

Analog Statistical Data	Description
Average	A time-weighted average calculated from values within a summary period. The average is calculated at the end of the summary period.
Count	A value count calculated from values within a summary period. The count is calculated at the end of the summary period.
First	The first value that occurs within a summary period based on the actual timestamp within the summary period.
Integral	An integral value calculated from values within a summary period. The integral is calculated at the end of the summary period.
Maximum	The first maximum value that occurs within a summary period.

<b>Analog Statistical Data</b>	<b>Description</b>
Minimum	The first minimum value that occurs within a summary period.
PercentGood	The ratio of labeled "good" quality data to all data within the summary period. The ratio is expressed as a percentage in the range 0 to 100. PercentGood is calculated at the end of the summary period.
StdDev	Time weighted standard deviation calculated from values within a summary period. The value is calculated using time weighted sums (Integrals) and time weighted sums of squares (IntegralOfSquares) values.
Last	The last value that occurred in the summary period based on the actual timestamp within the summary period.

## State Statistical Summary Data

The Historian stores and retrieves values, where every value gets stored with some timestamp and associated quality. This triplet of value, timestamp, and quality is called VTQ and constitutes the smallest addressable piece of data in the Historian data model.

State summary statistics summarize the states of a value. Four different state value data types are possible: analog (integer), Boolean, string, and Null.

---

**Note:** Possible Boolean state values are 0 or 1. True or False strings are not supported.

---

A custom property state reference can subscribe to state statistics from the Historian as static text, an expression or reference, an aggregate function name, minutes, and state value.

The Historian returns the VTQ for one cycle of a specified state. The quality returned is always OpcQuality. The time returned is always the summary period start time. Value and Time differ based on the aggregate function.

The following table lists state historical statistical data that can be specified for a custom property.

<b>State Statistical Data</b>	<b>Description</b>
Average	Average time a state occurred and completed within a summary period. A partial state within a summary period is ignored for an average calculation. (StateTimeAvgContained)
Minimum	Minimum time a state occurred and completed within a summary period. A partial state is ignored. (StateTimeMinContained)
Maximum	Maximum time a state occurred within a summary period. (StateTimeMax)

State Statistical Data	Description
Count	Number of times a state occurred and completed within a summary period. A partial state is not counted. (StateCountContained).
Percent	Percentage of the summary period that a state occurred. (StateTimePercent)
Total	Total time a state occurred within a summary period. (StateTimeTotal)

## Historical Summary Period

An Industrial graphic custom property can show historical statistics from Historian data over a defined summary period. During design time, you must specify the summary period to collect Historian summary data by entering values for the **Duration** and **StartTime** options shown on the **Edit Custom Properties** dialog box.

These assigned values are passed as input parameters of the new custom property. A value in minutes must be assigned to the **Duration** option.

The **StartTime** option can be left blank. Auto refresh is applied if a **StartTime** value is not specified.

- If a start time is not specified, then the start and end times of the summary period are calculated as:

Start Time = Current Time - Duration

End Time = Current Time

- If a start time is specified, then the start and end times of the summary period are calculated as:

Start Time = StartTime option value

End Time = StartTime + Duration

The **Duration** option can accept a negative number when a start time is specified. When **Duration** is assigned a negative number, the start time input parameter value becomes the end time of the summary period. The start time is calculated using the formula shown below:

End Time = StartTime assigned option value

Start Time = End Time + Duration (in this case it is negative value)

**Duration** can accept values from 1 minute to 10080 minutes, which is one week. **StartTime** must be within datetime Min and Max Value. During run time, history summary data is auto refreshed at an interval that is 25 percent of its duration length when a **StartTime** value is not specified.

## Showing Statistical Summary Data

The following procedure explains how to specify custom property options to subscribe to historical statistical data. Before completing this procedure, you must have object or other data that is saved to the Historian and enabled for statistics.

### To show historical summary data using custom properties

1. Open the Industrial Graphic Editor.
2. Add a custom property to an Industrial graphic.
  - a. Click the canvas to cancel any selected elements.

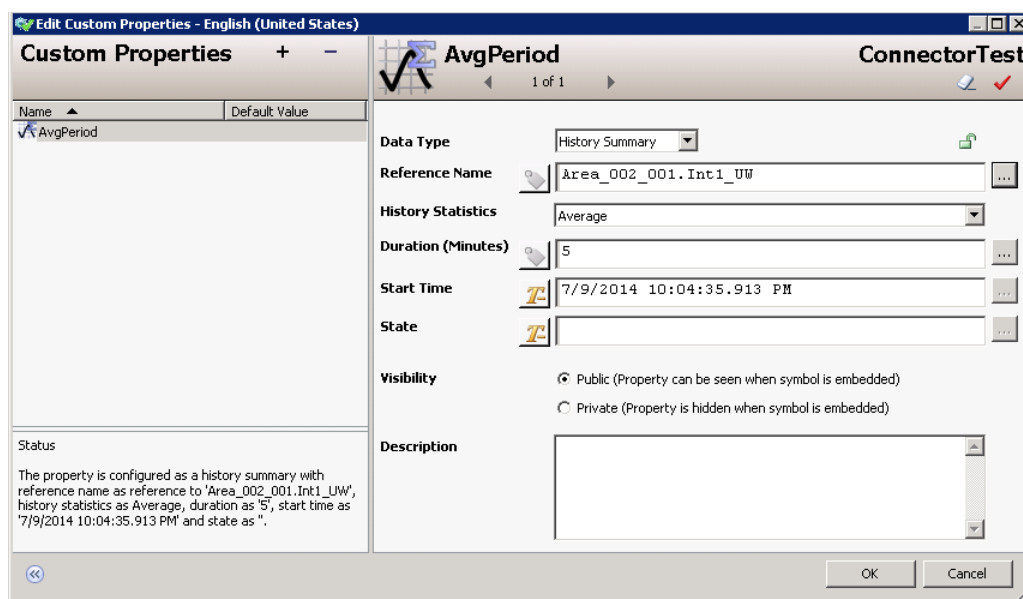
- b. On the **Special** menu, click **Custom Properties**. The **Edit Custom Properties** dialog box appears.
  - c. Click the **Add** icon. A new line is added in the custom properties list.
  - d. Type a name for the new historical summary custom property and click **Enter**.
  - e. You can see the names of the graphic and custom property in the header of the dialog box.
3. Select **HistorySummary** from the **Data Type** field.

---

**Important:** The History Summary data type only works with ArchestrA object attributes intended for AVEVA OMI ViewApps or InTouch HMI managed applications. Do not attempt to use custom properties assigned the History Summary data type in InTouch HMI Modern applications.

---

The **Edit Custom Properties** dialog box updates to show fields specific to the **HistorySummary** data type.



4. Enter a reference to data saved in the Historian in the **Reference Name** field.
- The icon to the left of the **Reference Name** field toggles input to the field as **Static Text** or **Expression or Reference** mode.

#### Auto-Detect

The Historian server is auto-detected from the AppEngine on which the reference attribute is running. For example, if the **Reference Name** field is set to UDO.UDA1, the reference is set to the Historian server name configured for the AppEngine on which UDO is running.

#### Expression

When an expression or reference is typed in the **Reference Name** field, a connection is made to the specified Historian Server. The reference can be an external reference like an object attribute or a custom property.

---

**Note:** A reference cannot be made to historical data from an InTouch tag.

---

5. Select the type of historical statistics by selecting an option from the drop-down list of **History Statistics**.

Average is the default type of historical statistic. The following table shows the historical statistics options for analog and state summary data.



Historical Statistics	Analog Historical Data	State Historical Data
Average		
First		
Minimum		
Maximum		
Count		
StdDev		
Integral		
PercentGood		
Percent		
Total		
Last		

- Set the length of the summary historical period in minutes by entering a value in the **Duration** field.

Acceptable **Duration** values are from 1 to 10080 and the default is 5. Duration can be specified as an integer, an expression, or a reference. For more information about possible **Duration** values, see *Historical Summary Period* on page 207.

- Set the start time of the of the summary period in the **StartTime** field.

A start time can be specified as static text, an expression, or a reference. The default start time is the current time.

A time for **StartTime** is optional and can be left blank. Auto refresh is applied if a **StartTime** value is not specified.

For more information about setting a start time, see *Historical Summary Period* on page 207.

- Set the type of Historian summary data in the **State** field.

A **State** value can be expressed as an integer constant, static text, an expression, or a reference.

If a string value is provided, then string state summary data is queried from the Historian. If an integer value is entered, the Historian query is for analog state summary data. If a Boolean state summary value is provided, the **State** value must be 0 or 1.

**State** can be left empty. If empty, the default query is for analog summary data.

To get summary historical data for a Null state, enter "NULL" in the **State** field. The query checks for OpcQuality equal to opcnull and StringValue "NULL" in the result.

## Using Binding in Custom Properties

ArchestrA object scripting supports a type called "Indirect". It enables you to bind a variable to a reference and read and write to it. This is done using the BindTo() method.

**Note:** The BindTo() method binds a variable to a reference as long as the graphic is shown.

For example, the local script variable ptr is defined and bound to the reference ud1\_001.Int1.

```
dim ptr as indirect;  
ptr.BindTo("ud1_001.Int1");
```

Within the same script you can use the indirect variable pointer to read from and write to the attribute ud1\_001.Int1.

Industrial Graphics also use scripting in the same way as the scripting of Application Server.

However, as an Industrial graphic can be embedded into an InTouch window and run anonymously, the time it takes to connect to the reference can be longer than one scan cycle.

For that reason, you cannot use the indirect variable immediately after it is bound to a reference to read from and write to it.

```
dim ptr as indirect;  
ptr.BindTo("ud1_001.Int1");  
ptr = 42;
```

In the example, the value 42 cannot be written to the reference ud1\_001.Int1 as the binding takes longer.

To avoid this problem, you can modify your Industrial graphic script, using the Industrial Graphic Script Editor to write the value after it is ensured that the binding is complete. The completion of the binding is indicated by the quality of the indirect variable.

Create a named script, with a trigger set to 'While True' and query for the quality and use the indirect variable to read from and write to the reference when its quality is good. This script will try again every second until the indirect attribute returns with good quality. If the quality is good, then the script exits from the while loop. Once the quality is confirmed, you can use the indirect variable reference in other scripts or button animations.

---

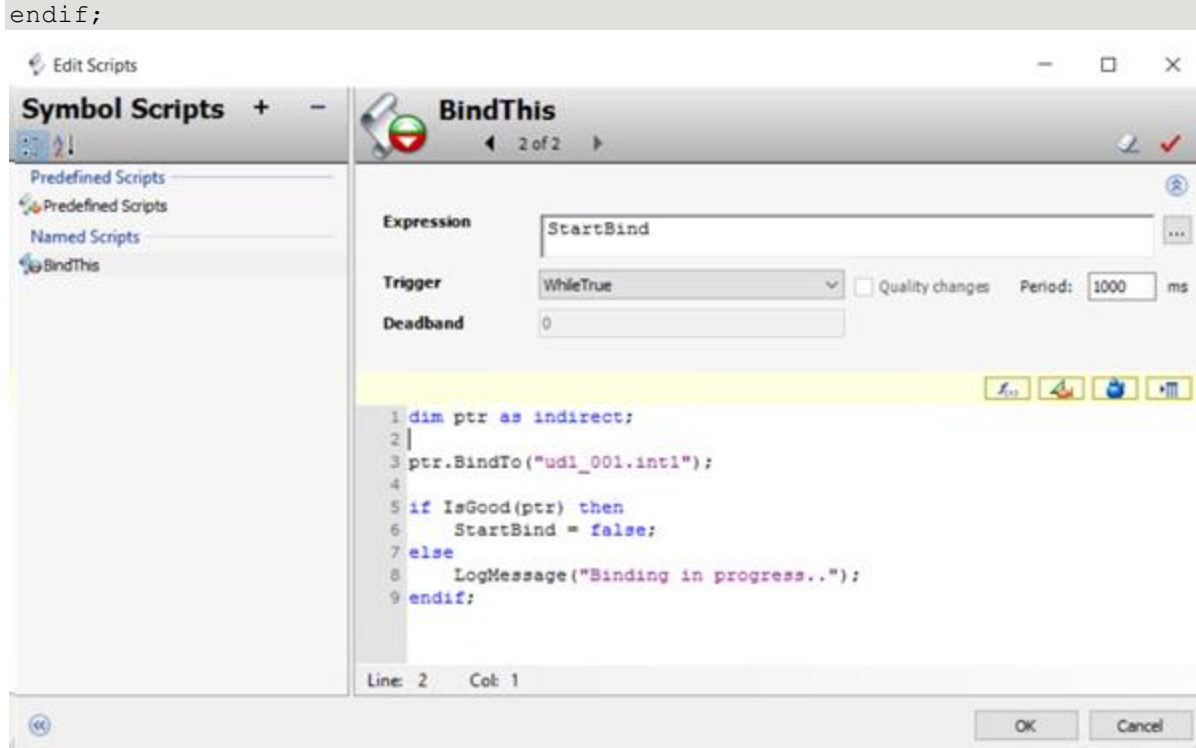
**Note:** Make sure to include an exit condition in your script, so that the script does not hang if the binding cannot be made.

---

The following example script shows you how to do this:

- Expression: Create a Boolean custom property StartBind with an initial value of True.
- Trigger: WhileTrue
- Period: 1000 ms

```
dim ptr as indirect;  
ptr.BindTo("ud1_001.Int1");  
if IsGood(ptr) then {if quality not good}  
    StartBind = false;  
else  
    LogMessage("Bidnding in progress..");
```



**Note:** Similar behavior can occur when you try to bind to a reference of an object that is hosted on a different AppEngine.

## Changing the Expression or Reference of a Custom Property at Run Time

You can change the expression or reference of a custom property at run time by calling the `SetCustomPropertyValue()` method on the graphic using a client script:

```
SetCustomPropertyValue(System.String name, System.String value, System.Boolean isConstant);
```

You can select this method using the Element Browser from within the Industrial Graphic Editor. This method is supported only for ArchestrA client scripts.

This method has three parameters:

- *name* - Name of the custom property to be modified on the graphic. This parameter is of type string, and it can be a reference or a constant.
- *value* - The new value to be set. This parameter is of type string, and it can be an expression, reference, or constant. If the value is given in quotes ("), then the value is considered a constant. If the value is given without quotes, then the value of the expression is considered a reference.
- *isConstant* - A flag that indicates whether the new value will be evaluated as a constant or a reference. This parameter is of type Boolean. If it is set to True (1), then the new value will be treated as a constant. If it is set to False (0), then the new value will be treated as a reference. This parameter only applies when the *value* parameter is a reference or constant and the custom property specified in the *name* parameter is a string or time type. This parameter has no meaning if the custom property is an integer, float, Boolean, or double type.

---

**Note:** The *isConstant* parameter does not override the type of input for the *value* parameter. The *value* parameter itself can be either a constant or a reference depending on whether it is enclosed in quotes. The *isConstant* parameter is only determining how the actual value (coming from the *value* parameter) is evaluated.

---

The whole expression or reference of the custom property is replaced with the new value, regardless if it is overridden or not. No partial replacement is supported.

Only public custom properties on the graphic can be changed.

When the method executes, it overrides any modifications done by previous `IOSetRemoteReference()` calls from a native `InTouch` script.

For an example of configuring the custom property as a reference, say you have a `Motor_001` object with the string field attribute `name State` that stores the current state of the motor ("Running" or "Stopped"). You also have an Industrial graphic that has the string data type custom property `MotorState`. The following script code will set the `MotorState` custom property to `Motor_001.State` in run-time:

```
GraphicA.SetCustomPropertyValue("MotorState", "Motor_001.State", False);
```

As a result of the call, the function will set the string custom property `GraphicA.MotorState` to "`Motor_001.State`" as a reference. The string custom property `GraphicA.MotorStatus` will resolve that reference and update its value with the reference value ("Running" or "Stopped").

For an example of configuring the custom property as a constant, say you have a `Motor_001` object with the Boolean field attribute `State` that reflects the current state of the motor (True or False). You also have an Industrial graphic that has the string data type custom property `MotorState`. The following script code causes the `MotorState` custom property to hold the state of equipment—"Running" or "Stopped"—as text based on the value returned for `Motor_001`:

```
IF Motor_001.State THEN
    GraphicA.SetCustomPropertyValue("MotorState", "Running", True);
ELSE
    GraphicA.SetCustomPropertyValue("MotorState", "Stopped", True);
ENDIF;
```

As a result of the call, the function will set the string custom property `GraphicA.MotorState` to "Running" or "Stopped," depending on the value of `Motor_001.State`.

## Working with the `SignedWrite()` Function for Secured and Verified Writes

This section provides information about the `SignedWrite()` function, `SignedWrite()` run-time behavior, scripting tips, and in-depth script examples.

For `SignedWrite()` scripting information including script syntax, parameters, and basic script examples, see the *Application Server Scripting Guide*.

You can write to an Automation Object attribute that is configured for Secured Write or Verified Write security classification by means of the Industrial Graphics `SignedWrite()` script function.

The `SignedWrite()` function can be used only in ArchestrA client scripts, not in Application Object scripts, and only on Attributes that have been configured for Secured Write or Verified Write. Attempting to use the function on an Attribute not so configured will result in an error message at run time.

### SignedWrite() Run-time Behavior

At run time, the `SignedWrite()` function does the following:

1. Checks the target Attribute for Signed Write or Verified Write configuration.

If not so configured, the following error message appears: **Operation Failed. Attribute does not have the correct security classification.**

2. Checks Galaxy security.

If the Galaxy is not secured, the following error message appears: **Operation Failed. Galaxy is not secured.**

3. There are several ways to write to an Attribute configured with Secured Write or Verified Write security classification, it is possible to have multiple SignedWrite() and other Secured/Verified writes pending from the same script, or even from multiple scripts running side-by-side.
4. Determines which dialog is required—Secured Write or Verified Write—and pops up the appropriate dialog box.

If Smart Cards are enabled, the function displays different versions of the Secured Write and Verified write dialog boxes.

5. Lists the predefined comments, if any, from the configured Predefined Comments list. Up to 20 comments are supported.
6. Enables comment editing if the Comment\_Is Editable parameter is configured and comment enforcement is other than PredefinedOnly.
7. Acquires the user credentials and authenticates them.

If the user credentials are invalid, an error message appears. The function will attempt the write only if the credentials are valid.

8. Checks if comment enforcement is mandatory, and displays an error message if the comment is empty.
9. Performs the write if user credentials are valid and the comment entry satisfies the comment enforcement parameter.
10. Provides a return status.

11. Following a Secured or Verified Write a security Event is written to the event log, including the signee name, verifier name, if any, Type of write: "Secured Write" or "Verified Write", Comment, if any entered by user, Reason Description, if any provided, Field Attribute description, if any, or the Short Description of the Application Object, if no Field Attribute description exists.

Each call to SignedWrite() is distinct from any other. The success or failure of any individual write does not affect other attempted writes.

Entering user credentials for SignedWrite() is distinct from logging on to the client application. The user can modify attributes configured with Secured or Verified Write even if another user is logged on, without affecting the session of the logged-on user.

## SignedWrite() Script Execution Sequence at Run Time

The SignedWrite() function goes into a queue and the script continues executing. The function is queued for operator entry. The script may complete prior to the operator completing the Secured or Verified Write operation.

By contrast, the SignedAlarmAck() script function executes completely synchronously, and waits for user input before proceeding to the next line in the script.

## SignedWrite() Scripting Tips

### Using Bound References in SignedWrite()

If the Attribute parameter string evaluates to the name of a Custom Property and that Custom Property is a bound reference to an Attribute, the SignedWrite() function will write to that indicated Attribute. The Attribute must have the security classification of Secured Write or Verified Write.

The SignedWrite() function supports Custom Properties that are nested bound references. That is, if the string evaluates to the name of a Custom Property and that Custom Property is a bound reference to another Custom Property which itself is a bound reference, the SignedWrite() function will follow through the chain of bound references until it finds an item that is a value. If that item is an Attribute that has the security classification of Secured Write or Verified Write, the SignedWrite() function will write to that item.

**Using SignedWrite() in WhileTrue, WhileFalse, or Periodic Type Scripts**

Using the SignedWrite() function with WhileTrue, WhileFalse, or Periodic type scripts can repeatedly execute the script, causing another secured write dialog box to pop up with each trigger. We do not recommend using the SignedWrite() function with WhileTrue, WhileFalse, or Periodic types.

**Using SignedWrite() with OnShow and OnHide Scripts**

We do not recommend using the SignedWrite() function with OnShow and OnHide scripts. This can cause issues with window functionality, including the window title bar, windows losing correct focus, and windows opening on top of one another.

## Examples of Using the Attribute Parameter in the SignedWrite() Function

Working from the overall syntax of the SignedWrite() function, the script examples in the following table illustrate a number of approaches to using the Attribute parameter in the SignedWrite() function.

The following String, Boolean, and Integer user defined attribute conditions apply to the script examples:

- User Defined object UD1\_001
- String attribute UD1\_001.udString1 with the value "WW". Security classification is set to Secured Write.
- Boolean attribute UD1\_001.secBool1 with the value false. Security classification is set to Secured Write.
- Integer attribute UD1\_001.seclnt1 with the value 24. Security classification is set to Secured Write.
- User Defined object UD2\_002

String attribute UD2\_002.udString2 with a value "UD1\_001.udString1" The following custom property conditions apply to the script examples:

- String custom property CP1 with a reference to UD1\_001.udString1
- String custom property CP2 with a value "UD1\_001.udString1"
- String custom property CP3 with a value "UD1\_001"
- Boolean custom property CP4 with a reference to UD1\_001.secBool1
- Integer custom property CP5 with a reference to UD1\_001.seclnt1
- String custom property CP6 with a reference to an attribute on Owing Object UD1\_001 as me.udString1

Script Example	Function and Result
<pre>SignedWrite("CP1", "AVEVA", "using redirect", true, 0, null);</pre>	<p>Uses the CP1 reference UD1_001.udString1 and pokes to it the value "AVEVA".</p> <p>Result: The value in UD1_001.udString1 will change from "WW" to "AVEVA".</p>

Script Example	Function and Result
<pre>SignedWrite(CP2, "AVEVA", "using string value", true, 0, null);</pre>	<p>Resolves CP2 string value "UD1_001.udString1" to a reference and pokes to it the value "AVEVA".</p> <p>Result: The value in UD1_001.udString1 will change from "WW" to "AVEVA".</p>
<pre>SignedWrite(CP3+".udString1", "AVEVA", "using string expression", true, 0, null);</pre>	<p>Resolves the string "UD1_001.udString1" to a reference and pokes to it the value "AVEVA".</p> <p>Result: The value in UD1_001.udString1 will change from "WW" to "AVEVA".</p>
<pre>SignedWrite("UD1_001.udString1", "AVEVA", "using constant string", true, 0, null);</pre>	<p>Resolves the string "UD1_001.udString1" to a reference and pokes to it the value "AVEVA".</p> <p>Result: The value in UD1_001.udString1 will change from "WW" to "AVEVA".</p>
<pre>SignedWrite(UD2_002.udString2, "AVEVA", "using attribute containing string", true, 0, null);</pre>	<p>Resolves the UD2_002.udString2 string value "UD1_001.udString1" to a reference and pokes to it the value "AVEVA".</p> <p>Result: The value in UD1_001.udString1 will change from "WW" to "AVEVA".</p>
<pre>SignedWrite("CP" + "1", "AVEVA", "using redirect from string expression", true, 0, null);</pre>	<p>Resolves the expression to "CP1" to use the CP1 reference UD1_001.udString1 and pokes to it the value "AVEVA".</p> <p>Result: The value in UD1_001.udString1 will change from "WW" to "AVEVA".</p>
<pre>SignedWrite("CP4", true, "using redirect", true, 0, null);</pre>	<p>Uses the CP4 reference UD1_001.secBool1 and pokes to it the value true.</p> <p>Result: The value in UD1_001.secBool1 will change from false to true</p>
<pre>SignedWrite("CP5", 37, "using redirect", true, 0, null);</pre>	<p>Uses the CP5 reference UD1_001.secInt1 and pokes to it the value 37.</p> <p>Result: The value in UD1_001.secInt1 will change from 24 to 37</p>
<pre>SignedWrite("CP6", "AVEVA", "using redirect using relative reference", true, 0, null);</pre>	<p>Uses the CP6 reference me.udString1 and resolves it to UD1_001.udString1 and pokes to it the value "AVEVA".</p> <p>Result: That the value in UD1_001.udString1 will change from "WW" to "AVEVA"</p>

## Secured and Verified Write Applied Examples

You can create a dashboard application to automate routine use of Secured and Verified Write by means of the SignedWrite() function.

## To configure the SignedWrite() script function

1. Open the System Platform IDE.
2. Create a symbol and associate it with an attribute configured with Secured Write or Verified Write. For more information on associating attributes with symbols, see Application Server User's Guide, "Creating and Working with UDAs" topic.
3. Add the SignedWrite script function to the symbol. The following editor detail shows the buttons configured with scripts in the applied example:
4. Configure the scripted functionality you require. Scripts for the buttons shown in the example are as follows:

- a. Hard-coded DataUDO.SecUDA: The following example sets the value of 23 to DataUDO.SecUDA. The user optionally can enter a comment, but no pre-defined comment list is available.

```
DataUDO.RetStatus=SignedWrite("DataUDO.SecUDA", 23, "Set the Value",
True, 0, null);
```

- b. Attribute Pointer has DataUDO.SecUDA: The source to be written to is passed as a parameter to the function. Attribute\_Pointer is a custom property whose value is set to DataUDO.SecUDA. The following example sets the value of 23 to DataUDO.SecUDA. The user optionally can enter a comment, but no pre-defined comment list is available.

```
DataUDO.RetStatus=SignedWrite(Attribute_Pointer, 23, "Set the Value",
True, 0, null);
```

- c. Attribute Pointer and Pre-Defined List: The pre-defined comment list is an array. This example extends the functionality of example b to force the user to enter a comment (Comment\_Enforcement parameter set to 1) and also presents a pre-defined set of comments linked to the DataUDO.PreDefComments[ ] array.

The following example will set the value of 23 to DataUDO.SecUDA. The user must enter a comment and may use one from the pre-defined comment list.

```
DataUDO.RetStatus=SignedWrite(Attribute_Pointer, 23, "Set the Value",
True, 1, DataUDO.PreDefComments[]);
```

- d. Variable Array: The pre-defined list is a pointer to an array. This example extends the functionality of example c to force the user to enter a comment (Comment\_Enforcement parameter set to 1) and also presents a predefined set of comments linked to DataUDO.PreDefComments[ ] array.

The value of custom property CP1 is "DataUDO.PreDefComments[ ]".

The following example will set the value of 23 to DataUDO.SecUDA. The user must enter a comment and may use one from the pre-defined comment list.

```
dim xInd as Indirect;
xInd.BindTo(CP1);
DataUDO.RetStatus=SignedWrite(Attribute_Pointer, 23, "Set the Value",
True, 1, xInd);
```

- e. All Parameters Variable: The predefined list array is built into the script. All parameters are passed as variables.

The following example will set the value of 23 to DataUDO.SecUDA . The user must enter a comment and may use one from the pre-defined comments list.

```
dim MyList[5] as string;
MyList[1] = "Batch Accepted";
```



```

MyList[2] = "Batch Rejected";
MyList[3] = "Batch on Hold";
MyList[4] = "Batch Resumed";
MyList[5] = DataUDO.PreDefComments[4];

DataUDO.RetStatus=SignedWrite(Attribute_Pointer,
SignedWrite_Value_Ptr, SignedWrite_Reason, Enable_Edit_Comment,
Comment_Options, MyList[]);

```

## Working with the Show/Hide Content Script Functions

### About the ShowContent() Function

---

**Note:** The Show/Hide Content() script functions are for use with AVEVA OMI ViewApps only. These functions cannot be used with InTouch HMI applications.

---

The Show/Hide Content() script functions let you write scripts for AVEVA OMI layouts, Industrial Graphics, Application Server objects to display a graphic, layout, or external content item (for example, a web page, video, or text file) inside a specific pane within an AVEVA OMI ViewApp. The Show/Hide Content() script functions are complementary to the Show/Hide Graphic script functions and the Show/Hide Symbol animation feature.

- Use the Show/Hide Graphic script functions to load or hide a graphic in a modal or modeless popup, in an InTouch HMI application or an AVEVA OMI ViewApp.
- Use the Show/Hide Content script functions to load or hide a graphic, layout, or an external content item into a pane in an AVEVA OMI ViewApp. When a Layout linked to an asset is added to a pane in another layout, any empty panes in the linked Layout that have the "Use for Autofill" flag enabled will attempt to fill the empty panes with content from the linked asset, using the "Current Only" autofill setting.

The Show/Hide Content functions let you specify:

- The specific content to show or hide in a pane.
- The pane in which to show or hide the content.
- The screen that contains a pane in which to show or hide the content.
- The type of content you want to show or hide.
- Any property overrides to apply when showing the content.
- Whether to confine the search for a pane in which to show or hide the content to the source screen, the primary screen, or to search all screens, and whether to search in the top level layout or a nested (embedded) layout.

### Configuring the Show/Hide Content Script Functions

Use the Show/Hide Content functions in layout scripts, Industrial Graphic scripts, or Application Server object scripts to populate a pane with a specific graphic, layout, or external content object in an AVEVA OMI ViewApp. You can call the same script multiple times to refresh content, or to show the content in a different pane. If you choose to display the same content in different panes, you can alter its settings through the ParameterOverride parameter.

If ShowContent calls the same content item while it is open, ShowContent searches for open content items that match the specified parameters in the call. Any matching items are closed and then reopened. If the content has changed, the latest changes are shown when the content is redisplayed.

---

**Important:** The Show/HideContent functions can be used in a graphic or layout action script, named script and pre-defined script. Although the system allows you to include it in a server script, such as Start Up, On Scan, Off Scan, Shut Down and Execute, you will not be able to execute the function at run time.

---

**To include a script that contains the Show/Hide Content functions within a script**

1. Open the System Platform IDE.
2. Open a new or existing graphic or layout.
  - To add a layout script, click the **Script** tab.
  - To add a graphic script, create a graphic, and then double-click it to open the **Edit Animations** page, then open the script editor and click the **Display Script Function Browser** icon. The **Script Function Browser** appears. Click the ShowContent() script function in the **Graphic Client** list, and then click **OK**. The following code snippet is added:

```
Dim contentInfo as aaContent.ContentInfo;
contentInfo.Content = "<ContentName>";
ShowContent( contentInfo );
```

---

**Note:** You can click **Help** to view the Help after you have selected any Graphic Client script function.

---

3. Modify the script as needed. ContentInfo is a predefined structure that contains data members listed below. Content is the only required parameter. For more information about the Show/Hide Content, see the AVEVA Scripting Guide.
  - **Content:** name of the content (graphic, layout, or external content object) to be loaded into the pane. Content is a string and is the only required parameter.
  - **Name:** name property of the content. This is automatically created when the content is added to the layout.
  - **ScreenName:** name of the screen that contains the pane in which to load the content. ScreenName is a string.
  - **SearchScope:** when ScreenName is not specified, **SearchScope** determines which screens within the Screen Profile are searched for a pane in which to place the content. SearchScope is an enum that can be set to one of the following values:
    - **Self:** This is the default. Searches for matching content within the panes of the layout from which the ShowContent call was made. Self will search from a nested (embedded) layout, if the call was made from that layout. The remaining SearchScopes search ONLY the top level if there are nested layouts.
    - **AllScreens:** Searches the top level layout on all screens, starting with the source screen, then the primary screen, and then any remaining screens in alphabetical order. The search stops as soon as a matching pane is found. If a matching pane is not found, Content is placed in the default pane of the source screen.
    - **SourceScreen:** Searches the top level layout only in the source screen (the screen that made the ShowContent call).
    - **PrimaryScreen:** Searches the top level layout only in the primary screen, as designated in the Screen Profile.
  - **PaneName:** name of the pane in which to load the content. PaneName is a string.
  - **ContentType:** specifies the type of content to be loaded. The ContentType can be matched against the Content Type designation that was set for a pane. ContentType is a string.
  - **PropertyOverrides:** specifies overrides for custom properties. This parameter is only valid if a graphic is the designated content. It is not valid if a layout has been designated. PropertyOverrides is a ValuePair structure.

- **OwningObject:** specifies an automation object as an owning object.

### Example

```
Dim contentInfo as aaContent.ContentInfo;

Dim cpValues [2] as aaContent.PropertyOverrideValue;
cpValues[1] = new aaContent.PropertyOverrideValue("CP1", "20", true);
cpValues[2] = new aaContent.PropertyOverrideValue("CP2", "Pump.PV.TagName",
false);

contentInfo.Content = "S1";
contentInfo.ContentType = "Overview";
contentInfo.OwningObject = "Enterprise";
contentInfo.PaneName = "Panel";
contentInfo.ScreenName = "Wall";
contentInfo.PropertyOverrideValues = cpValues;

contentInfo.SearchScope = aaContent.SearchScope.PrimaryScreen;

ShowContent ( contentInfo );
```

## Best Pane Match Algorithm

When some or all of the optional parameters are not provided or are invalid, ShowContent follows a set of rules to determine in which pane the specified content should be placed. In addition, the SearchScope parameter can be used to limit which screens will be searched for a pane.

- If SearchScope is set to SourceScreen, only the screen that made the ShowContent call is searched. If no matching pane is found, Content is loaded into the default pane of the SourceScreen.
- If SearchScope is set to PrimaryScreen, only the primary screen in the Screen Profile is searched. If no matching pane is found, Content is loaded into the default pane of the PrimaryScreen.
- If SearchScope is set to AllScreens, every screen contained in the Screen Profile can be searched. This is the default behavior. Searches progress as follows and stop at the first match:
  - SourceScreen
  - PrimaryScreen
  - Any additional screens are searched in alphabetical order.
  - If no matching pane is found, Content is loaded into the default pane of the SourceScreen.
- If SearchScope and ScreenName are not specified and ShowContent is called from an embedded layout:
  - ShowContent searches locally for a matching pane within the embedded layout.
- If SearchScope or ScreenName is specified and ShowContent is called from an embedded layout:
  - ShowContent searches for a pane globally and uses the SearchScope and/or ScreenName settings to determine which screens within the Screen Profile associated with the ViewApp will be searched.

### Order of Precedence for Determining Content Placement

Content is the only required parameter. Three values are also checked according to the following order of precedence to determine placement of the content:

1. PaneName: optional parameter
2. ContentType: optional parameter

3. Type of content: this is part of the Content definition and is not specified in the ShowContent call.

**Search Steps (SearchScope not specified or set to AllScreens)**

**Scenario 1: PaneName** Assumes SearchScope not set or set to AllScreens. Otherwise, search is confined to the named screen.  
 PaneName is specified

1. Search SourceScreen for PaneName.
2. If not found, search PrimaryScreen for PaneName.
3. If not found, search any additional screens in alphabetical order for PaneName.

The search stops as soon as a matching PaneName is found.

**Scenario 2: ContentType** Assumes SearchScope not set or set to AllScreens. Otherwise, search is confined to the named screen.  
 PaneName is not specified OR matching PaneName not found AND ContentType is specified

1. Search SourceScreen for a pane that supports the ContentType.
2. If not found, search PrimaryScreen for pane that supports the ContentType.
3. If not found, search any additional screens in alphabetical order for a pane that supports the ContentType.

The search stops as soon as a pane that supports the ContentType is found.

**Scenario 3: Type of content** Assumes SearchScope not set or set to AllScreens. Otherwise, search is confined to the named screen.  
 PaneName is not specified OR matching PaneName not found AND ContentType not specified or not found

1. Evaluate type of content for the specified Content.
2. Search SourceScreen for a pane that supports the type of content.
3. If not found, search PrimaryScreen for pane that supports the type of content.
4. If not found, search any additional screens in alphabetical order for a pane that supports the type of content.

The search stops as soon as a pane that supports the type of content is found.

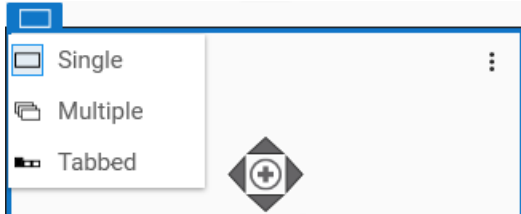
**Scenario 4: No matching pane** When SearchScope is not set or set to AllScreens:  
 If no match is found using PaneName, ContentType, or type of content, Content is loaded into the default pane of the SourceScreen.  
 No matches found from previous searches (scenarios 1 through 3)

When SearchScope is set to SourceScreen:  
 Only the SourceScreen is searched and if no match is found, Content is loaded into the default pane of the SourceScreen.

When SearchScope is set to PrimaryScreen:  
 Only the PrimaryScreen is searched, and if no match is found, Content is loaded into the default pane of the PrimaryScreen.

## Content Display Rules

The way content is displayed differs slightly between panes that are configured as single content panes versus those configured as multi-content panes. The differences are described below. Pane configuration (single or multi-content) is set by opening the **Presentation Style** selector at the top left of the pane in the **Layout Editor**. See *Set the Presentation Style of a Pane* in the *System Platform Help* for additional information about single and multiple content modes.



If there are multiple ShowContent() calls, but with different parameters, each ShowContent() call opens a different pane or tab. The content in each pane or tab is configured with the parameters that are specified in the call.

### Content Display Rules – Single Content Pane

If the pane already contains content, the existing content is closed and the content specified by ShowContent() then is loaded into the pane. This occurs even if the existing content is the same as the specified content.

### Content Display Rules – Multi-Content or Tabbed Pane

For panes that are configured as multiple-content or tabbed panes, the exact behavior will depend on the structure of the pane.

- If the specified Content is already open in the pane or a tab, the pane or tab remains open and the existing Content is replaced with a new instance of the same Content.
- If the pane or tab has been split (that is, contains child panes), or contains a nested layout, existing panes and/or tabs remain open, and the specified content is loaded into a new tab or pane and the focus is set to the new tab or pane.
  - This is true as long as the number of tabs or child panes is less than the maximum available. The default setting allows up to 20 panes or tabs, but you can increase this to 50.
  - If the maximum number of tabs or child panes has been reached, the pane that has been updated last (oldest content) is closed and a new tab or pane is opened in its place.




# CHAPTER 6

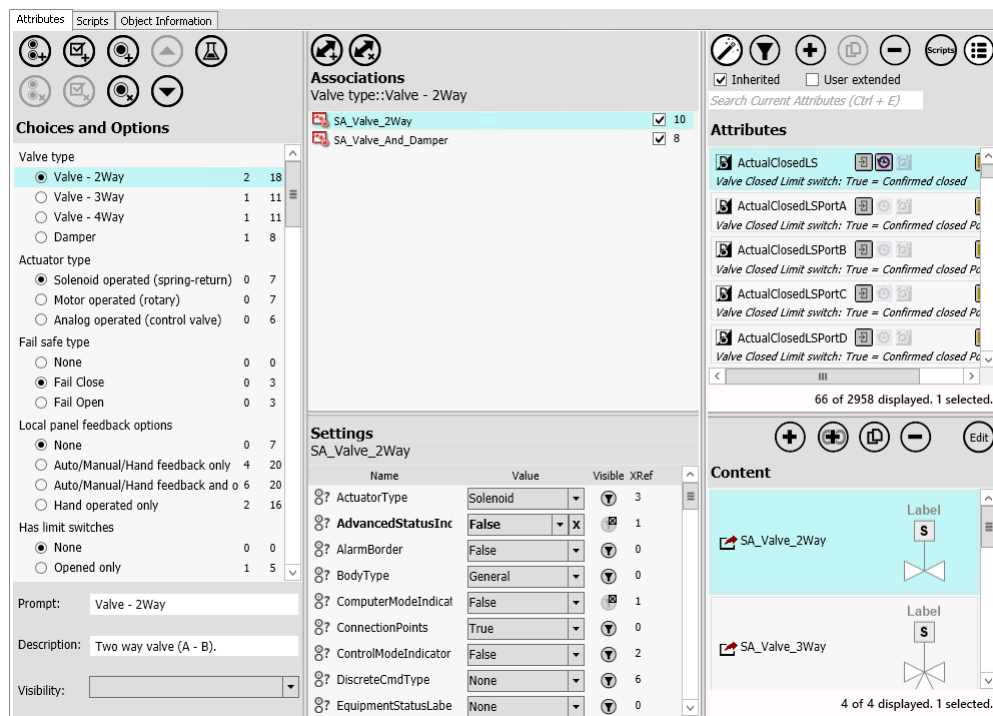
## Using Object Wizards

### About Object Wizards

Object Wizards consist of a series of user-selectable choices and options that are used to customize a deployable instance. Each choice and option may have one or more attributes, symbols, other content types and/or scripts associated with it.

An Object Wizard can be added to any derived template, and provides a simplified user interface for configuring instances (assets) from the template. To access the Object Wizard editor:

1. Open a template in the **Object Editor**.
2. If the template does not open to the **Attributes** page by default, select it now.
3. If necessary, select the **Object Wizard button**  to open the Object Wizard editor. The Attributes tab divides into multiple panes that are used for building the Object Wizard.



Object Wizards can be used either within the IDE, or from the **Configure New Asset** editor that also lets you configure an associated graphic as you configure the object.

**Note:** The **Configure New Asset** editor can only be used if 1) the template contains a symbol, and 2) your security setting allows you to edit graphics. It cannot be used with other content types, such as layouts and external content.

A single template with an Object Wizard can replace a number of derived templates for configuring a variety of similar instances. You can add an Object Wizard to any derived template.

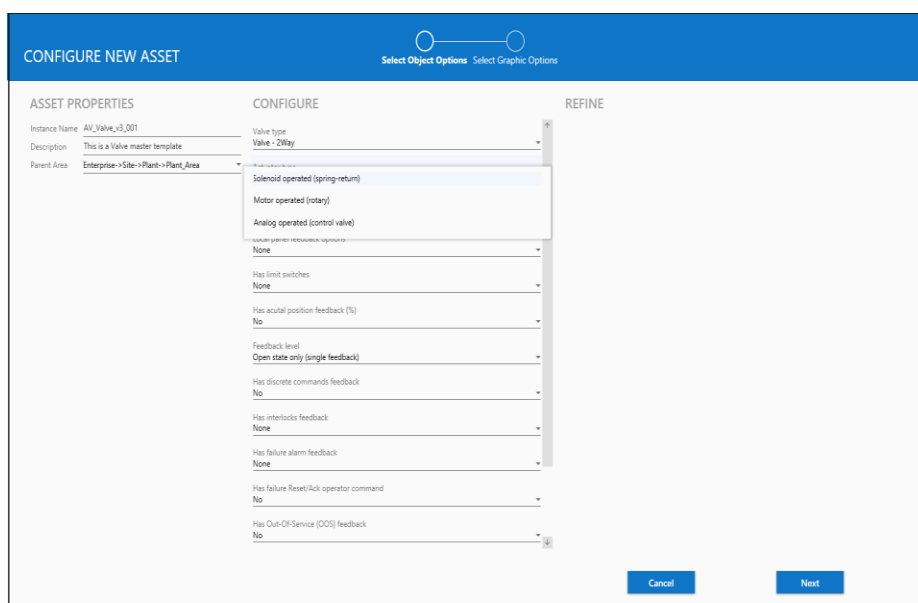
Depending on their level of permissions, users can:

- Create and configure instances from within the IDE, and add them to the Galaxy, or modify existing instances. A user simply opens an instance and answers a series of prompts or questions. See *Configure Instances* on page 286 for additional information.
- Use the **Configure New Asset** editor to create and configure instances and a representative symbol, simply by dragging an associated (linked or embedded) symbol into a graphic, and then answering a series of prompts or questions. See *Create a New Instance Graphically* on page 290 for additional information.

**Note:** The drag and drop method is only valid for symbols, and not for other linked content types such as layouts and external content.

- For more information about linking to a symbol, see *Link to Shared Content in the Graphic Toolbox* on page 236.
- For more information about embedding a symbol, see *Add a New Object-Owned Symbol* on page 235.

**Important!** InTouch HMI does not support directly adding assets to an InTouch HMI window if the asset is derived from an object wizard with a linked symbol wizard, such as the symbols in the Situational Awareness Library. **With InTouch HMI, the symbol wizard must be contained in an overview symbol** when deriving assets from an object wizard. In contrast, AVEVA OMI does support the the use of linked symbol wizards when adding assets to AVEVA OMI layouts from an object wizard.



## Object Wizard Features

An Object Wizard contains choices and options that guide users through the process of configuring instances. You can replace many derived templates by using a single template with an Object Wizard, since the Object Wizard can contain a wide variety of configuration possibilities. Attributes and features contained in the Object Wizard that are not needed in a particular instance are removed and are not a part of the run-time object. Finally, and perhaps most importantly, an Object Wizard is easy to use and can reduce user training and system maintenance costs.

Object Wizards provide the following benefits:

- Simplified workflow for configuring instances
- Automatic propagation of changes, without having to lock templates



- Unnecessary attributes can be trimmed from instances to reduce the footprint of deployed objects
- Unused symbols can be hidden from users to reduce clutter
- Reduced system maintenance
- Reduced training requirements for plant personnel

## Create a Basic Object Wizard


An Object Wizard can include Choice Groups, each with two or more Choices, and/or Options. A Choice Group is a radio button list that lets the user select one choice. An Option contains a checkbox that a user can select or deselect. To add an Object Wizard to a template:

1. In the IDE, open a derived template in the **Object Editor**.
2. If the **Attributes page** is not already selected, select it now. See *Create Derived Templates* on page 57.

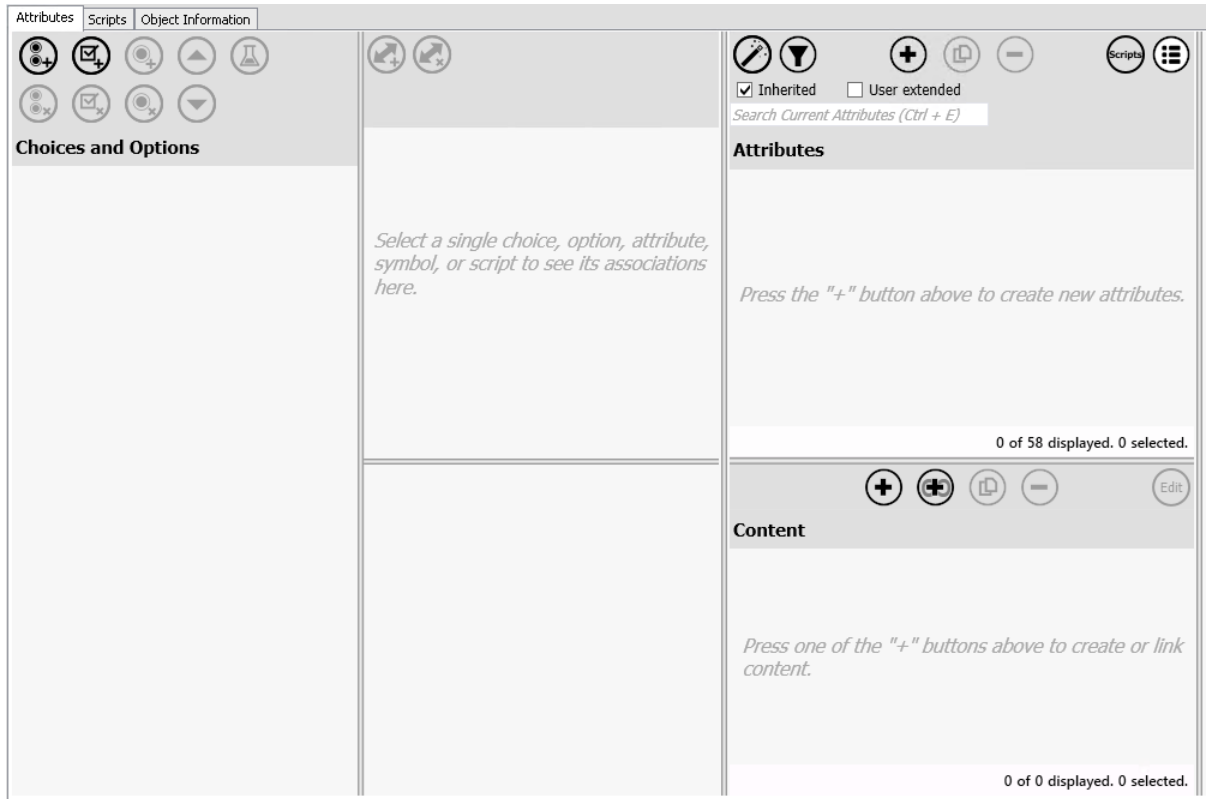
---

**Note:** Only templates derived from the \$UserDefined base template open in the **Object Editor** with the **Attributes page** selected. The Object Editor always opens the first page of the object. For example, the **Object Editor** opens the **General page** of the \$AnalogDevice template.

---

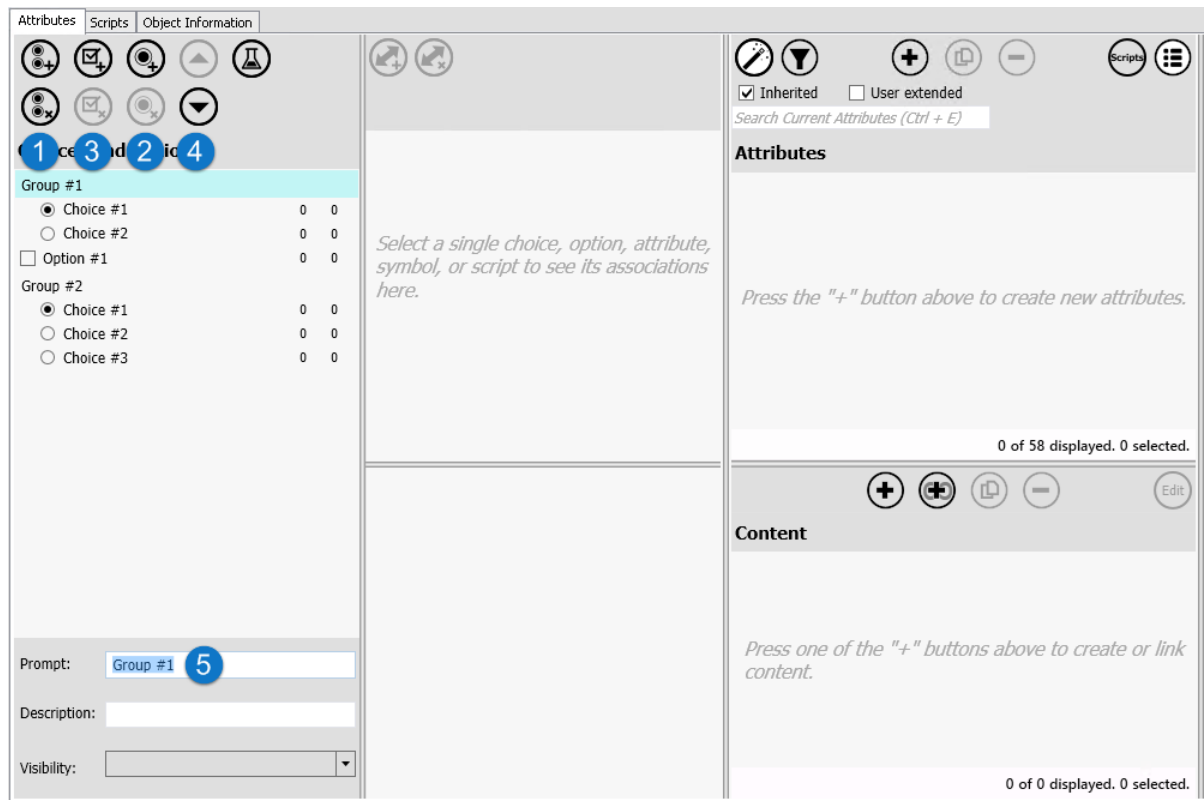
3. Configure the template by adding:
  - Attributes
  - Symbols or other content such as links to documents and videos
  - ScriptsSee Add Attributes, Symbols and Scripts to an Object Wizard for additional information.
4. If the Object Wizard editor is not already open, select the **Wizard**  button.

With Object Wizard editor open, the IDE should look like this:



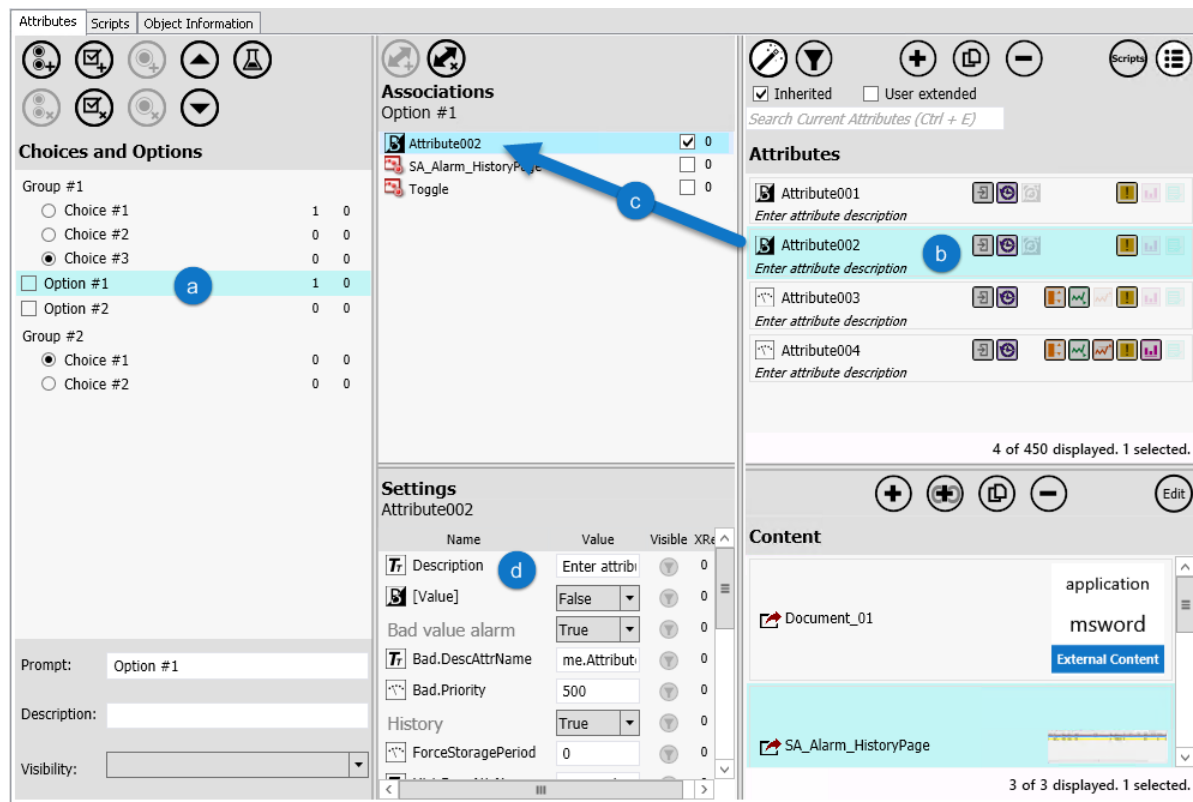
See *Object Wizard GUI Panes* on page 229 for additional information.

5. Add a Choice Group by selecting the **Add Choice Group (1)** button. With the Group name selected, change the name of the Group shown in the **Prompt (5)** text field. See Add a Choice Group for more information.



6. To add more Choices to a Choice Group, select the Choice Group and then press the **Add Choice (2)** button. To change the name of a Choice, select the Choice and enter the new name in the **Prompt** field. See Add a Choice to a Choice Group for more information.
7. Add Options by selecting the **Add Option (3)** button. Change Option names the same way that you change Choice Group and Choice names. See Add an Option for more information.
8. Use the **Up and Down arrow (4)** buttons to change the order of Choice Groups, Choices (within a Choice Group), or Options.
9. To associate an attribute or content with a Choice or Option:
  - a. Select the Choice or Option.
  - b. Select the attribute or content (symbol or other type of content) you want to associate with the Choice or Option.
  - c. Select the **Add Association** button (or drag the attribute or content to the **Associations** pane).
  - d. Settings for the selected attribute or content are shown in the **Settings** pane, once the association is made. Configure visibility and value overrides, as needed.

**Note:** No settings are shown for symbols that do not contain custom properties or a Symbol Wizard.



## Object Wizard Configuration Best Practices

An Object Wizard can be added to any derived template. However, to maximize the benefits of using Object Wizards, add your Object Wizard to the template as close to the top level of the derivation hierarchy as possible. To build an Object Wizard you must:

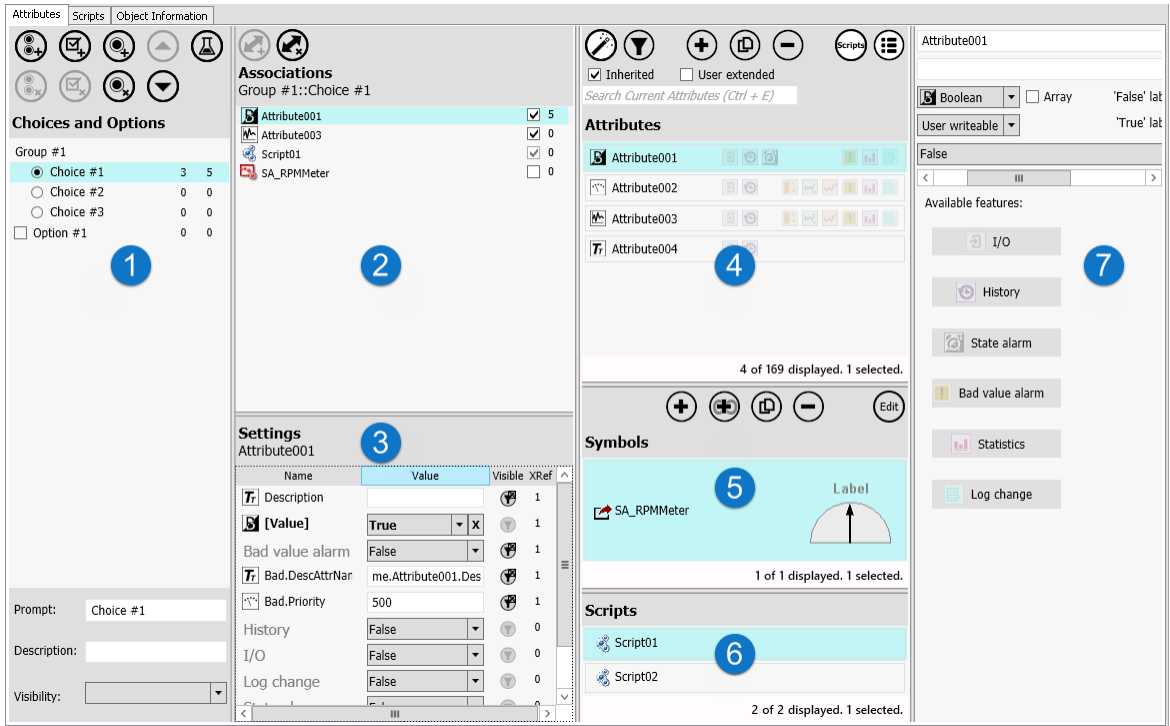
- Add and configure attributes, symbols, links to external content, and scripts before you build the Object Wizard.
- An Object Wizard requires at least one Choice Group or Option. See *Object Wizard Components* on page 234 for additional information.
- Configure Choices and Options by associating attributes, symbols/content, and scripts with Choices and Options. We recommend that you limit the number of items you associate with each Choice or Option to avoid configuration mismatches.
- When configuring Choices and Options, keep the workflow as simple and direct as possible. Remember that users will be working sequentially through the Object Wizard. If you enable a feature at one level of the Wizard hierarchy, you cannot disable the same feature at a subsequent level.

For example, if a symbol has Custom Property "X" enabled and is associated with a Choice, do not associate the same symbol with "X" disabled to an Option further down in the hierarchy. Instead, leave "X" disabled and override the setting when deriving an instance.



- Configure the *Conditional Visibility Expressions for Choice Groups, Choices, and Options* on page 244 of Choices and Options in derived instances.
- Configure settings for the associated attributes and symbols. Scripts and external content do not have settings that can be configured.

- Select whether each attribute, symbol, and other content can be *Trimming* on page 254 from derived instances. Scripts are always configured as trimmable.

## Object Wizard GUI Panes



The user interface for configuring Object Wizards is accessed through the **Attributes** page within the Object Editor of the Integrated Development Environment (IDE).

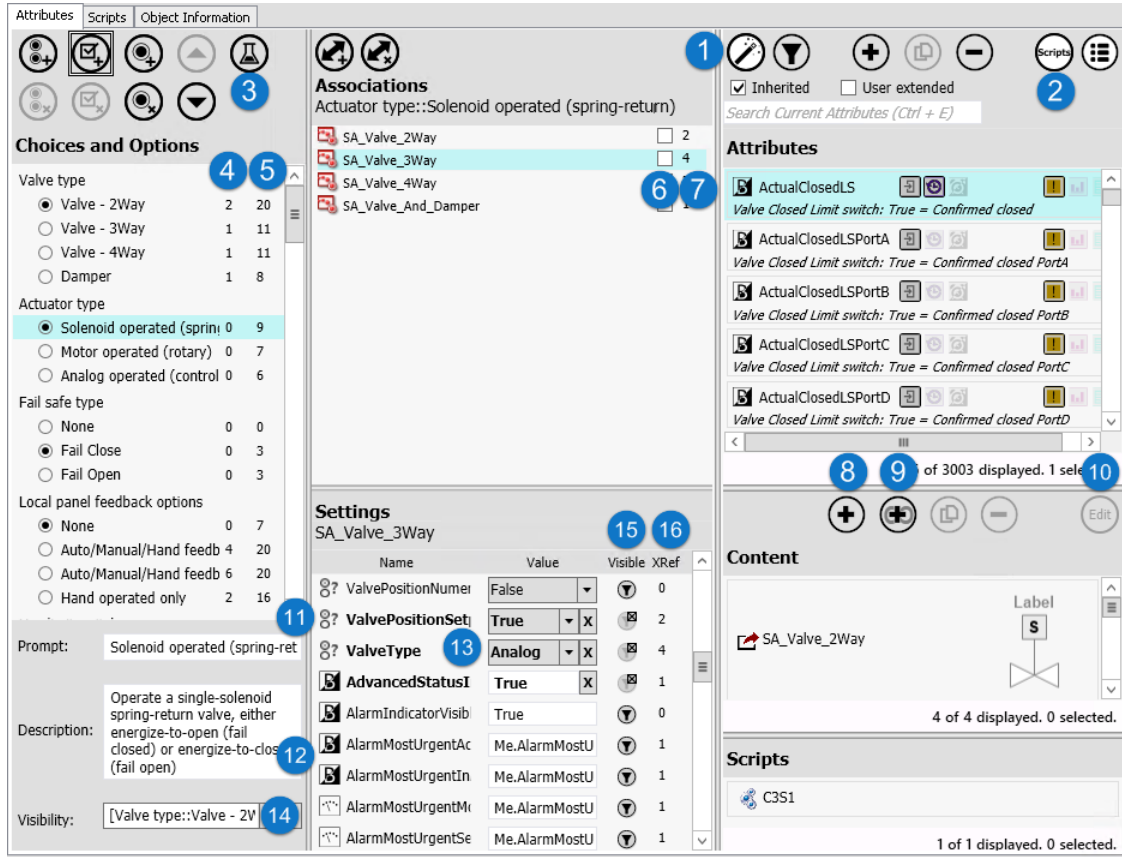
	<p>Select the <b>Object Wizard</b> button to open the Object Wizard user interface. The <b>Attributes</b> page divides into multiple panes for use in creating and editing an Object Wizard. See <i>Object Wizard User Interface Details</i> on page 231 for additional information.</p>
	<p>Select the <b>Scripts</b> button (located at the top right of the <b>Attributes</b> pane) to open the <b>Scripts</b> pane.</p>

Ref	Name	Description
1	<b>Choices and Options pane</b>	Add, rename, reorder, and delete Choice Groups, Choices, and Options that comprise the Object Wizard in the <b>Choices and Options</b> pane (1). You can also set the logic for displaying Choices and Options.
2	<b>Associations pane</b>	<p>Drag and drop attributes, content, and scripts to the <b>Associations</b> pane (2) to associate them with a choice or option; you can use the <b>Add</b> button at the top of the pane instead of dragging and dropping. You can select multiple attributes, symbols, or scripts at once. You will also enable and disable trimming of associated attributes and symbols in this pane. See <i>Trimming</i> on page 254 for additional information.</p> <p>You can view associations for individual attributes, symbols, and scripts in this pane by selecting an attribute, symbol, or script. Every association of the item is displayed. Select an association to view the item's configuration.</p>

3	<b>Settings pane</b>	<p>Configure attribute and symbol overrides in the <b>Settings pane (3)</b>. Overrides are set individually for each choice or option with which the attribute or symbol is associated. Default settings for attributes are set in the <b>Details pane (7)</b>.</p> <p>Columns can be resized by selecting a column header (in this case, <b>Value</b>) and moving its edges to widen or narrow it.</p>
4	<b>Attributes pane</b>	<p>The <b>Attributes pane (4)</b> lists the attributes contained in the template. You can create new attributes in this area. To associate attributes with the Object Wizard, highlight a Choice or Option, and then select and drag one or more user-created attributes from the <b>Attributes pane</b> to the <b>Associations pane (2)</b>. See <i>Associate Attributes with Choices and Options</i> for additional information.</p> <p>Default configuration settings for a selected attribute are displayed to the right of this pane, in the <b>Details pane (7)</b>. See <i>About the Attributes Page</i> on page 74 for additional information. Configured settings for the selected attribute that apply to a selected Object Wizard Choice or Option are shown in the <b>Settings pane (3)</b>. You can also enable attribute features (historization, I/O, alarms, etc. in the <b>Attributes pane</b>. Multi-select can be used to enable features for attributes, as well as to associate attributes. To use multi-select for features, select the attributes in the <b>Attributes pane</b> and enable the feature in the <b>Details pane</b>.</p>
5	<b>Content pane</b>	<p>The <b>Content (5)</b> pane lists symbols and external content items contained in, or linked to the object. To associate symbols with the Object Wizard, highlight a Choice or Option, and then select and drag one or more items from the <b>Content pane</b> to the <b>Associations pane (2)</b>. You can add a new symbol to the object from this pane, or link to a symbol in the Graphic Toolbox. See <i>Associate Content with a Choice or Option</i> on page 237 for additional information.</p>
6	<b>Scripts pane</b>	<p>To display the <b>Scripts pane (6)</b>, select the <b>Scripts</b> button. The <b>Scripts pane</b> lists scripts contained in the object. To associate a script with the Object Wizard, highlight a Choice or Option, and then select and drag one or more scripts from the <b>Scripts pane</b> to the <b>Associations pane (2)</b>. See <i>Associate Content with a Choice or Option</i> on page 237 for additional information.</p> <p>To add a script to the object, select the <b>Scripts tab</b> at the top left of the <b>Object Editor</b>.</p>
7	<b>Details pane (Attribute Editor)</b>	<p>The <b>Details pane (7)</b> displays details of a selected attribute, symbol, or other content. This pane lets you configure individual attributes, but is not usually used while configuring an Object Wizard. See <i>Configuring Objects</i> on page 121 for additional information.</p>

## Object Wizard User Interface Details

The Object Wizard user interface provides a number of visual elements to help you as you build an Object Wizard.



Ref	Icon	Name	Description
1		<b>Object Wizard button</b>	Opens/closes the <b>Object Wizard</b> editor ( <b>Choices and Options</b> , <b>Associations</b> , <b>Settings</b> , <b>Attributes</b> , and <b>Content</b> panes).
2		<b>Scripts button</b>	Opens/closes the <b>Scripts</b> pane so you can associate scripts with Object Wizard Choices and Options.
3		<b>Test mode button</b>	Changes the <b>Object Wizard</b> editor to test mode. This lets you verify the behavior of the Object Wizard. See <i>Object Wizard Test Mode</i> on page 267 for additional information.
4	<#>	<b>Number of associations</b>	Number of trimming-enabled attributes, symbols/content, and scripts associated with a Choice or Option. This can help in the maintenance of complex Object Wizards by letting you quickly identify where trimming has been enabled.
5	<#>	<b>Total configured settings</b>	Total number of configured settings for all attributes and symbols that are associated with each <b>Choice or Option</b> . This can help in the maintenance of complex Object Wizards by letting you quickly identify where overrides have been configured.

6	<input type="checkbox"/> <input checked="" type="checkbox"/>	<b>Trimming checkbox</b>	Place a check in the checkbox to enable trimming for an attribute or symbol. Trimming is always enable for scripts, and is enabled by default for attributes. See <i>Trimming</i> on page 254 for additional information.
7	<#>	<b>Number of configured settings</b>	Number of configured settings for each <b>attribute</b> or <b>symbol</b> that is associated with the selected Object Wizard Choice or Option. Scripts and external content items do not have configurable settings.
8		<b>Add symbol button</b>	Add (create new) symbol button. The new symbol is owned by the object. See Add Attributes, Symbols and Scripts to an Object Wizard for additional information.
9		<b>Link content button</b>	Link to an existing symbol or external content item in the Graphic Toolbox. The symbol/content can be shared with other objects. See Add Attributes, Symbols and Scripts to an Object Wizard for additional information.
10		<b>Edit symbol button</b>	Opens the selected symbol in the <b>Graphic Editor</b> .
11		<b>Symbol Wizard icon</b>	Symbol Wizard settings are indicated by a unique icon. See Attribute and Symbol Icons, below.
12		<b>Attribute / Custom Property icons</b>	Attribute and symbol custom property icons are displayed for Boolean, integer, float, double, string, time, elapsed time, and internationalized string. An icon is also displayed for Symbol Wizard settings. See <b>Attribute and Symbol Icons</b> , below.
13		<b>Setting Override indicator</b>	Attribute, custom property, and Symbol Wizard settings that contain an override are indicated by <b>bold</b> text and an X next to the value. To remove the override, select and clear the X. See <b>Override Icons</b> , below.
14	<text field>	<b>Visibility Expression</b>	Sets the conditional visibility of Object Wizard elements. See <i>Conditional Visibility Expressions for Choice Groups, Choices, and Options</i> on page 244 for additional information.
15	<b>Visibility indicator icons</b>		When the icon is unmuted, the setting is visible to the user when configuring instances. When muted, the setting is hidden. See <i>About Attribute and Symbol Overrides</i> on page 240 and <i>Configurable Settings for Attribute and Symbol Values</i> on page 261 for additional information. A small x in the top right corner of the icon indicates that the visibility setting is an override, and may differ from the setting in the parent template.  <b>Note:</b> In derived templates, visibility indicators replace visibility expressions for Object Wizard choices and options. See <i>Object Wizard Derivation</i> on page 263 for additional information.
		Visible	
		Visible - override	
		Hidden	
		Hidden - override	
16		<b>XRef</b>	Total number of configured overrides for the selected symbol or attribute. If the attribute or symbol is associated with multiple Object Wizard Choices or Options, hover over the XRef number to see which Choices and Options have overrides configured.

**Attribute and Symbol Icons**



Icon	Description / Data Type	Attribute	Symbol
	Symbol Wizard Choice or Option		X
	Boolean data type	X	X
	Integer data type	X	X
	Float data type	X	X
	Double data type	X	X
	String data type	X	X
	Time data type	X	X
	Elapsed time data type	X	X
	Internationalized string data type	X	X
[No icon]	Large type with no icon indicates that the item is an attribute feature. Attribute features are selected in the <b>Details pane</b> (see <i>Object Wizard GUI Panes</i> on page 229) and include I/O, History, State Alarms, Bad Value Alarms, Hi/Lo Alarms, etc.	X	

**Note:** For custom properties, string, time, and elapsed time data types can be toggled between reference/expression and string literal. This does not apply to attributes.



reference/expression



string literal

### Override Indicators

Indicator	Description
	This is the default setting (no override). If the setting is changed in a parent template, this value will also change.
	Setting has been overridden. Changes made in a parent template will not cause the this value to change. The value is shown in <b>bold</b> and an X appears next to the value.
	No override: Setting is visible and is not overridden. If the visibility of this item is changed in a parent template, the visibility of this setting will also change.
	No override: Setting is hidden and is not overridden. If the visibility of this item is changed in a parent template, the visibility of this setting will also change.
	Override has been applied: The visibility setting ("visible") is locked in this template. If this item is hidden in a parent template, this setting will remain visible (setting will not change).
	Override has been applied: The visibility setting ("hidden") is locked in this template. If the this item is made visible in a parent template, this setting will remain hidden (setting will not change).

## Object Wizard Components

An Object Wizard consists of Choice Groups and Options. Choice Groups contain at least two Choices, and Object Wizard users must select one Choice from every Choice Group.

Once you create Choice Groups, Choices and Options, you can associate attributes, symbols, layouts, external content, and scripts with the wizard to create an Object Wizard workflow. Typically, you create and configure the required attributes, scripts and other content prior to adding an Object Wizard to the template. See *Configuring Objects* on page 121 for additional information.

Choices are chosen by selecting the radio button next to the Choice. Only one Choice can be selected from a Choice Group. When users select a Choice, they are also selecting any elements (attributes, etc.) associated with the Choice, even though those elements may not be visible to them.

Options are selected by placing a check in the Option's checkbox. Options can be either true (checked) or false (unchecked). Attributes, symbols, other content, and scripts can only be associated with the true state. No action is implemented when an Option is set to false.

You can rename Choice Groups, Choices, and Options as needed and add a description (the description is visible to users as a tooltip). Choice Groups and Options must be uniquely named; a Choice Group and an Option cannot have the same name. Choices within a single Choice Group must also be uniquely named.

## Add Content for Use with an Object Wizard

You add attributes, symbols, layouts, scripts, and other content to a template object with an Object Wizard the same way that you add content to any other object. After adding the content, you can associate individual attributes and other content to individual Choices and Options. This association allows content that is not needed to be trimmed when you derive instances from the template. This helps to reduce the footprint of run-time objects. See *Trimming* on page 254 for additional information.

---

**Note:** Only user-created attributes can be associated with Object Wizard Choices and Options.

---

When you associate content with Choices and Options, the propagation of these items to derived instances is determined by:

- User selection of Choices and Options during configuration.
- *Trimming* on page 254 configuration for attributes and symbols, as they apply to the selected Choices and Options.

Content is trimmed from a derived instance when the instance is saved. Note that all items are propagated to derived templates, regardless of configuration and trimming settings, since unlike instances, templates are not run-time objects.

Both attributes and symbols can have unique settings for each association. The visibility of these settings, which determines if the setting can be overridden by a user deriving instances from the template, can also be configured.

For detailed information about Symbol Wizards and custom properties, see "Using Custom Properties" and "Working with Symbol Wizards."

Content can either be added or linked to an object.



When you select the **Add Content** button, a new symbol is created that is owned by the object. Any changes made to the symbol are only applicable to the owning object. Changes to the symbol in the parent object will cascade to child objects. The **Add Content** button does not let you create an object-owned external content item.



When you select the **Link Content** button, the Galaxy Browser opens the Graphic Toolbox (GTB). Selecting a symbol, layout or external content object from the Graphic Toolbox creates a link between the object and the content, but the object does not own the content. Note that relative references will resolve correctly for the linked symbol or layout. Changes to the content will cascade to every object with which the item is linked, regardless of the hierarchical relationship of the objects.


For information on adding scripts to a template, see *Writing and Editing Scripts* on page 160.

The settings configured in the **Attribute Editor**, **Graphic Editor**, and **Layout Editor** provide the default settings for each attribute and content item. You can override the default values for attributes and symbol custom properties in the **Settings pane**. Note that you can set different values for each Choice or Option with which the content is associated.

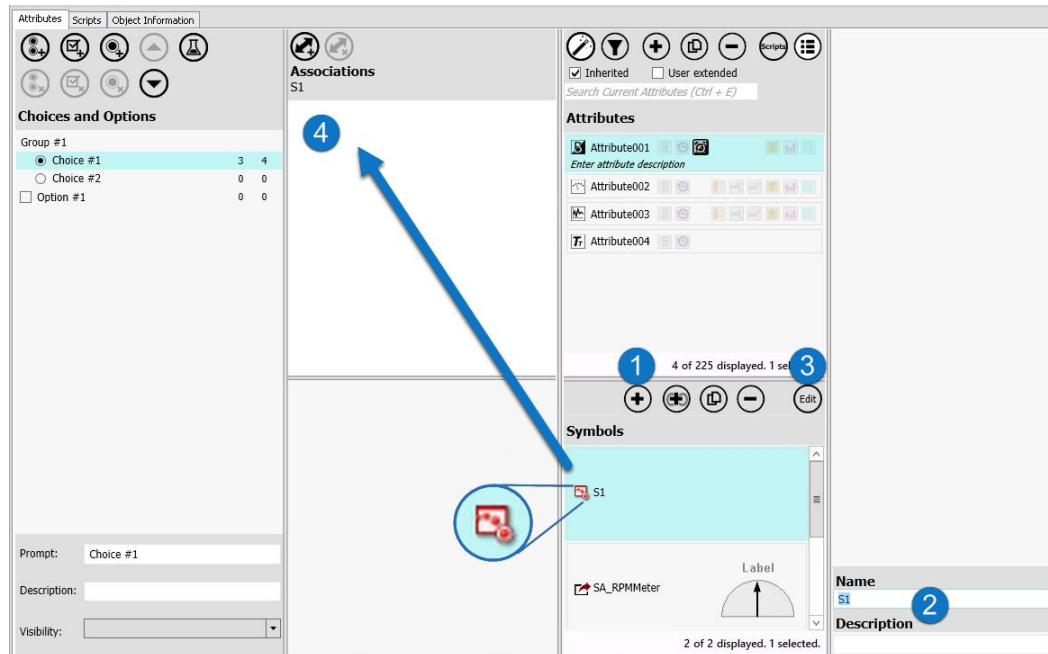
In addition, you can allow the user to override the default values in derived instances. See "About the Attributes Page on page 74" and "Add Attributes to an Object on page 121" for information about adding and configuring attributes.

## Add a New Object-Owned Symbol

To add a new object-owned symbol:

1. In the **Content pane**, select the **Add Content button** . The symbol is added with a default name (based on the previously added symbol, if any).
2. In the **Details pane**, edit the name of the symbol and add a description.
3. In the **Content pane**, select the **Edit button** to open the **Graphic Editor**. See *Creating and Managing Industrial Graphics User Guide* for information about using the editor to create symbols. When you done editing the symbol, save and close the Graphic Editor.
4. Associate the symbol with one or more Choices or Options, as described in *Associate Content with a Choice or Option* on page 237.

The icon next to the symbol name in the **Content pane** indicates that the symbol is owned by the object.



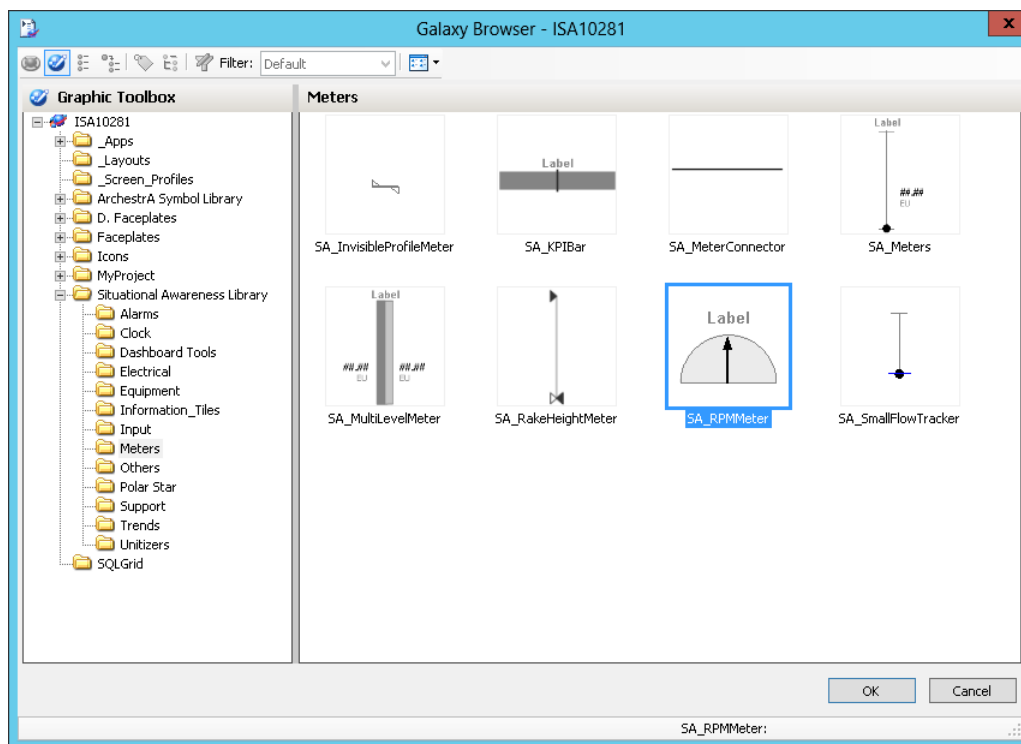
## Link to Shared Content in the Graphic Toolbox

You can link an object to content in the Graphic Toolbox. Content can be a symbol, a layout or an external content object.

### To link to an existing content item in the Graphic Toolbox:

1. In the **Content pane**, select the **Link Content button** .

The **Galaxy Browser** opens. Select an existing symbol, layout, or external content object from the **Graphic Toolbox**.



2. In the **Details pane**, edit the default name of the content as needed. The edited name applies only to the object to which it is linked. The name of the content item remains unchanged in the Graphic Toolbox. You can also add a description .

---

**Note:** If you choose to link to the same content item more than once, it is automatically renamed within the object with an incremental number appended to the original name. Its name in the Graphic Toolbox is not changed, and is displayed in brackets in the following format: `OriginalName1 [OriginalName]`. In most cases, linking to the same item more than once is not an efficient use of an Object Wizard.

---

3. **Optional:** To edit the linked content, select the **Edit button** to open the **Graphic Editor** (for symbols), the **Layout Editor** (for layouts), or the **External Content Editor** (for external content items). Any change you make to the linked content item changes the configuration of the content, and the change affects any object that links to the item.

---

**Note:** Editing protected symbols, such as symbols contained in the **Situational Awareness Library**, is not permitted. If you edit an item in the **Graphic Toolbox**, the changes will affect every object that links to the item.

---

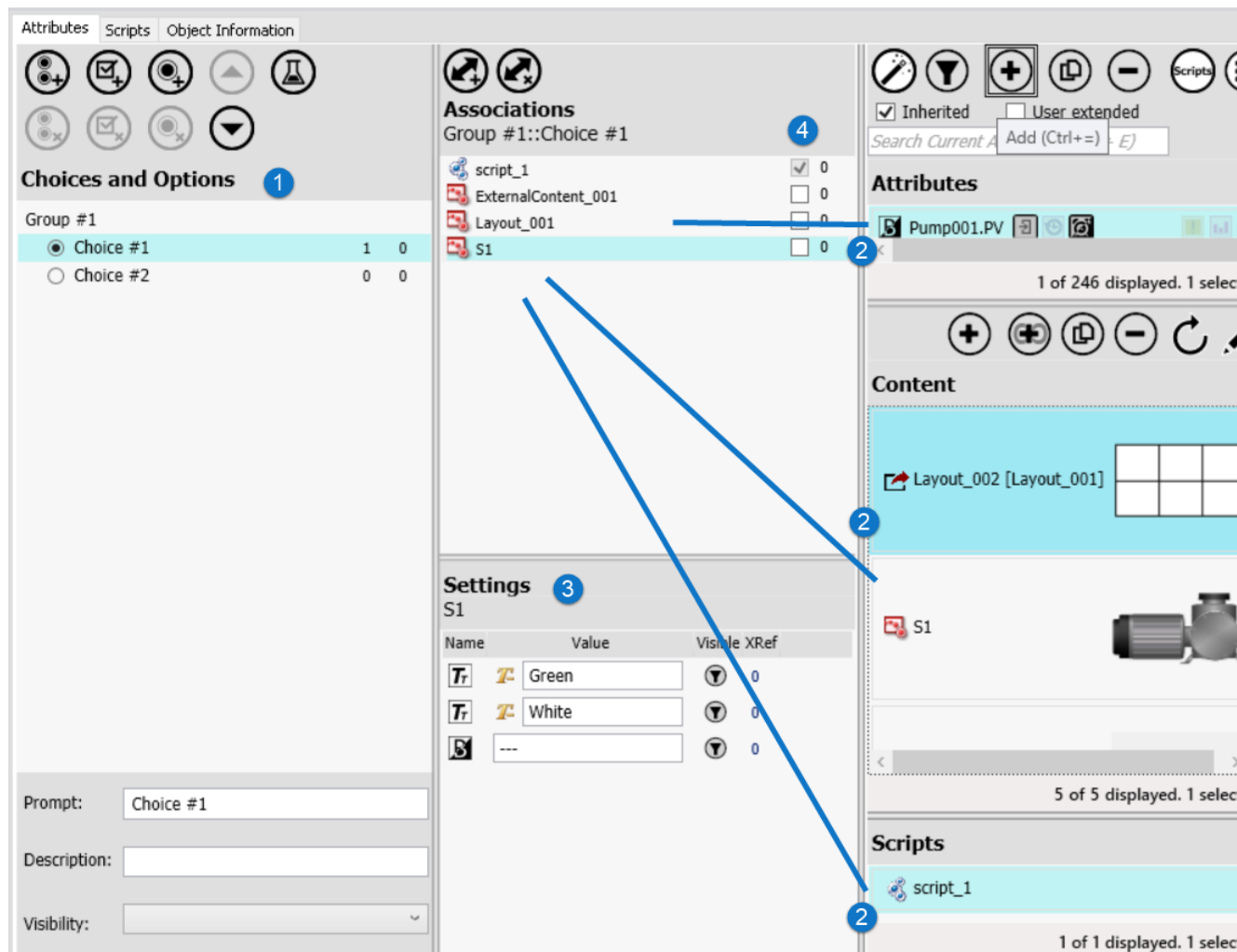
## Associate Content with a Choice or Option

In most cases, you will have added content to the object prior to adding the Object Wizard. Once you have begun configuring the Object Wizard, you can associate the following types of content with Object Wizard Choices and Options:

- Attributes: shown in the **Attributes pane**
- Symbols: shown in the **Content pane**
- External Content Objects: shown in the **Content pane**
- .NET Controls: shown in the **Content pane**
- Scripts: shown in the **Scripts pane**

### To associate content with a Choice or Option:

1. Highlight a Choice or Option in the **Choices and Options pane**.
2. Select and drag one or more item from the **Attributes**, **Content**, or **Scripts pane** to the **Associations pane**. Use the **Shift** or **Ctrl** key to enable multi-select for items in a single pane.



---

**Note:** Individual items can be associated with more than one Choice or Option. However, a Choice or Option does not have to have any items (attributes, etc.) associated with it. For example, a Choice labeled "No" or "None" typically would not have any content associated with it.

---

---

**Note:** A content item (attribute, symbol, layout, etc.) does not have to be associated with an Object Wizard Choice or Option. Unassociated items are simply a part of the object and are propagated to all derived objects.

---

3. If applicable, highlight an attribute or symbol and enter overrides for the default values. The overrides affect only the Choice or Option for which they are entered. For example, a symbol can have one label for one Choice, and a different label for another. The override applies only to the associated Choice or Option. See *Attribute and Symbol Overrides* on page 257 for additional information.

---

**Note:** You can resize columns in the **Settings pane** by selecting the column header and moving its edges to widen or narrow the column.

---

4. If applicable, override the default trimming setting for each associated item.
  - Trimming is disabled by default for symbols, layouts, and external content (checkbox is unchecked).
  - Trimming is enabled by default for attributes (checkbox is checked).
  - Trimming is enabled for scripts and cannot be disabled.
  - Note that the trimming defaults for symbols and attributes are different. See *Trimming* on page 254 for additional information.

## Remove an Association from a Choice or Option

To discontinue the association of a content item (attribute, symbol, etc.) with a Choice or Option:

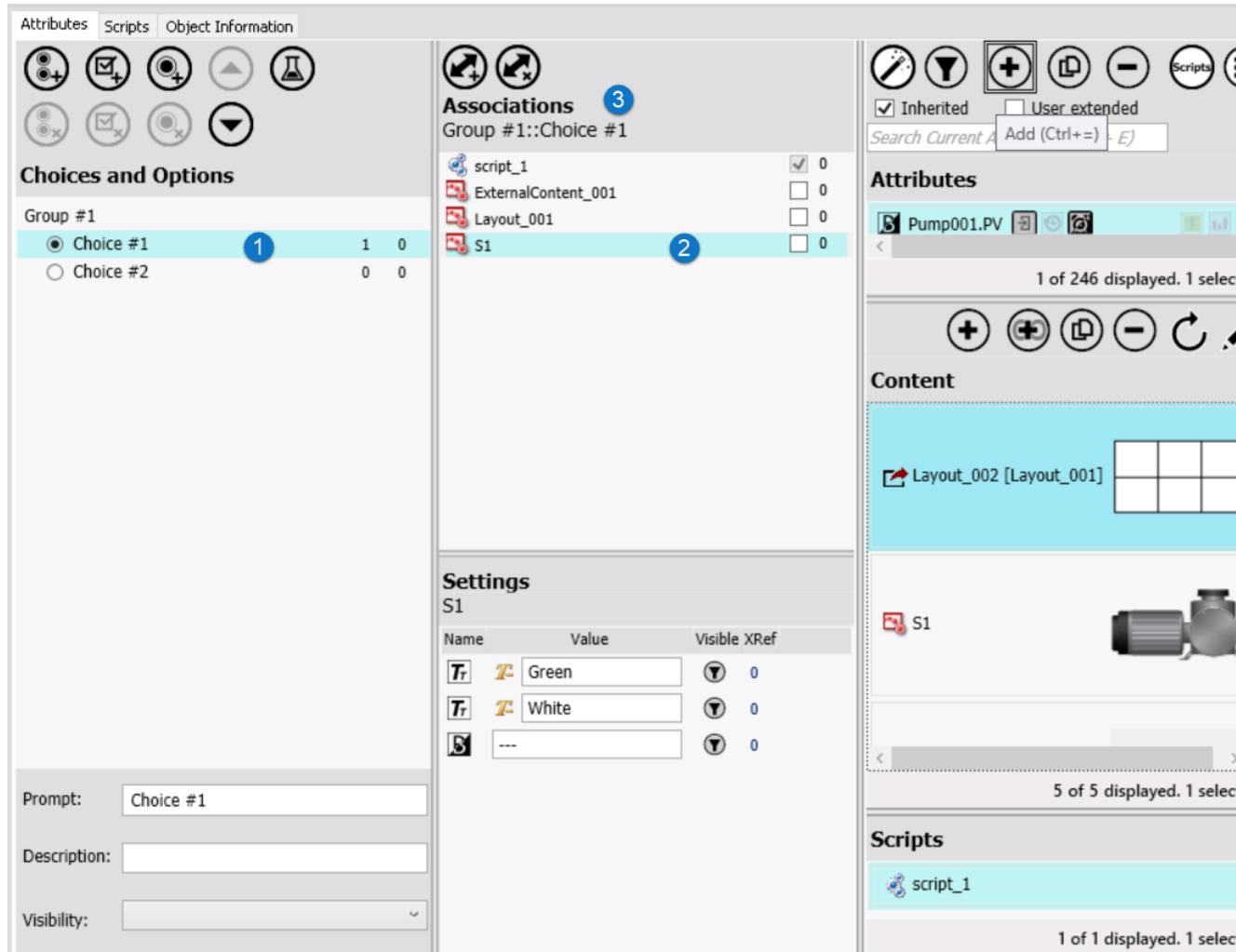
1. Locate the Option, or the Choice within its Choice Group (1), and highlight it.
2. In the **Associations pane**, select the content item (2) you want to remove from the Choice or Option.
3. Select the **Remove Association** button (3). The item is no longer associated with the Choice or Option.

---

**Note:** Removing an association does not remove the content item. The content remains a part of the template, and if the content is associated with other Choices or Options, those associations remain intact. However, removing an association may affect the trimming of content within instances (see *Trimming* on page 254 for additional information). If the content is completely unassociated with any Object Wizard Choice or Option, it is always included in derived instances.

---

To completely remove an attribute from a template, see *Delete Content from an Object* on page 241.



## Multiple Content Item Associations

Attributes, symbols, and other content can be associated with more than a single Choice and/or Option. Each association can utilize different override values (settings) for a particular attribute or symbol. To change a setting for a particular association, enter overrides of the defaults in the settings pane.

You can use different overrides for each Choice or Option with which an attribute or symbol is associated. For example, an integer attribute could have a default value of 10, but could have an override of 5 for one Choice and an override of 20 for a different Choice.

## Map Attribute and Symbol Configurations to a Choice or Option

You can associate a particular attribute, layout, or symbol configuration with an Object Wizard selection (Choice or Option). In this way, pre-configured attributes and content can be added to the derived instance, simply by selecting an Object Wizard Choice or Option. Different configurations can be applied to different Choices and Options.

Mapping these configurations to Object Wizard Choices and Options may require additional planning and testing. This is especially important if multiple configurations of the attribute or content are being mapped. Mismatches between configurations can result in different behaviors in the derived instance than intended. Use **Test mode** to catch potential mismatches.

## About Attribute and Symbol Overrides

Override values for an attribute or symbol (with a Symbol Wizard or custom properties) can be configured for each associated Choice or Option. When you associate an attribute or symbol, the default values are listed in the **Settings pane**. To configure values that apply to the selected Choice or Option, enter the values in the **Settings pane**. If the attribute or symbol is associated with multiple Choices or Options, different values can be used for each one. Values for the Choices and Options are saved when you save the object. If the setting is marked as visible, users can override your setting (or override the default, if you have not configured a setting) as they derive and configure instances from the Object Wizard.

Symbols with overrides that affect their shape, for example, certain Situational Awareness Library symbols, are rendered with the applicable shape-changes as

---

**Note:** If you override a Symbol Wizard setting, the only way to make the setting visible again is to remove the override. Attributes and custom properties with overrides can be made visible again by selecting the visibility checkbox.

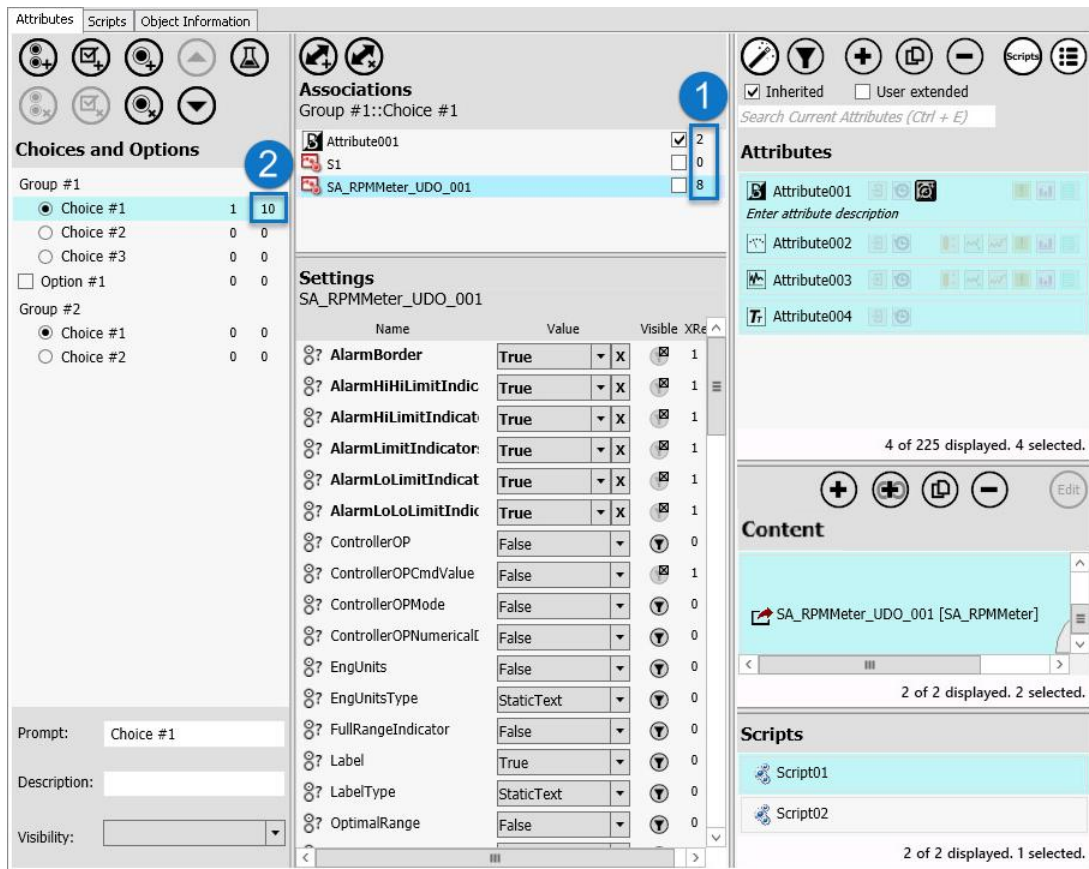
---

A configured setting is defined as any change from the default configuration of the attribute or symbol. This includes changing a value or visibility. For example, you might change the default Boolean value from true to false, or you might change an integer value from 5 to 10, or change engineering units from liters to cc. Changing the visibility of a setting also counts as a configured setting.

- The default visibility setting for attribute values is hidden (Visibility icon is muted (grayed-out) in the **Settings pane**).
- The default visibility setting for Symbol Wizard and custom property values is visible (Visibility icon is unmuted in the **Settings pane**).
- Visibility overrides are indicated by a small X in the upper right quadrant of the visibility icon.



The total number of configured settings for each individual attribute and symbol is displayed in the **Associations pane (1)**. The total number of configured settings for ALL attributes and symbols associated with an individual Choice or Option is displayed in the **Choices and Options pane (2)**.



## Delete Content from an Object

You can delete (completely remove) a content item from an object. When you delete content, all settings and associations to the items are also removed.

If there are any instances or templates derived from the template that use the deleted item, the deletion of the item will propagate to all the derived objects. That is, all objects derived from the template will no longer contain the deleted content.

## Delete an Attribute or Symbol from a Template

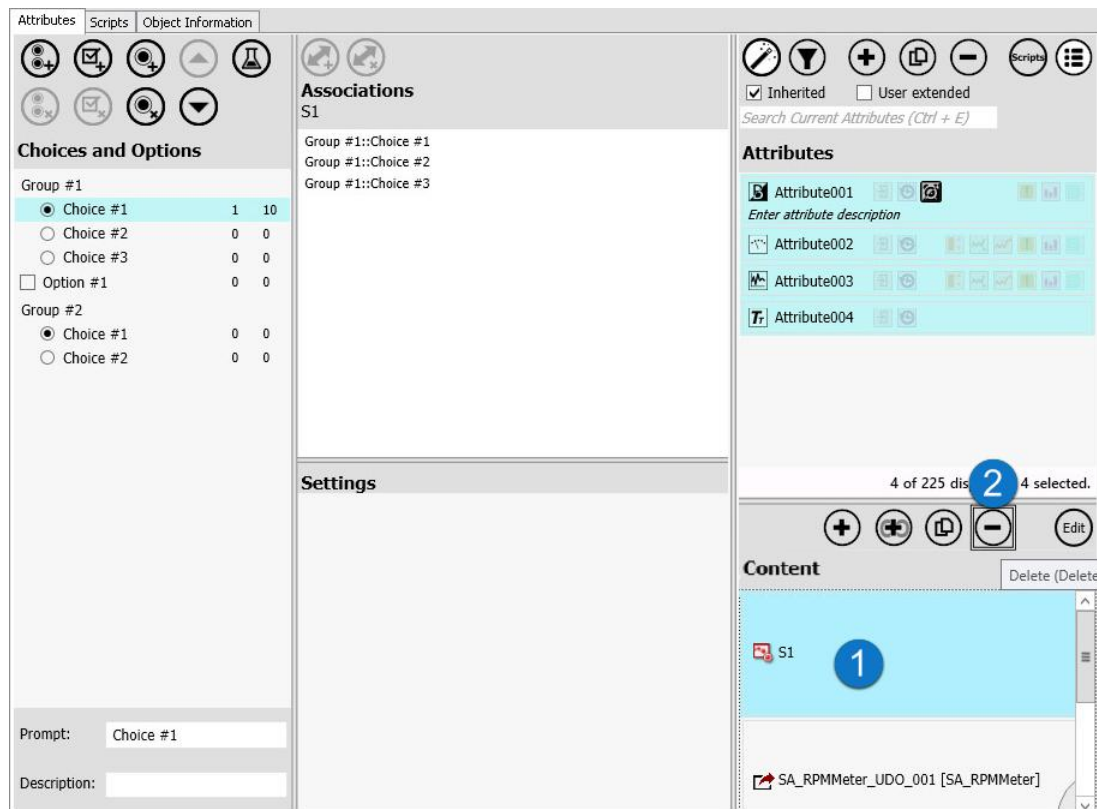
You can delete an attribute or content (object-owned or linked) from a template that has an Object Wizard. If there are objects derived from the Object Wizard that use the deleted attribute or content, the deletion propagates to all the derived objects.

Linked symbols, layouts, and external content items will still exist in the Graphic Toolbox. Deleting an object-owned symbol completely removes it. Once you save the template that contains the deleted object-owned symbol, the symbol cannot be retrieved.

### To delete an attribute or content from a template:

1. Select the content item from the **Content pane**. To delete an attribute, select it from the **Attributes pane**. You can select more than one attribute or symbol at a time.
2. Select the **Delete** button. This removes the selected symbol from the object. Use the **Delete** button in the **Attributes pane** to delete attributes.

**Note:** Added (object-owned) symbols are permanently deleted once the object is saved. Linked content is removed from the object but remain in the Graphic Toolbox.



## Example 2: Configure a Linked Symbol Associated with Multiple Choices

This example shows the steps required to link and configure a symbol from the **Graphic Toolbox**, set configurations, and associate different configurations with Choices and Options. This example uses a symbol from the **Situational Awareness Library**. You could also link other types of content, such as layouts and external content items. However, only symbols allow you to configure and associate different configuration settings within the Object Wizard.

**Note:** Most symbols in the **Situational Awareness Library** are Symbol Wizards.

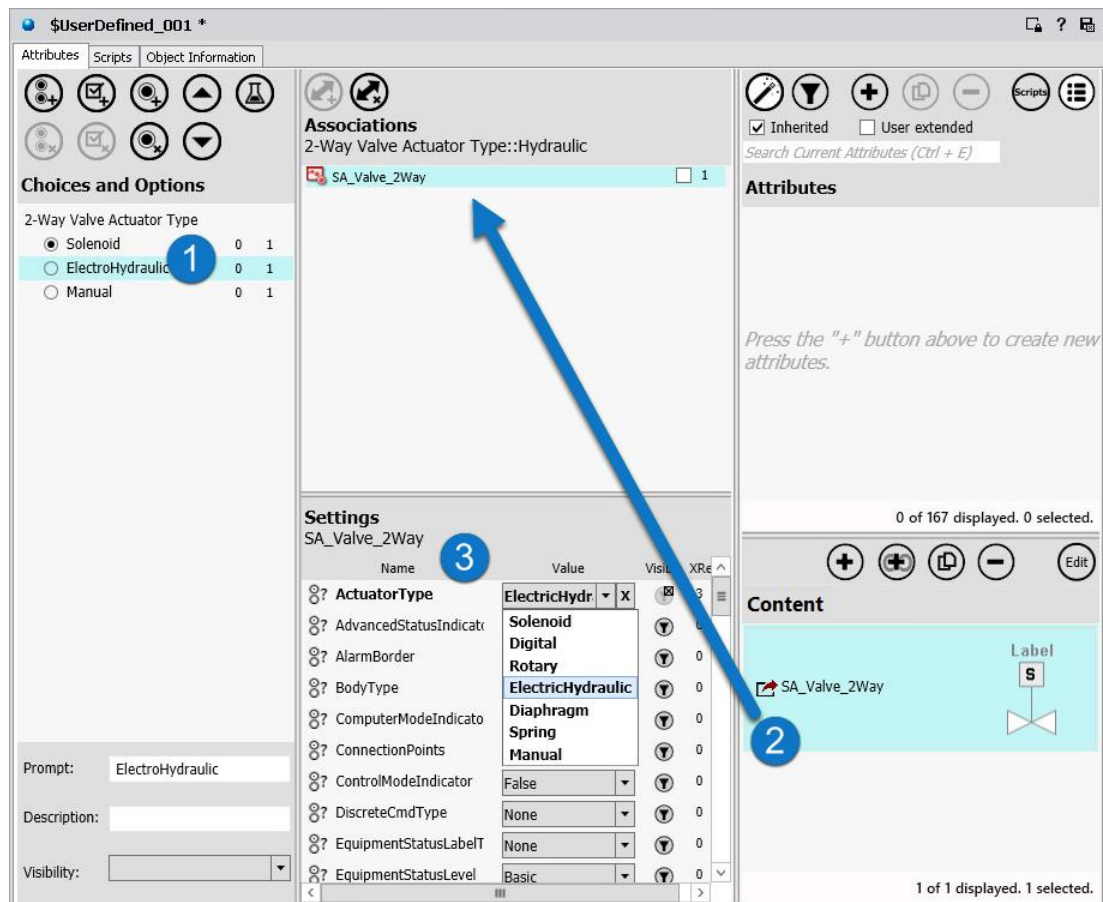
### To configure a linked symbol associated with multiple choices

1. Derive a template from the \$UserDefined template, then:
  - a. Add an Object Wizard Choice Group and name it "2-Way Valve Configuration."
  - b. Add three Choices, named: "Solenoid," "ElectroHydraulic," and "Manual."
2. Navigate to the Equipment folder of the Situational Awareness Library (in the **Graphic Toolbox**), and link to the SA\_Valve\_2Way symbol. The symbol is added to the **Content** pane.

Associate the linked symbol with each of the Choices created in step 2, and set the ActuatorType in the **Settings** pane to match the associated Object Wizard Choice.

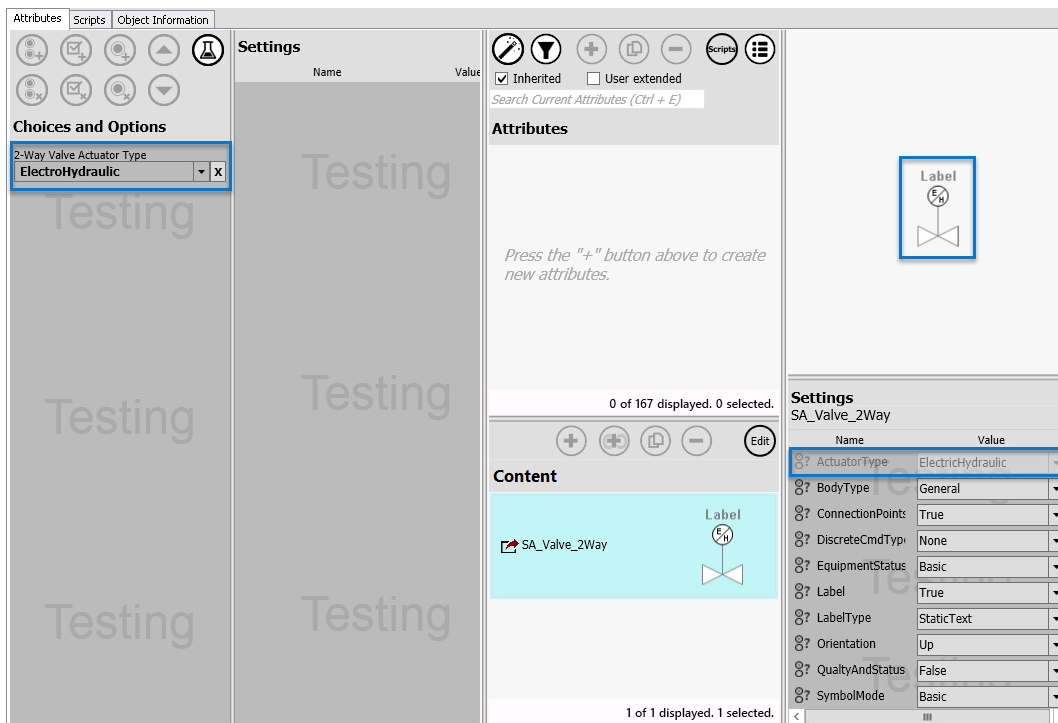
3. Set visibility and overrides for each association:
  - For Solenoid:
    - Select the Solenoid Choice from the **Choices and Options** pane.

- Select SA\_Valve\_2Way from the Associations pane.
- In the Settings pane, make sure that ActuatorType is set to the default (Solenoid), then turn off visibility.
- For ElectroHydraulic and Manual:
  - Select the ElectroHydraulic or Manual Choice from the **Choices and Options pane**.
  - Select SA\_Valve\_2Way from the Associations pane.
  - In the **Settings pane**, set the ActuatorType to match the Choice in the Choices and Options pane. Since both ElectroHydraulic and Manual are overrides, visibility of these settings automatically changes to hidden.



4. Switch to **Test mode** and check each of the Choices in the **Choices and Options pane**.
  - ActuatorType under Symbol Settings should be grayed-out to indicate it is hidden
  - The symbol setting should match what you have selected in the **Choices and Options pane**.

- o The symbol thumbnail should match the selected ActuatorType.



## Conditional Visibility Expressions for Choice Groups, Choices, and Options

You can set whether or not a user will be able to see individual Choice Groups, Choices and Options as they use the Object Wizard. Visibility expressions are scripted and use simple logical expressions to determine if users will be able to see a particular Option or Choice when configuring instances. The visibility expression determines if the Object Wizard component will be visible or hidden, based on the user's previous Object Wizard selections. For example, selecting Choice #1 may hide (or show) Choice Groups, Choice or Options that follow that Choice in the Object Wizard.

**Note:** You cannot add a visibility expression to the first listed item (Choice Group or Option) in an Object Wizard; the first item is always visible.

When a user uses the Object Wizard to configure an instance, the default behavior of any elements hidden by visibility expressions remains in effect. For example, if Choice #1 is the default of a Choice Group, and the Choice Group is hidden, Choice #1 is still selected for the configured instance, even though it is not visible to the user. If the default state of an Option is True but the Option is hidden, the Option remains True for the configured instance.

Visibility is set through logic expressions assigned to Choices and Options. These logic expressions use the Boolean operators AND, OR, and NOT.

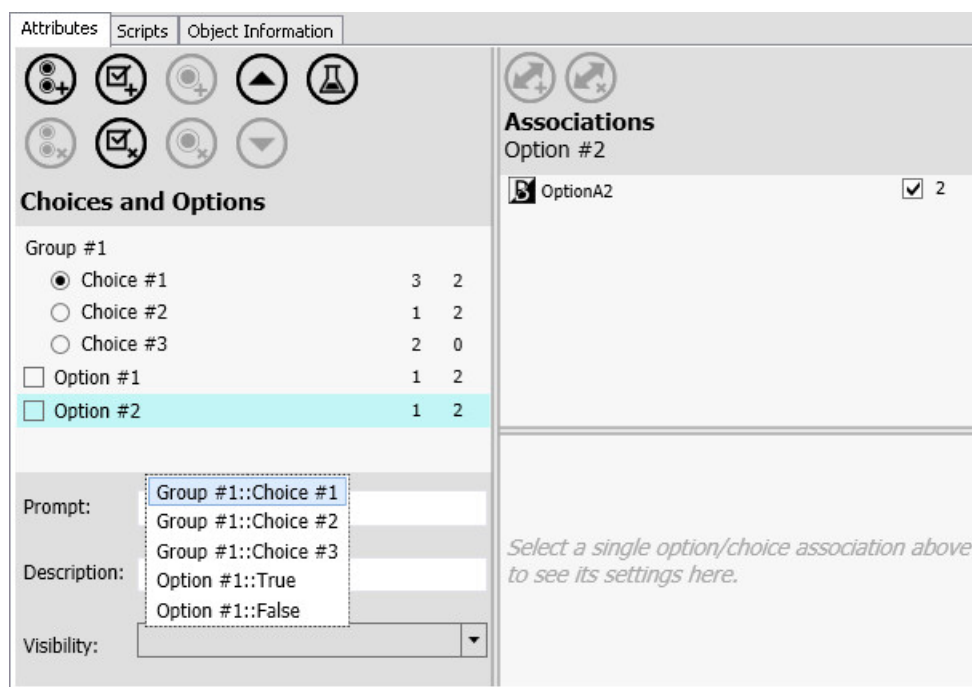
In derived templates that contain an Object Wizard from the parent object, icons are used to implement visibility settings that are in addition to the settings made in the parent object. Note that the visibility settings made in the parent Object cannot be changed in the derived template. See *Object Wizard Visibility Settings in a Derived Template* on page 248 for additional information.

## Add a Basic Visibility Expression

### To add a basic visibility expression to a Choice Group, Choice or Option:

1. Highlight the item for which you are setting visibility (Choice Group, Choice or Option). You can select any item except the initial one. That is, you cannot make the visibility conditional of the first item (Choice Group or Option) in the Object Wizard. If the first item is a Choice Group, you cannot make it or any of its Choices conditional.
2. Place the cursor in the Visibility text box of the selected item, and type " [" (open bracket character). A drop-down list of all preceding items (Choices and Options) is displayed.

**Note:** Choices are listed in format "*Choice Group Name*::*Choice Name*", for example, *Choice Group #1::Choice #1*. Options are listed in the format "*Option::State*", for example *Option #1::True*.



**Note:** While you can simply select an item for a simple visibility expression without typing an open bracket, we recommend that you always type the open bracket as a matter of habit. As you begin building more complex visibility expressions, you risk deleting the expression if you select an item from the drop-down without first typing an open bracket.

3. Select the item from the drop-down list that will be used to control the visibility of the selected item. For example, you can make Option #2 visible only if Choice #1 of Choice Group #1 has been selected (Group #1::Choice #1), and/or if Option #1 is set to True (Option #1::True).

**Note:** When a user configures an instance, the default behavior of elements remains in effect, even if they are hidden from the user. The default Choice of a hidden Choice Group is still selected, and hidden Options remain in their default states.

For additional information, see *Operators, Rules, and Behavior for Visibility Expressions* on page 247.

## Add a Visibility Expression with Logical Operators

**To add a visibility expression that uses logical operators to a Choice Group, Choice or Option:**

1. Highlight the item for which you are setting visibility (Choice Group, Choice or Option). You can select any item except the initial one. That is, you cannot make the visibility conditional of the first item (Choice Group or Option) in the Object Wizard. If the first item is a Choice Group, you cannot make it or any of its Choices conditional.
2. Place the cursor in the Visibility text box of the selected item, and type " [" (open bracket character). A drop-down list of all preceding items (Choices and Options) is displayed.

---

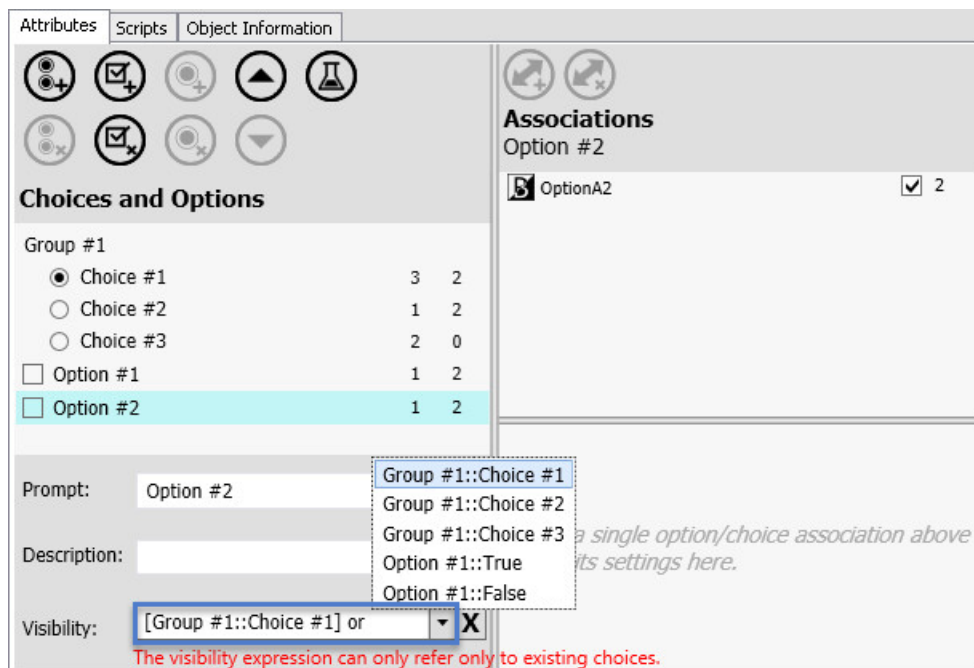
**Note:** Always type an open bracket before selecting an item from the drop-down list. Simply selecting an item without first typing an open bracket will overwrite the existing expression.

---

**Note:** Choices are listed in format "Choice Group Name::Choice Name", for example, Choice Group #1::Choice #1. Options are listed in the format "Option::State", for example Option #1::True.

---

3. Select the item upon which you are basing the visibility of the selected item. It will be entered in the visibility expression. To expand the expression:
  - o Place the cursor at the front of the expression (before the open bracket) and enter the NOT operator, or
  - o Place the cursor at the end of the expression (after the close bracket) and enter the AND or OR operator.
4. Type " [" (open bracket character) and select another item from the drop-down. All Object Wizard items must be enclosed in brackets. Enclose items and operators in parentheses as needed to ensure that the expression resolves as intended.



5. Add additional operators, parentheses, and Object Wizard items as needed to complete the visibility expression. For example:

```
[ChoiceGroup1::Choice1] AND ([Option1::True] OR [Option2::True] OR [Option3::False])
```

**Note:** When a user configures an instance, the default behavior of elements remains in effect, even if they are hidden from the user. The default Choice of a hidden Choice Group is still selected, and hidden Options remain in their default states.

For additional information, see *Operators, Rules, and Behavior for Visibility Expressions* on page 247.

## Operators, Rules, and Behavior for Visibility Expressions

As you build an Object Wizard, you can create visibility expressions that selectively hide Choice Groups, Choices, and Options. This can simplify the workflow by only showing Choices and Options that are applicable, based on Choices and Options that the user has already made. When an Option or Choice Group is hidden, the system applies the configured default values for attributes, symbol and scripts associated with the hidden Option or Choice Group. You can leverage this capability to hide advanced functions from users tasked with configuring instances from templates that contain Object Wizards.

The **Visibility** prompt lets you assemble expressions to hide or show Choice Groups, Choices, or Options based on what the user has previously selected. Note that Visibility cannot be conditional for the first Object Wizard item (Choice Group or Option). The first Object Wizard item is always visible. Available logical operators are AND, OR, and NOT.

Observe these basic rules and behaviors when building visibility expressions:

- Always start by typing the "[" (open bracket character) in the visibility text box. This will bring up a list of all Choices (from other Choice Groups) and Options that precede the selected item. Do not simply highlight an element without first typing the open bracket character.
- Choices are listed with their Choice Group. Each Option is listed twice, for both states (True and False). Use fully qualified Option/Choice selections surrounded by brackets. The separator between Choice Groups and Choices, and between Options and their states is "::". For example:

```
[Valve Type::3-Way] OR [Valve Type::4-Way] AND [Option1::True]
```

- Valid logical operators:
  - Valid binary operators are "AND" and "OR".
  - Valid unary operator is "NOT".
  - "AND" has a higher precedence than "OR".
  - "NOT" has a higher precedence than "AND" or "OR".
  - You can use "&" or "&&" instead of "AND".
  - You can use "|" or "||" instead of "OR".
  - You can use "!" instead of "NOT".
- Parentheses can be used to build complex expressions. For example:

```
[ChoiceGroup1::Choice1] AND ([Option1::True] OR [Option2::True] OR [Option3::False])
```

Note that without parentheses, this expression would resolve very differently, since "AND" has a higher precedence than "OR". If parentheses were not used, the expression would resolve this way:



```
[ChoiceGroup1::Choice1] AND [Option1::True] OR [Option2::True] OR [Option3::False]
```

- Literals and operators are not case sensitive.
- Spacing between literals and operators is optional.

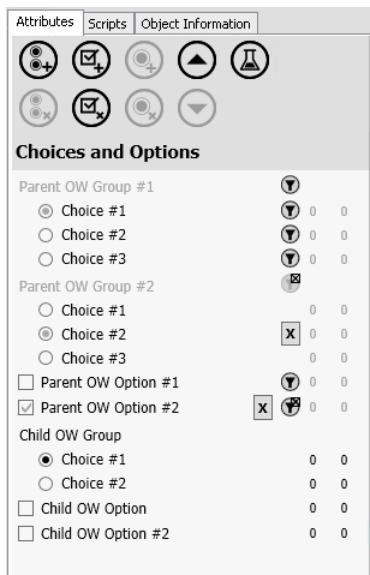
- The visibility expression editor validates the logical expression as you enter it, but does not check if the expression is possible. For example, the expression "[Valve Type::3-Way] AND [Valve Type::4-Way]" would not be flagged since it is logically correct, even though the condition is impossible (you cannot select more than one Choice from a single Choice Group).


## Object Wizard Visibility Settings in a Derived Template

When you derive a template from a template with an Object Wizard, derived Object Wizard elements can no longer be edited, and Object Wizard group names are grayed out. Other noticeable changes are:

- A visibility icon  appears next to each derived wizard item.
- You can override visibility settings by clicking on the visibility icon.
- The item counts are reset to zero; changes made in the derived object will increment the counters.
- If you change a default selection, the setting override indicator  appears next to the changed setting.

See *Object Wizard User Interface Details* on page 231 for more information about visibility icons and the setting override indicator. For additional information about derivation and visibility, see *Object Wizard Derivation* on page 263.



The visibility expressions that were configured in the parent Object Wizard remain in effect. You can, however, hide Choice Groups, Choices, and Options. Instead of using Boolean expressions to set the visibility of Groups, Choices, and Options, use the visibility icon . When changing the visibility settings of a derived Object Wizard:

- The behavior of a hidden Choice or Option does not change when configuring an instance from the derived Object Wizard. The default state of a hidden Option remains in effect when the instance is configured. If the Option's default state is True, any attributes, symbols, or scripts associated with it are automatically configured for the instance. Similarly, the default Choice in a hidden Choice Group is automatically selected, along with its associated attributes, symbols and scripts.
- You cannot hide (turn off the visibility of) a default Choice. You can, however, hide the entire Choice Group.
- If every Choice except the default is hidden, the entire Choice Group is hidden.



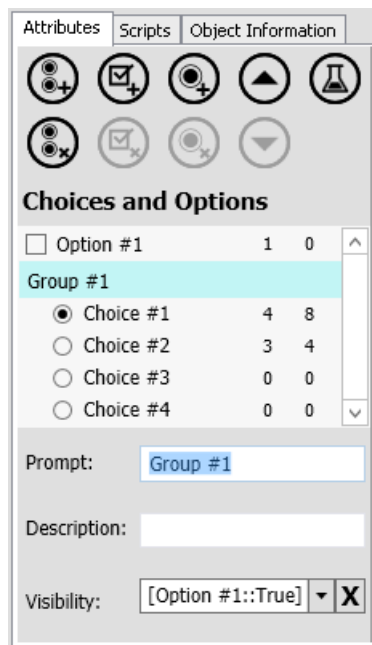
- If you change a default choice, the entire Choice Group is automatically hidden. You can turn visibility on again for the Choice Group. Once the Choice Group has been made visible, you can hide individual Choices.

## Example 1: Basic Visibility Expressions

This example shows how to build a basic visibility expression. In this case, we want to hide a Choice Group unless an Option is True.

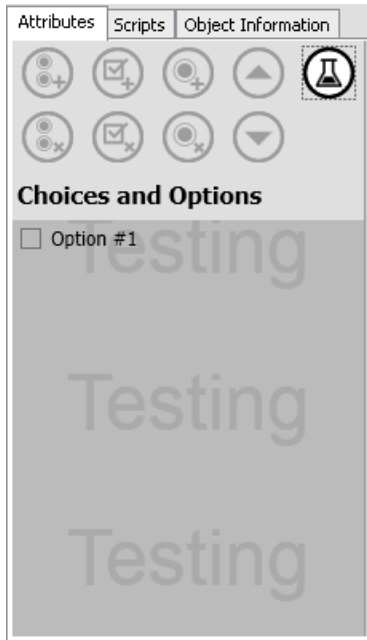
1. Select Group #1 and type [ (open bracket) in the Visibility text box.
2. Select Option #1::True from the drop-down list. The Visibility text box contains:

```
[Option #1::True]
```

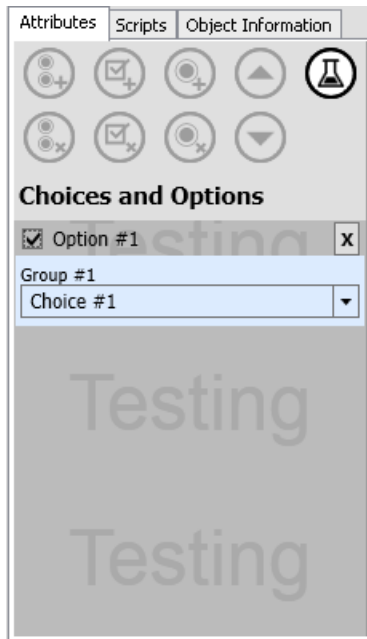


3. Select Test mode.

- With Option #1 unselected (false), Group #1 is hidden.



- Set Option #1 to True. Group #1 is now visible.

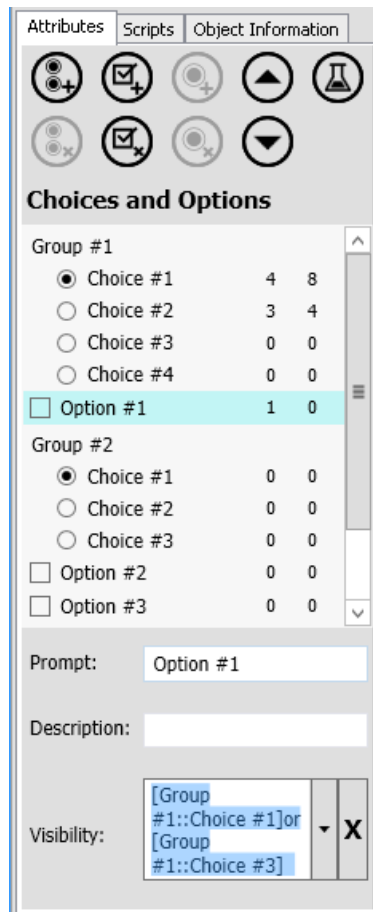


## Example 2: Complex Visibility Expressions

Setting more complex rules for the visibility of Choices and Options requires the use of the logical operators AND, OR, or NOT, and may also necessitate the use of parentheses to ensure that the expression is parsed correctly. The rules for using logical operators are listed in *Operators, Rules, and Behavior for Visibility Expressions* on page 247.

## Scenario 1: Visibility of option is contingent on specific choices

Consider the following scenario. We want to allow a user to select Option #1 only when Choice #1 or Choice #3 is selected from Group #1.



1. Select Option #1 and type [ (open bracket) in the Visibility text box.
2. Select Group #1::Choice #1 from the drop-down list.
3. Place the cursor in the text box after ] (close bracket) and type OR.
4. Type [ and select Group #1::Choice #3 from the drop-down list. The Visibility text box should contain:  

```
[Group #1::Choice #1] or [Group #1::Choice #3]
```
5. Select Test mode.
6. Select Group #1::Choice #1. Option #1 is visible
7. Switch to Group #1::Choice #3. Option #1 remains visible.
8. Switch to Group #1::Choice #2. Option #1 is now hidden.

## Scenario 2: Visibility of option is contingent on a combination of choices and options

In this case, we want allow a user to select Option #3 only when Group#1::Choice #1 and Group#2::Choice #1 or Group#2::Choice #2 is selected and Option #2 is true. This expression can be interpreted in different ways:

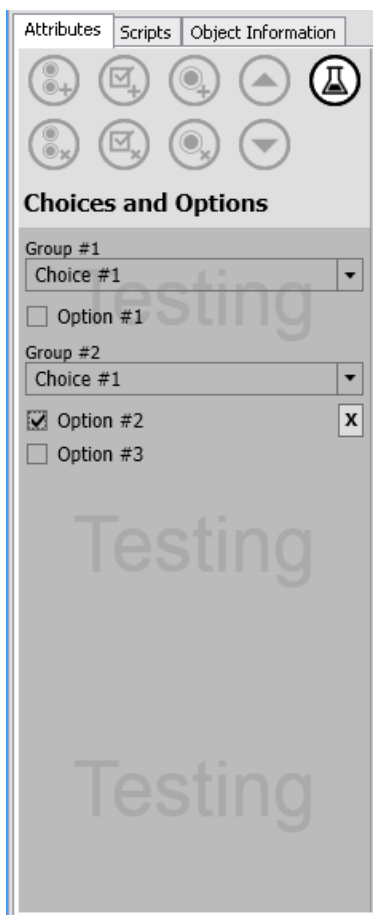
- (Group#1::Choice #1 and Group#2::Choice #1) or (Group#2::Choice #2 and Option #2::True)
- Group#1::Choice #1 and (Group#2::Choice #1 or Group#2::Choice #2) and Option #2::True

Let's assume that we want the latter interpretation.

1. Select Option #3 and type [ (open bracket) in the Visibility text box.
2. Select Group #1::Choice #1 from the drop-down list.
3. Place the cursor in the text box after ] (close bracket) and type AND.
4. Type [ and select Group #2::Choice #1 from the drop-down list.
5. Place the cursor in the text box after ] (close bracket) and type OR.
6. Type [ and select Group #2::Choice #2 from the drop-down list.
7. Place the cursor in the text box after ] (close bracket) and type AND.
8. Type [ and select Option #2 from the drop-down list.
9. Position the cursor in the text box enclose Group#2::Choice #1 or Group#2::Choice #2 in parentheses.
10. The Visibility text box should contain:  

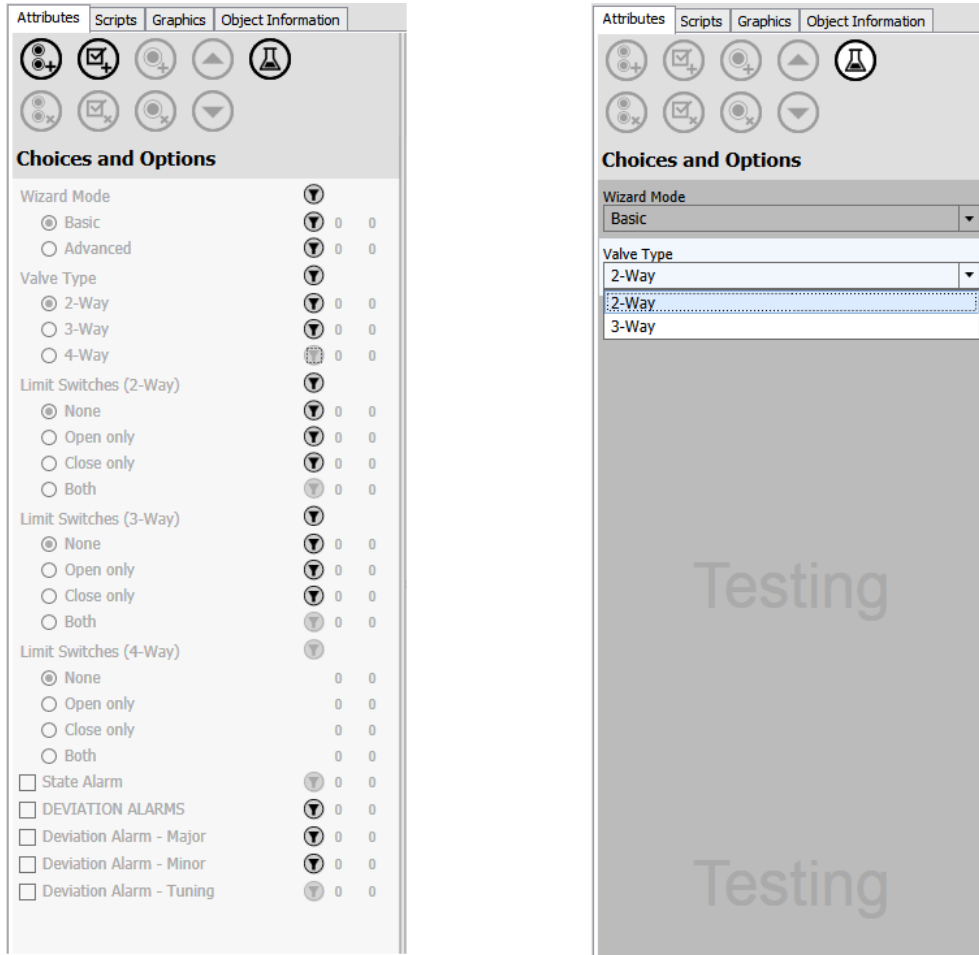
```
Group#1::Choice #1 and (Group#2::Choice #1 or Group#2::Choice #2) and Option #2::True
```
11. Select Test mode.
12. Select Group #1::Choice #1. and Group#2::Choice #1. Option #3 is hidden.

13. Set Option #2 to True. Option #3 is now visible.



### Example 3: Visibility of Choice Groups, Choices, and Options in a Derived Object Wizard

This example illustrates how a derived Object Wizard with hidden Choices and Options is translated at run time and test mode:



In the example above, the following items are hidden:

- Group: Limit Switches
- Choice: 4-Way (in the Group Valve Type)
- Choice: Both (in the Groups Limit Switches (2-Way) and Limit Switches (3-Way))
- Option: State Alarm
- Option: Deviation Alarm - Tuning

### Trimming

Trimming refers to the ability of the Object Wizard to remove unneeded attributes, symbols, and scripts from derived instances. By removing unneeded items, the size of instances can be minimized, thus allowing more efficient utilization of run-time system resources. This can also result in faster deployments.

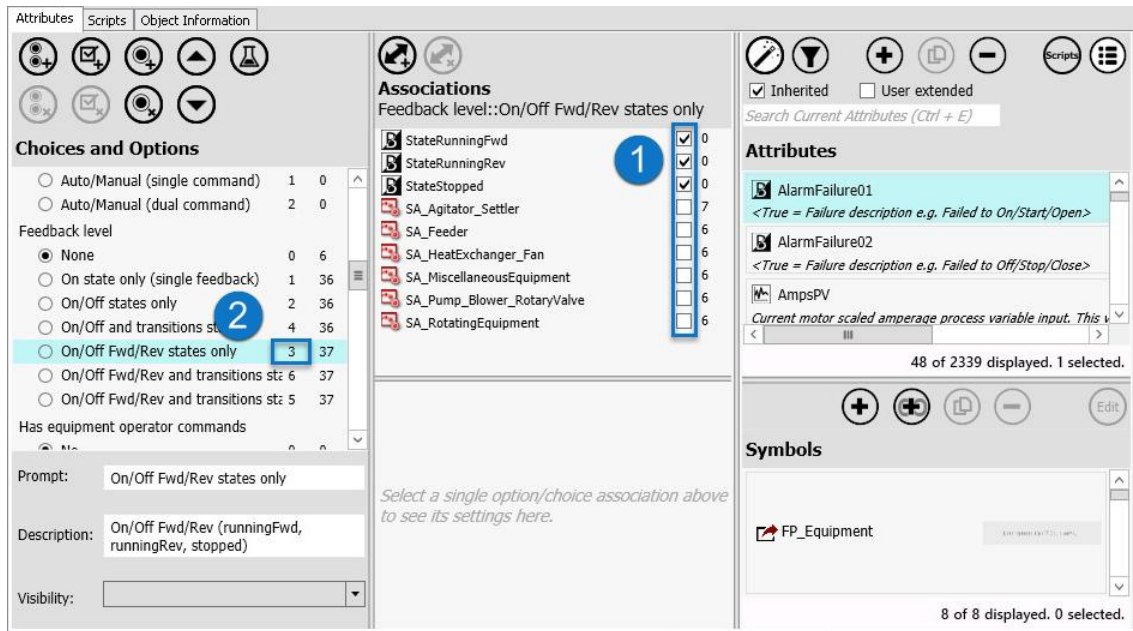
- When you associate an attribute with a Choice or Option, the **Trim checkbox** (1) is checked by default. This means that trimming is enabled, and the attribute will be evaluated for trimming.

- When you associate a symbol with a Choice or Option, the **Trim checkbox** is unchecked by default. This means that trimming is DISabled, and the symbol will NOT be evaluated for trimming.
- When you associate a script with a Choice or Option, the **Trim checkbox** is checked. Trimming cannot be disabled for scripts.

**Note:** Symbols do not exist as run-time objects, but do exist as part of a ViewApp. Trimming unnecessary symbols can help to reduce system resource requirements.

The total number of trimmable items (attributes, symbols, and scripts) associated with each Object Wizard Choice and Option is displayed in the **Choices and Options pane** (2). Items that are not enabled for trimming are not included in this count. The default trim state of associated symbols is disabled.

**Note:** The count of trimmable items resets to 0 in derived templates.



## Enable Trimming

Whether or not an attribute or symbol associated with a Choice or Option will be evaluated for trimming is determined by the state of its trimming checkbox in the **Associations pane**. See *Object Wizard User Interface Details* on page 231 for additional information.

**Note:** Scripts are always evaluated for trimming.

### To enable trimming:

1. Select the **Trim checkbox** in the **Associations pane**.
  - When checked, trimming is enabled.
  - When unchecked, trimming is disabled.
2. Symbols and attributes are evaluated for trimming when an instance is derived, in accordance with the following behaviors (scripts are always evaluated for trimming):
  - If none of its associations have trimming enabled, the item is not evaluated for trimming and is always propagated to derived instances.

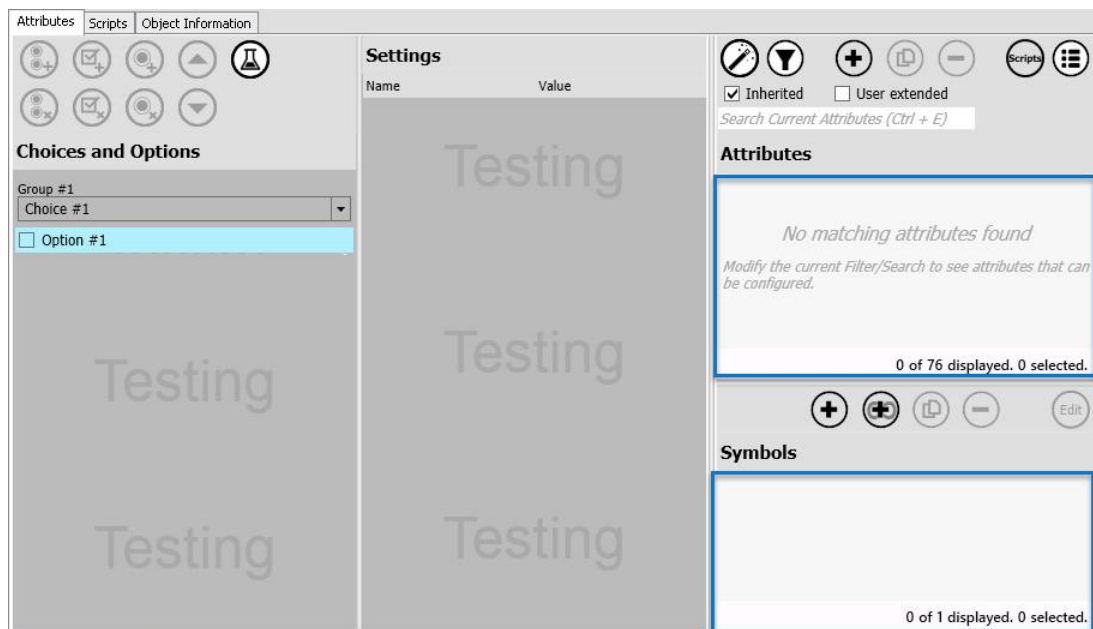
- The item is propagated to the instance if any Choice or Option where trimming is enabled is selected.
- The item is trimmed if at least one of its associations has trimming enabled, and none of the associations with trimming enabled are selected (even if associations with trimming disabled have been selected).
- Trimming occurs only when the instance is saved.

**Note:** Trimming is not applicable to derived templates, since these are not run-time objects.

## Example: Enable Trimming for Attributes and Symbols

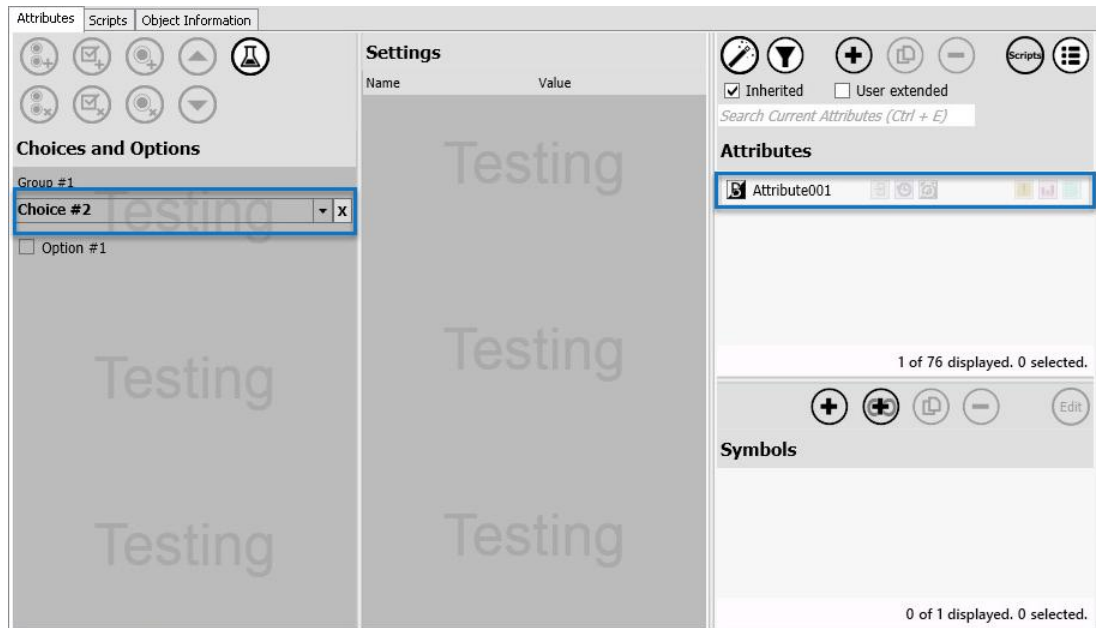
When you associate an attribute with a Choice or Option and set it as trimmable, you can use Test mode to see how attributes and symbols will be added to, or removed from, instances that are configured from the Object Wizard. For example:

1. Create a derived template and open it in the Object Editor.
2. Add an attribute: Attribute001.
3. Add a symbol: S1.
4. Add a Choice Group (with two Choices) and one Option. Leave Choice #1 as the default choice, and leave the default state of Option #1 as not selected.
5. Associate Attribute001 and S1 with Choice #1, and disable trimming for both.
6. Associate Attribute001 with Choice#2, and enable trimming for the attribute.
7. Associate S1 with Option #1, and enable trimming for the symbol.
8. Select **Test mode**. Note that with the default selections, Attribute001 does not appear in the **Attributes pane**, and S1 does not appear in the **Symbols pane**.

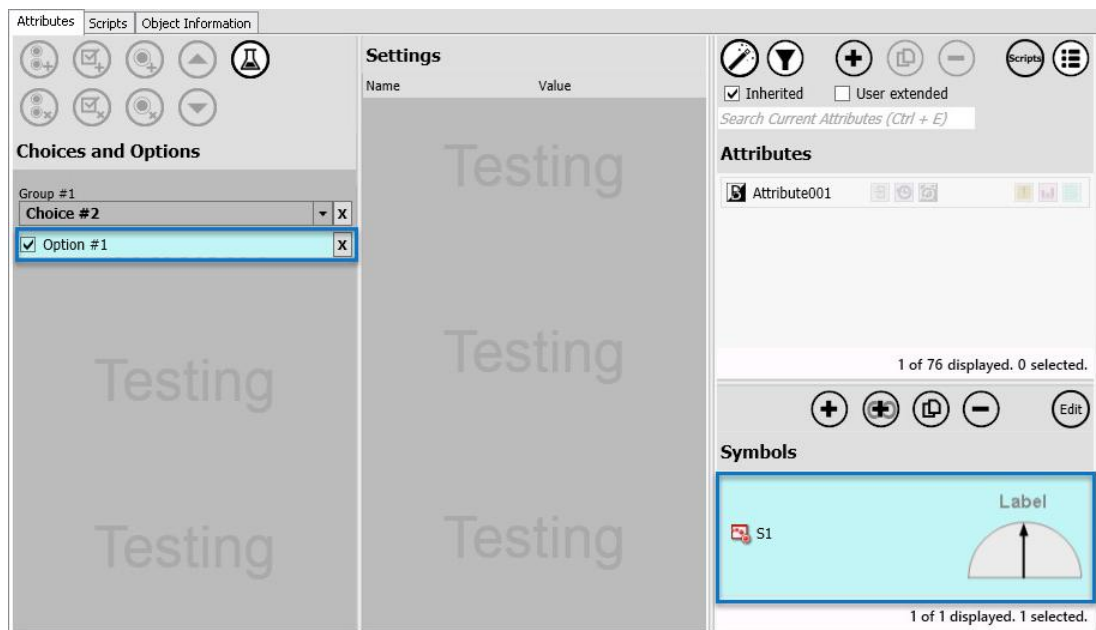




9. Select Choice #2. Note that Attribute001 now appears in the **Attributes** pane.



10. Enable Option #1. Note that S1 now appears in the **Symbols** pane.



## Attribute and Symbol Overrides

You can associate different configurations of symbols and attributes with different Object Wizard Choices and Options. You can also allow users to override values that you previously set as they configure derived objects. See *Configurable Settings for Attribute and Symbol Values* on page 261 for additional information about setting overrides and visibility for symbol wizard and custom property values. For detailed information about Symbol Wizards and custom properties, see "Using Custom Properties" and "Working with Symbol Wizards" in the *Creating and Managing Industrial Graphics User Guide*.

The **Settings** pane lets you configure setting overrides and visibility for associated attributes and symbols.

## Attribute Overrides

For attributes, you can override default settings and set visibility for:

- Attribute description
- Data type
- Initial value
- Engineering units (if applicable)
- Activate/deactivate features, such as I/O and alarms
- Feature values

## Attribute Setting Visibility

When you associate an attribute with an Object Wizard Choice or Option, its settings are **hidden by default**. You can make settings visible by selecting the checkbox next to the setting. You can also make overrides visible.

## Symbol Overrides

For symbols, you can override default settings and set visibility for:

- Symbol Wizard Choices and Options
- Custom properties

## Symbol Setting Visibility

When you associate a symbol with an Object Wizard Choice or Option, Symbol Wizard and custom property settings (if any) are **visible by default**. If you override a symbol custom property or Symbol Wizard setting, its visibility automatically turns off (the setting is hidden). You can make an overridden custom property setting visible again by selecting the checkbox next to the setting.

---

**Note:** If you override a Symbol Wizard setting, the only way to make the setting visible again is to remove the override.

---

Custom properties can be Boolean, integer, float, string, etc. See *Object Wizard User Interface Details* on page 231 for additional information. For detailed information about custom property data types, see "Using Custom Properties" in the *Creating and Managing Industrial Graphics User Guide*. When configuring string, time, and elapsed time custom properties, a toggle is provided in the **Settings pane** to configure the custom property as either:

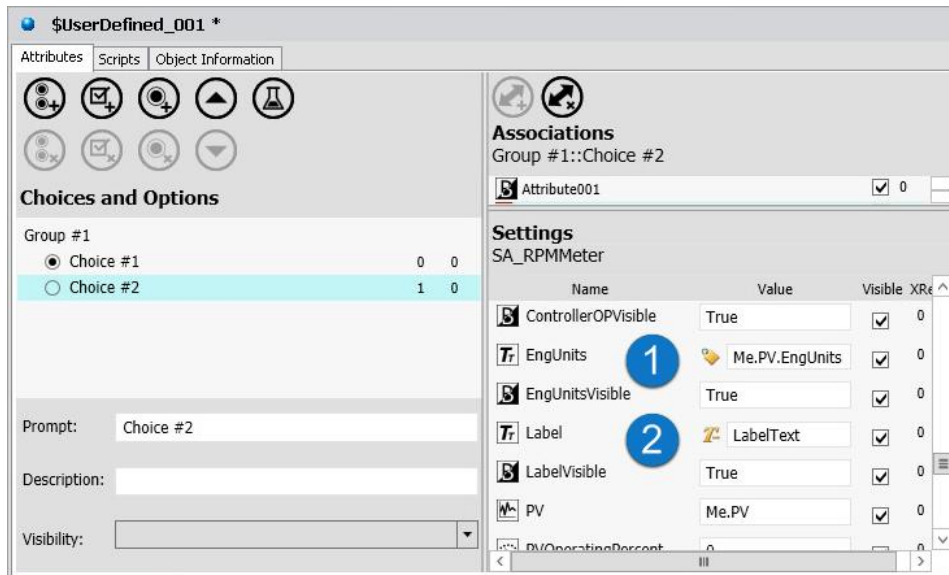


Reference string or an expression. Reference strings will be evaluated to determine their meanings. For example, `Me.Tank1.InletValve.PV`



String literal. String literals are passed as static text.

To change the string type, select the icon next to the text field.



## Enter Overrides and Set Visibility for Attribute and Symbol Values

For both attributes and symbols that contain a wizard or custom properties, you can enter overrides that apply only to the associated Object Wizard Choice or Option. While the following procedure shows how to configure a symbol, the procedure for configuring attributes is virtually identical. See also *About Attribute and Symbol Overrides* on page 240.

### To configure a Overrides for a Choice or Option

1. Select the Choice or Option you want to configure (**Choices and Options pane**).
2. Select a symbol or attribute (**Associations pane**) associated with the Choice or Option. See Add Attributes, Symbols and Scripts to an Object Wizard for information about associating symbols with Object Wizards.

This does not apply to scripts since they can only be associated with a Choice or Option. There are no configuration or visibility settings that can be configured.

3. In the **Settings pane**, override the default values as needed. Overrides are shown in **bold** and by an **X** next to the new value. See
  - o You can resize columns in the **Settings pane** as needed by selecting and moving the boundary between column headers.
  - o Symbol Wizard and Custom Property settings are visible by default. Overrides are automatically hidden from users.
    - You can only restore the visibility of a Symbol Wizard setting with an override by removing the override.

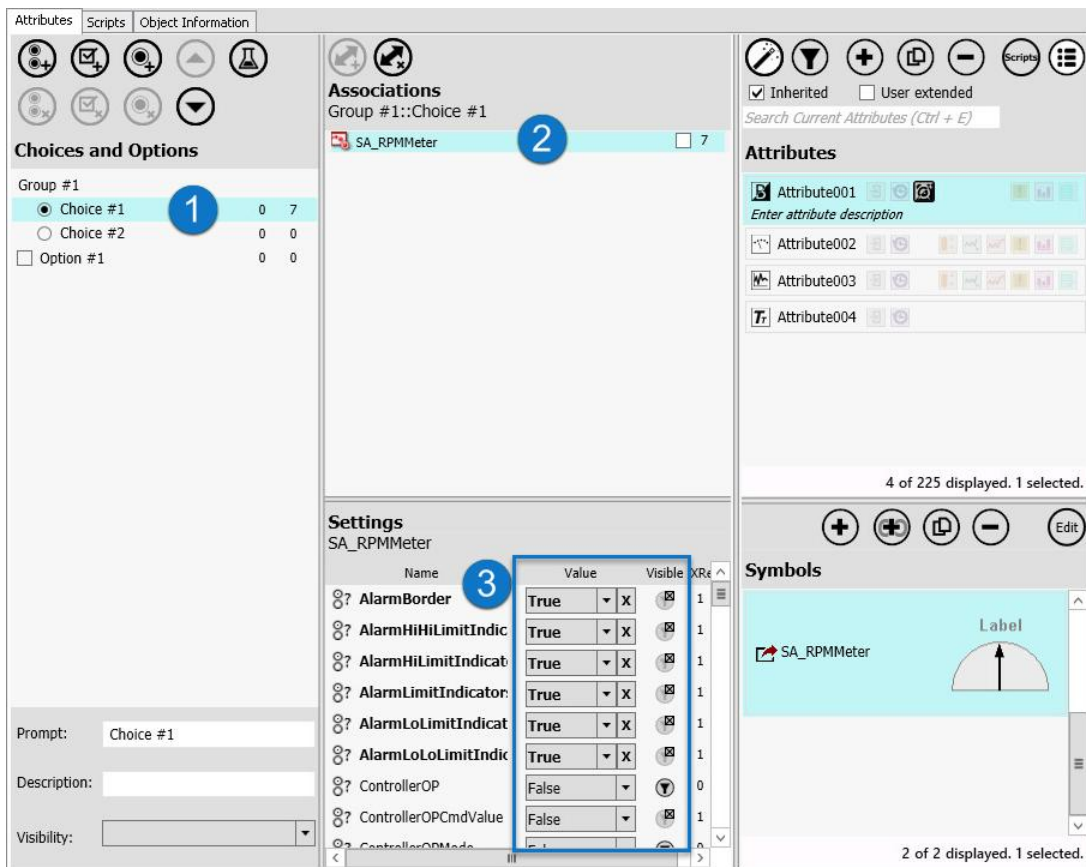
---

**Note:** This behavior is unique to Symbol Wizard overrides. You can change visibility settings for both Custom Property overrides and attribute overrides.

---

- You can restore the visibility of an Custom Property with an override by selecting the Visibility icon until it is unmuted. The icon will have a small X in its upper right quadrant to indicate that Visibility has been overridden.
- o Switching a Custom Property string type between expression/reference and static text is an override (value is displayed with **bold** type with an **X** next to the value and in the Visibility setting).

- o Attribute settings are hidden by default. You can override the Visibility settings for attributes that use their default setting, as well as for attributes that have an override.



**Note:** Attribute, Symbol Wizard, and Custom Property values that are marked as visible can be overridden by users as they configure instances.

**Note:** The total number of configured settings (visibility settings and value settings) is shown in the **Associations pane**, and is reflected in the **Choices and Options pane**. See *About Attribute and Symbol Overrides* on page 240 for additional information.

## About Attribute and Symbol Overrides

Override values for an attribute or symbol (with a Symbol Wizard or custom properties) can be configured for each associated Choice or Option. When you associate an attribute or symbol, the default values are listed in the **Settings pane**. To configure values that apply to the selected Choice or Option, enter the values in the **Settings pane**. If the attribute or symbol is associated with multiple Choices or Options, different values can be used for each one. Values for the Choices and Options are saved when you save the object. If the setting is marked as visible, users can override your setting (or override the default, if you have not configured a setting) as they derive and configure instances from the Object Wizard.

Symbols with overrides that affect their shape, for example, certain Situational Awareness Library symbols, are rendered with the applicable shape-changes as

**Note:** If you override a Symbol Wizard setting, the only way to make the setting visible again is to remove the override. Attributes and custom properties with overrides can be made visible again by selecting the visibility checkbox.

A configured setting is defined as any change from the default configuration of the attribute or symbol. This includes changing a value or visibility. For example, you might change the default Boolean value from true to false, or you might change an integer value from 5 to 10, or change engineering units from liters to cc. Changing the visibility of a setting also counts as a configured setting.

- The default visibility setting for attribute values is hidden (Visibility icon is muted (grayed-out) in the **Settings pane**).
- The default visibility setting for Symbol Wizard and custom property values is visible (Visibility icon is unmuted in the **Settings pane**).
- Visibility overrides are indicated by a small X in the upper right quadrant of the visibility icon.

The total number of configured settings for each individual attribute and symbol is displayed in the **Associations pane (1)**. The total number of configured settings for ALL attributes and symbols associated with an individual Choice or Option is displayed in the **Choices and Options pane (2)**.

The screenshot displays the Object Wizard interface for configuring an attribute. The interface is divided into several panes:

- Associations pane (1):** Shows a table with columns for Name, Value, Visible, and XRe. The total number of configured settings for the selected attribute is 2.
- Choices and Options pane (2):** Shows a table with columns for Choice/Option name and a total of 10 settings for the selected choice.
- Settings pane:** Shows a table with columns for Name, Value, and Visible. The total number of configured settings for the selected attribute is 22.
- Attributes pane:** Shows a list of attributes with a total of 4 settings for the selected attribute.
- Content pane:** Shows a list of content items with a total of 2 settings for the selected content item.
- Scripts pane:** Shows a list of scripts with a total of 2 settings for the selected script.

## Configurable Settings for Attribute and Symbol Values

Symbols and attributes can be configured for each Choice or Option with which they are associated. When you initially associate an attribute or symbol with a Choice or Option, the default settings of the attribute or symbol are listed in the **Settings pane**.

**Note:** Icons for attribute and symbol settings are defined in *Object Wizard User Interface Details* on page 231.

To override default values, enter the new values in the **Settings pane (1)**. If the attribute or symbol is associated with multiple Choices or Options, different overrides can be used for each one. These configured (override) values are saved when you save the template.

An override is defined as any change from the default symbol configuration. For example, you might change the default Choice of a Symbol Wizard, or change a custom property or attribute value. Changing the visibility of a setting is also considered an override. By default, symbol settings are visible in derived objects, while attribute settings are hidden by default.

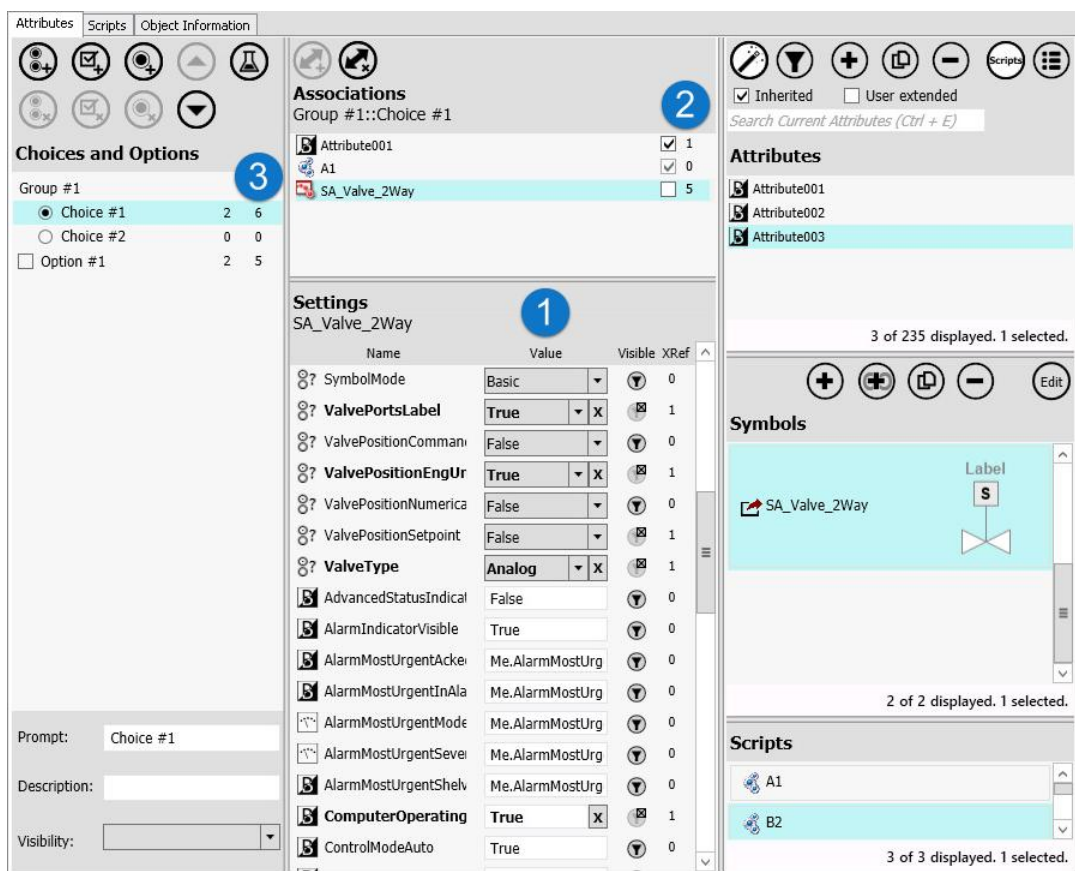
If the setting is marked as visible, users can enter overrides when they use the **Create New Asset** editor to configure a new instance. If the setting is hidden, it is not exposed to users and therefore the default value cannot be overridden.

Symbol Wizard overrides are automatically hidden and cannot be changed back to visible, except by restoring the original setting. The names of configured settings are displayed in bold type, and have an "X" next to the setting. To restore the original setting, clear the X.

Configured visibility settings have a small x in the upper right quadrant of the visibility icon. To restore the original, click on the icon twice to clear the override. If the original visibility state was visible, the icon will cycle through override - hidden, override - visible, and then return to its original state of visible (no override).

Custom property and attribute overrides are also hidden automatically. However, you can override the visibility setting for a setting that has an override value and make it visible again.

The total number of configured settings for each individual attribute and symbol is displayed in the **Associations pane (2)**. The total number of overrides for ALL attributes and symbols associated with a Choice or Option is displayed in the **Choices and Options pane (3)**.



## Object Wizard Derivation

You can derive child templates from templates with Object Wizards. Creating a second (or deeper) level of derivation from a template with an Object Wizard lets you extend the functionality of the original Object Wizard, and make specialized versions for specific applications. See *Reasons to Derive a Template from an Object Wizard* on page 264 for additional information.

### Derive a Child Template from a Template with an Object Wizard

To create a derived template from a template with an Object Wizard:

1. Select the parent template with Object Wizard in the Template Toolbox.
2. Do one of the following:
  - Right click on the template and select **Derived Template**.
  - Use the keyboard shortcut: **Ctrl + Shift + N**.

You will notice that the child template looks different than the parent template.

Choice Group	Option	Count	Value
Wizard Mode	Basic	0	0
	Advanced	0	0
Valve Type	2-Way	0	4
	3-Way	0	8
	4-Way	0	8
Limit Switches (2-Way)	None	0	0
	Open only	0	0
	Close only	0	0
	Both	0	0
Limit Switches (3-Way)	None	0	0
	Open only	0	0
	Close only	0	0
	Both	0	0
Limit Switches (4-Way)	None	0	0
	Open only	0	0
	Close only	0	0
	Both	0	0
State Alarm	State Alarm	0	0
	DEVIATION ALARMS	1	8
	Deviation Alarm - Major	1	2
	Deviation Alarm - Minor	1	3
	Deviation Alarm - Tuning	1	9

*Original Object Wizard*

Choice Group	Option	Count	Value	Visibility
Wizard Mode	Basic	0	0	▼
	Advanced	0	0	▼
Valve Type	2-Way	0	0	▼
	3-Way	0	0	▼
	4-Way	0	0	▼
Limit Switches (2-Way)	None	0	0	▼
	Open only	0	0	▼
	Close only	0	0	▼
	Both	0	0	▼
Limit Switches (3-Way)	None	0	0	▼
	Open only	0	0	▼
	Close only	0	0	▼
	Both	0	0	▼
Limit Switches (4-Way)	None	0	0	▼
	Open only	0	0	▼
	Close only	0	0	▼
	Both	0	0	▼
State Alarm	State Alarm	0	0	▼
	DEVIATION ALARMS	1	8	▼
	Deviation Alarm - Major	1	2	▼
	Deviation Alarm - Minor	1	3	▼
	Deviation Alarm - Tuning	1	9	▼

*Derived Object Wizard*

See *Object Wizard Visibility Settings in a Derived Template* on page 248 for additional information.

### Set Visibility of Choices and Options in a Derived Template

The original Object Wizard cannot be changed in the child derived template. However, you can set the visibility of Choice Groups, Choices, and Options. You can also set different defaults for Choice Groups, and you can turn Options on or off.

To set Object Wizard visibility in a derived template, do any of the following:

1. Select the visibility icon next to a Choice Group, Choice, or Option.
  - When you hide a Choice Group, all its Choices are hidden as well.
  - When you hide a Choice, the remaining Choices in the Choice Group are still visible.
  - When only one Choice is visible in a Choice Group, the entire Choice Group is hidden.

---

**Note:** The system applies the default values and actions of hidden Options and Choice Groups when a user derives an instance from the template. Visibility does not change configuration, it only affects aspects of the wizard that are visible to the user.

---

2. Change the default Choice in a Choice Group. This hides the Choice Group.

You can select the visibility icon to make the Choice Group visible again. You can then set visibility of the Choices individually.
3. Change the default state of an Option. This hides the Option.

You can select the visibility icon to make the Option visible again.

## Reasons to Derive a Template from an Object Wizard

There are two main reasons for deriving a child template from a template that contains an Object Wizard:

1. To extend the existing Object Wizard and thus, make it more capable. Available actions to accommodate the specific needs of a target solution include:
  - Add Choice Groups or Options.
  - Add attributes, symbols, or associations.
2. To create specialized versions of templates that preconfigure choices, and thus simplify the task of configuring instances. For example, you can define common configuration aspects that are applicable to a set of similar instances, or modify some of the characteristics of the existing Object Wizard. Available actions to simplify configuration include:
  - Change default Choices or settings.
  - Hide Choice Groups, Choices, or Options.
  - Enter overrides for attribute or symbol settings, or change their visibility settings.

Using derived templates lets you propagate changes made to a parent template, as long as the child template does not have an override for what has changed. See *Propagation of Attribute and Symbol Settings* for additional information.

If you are familiar with using Application Server, you will notice one other advantage of using derived Object Wizards: there is no need to lock objects. Changes automatically propagate to derived objects (provided the child template does not have an override for what changed in the parent), without explicitly locking attributes in the parent object.

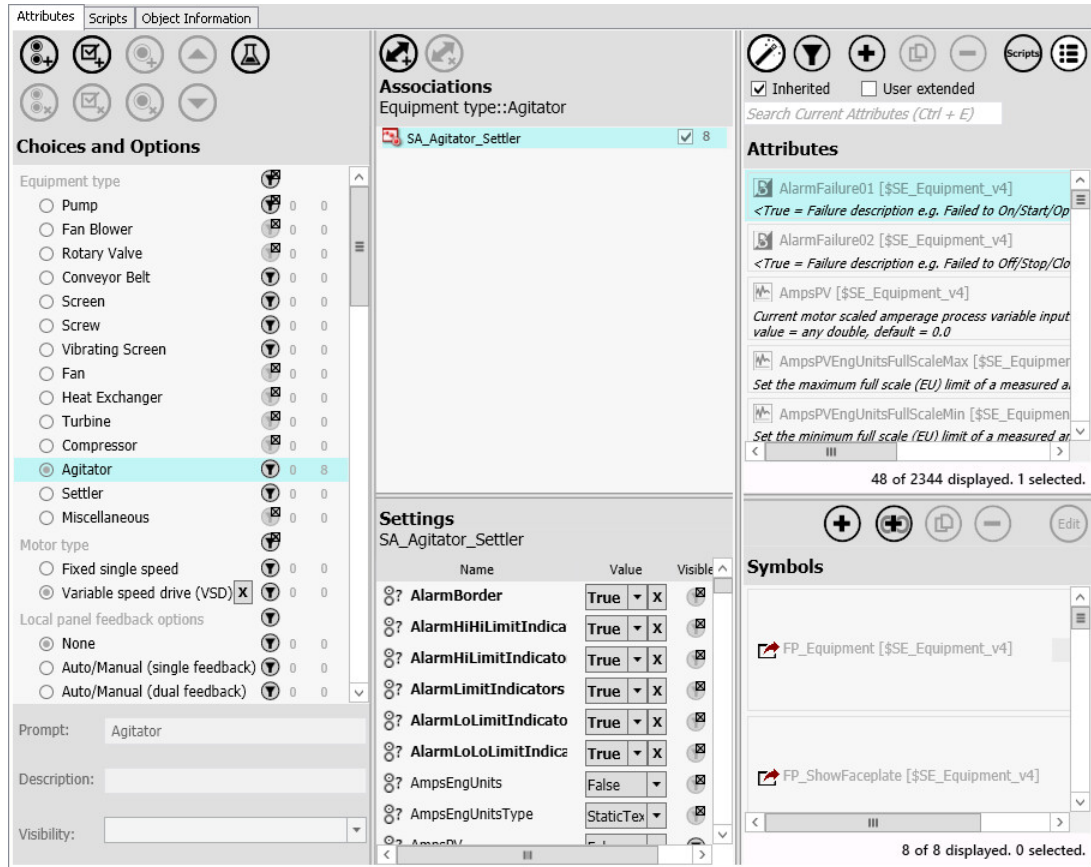
## Changes to Object Wizards in Derived Templates

After you create a derived template from a template with an Object Wizard, you can only make certain changes to the existing Object Wizard in the derived object.

- You can set overrides for attributes and symbols.
- You can hide or show settings for attributes and symbols.
- You can change visibility and defaults for Choices and Options.
- You can add new Choice Groups and Options (you cannot add new Choices to an existing Choice Group).



- You can add new attributes, symbols, and scripts, and associate them with the existing Object Wizard.
- You cannot remove existing associations from the Object Wizard.
- You cannot remove any existing Choice Groups, Choices, or Options.
- You cannot change the order of any existing Groups, Choices, or Options.
- You cannot change attribute, symbol, or script associations with any existing Choices or Options.



When a Choice or Option is changed from the default, the change is indicated by an **X** next to the modified Choice or Option. To revert to the default, select the **X**.

Similarly, when a visible setting for an associated attribute or symbol has been changed from the default, the change is indicated by an **X** next to the modified setting. To revert to the default, select the **X**.

## Behavior of Object Wizard Components in a Derived Template

The following list summarizes actions that you can and cannot perform in derived templates that contain an Object Wizard.

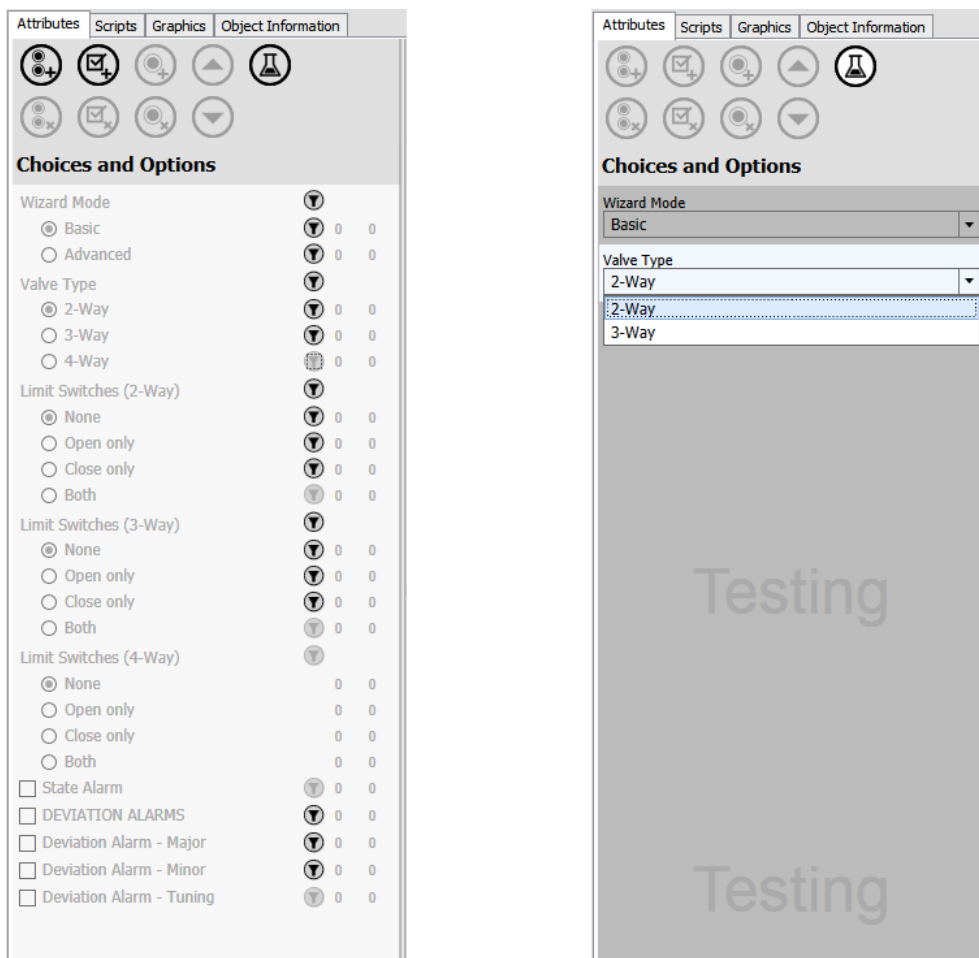
- Inherited Groups, Choices and Options cannot be edited.
  - Groups, Choices and Options cannot be renamed or deleted.
  - Their order cannot be changed (cannot be moved up or down).
  - Their prompts, descriptions, and visibility rules cannot be edited.
- You can change the default Choice, or the default state of an Option.

- Associations of inherited items (attributes, symbols, and scripts) with Choices and Options cannot be changed, and inherited associations cannot be removed.
- You can associate an inherited item with additional inherited Choices and Options
- Inherited attribute and symbol values and visibility settings can be overridden.
- Inherited values and visibility settings with overrides can be overridden.

**Note:** Even if a value is overridden in the parent object, the value will appear as a default value in the child object. This is because the override from the parent becomes the default for the child.

### Example: Set Visibility of Derived Wizard Choices and Options

The following example illustrates how a derived Object Wizard with hidden Choices and Options is interpreted in test mode. This simulates instance configuration.



In the example, the following items are hidden:

- Group: Limit Switches
- Choice: 4-Way (in the Group Valve Type)
- Choice: Both (in the Groups Limit Switches (2-Way) and Limit Switches (3-Way))
- Option: State Alarm
- Option: Deviation Alarm - Tuning

## Object Wizard Test Mode

Test mode lets you simulate how an Object Wizard will behave when a user configures a derived instance. Use Test mode as you develop your Object Wizard to ensure that it behaves as expected. For complex Object Wizards, associated attributes and symbols may not always appear as you had planned. Test mode lets you verify that derived instances will behave as intended.

- Select the **Test** button to enter Test mode.
- Deselect the button to exit Test mode.



Visibility expressions determine which Choice Groups, Choices, and Options are visible or hidden (along with associated symbols, attributes, settings, and configured overrides), based on previous selections.

## Verify Behavior of Visibility Expressions

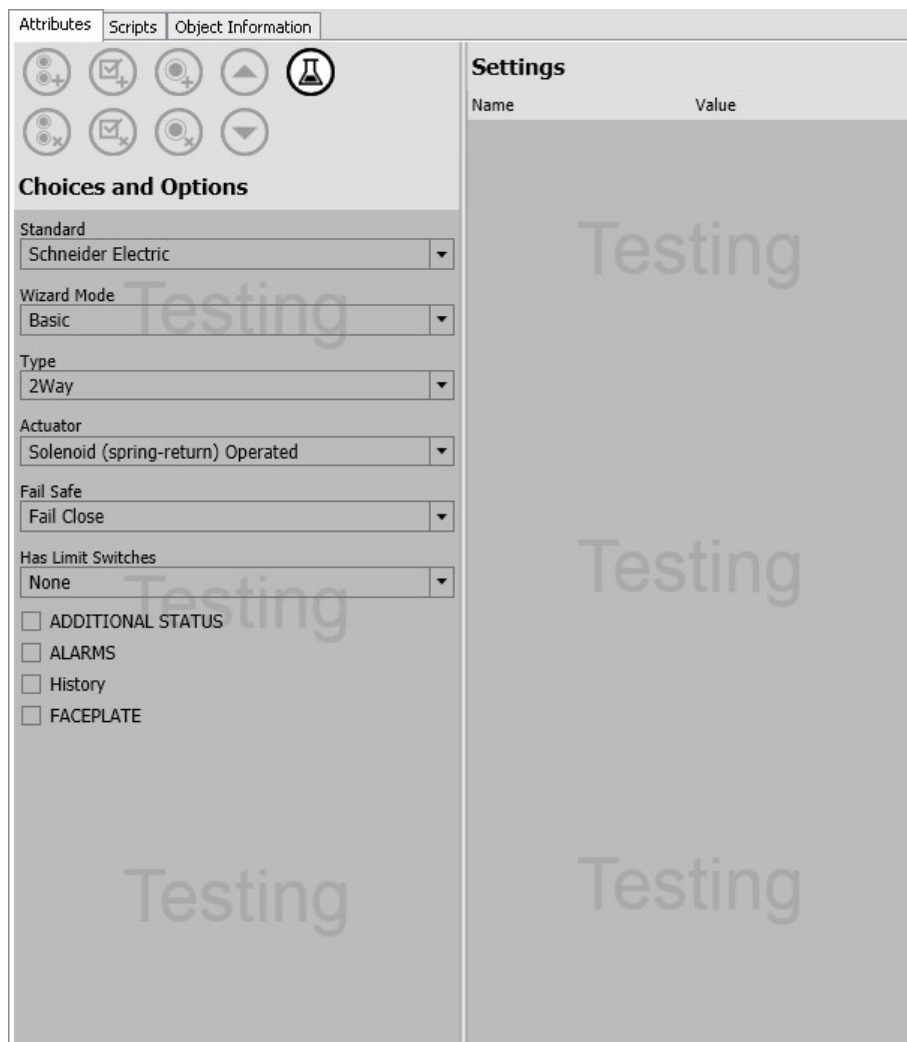
1. Open the template, containing the Object Wizard to be tested, in the **Object Editor**.

The screenshot shows the Object Editor interface with the following sections:

- Attributes | Scripts | Object Information** (Tabs)
- Choices and Options**
  - Standard**

<input checked="" type="radio"/> Schneider Electric	0	0
<input type="radio"/> Rockwell	0	0
<input type="radio"/> Foxboro	0	0
<input type="radio"/> Siemens	0	0
<input type="radio"/> Eurotherm	0	0
<input type="radio"/> Generic	0	0
  - Wizard Mode**
    - Basic (0 0)
    - Advanced (0 0)
  - Type**
    - 2Way (2 25)
    - 3Way (1 0)
    - 4Way (1 0)
  - Actuator**
    - Solenoid (spring-return) Operated (1 8)
    - Motor (rotary) Operated (3 8)
    - Analog Operated (Control Valve) (2 4)
    - Manual (hand) Operated (1 8)
  - Control Output Signal**
    - ...
- Associations**
  - Standard::Schneider Electric
  - None
  - Drag selected attributes/symbols/scripts here or click on the '+' above to associate them.*
  - Select a single option/choice association above to see its settings here.*
- Prompt:** Schneider Electric
- Description:** [Empty field]
- Visibility:** [Dropdown menu]

2. Select the **Test** mode button. The background color of the **Choices and Options** and **Settings** panes darkens, and the **Associations** pane is no longer shown. Choice Groups and Options are displayed, based on their visibility settings.



3. Check that the Choice Groups and Options that you expect to be visible are shown.
  - o Regardless of the construction of the Object Wizard, the first item (Choice Group or Option) in the Object Wizard is always visible.
  - o Choice Groups are displayed as drop down lists, with only the default Choice visible.
4. Check that the expected default Choice for each visible Choice Group is shown.
5. Check that the default state (checked or unchecked) of each visible Option is correct.
6. Cycle through different Choice and Option selections, and verify that subsequent dependent items are hidden or shown as designed.
7. Verify that the settings for associated attributes and symbols are shown in the **Settings** pane, and that settings that should be visible are marked as visible. Verify that settings with visibility turned off are marked as hidden.

## About Object Wizard Behavior in Test Mode

At the most basic level, you can check that Object Wizard elements (Choice Groups, Choices, and Options) operate as intended. Choices and Options should appear in their default states. That is, Options that should be enabled by default should appear with their checkbox checked, and Choice Groups should show the default Choice. You can change these selections while in Test mode, to ensure that Choices and Options are correctly configured, including visibility settings and associated attributes, symbols, and scripts. Attribute and symbol settings that should not be changeable by consumers of the Wizard should not be visible.

- Check conditional visibility expressions. See *Conditional Visibility Expressions for Choice Groups, Choices, and Options* on page 244 for additional information.
- Check that attributes and symbols are shown in their correct configuration when the corresponding Choice or Option is selected. See *Enter Overrides and Set Visibility for Attribute and Symbol Values* (see "Enter Overrides and Set Visibility for Attribute and Symbol Values" on page 259) for additional information.
- Check that attributes and symbols are correctly trimmed. See *Trimming* on page 254 for additional information.

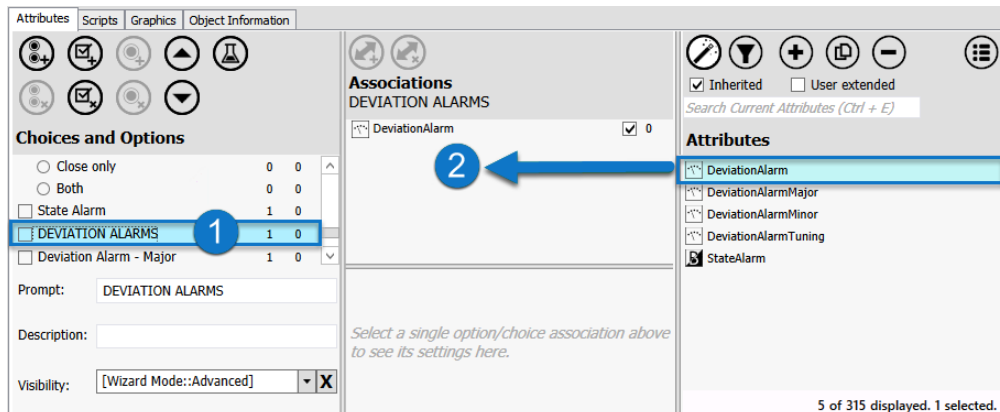
### Example 1: Verify Behavior of Associated Attributes

Here, we examine how the selection of Choices and Options affects attributes that are associated with them. For an overview of how an Object Wizard behaves in Test mode, see *About Object Wizard Behavior in Test Mode* on page 269.

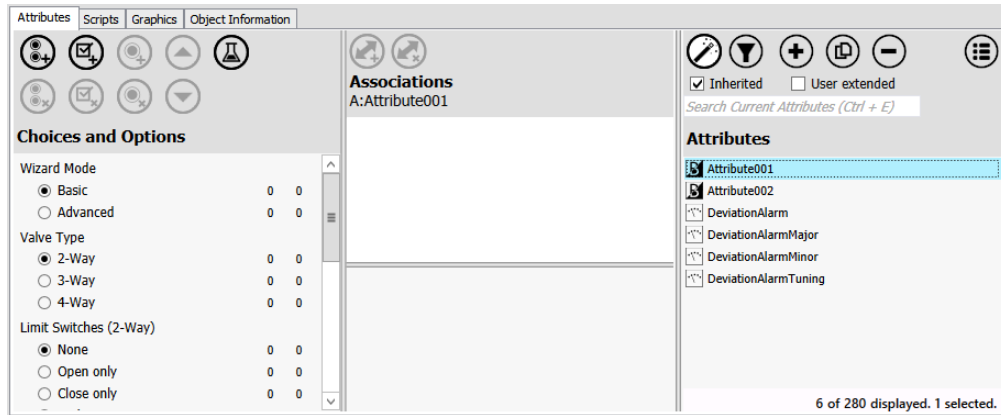
Let's start by creating new attributes and associate them with Options.

1. Create four new attributes.
2. Name the attributes as follows:
  - o DeviationAlarms
  - o DeviationAlarmMajor
  - o DeviationAlarmMinor
  - o DeviationAlarmTuning
3. Highlight one of the Deviation Alarm Options (1), then select the corresponding attribute and drag it to the **Associations pane** (2). For example, with the DEVIATION ALARMS option highlighted in the **Choices and Options pane**, drag and drop the DeviationAlarms attribute to the **Associations pane**.

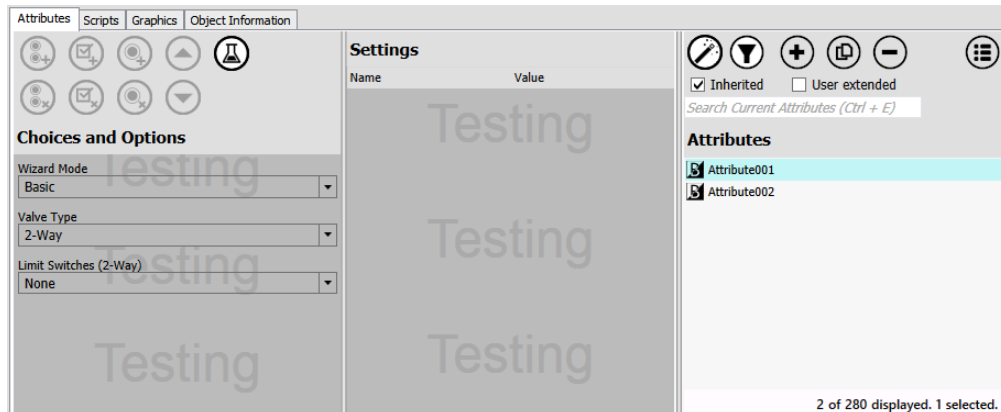
Make sure that the **Trimming checkbox** is selected in the **Associations pane**. See *Trimming* on page 254 for additional information.



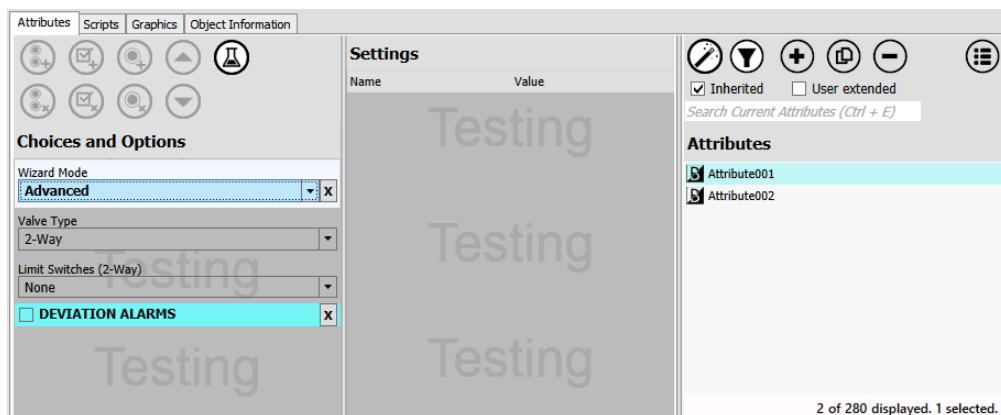
4. Repeat for the remaining Deviation Alarms with trimming enabled.
5. Create two more attributes: Attribute001 and Attribute002. We will leave these attributes unassociated with any particular Choice or Option. Since they are not a part of the Object Wizard, these attributes will be included in all objects that are derived from the template; these two attributes will not be trimmable.



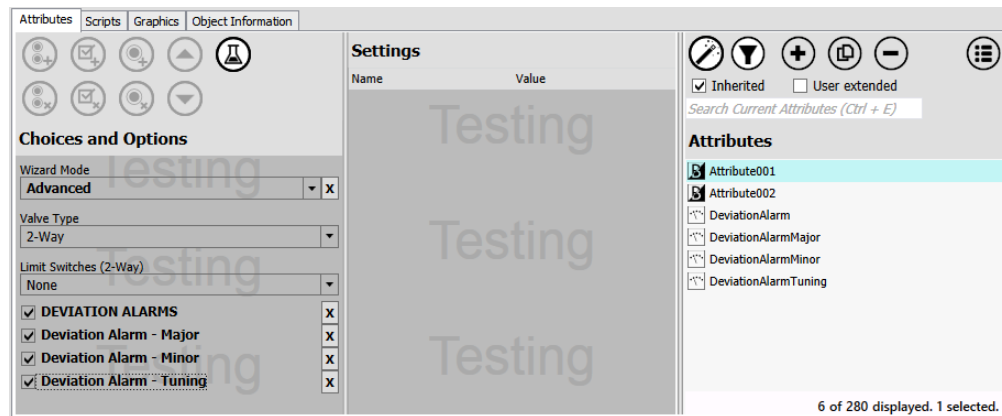
6. To verify attribute behavior, select the **Test** button.
  - o With no Object Wizard selections made, the attributes that are not associated with any Wizard Choices or Options (Attribute001 and Attribute002) are shown in the **Attributes** pane. Attributes associated with Choices or Options (DeviationAlarm attributes) are not shown.



- o Change the Wizard Mode to **Advanced**. This displays the DEVIATION ALARMS option (unchecked). Note that the associated DeviationAlarms attribute is not shown.



- To display the DeviationAlarm attribute, set the DEVIATION ALARMS option to TRUE (checked). Continue setting the remaining Deviation Alarm options to TRUE. Note that as each Option is selected, its corresponding attribute is displayed.



- Deselecting an Option (setting it to FALSE) turns off the attribute.

---

**Note:** When configuring an instance from an Object Wizard, unused attributes that have trimming enabled are not propagated to the instance. Only attributes that are not associated with the Object Wizard, and attributes with trimming enabled that evaluate as required, are present in the instance after the instance is saved. Trimming does not occur until after an object is saved. Any configured overrides for trimmable attributes remain in memory until the user saves the instance.

---

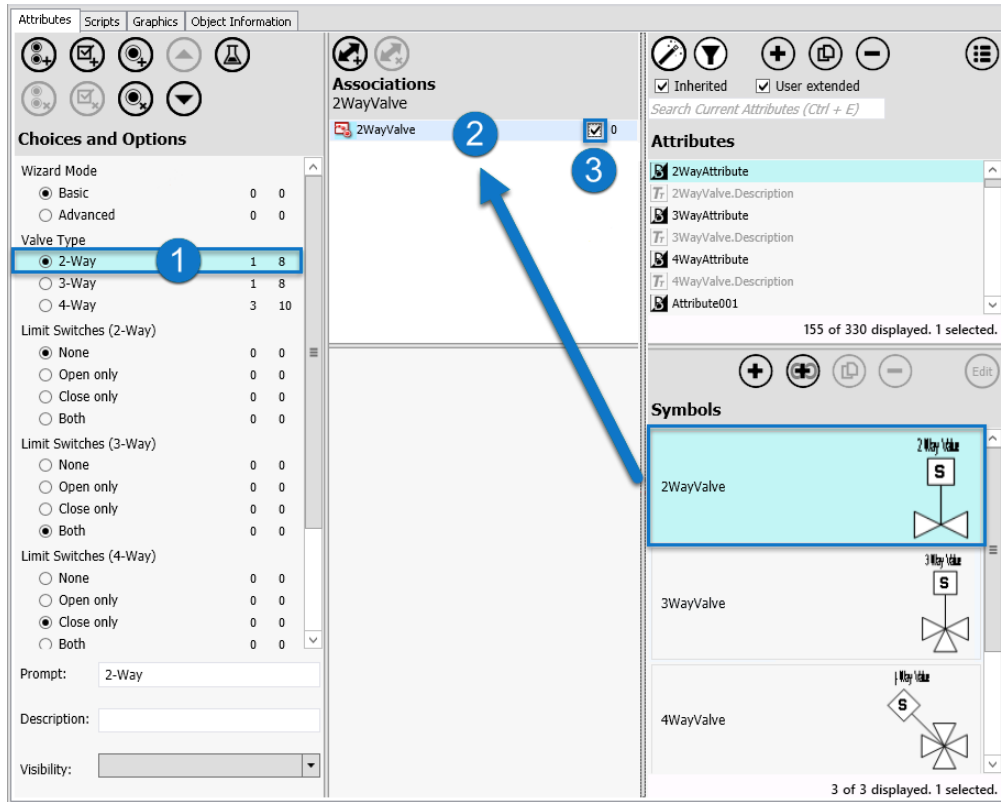
## Example 2: Verify Behavior of Associated Symbols

Here, we examine how the selection of Choices and Options affects symbols that are associated with them. For an overview of how an Object Wizard behaves in Test mode, see *About Object Wizard Behavior in Test Mode* on page 269.

Start by creating new symbols and associate them with some Object Wizard Choices.

1. Create three new symbols. In this example, each symbol contains an embedded Situational Awareness Library graphic.
  - 2WayValve
  - 3WayValve
  - 4WayValve
2. Highlight one of the Choices in the Valve Type Choice Group (1), then select the symbol with the corresponding name and drag it to the **Associations pane** (2). For example, with the 2-Way valve type Choice highlighted in the **Choice s and Options pane**, drag and drop the 2WayValve symbol from the **Symbols pane** to the **Associations pane**.

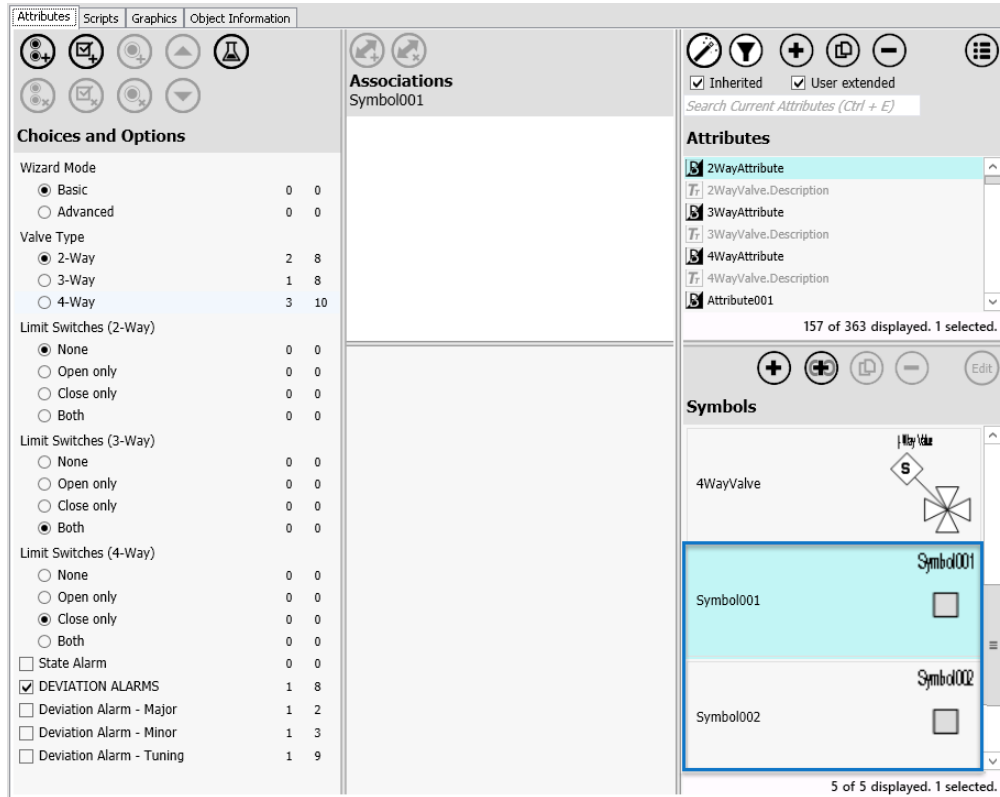
Make sure that the **Trimming checkbox** (3) is selected in the **Associations pane**; the default state for symbols is unselected. See *Trimming* on page 254 for additional information.



3. Repeat for the remaining valve types (with trimming enabled).

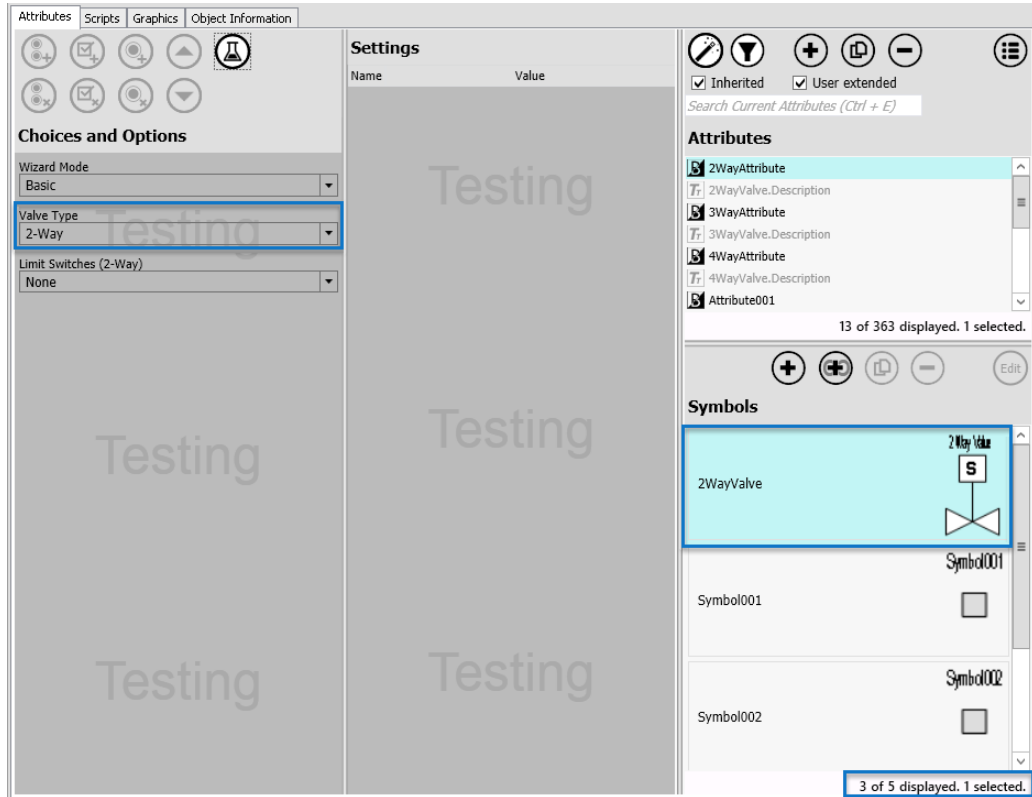


4. Create two more symbols: Symbol001 and Symbol002. We will leave these symbols unassociated with any particular Choice or Option. Since they are not a part of the Object Wizard, these symbols will be included in all objects that are derived from our Object Wizard. It does not matter if trimming is enabled for these, since without any Object Wizard association, they are considered to be owned by the object.

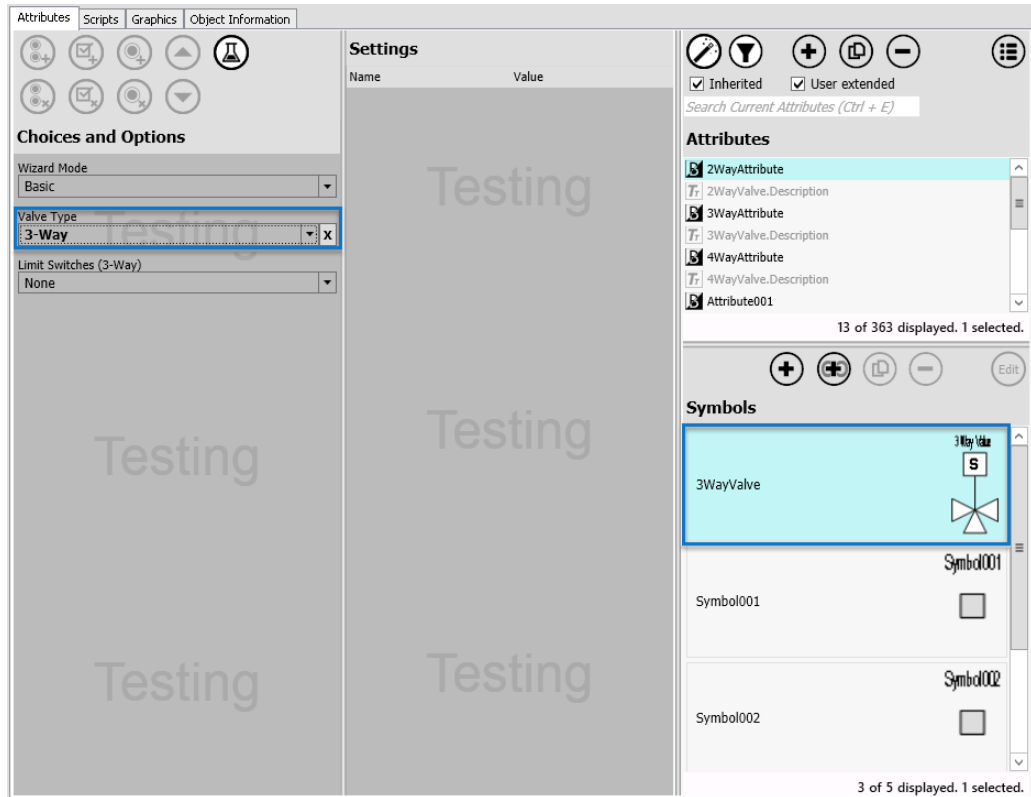


5. To verify attribute behavior, select the **Test** button.

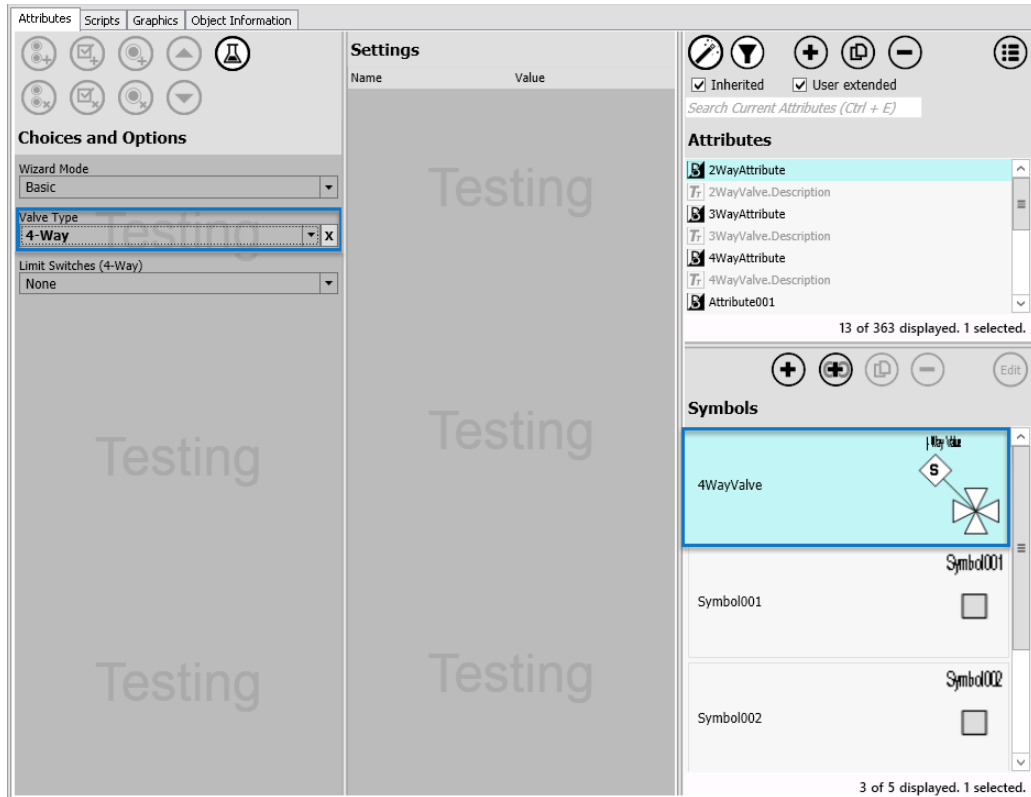
- With the default Object Wizard selections made (in this case, 2-Way Valve), the 2WayValve symbol associated with this choice is shown, along with the two symbols not associated with any Wizard Choices or Options (Symbol001 and Symbol002). These can be seen in the **Symbols pane**. Note the message at the bottom of the **Symbols pane** that states "3 of 5 displayed." This is because the 2WayValve and 3WayValve symbols are hidden.



- Change the Valve Type to **3-Way**. This displays the 3WayValve symbol and hides the 2WayValve and 4WayValve symbols.



- Change the Valve Type to **4-Way**. This displays the 4WayValve symbol and hides the 2WayValve and 3WayValve symbols.



**Note:** When configuring an instance from an Object Wizard, unused attributes that have trimming enabled are not propagated to the instance. Only attributes not associated with the Object Wizard, and attributes with trimming enabled that evaluate as required, are present in the instance after the instance is saved. Trimming does not occur until after an object is saved, and any overrides a user enters for a trimmable attribute remain in memory until the user saves the object.

## Propagation of Object Wizard Changes to Derived Objects

Most changes made to an Object Wizard propagate through the object hierarchy to all objects derived from the Wizard. Examples of changes that are propagated include:

- Adding, editing, renaming, changing the order of, or deleting Choice Groups, Choices, or Options.
- Adding, removing, or modifying an association (attribute, symbol, or script) with a Choice or Option
- Changing the default state (setting) of an Option.
- Changing the visibility setting of a Choice or Option.
- Changing the default value or visibility setting of an attribute or symbol associated with a Choice or Option.
- Changing the visibility setting of a value in the **Settings** pane.

**Important:** With the exception of the first two items in the above list, changes DO NOT propagate to derived templates if the setting is marked as overridden in the derived template.

## Object Wizard Behavior when a Default Choice is Deleted

If you remove a default Choice from a Choice Group, the topmost Choice automatically becomes the new default, unless you explicitly select a different one. The new default propagates to child objects (templates and instances).

You cannot delete a Choice if there are only two Choices in the Choice Group; you must delete the entire Choice Group (a Choice Group, by definition, must contain at least two Choices). In this case, any associated attributes or symbols that were associated with the deleted Choice Group become a part of the object; they no longer have an association with the Object Wizard. Attributes and symbols that are not associated with the Object Wizard are not trimmed from derived instances.

## Limitations Associated with Object Wizard Propagation

You cannot delete, rename, or change the order of Choice Groups, Choices, or Options in a derived Object Wizard. However, additional Choice Groups and Options can be added to a derived template with an existing Object Wizard. You can also selectively hide Wizard elements in the derived Object Wizard. Elements that are hidden will invoke a default behavior, or limit the Wizard consumer to choose between the remaining elements. For example, in a Choice Group with three Choices, you could hide Choice #2, leaving the Wizard consumer to choose between Choice #1 and Choice #3. See *Object Wizard Visibility Settings in a Derived Template* on page 248 for additional information.

See *Object Wizard Derivation* on page 263 for additional information about derived objects.

## Extend an Inherited Object Wizard

You can extend the capabilities of a template derived from an Object Wizard by adding different types of content to it. Actions you can take include:

- Expand the existing Object Wizard by adding new Choice Groups or Options.

---

**Note:** You cannot add Choices to an existing Choice Group.

---

- Associate attributes, symbols, and scripts with existing Choices and Options.
- Enter overrides for attribute and symbol settings.
- Change the default setting for Choice Groups.

---

**Note:** When you change an inherited default Choice, the entire Choice Group is automatically hidden. However, you can turn visibility back on for the Choice Group. After that, you can selectively hide Choices within the Choice Group, if necessary.

---

- Selectively hide Choice Groups, Choices, and Options.
- Change the default default state of Options.

---

**Note:** When you change an Option's default state, the Option is automatically hidden. However, you can turn visibility back on.

---

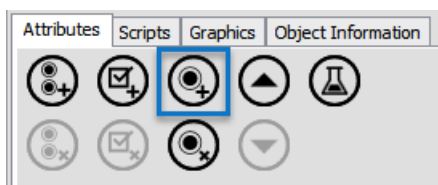
## Add a Choice Group to a Derived Template

To add a new Choice Group to a derived template with an Object Wizard:

1. Select the **Add Choice Group** button to add a Choice Group.



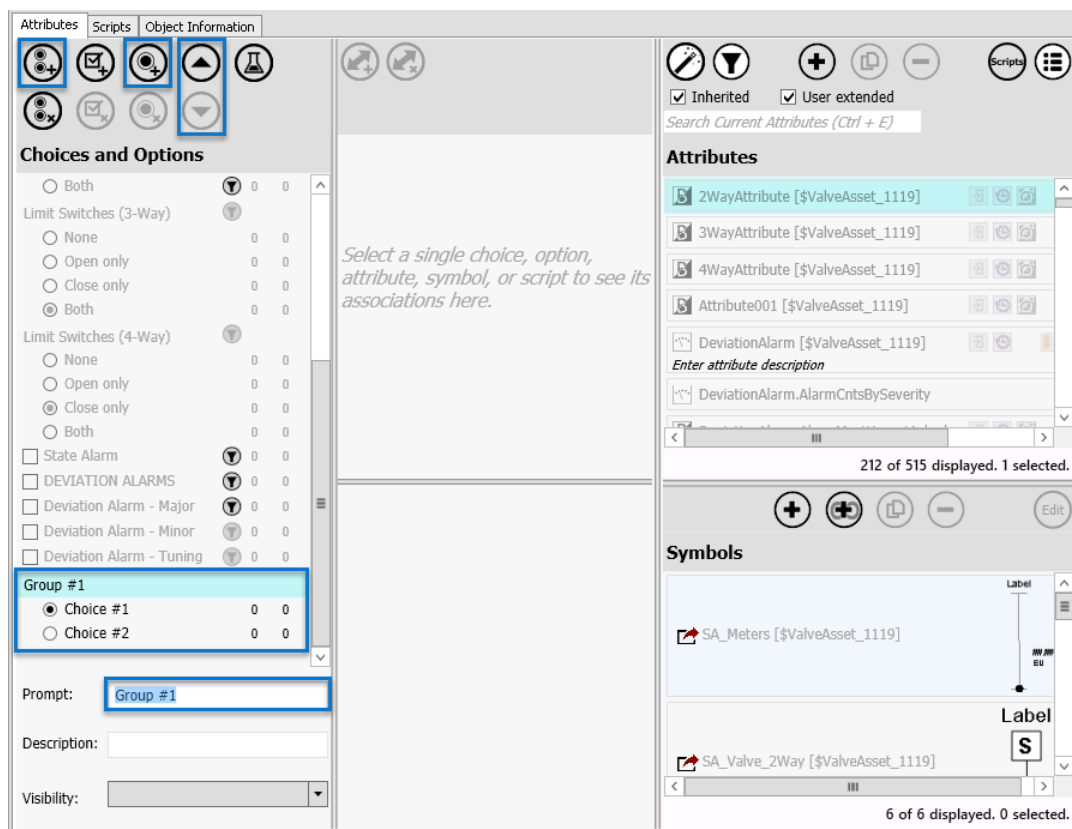
2. Rename the Choice Group by entering the new name in the **Prompt** textbox.
3. You can change the relative position of the new Choice Group as needed by using the **Move Up** and **Move Down** buttons.
4. Select the **Add Choice Group** button to add additional Choices, if needed. Choice Groups include two Choices by default.



Choices within a Choice Group are mutually exclusive; one, and only one, Choice can be selected from a Choice Group. Choices are chosen by selecting the radio button next to the Choice. For example, an Object Wizard user must choose Choice #1 OR Choice #2 (but not both or none).

5. Rename the Choice by entering the new name in the **Prompt** textbox.

**Note:** You cannot add Choices to an existing Choice Group in a derived template.



## Add an Option to a Derived Template

To add a new Option to a derived template with an Object Wizard:

1. Select the **Add Option** button to add a Choice Group.

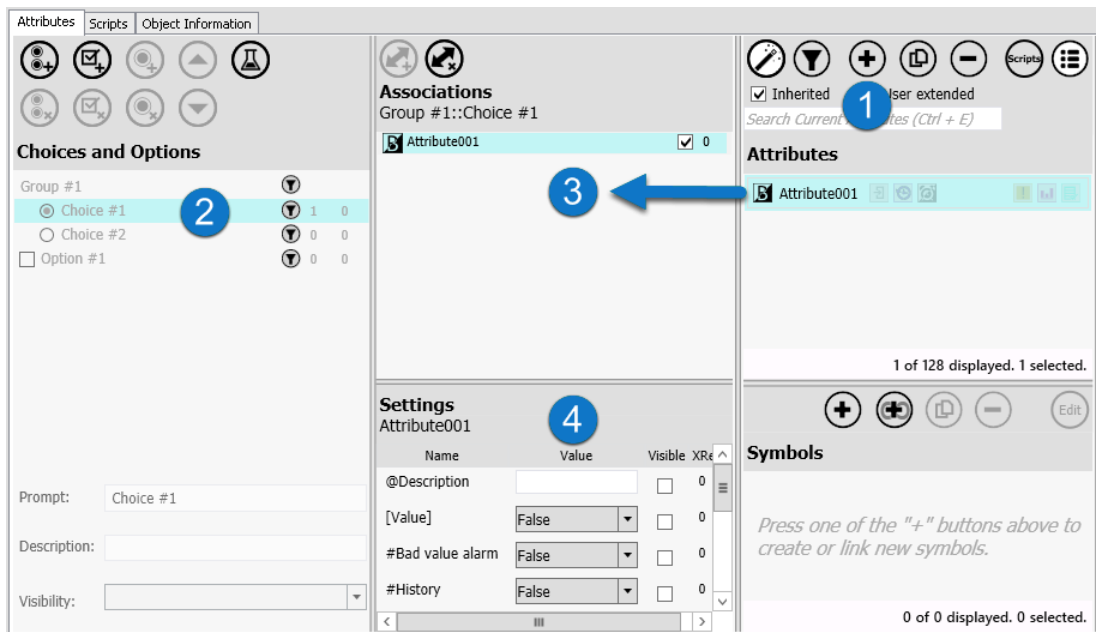


2. Rename the Option by entering the new name in the **Prompt** textbox.

## Add an Association to an Inherited Object Wizard

To add an association (attribute, symbol, or script) to the inherited Object Wizard:

1. Create a new attribute, symbol or script (as needed) and configure any of its base settings, as needed.
2. Highlight a Choice or Option in the **Choices and Options** pane.
3. Add the new item in the **Associations** pane.
4. Enter overrides (for attribute and symbol settings) as needed for associated the Choice or Option.



5. Add and configure additional items and associate them with the Object Wizard, as needed.

**Note:** A Choice or Option does not have to have any associations. An individual attribute, symbol, or script can be associated with more than one Choice or Option. Symbol and attribute override settings can override the default values of an attribute; the overrides affect only the Choice or Option for which they are entered. For example, an attribute can have a value of 5 for one Choice, and a value of 10 for another.

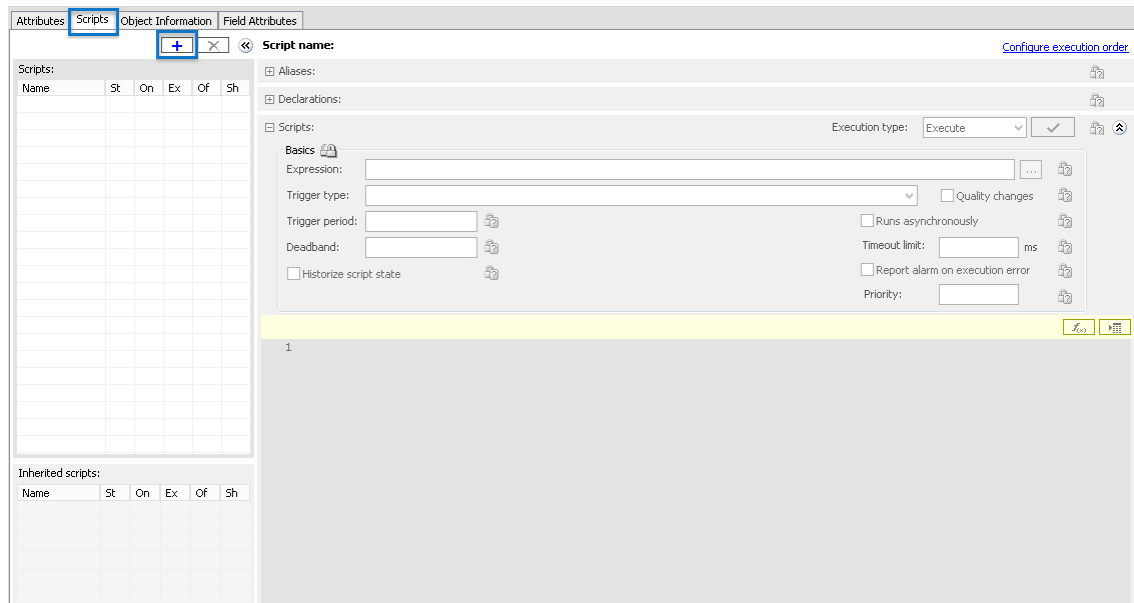
## Add a Script to a Derived Template

As with attributes, you can also add new scripts to existing Choices or Options. The scripts can be pre-existing within the object, or you can add new scripts to the object, and then associate them with the Object Wizard.

See the *Archestra Scripting Guide* for additional information about scripts.

To add an script association with the Object Wizard:

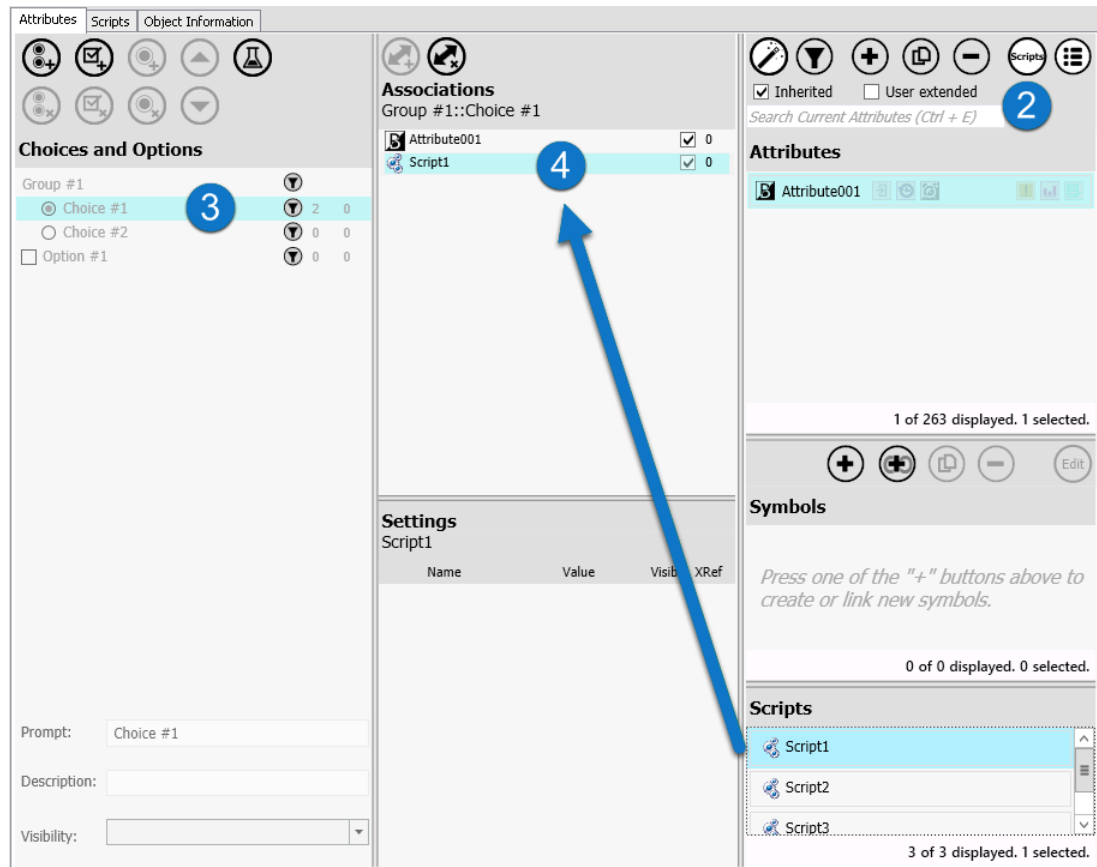
1. If you need to create a new script, select the **Scripts tab** and then select the **Add Script** button. See the Application Server User Guide, " *Writing and Editing Scripts on page 160,*" for additional information.



2. If necessary, select the Scripts button to show the **Scripts pane**.
3. Highlight a Choice or Option in the **Choices and Options pane**.
4. Drag and drop the script from the **Scripts pane** to the **Associations pane**.



No additional configuration is needed for scripts. If you need the script to modify a script for another Choice or Option, create a copy of the script and change it.



5. Add additional scripts and associate them with the Object Wizard, or associate the same script with other Choices or Options.

## Add a Symbol to a Derived Template

To extend or change the functionality of a derived template with an Object Wizard, you can add new symbols to existing Choices or Options.

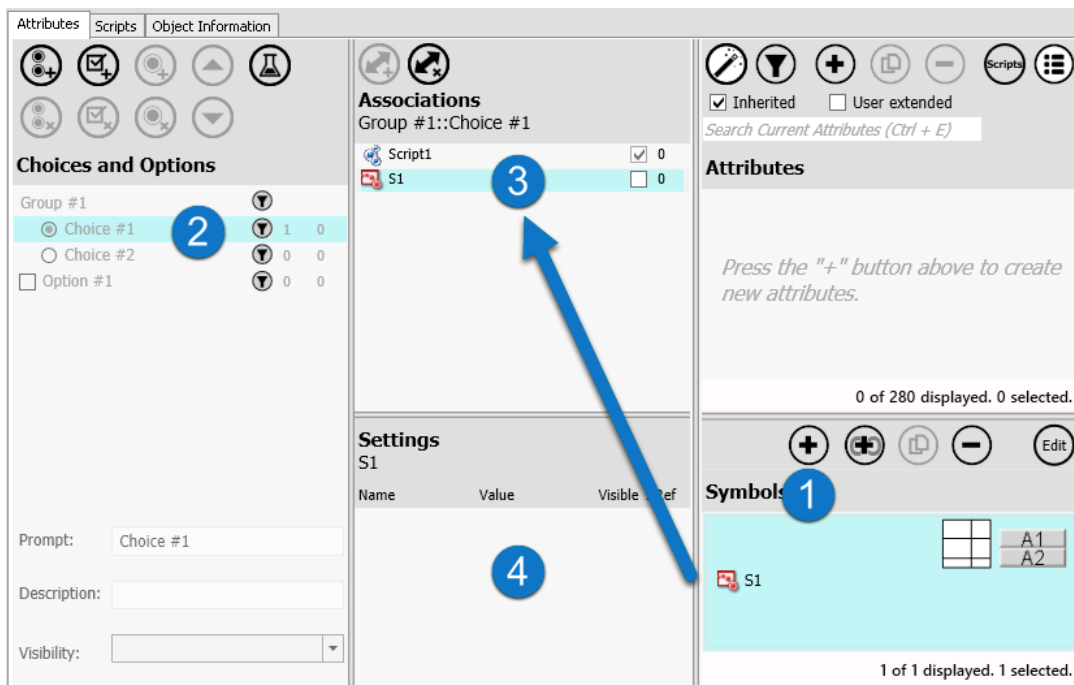
You can create a new symbol for the object, and then associate the symbol with the Object Wizard. You can also link symbols from the Graphics Toolbox to the object. See *Link to Shared Content in the Graphic Toolbox* on page 236 for additional information.

See the Application Server User Guide, "Creating and Working with Content on page 165," for additional information about symbols.

To add a new symbol for association with the Object Wizard:

1. Select the **Add Symbol** button to add a new symbol to the object.
2. Highlight a Choice or Option in the **Choices and Options** pane.
3. Drag and drop the symbol from the **Symbols** pane to the **Associations** pane.

- If the symbol contains a Symbol Wizard or custom properties, modify the settings as needed for the Choice or Option with which it is associated.



- Add additional symbols and associate them with the Object Wizard, or associate the symbol with other Choices or Options.
- If applicable, configure Symbol Wizard and custom property values for each symbol association in the **Settings** pane.

---

**Note:** A Choice or Option does not have to have a symbol associated with it. A symbol can be associated with more than one Choice or Option. You can override the default Symbol Wizard and custom property values; the overrides affect only the Choice or Option for which they are entered.

---

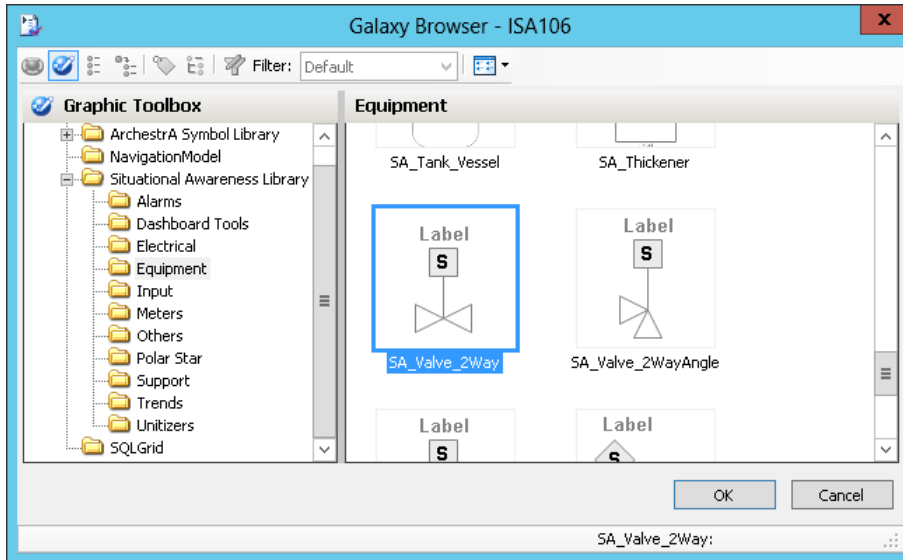
## Link to a Symbol in the Graphic Toolbox

To extend or change a derived Object Wizard's functionality, you can link a symbol from the Graphics Toolbox with the Object Wizard. Once you have linked the symbol, you can associate it with the Object Wizard.

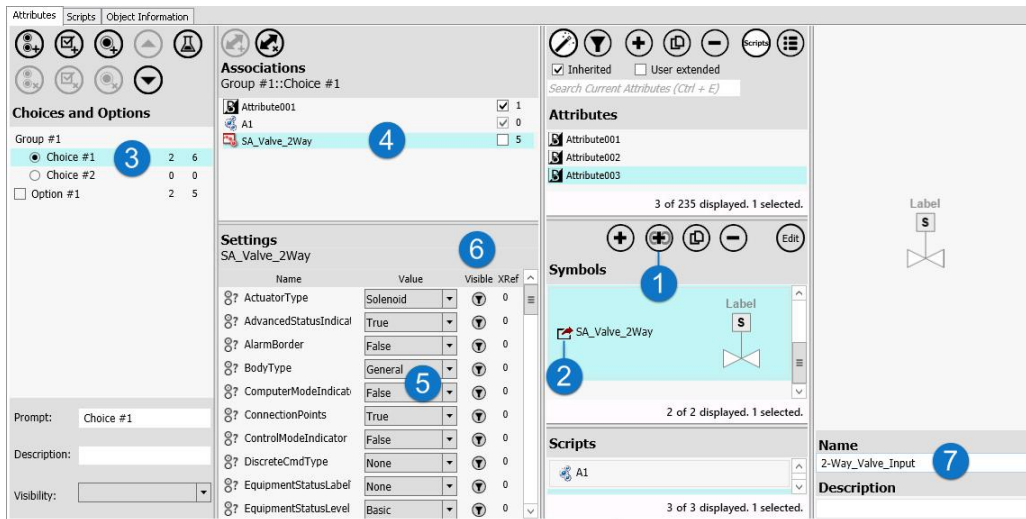
To link a symbol or other item from the Graphic Toolbox to an Object Wizard:

- In the **Symbols** pane, select the **Link Symbol** button (1) to link a symbol from the Graphic Toolbox to the object.

- The **Galaxy Browser** opens. Select a symbol or Symbol Wizard from the Graphic Toolbox and select **OK**.



2. The symbol is added to the object. The icon **(2)** next to the symbol name indicates the symbol is linked to the object.
3. Highlight a Choice or Option in the **Choices and Options** pane **(3)**.
4. Drag the symbol from the **Symbols** pane to the **Associations** pane **(4)**.
5. If the symbol is a Symbol Wizard, modify the settings **(5)** as needed for the Choice or Option with which it is associated.
6. Turn visibility of the Symbol Wizard settings **(6)** on or off, as needed. If the settings are visible, users can override your configuration settings.
7. In the **Details** pane **(7)**, edit the default name of the symbol as needed. You can also add a description (optional).



**Note:** A Choice or Option does not have to have a symbol associated with it. A symbol can be associated with more than one Choice or Option. You can override the default values of a Symbol Wizard; the overrides affect only the Choice or Option for which they are entered.

## Symbols and Object Wizards

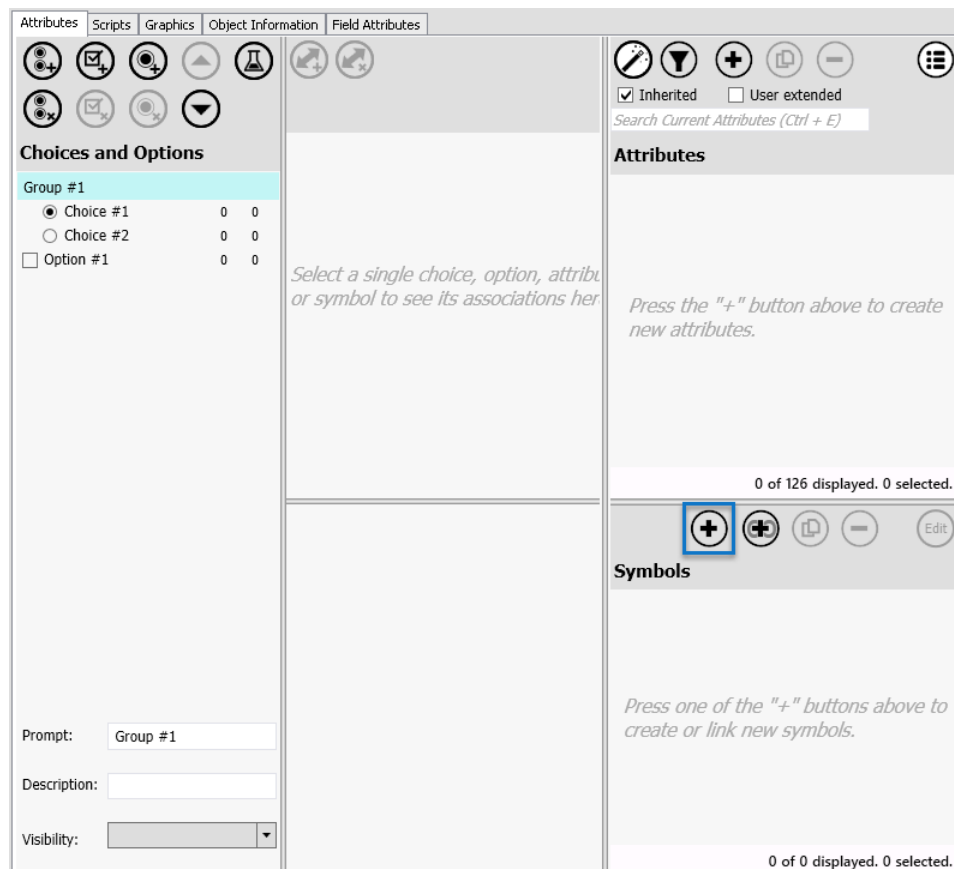
Two different methods are for associating symbols with Object Wizards.

- Add symbols into the template that you have created for that particular template. These symbols you owned by the object. While changes to the symbol in the object will propagate to child objects, non-related instances and templates that may use copies of the symbol will not be updated when the original symbol is edited.
- Link an external symbol to the template. Create a link from the template to external symbols in the Graphic Toolbox (GTB). Linked symbols can be shared between unrelated objects (objects that are not part of the same derivation hierarchy). Changes to the symbol will be reflected to each template and instance that link to the GTB symbol.

### Example 1: Add a Symbol to the Object

To add a custom symbol to a template:

1. In the **Symbols pane**, select the **Add Symbol** button.

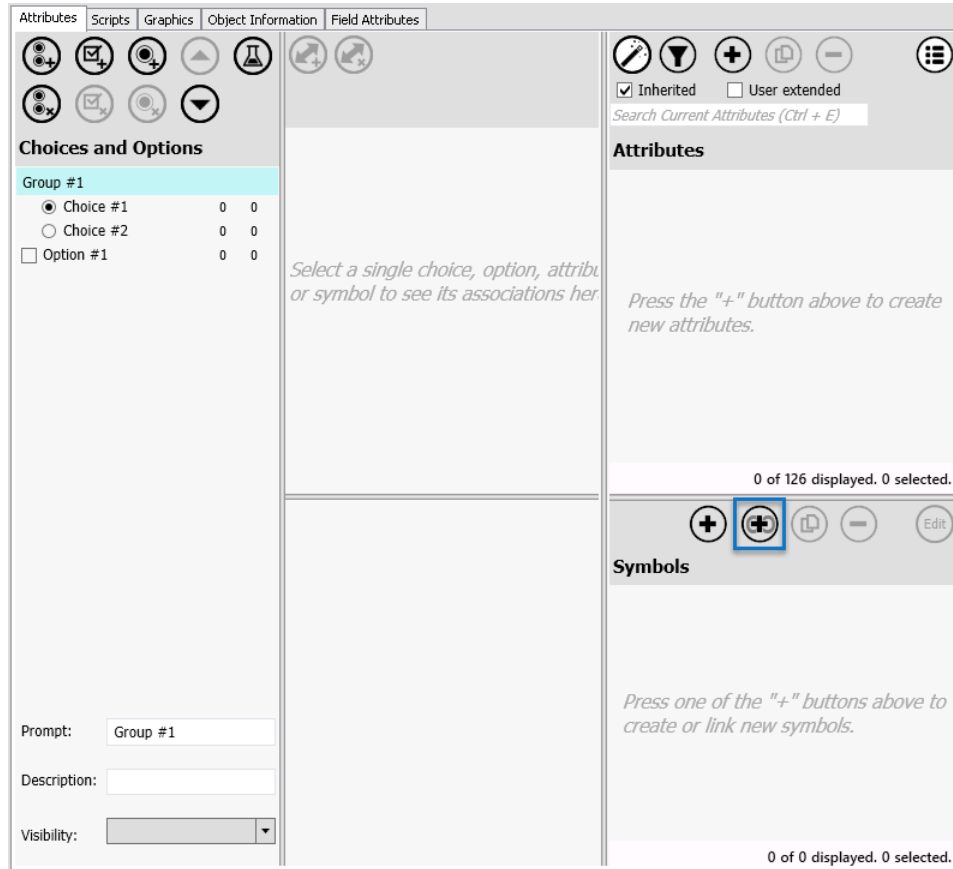


2. Rename the graphic in the **Details pane** of the **Object Editor**. You can also add a description of the symbol (optional).
3. Select the **Edit symbol** button. The **Graphic Editor** will open in a new window.
4. Select the **Graphic Editor** window and create the symbol to be added to the template.
5. Save the symbol and close the **Graphic Editor**.
6. To associate the symbol with an Object Wizard Choice or Option, see *Associate Content with a Choice or Option* on page 237.

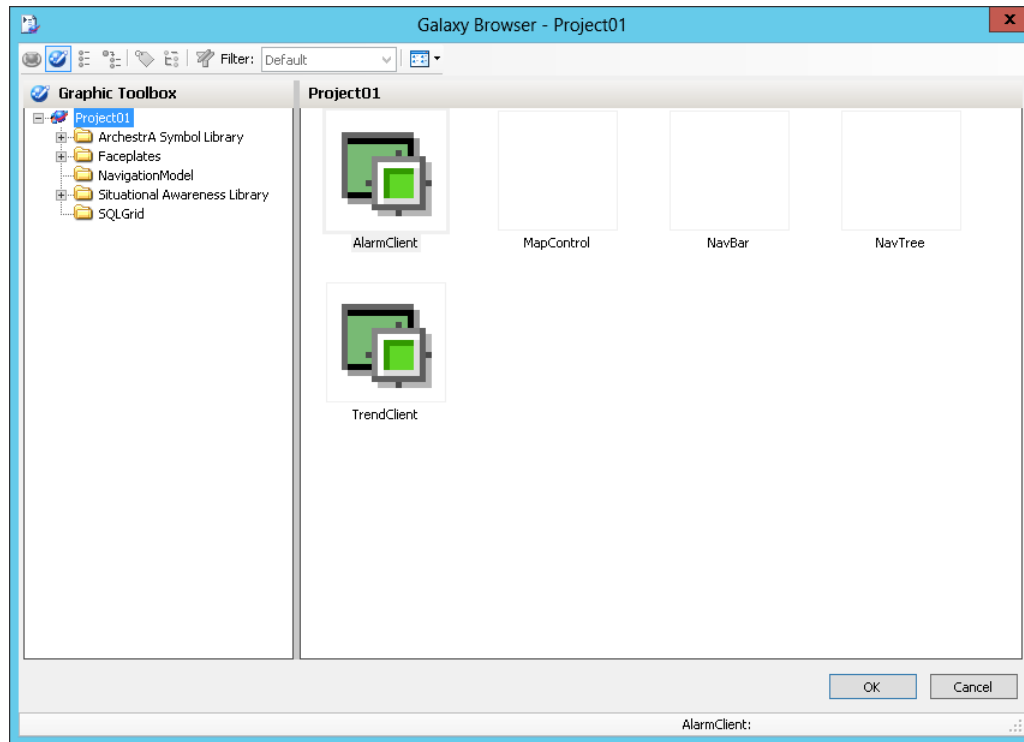
## Example 2: Link to a Symbol in the Graphic Toolbox

To link a Graphic Toolbox (GTB) symbol to an template:

1. In the **Symbols pane**, select the **Link External Symbol** button.



- The **Galaxy Browser** opens and displays the Graphic Toolbox.



- Select the Graphic Toolbox symbol from the navigation pane that you want to link to the template.
- Rename the graphic in the **Details pane** of the **Object Editor**. This name applies only to the current object. You can also add a description of the symbol (optional).
- Select the **Edit symbol** button. The **Graphic Editor** will open in a new window.
- Select the **Graphic Editor** window and edit the symbol as needed.
- Save the symbol and close the **Graphic Editor**. When you save the GTB symbol, changes will be reflected in every object to which the symbol is linked, including child objects.
- To associate the symbol with an Object Wizard Choice or Option, see *Associate Content with a Choice or Option* on page 237.

## Configure Instances

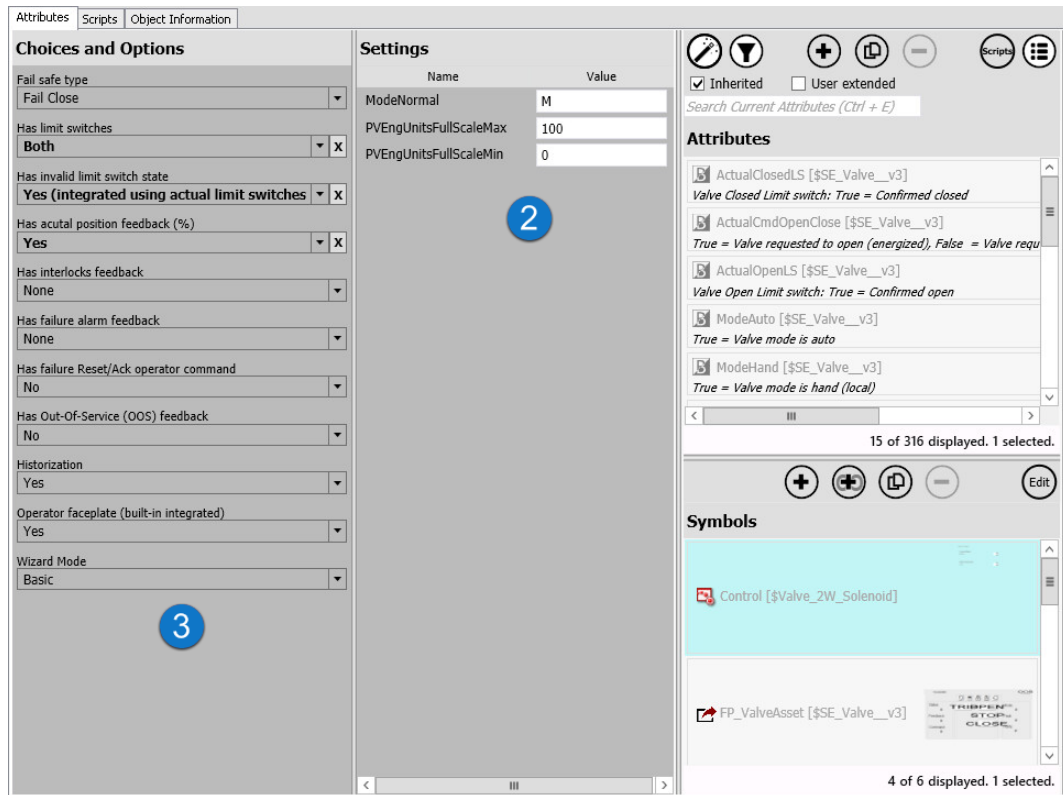
You can use the IDE to configure instances from a template with an Object Wizard, or you can graphically create instances using an editor that is started from the Graphic Editor. If you are using AVEVA OMI for your run-time environment, you can also access the editor through the ViewApp Editor or Layout Editor. See the AVEVA OMI online help for more information about these editors.

- From the IDE, the Object Wizard presents a series of prompts or questions, from which you select an answer to build a representation of a piece of field equipment, such as a pump, valve, PLC, etc.
- If you are using the Graphic Editor or one of the AVEVA OMI editors, select a symbol from the Toolbox tab and drag it onto an overview symbol. See *Create a New Instance Graphically* on page 290 for more information.

## Configure an Instance

To configure an instance using an Object Wizard in the IDE:

1. In the IDE, open the instance you need to configure.
  - To create and configure a new instance:
    - Select the template you want to use, and derive an instance from it. Then, open the instance. See *Create an Instance* on page 99 for additional information.
  - To change the configuration of an existing instance:
    - Select the instance you want to use and open it. See *Create an Instance* on page 99 for additional information.
2. Start at the top of the **Choices and Options pane**. Either select a Choice from the first Choice Group, or select/deselect the first Option as needed to match the physical asset you are modeling. Depending on the selections you make, attributes associated with the Choice or Option may appear in the **Attributes pane**.
  - If you change from the default Choice, the override is indicated by an **X** and the override is shown in **bold** type.
  - If you change the default state of an Option (selected or not selected), the override is indicated by an **X**.
  - You can restore a default setting by clearing the **X**.
3. If an associated attribute contains configurable settings, these are listed in the **Settings pane**. Overrides are also indicated by an **X** next to the changed setting and are indicated by **bold** type.



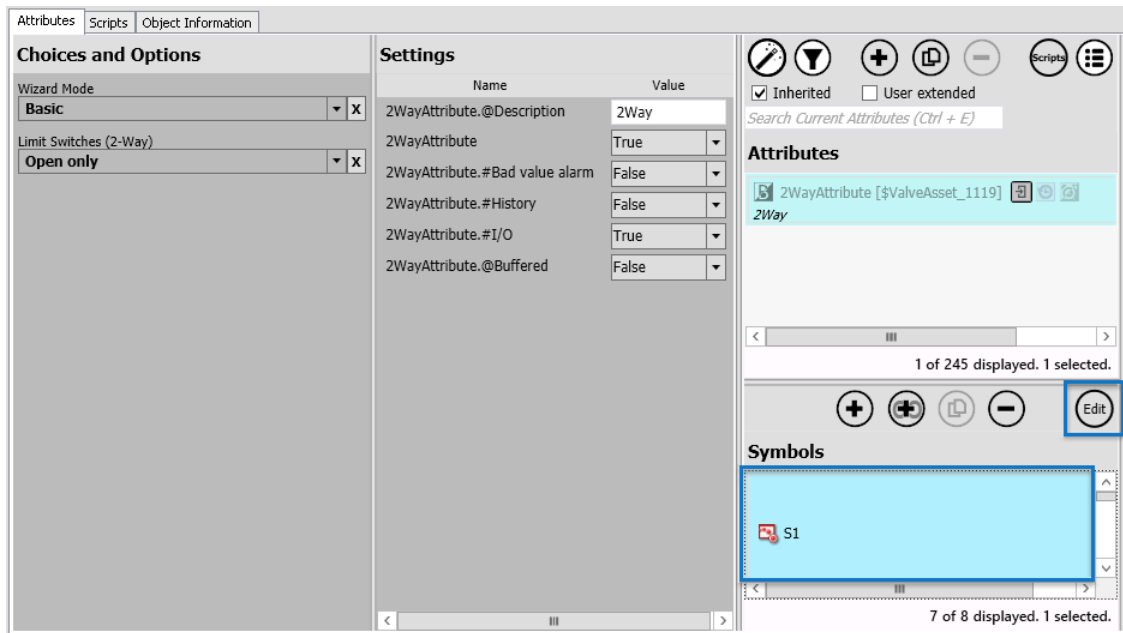
4. Continue sequentially down the list of Choices and Options.
5. Save and close the instance, and then deploy it.

## View and Configure Symbols

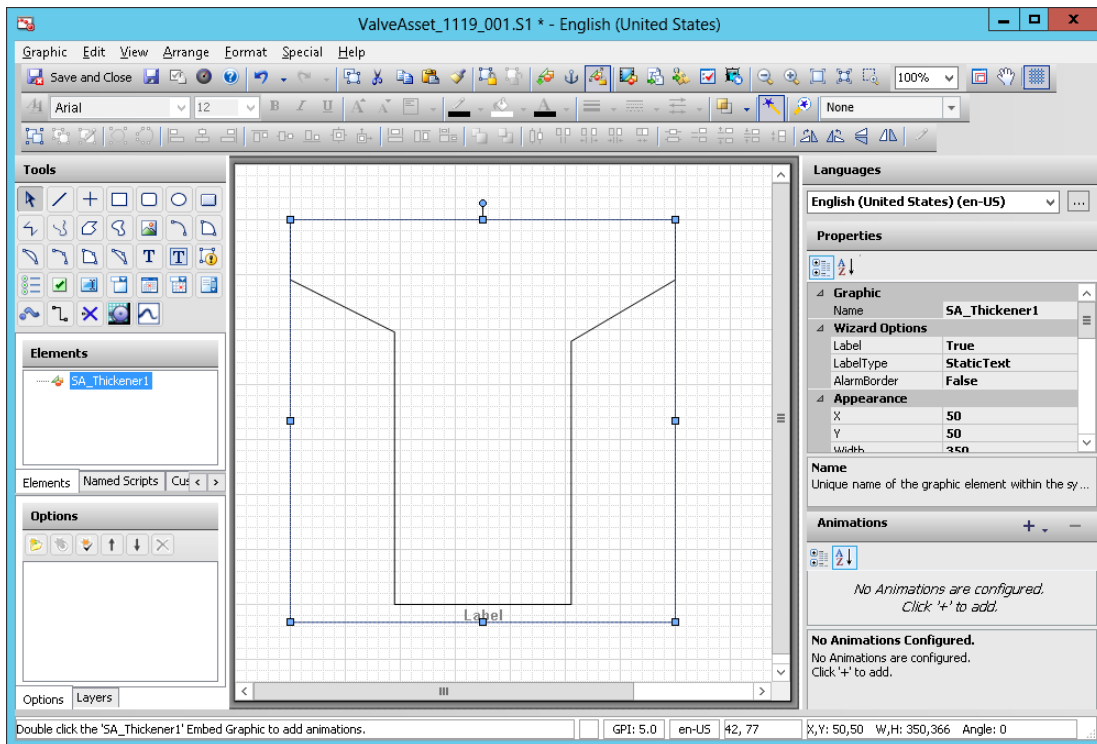
To view and configure symbols associated with an Object Wizard:

1. Select the symbol from the **Symbols pane**.

- Open the symbol by selecting the **Edit button**, or by double clicking on the symbol.



- The **Graphic Editor** opens.
  - If the symbol is inherited, it opens as read-only.
  - If the symbol is a Graphic Toolbox template, it opens as read-only.
  - You can only configure symbols that have been created for the object (object-owned).



- Configure Symbol Wizard settings and custom property settings as needed.
- Save and close the **Graphic Editor**.



See the Application Server User Guide, "Creating and Working with Content on page 165," for additional information about working with symbols.

## Guidelines for Configuring Instances with an Object Wizard

Keep in mind the following behaviors and guidelines when you use an Object Wizard to configure instances:

- Start at the top, with the first item of the Object Wizard.
- The answers you provide to each prompt may cause subsequent prompts to change. Therefore, if you skip selecting a choice or option, the subsequent selections you make could be changed or removed when you return to select a skipped step.
- Some prompts could be added or removed, based on the answers provided to previous prompts.
- If an attribute or symbol is associated with a selected Choice or Option, you may have the option to override certain settings.

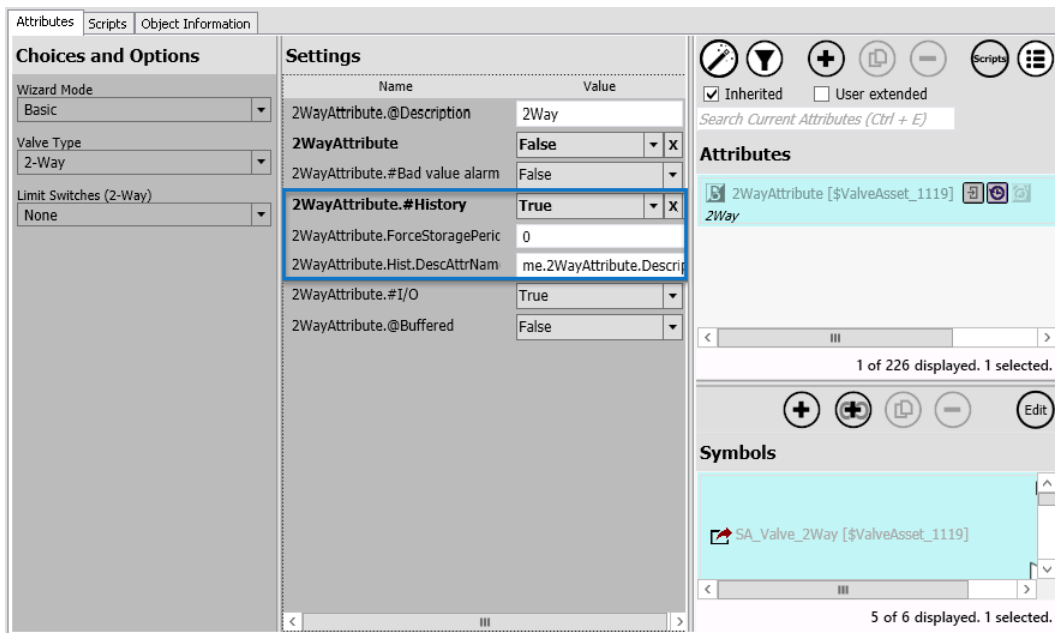
## Override Default Settings

When you override a default attribute or symbol setting in the Object Wizard, the override is marked by an X, and the setting and its description are displayed in bold.

The screenshot shows the configuration window for 'ValveAsset\_1119\_001 \*'. The 'Settings' tab is active, displaying a table of attributes and their values. The '2WayAttribute' attribute is highlighted, and its value 'False' is marked with an 'X' in a dropdown menu, indicating an override. The attribute name and description are also bolded. The 'Attributes' and 'Symbols' panels on the right show the selected attribute and symbol, respectively.

Name	Value
2WayAttribute.@Description	2Way
<b>2WayAttribute</b>	<b>False</b> X
2WayAttribute.#Bad value alarm	False
2WayAttribute.#History	False
2WayAttribute.#I/O	True
2WayAttribute.@Buffered	False

Settings can be different types, including Boolean, integer, string, etc. Settings can be used to turn on or off a feature for the object, such as I/O, Historization, or Alarms. Changing one setting can cause other settings to be shown or hidden.



## Create a New Instance Graphically

You can quickly and easily add and configure new instances from templates that contains an **Object Wizard** and *at least one graphic*, simply dragging a thumbnail image from the template onto the Graphic Editor canvas. Dropping the graphic thumbnail onto the canvas opens the **Configure New Asset** editor, where you can configure the Object Wizard settings. You can also configure symbol wizard and custom property settings, if applicable to the symbol you select. With this editor, you do not need to use the **System Platform IDE** to configure object properties, or spend a lot of time configuring symbol properties in the **Graphic Editor**.

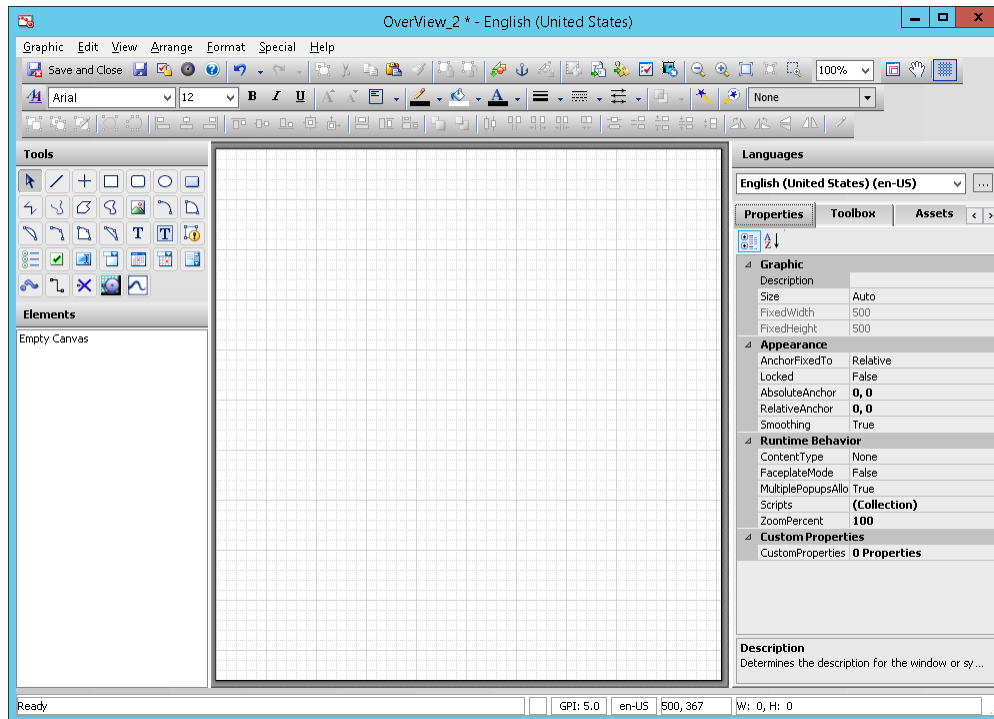
---

**Note:** You can add an asset configured from an object wizard with a linked symbol wizard directly to an AVEVA OMI pane. However, InTouch HMI does not support adding and configuring an asset in an InTouch HMI window that has been derived from an object wizard with a linked symbol wizard. If you are using InTouch HMI, you must instead embed the symbol in an overview symbol. InTouch HMI does not support directly linking an Object Wizard to symbols in the Graphic Toolbox that use a symbol wizard, such those in the Situational Awareness Library.

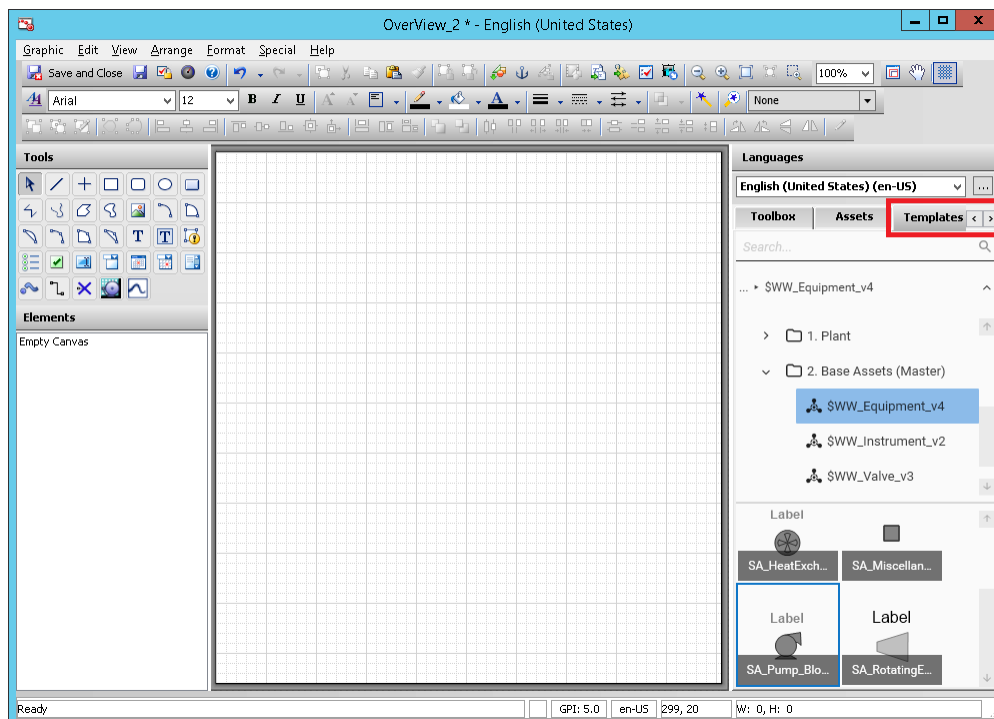
---

To start the **Configure New Asset** editor:

1. Select a new or existing symbol in the **Graphic Toolbox**. The **Graphic Editor** opens.

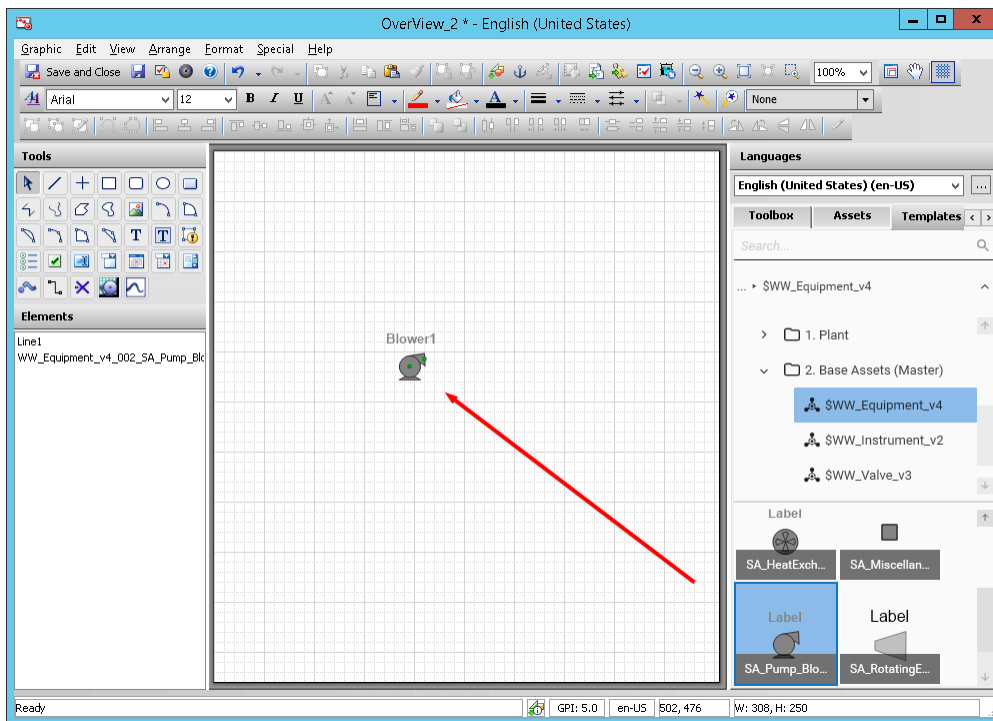


2. If necessary, use the arrow keys until the **Templates** tab is shown, then select it.

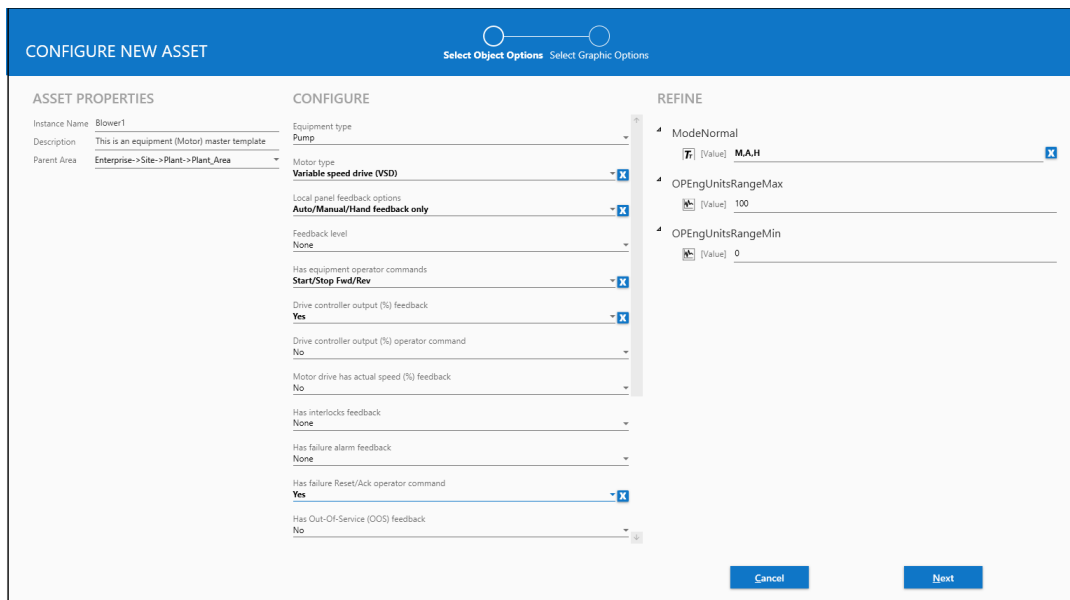


3. Navigate to the template you want to use. The template you select must contain at least one symbol.

4. Select the symbol you want to use from the template and drag it onto the canvas.



5. As soon as you drop the image onto the canvas, the **Configure New Asset** editor opens to the **Select Object Options** page.



Under **Asset Properties** (left column) edit the following entries:

- **Instance Name** (name of the new asset). The Instance Name is the same as the object tag name. Each Instance Name must be unique within the selected **Parent Area**.
- **Description** (a description of the new instance).
- **Parent Area** (select the appropriate parent area for the new instance).
- Configure the **Object Options** (center column).

- Configure the properties listed under **Refine** (right column), then click **Next**.

Object options determine which graphic options can be used. Only the applicable symbol (or symbols) will be available. For example, if an object includes options that allow it to be configured as a pump or as a conveyor belt, you will not be able to add a pump graphic if you have selected a conveyor belt asset on the **Object Options** page.

See *About Object Options* on page 294 for additional information.

6. The **Select Select Graphic Options** page options. If more than one symbol is displayed, select the one you want to use, then configure its graphic options. Click **Finish** when you are done.

The screenshot shows the 'CONFIGURE NEW ASSET' dialog box with the 'Select Graphic Options' page active. The 'AVAILABLE GRAPHICS' section shows a 'Label' icon selected. The 'GRAPHIC OPTIONS' section is expanded to show 'Advanced' options, including 'ControllerOP', 'ControllerOPNumericalDisplay', and 'AlarmBorder', all of which are checked. The 'CUSTOM PROPERTIES' section shows 'ControllerOPVisible' and 'LabelVisible', both set to 'True'. At the bottom right, there are 'Cancel', 'Back', and 'Finish' buttons.

Graphics options are symbol wizard options. Select the symbol wizard options applicable to the asset you are adding to the overview symbol. Once you select the graphic options, you can configure any applicable custom properties.

7. After you click **Finish**, the **Configure New Asset** editor closes. Adjust the size and position of the symbol on the **Graphic Editor** canvas as needed. Add additional symbols and graphics as needed.
8. When you have finished adding symbols and other edits, select **Save and Close** to exit from the **Graphic Editor**.

## About Object Options

The **Select Object Options** page consists of three columns for **Asset Properties**, configuration (**Configure**), and settings (**Refine**).

The screenshot shows the 'CONFIGURE NEW ASSET' wizard interface. It is divided into three main sections: ASSET PROPERTIES, CONFIGURE, and REFINE. The ASSET PROPERTIES section includes fields for Instance Name (Blower1), Description (This is an equipment (Motor) master template), and Parent Area (Enterprise->Site->Plant->Plant\_Area). The CONFIGURE section contains a series of prompts with drop-down menus for equipment type (Pump), motor type (Variable speed drive (VSD)), local panel feedback options (Auto/Manual/Hand feedback only), feedback level (None), and various feedback settings (Start/Stop Fwd/Rev, Drive controller output (% feedback), Drive controller output (% operator command), Motor drive has actual speed (% feedback), Has interlocks feedback, Has failure alarm feedback, Has failure Reset/Ack operator command, and Has Out-Of-Service (OOS) feedback). The REFINE section includes settings for ModeNormal (M.A.H), OPEngUnitsRangeMax (100), and OPEngUnitsRangeMin (0). At the bottom right, there are 'Cancel' and 'Next' buttons.

The **Asset Properties** column contains settings for:

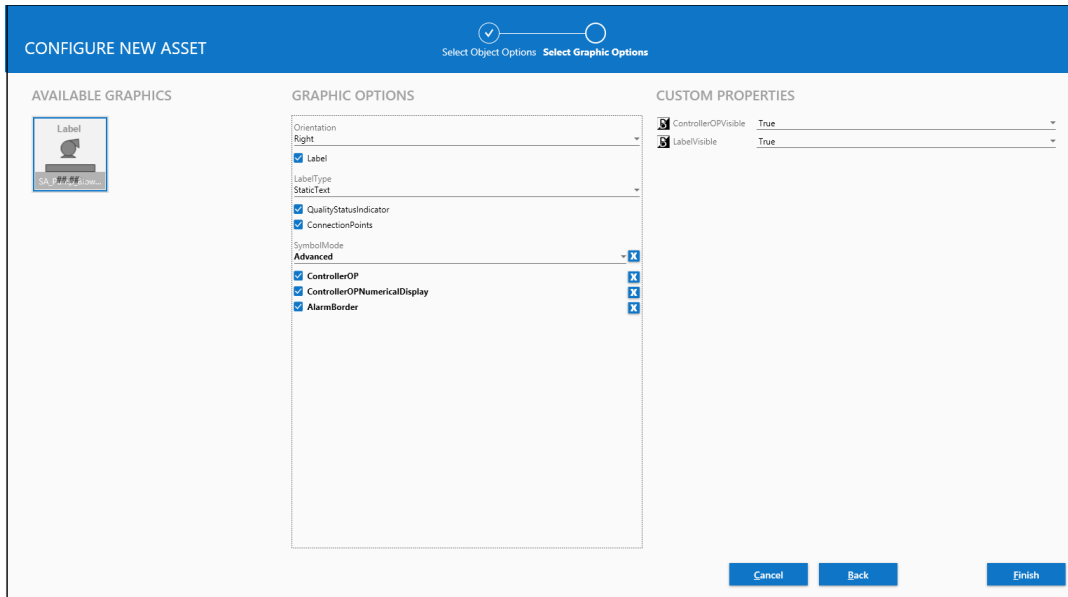
- **Instance name** (what the instance will be named within the Galaxy). The instance name is the object's tagname, and can be up to 32 alphanumeric characters, with at least one letter and no spaces. \$ cannot be used as the first character.
- **Description** (of the instance). There are no restrictions on the length or characters in the description.
- **Parent area**: This is a drop-down selection that lets you move the instance to the appropriate area of the Galaxy.

The **Configure** column contains object property selections, arranged as a series of prompts with drop-down answers. Start at the top and work down to the end. As you make selections, additional prompts may appear in the Configure area (always below the current prompt).

The **Refine** column contains additional settings that you can set. Typically, the settings will include things like upper and lower limits for scaling and alarms, on/off settings, or other items that further define settings that were selected in the **Configure** column. Some configurations do not contain any settings under the **Refine** column.

## About Graphic Options

The **Select Graphic Options** page consists of three columns for **Available Graphics**, **Graphic Options**, and **Custom Properties**.



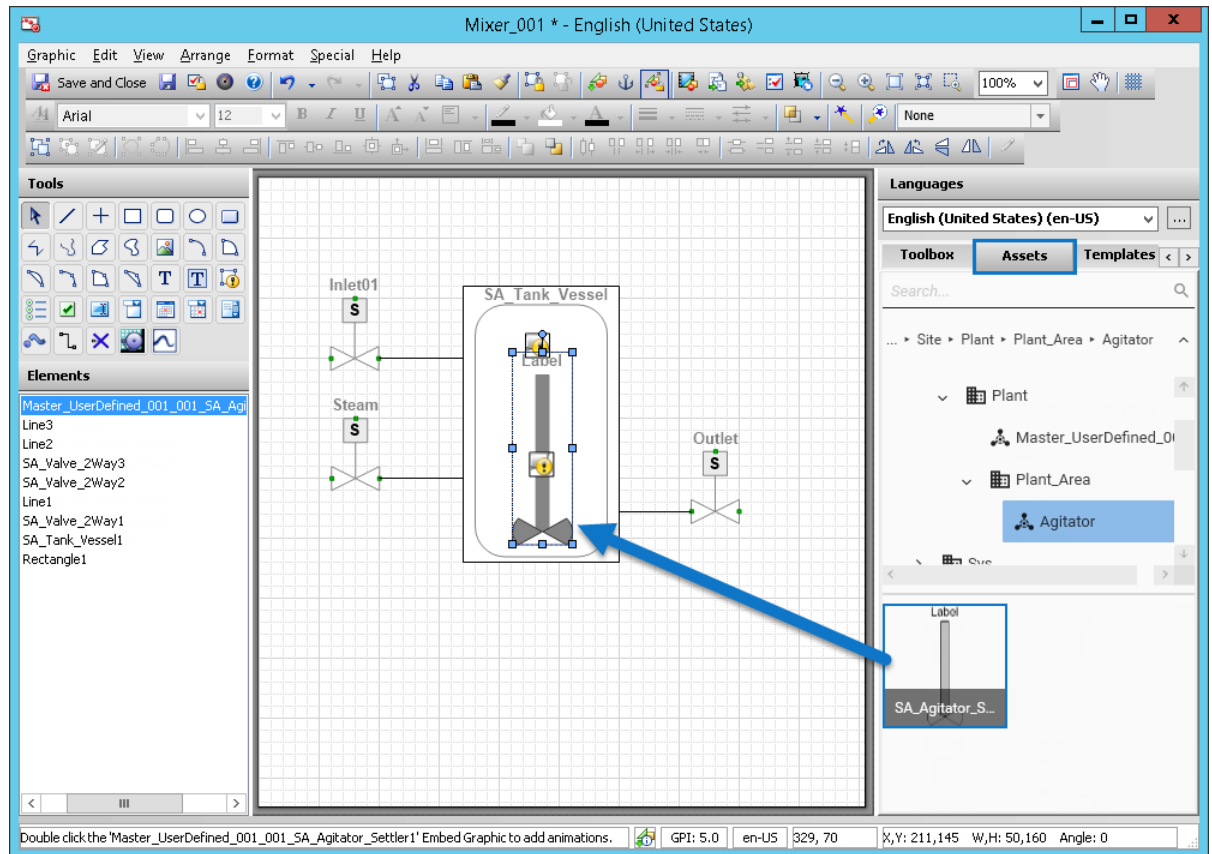
The **Available Graphics** column displays the symbol or symbols that are associated with the object as configured on the **Select Object Options** page. If more than one symbol is displayed, select the one you want to use. Only one graphic can be selected for each asset you configure.

The **Graphic Options** column contains symbol wizard configuration selections that apply to the symbol. Typically, these are used to control things like orientation of the symbol, whether or not it has a visible label, to add alarm borders, etc.

The **Custom Properties** column contains additional refinements to set parameters for the symbol. This includes items like alarm limits and priority levels.

## Add a Configured Asset to a Symbol

1. To add a symbol from a configured instance, select the **Assets** tab in the **Graphic Editor**.



2. Navigate to the asset you want to add to the open symbol., and drag it onto the canvas.
3. Adjust the size and position of the symbol on the **Graphic Editor** canvas as needed. Add additional symbols and graphics as needed.
4. When you have finished adding symbols and other edits, select **Save and Close** to exit from the **Graphic Editor**.

**Note:** InTouch HMI does not support adding a symbol derived from a symbol wizard linked to an object wizard directly onto an InTouch window. The symbol must be contained in an overview symbol.



# CHAPTER 7

## Configuring and Using the OPC UA Server

### About the Application Server OPC UA Server

Application Server supports the OPC UA (Unified Architecture) protocol for machine-to-machine communications.

This OPC UA Server functionality allows third party clients to interact with Application Server and leverage industry standards, such as OPC UA. When OPC UA Server functionality is enabled on Application Server, a client can utilize the built-in OI Gateway or a third party client to connect to Application Server, and use OI Gateway or the client to securely browse the OPC UA namespace and interact with Application Server.

### OPC UA Configuration Checklist

#### Required tasks for end-to-end configuration of the OPC UA server and OPC UA client

The configuration tasks are shown in the order in which they must be completed.

1. **Configure the System Management Server:** The System Management Server is used for establishing a trust relationship between machines, and must be configured to ensure secure communications between nodes. The System Management Server is normally configured during initial System Platform installation. See the *System Platform Installation Guide*, "Configuring the System Management Server," for details.
2. **Set Galaxy Security:** Enabling Galaxy security is not mandatory but is strongly recommended. The OPC UA Server supports the Galaxy security configuration, and where username and password are required, it will leverage the Galaxy security configuration. See *Configuring Security* on page 481 for details.
3. **Configure and deploy the OPC UA server:** Set the configuration options and deploy the OPC UA server to a run-time node.
4. **IT compliance/firewall validation:** Firewall configuration and verification must be completed at this point of the configuration. The node to which the OPC UA Server has been deployed must have Inbound Rules for the firewall configured and verified.

---

**IMPORTANT! A firewall test must be successfully performed before proceeding with the remaining configuration tasks.**

---

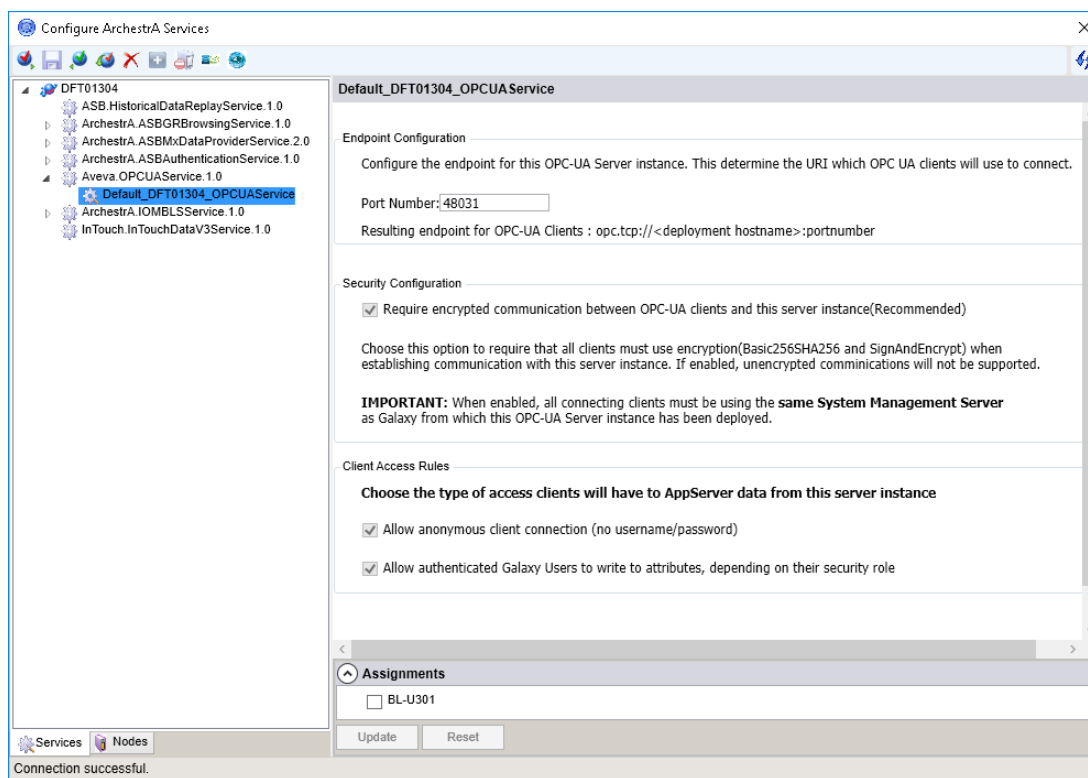
5. **Configure the OPC UA Client:** Client configuration may include the following:
  - Define the OPC UA server address (in the format `opc.tcp://<ServerName>:<PortNumber>`).
  - Select the correct OPC UA server security policy (`Basic256Sha256`).
  - Enter the configured OPC UA User Credentials (username and password).
  - Recommended: Disable Anonymous User. Anonymous User is enabled by default.
6. **Security Certificate:** Download and configure the OPC UA security certificate on the run-time node.
7. **Validate connectivity:** Open the OPC UA client and verify that you can connect to the OPC UA Server, and can view items in the namespace.

## Configuring and Deploying the OPC UA Service

The Archestra OPC UA Service provides access from an OPC UA client to Application Server data, without the need for the Galaxy Browser, a gateway, or other protocol translation mechanism.

### To configure and deploy the OPC UA Server Service

1. In the IDE, click **Galaxy** on the menu bar, select **Configure**, then select **ArchestrA Services**.
2. The **Configure ArchestrA Services** utility opens.



3. Right-click the instance name and select **Check-out** from the context menu.
4. Edit the **port number** for the OPC UA server instance. The default port is **48031**.
5. **Security Configuration (require encrypted communication):** It is strongly recommended that you enable this option, as this will encrypt the payloads across the connection. Note that the client must match this configuration.

---

**Important!** An OPC UA connection cannot be established if you do not enable this option while OS security is enabled, even if the option "Allow authenticated Galaxy Users to write attributes" (Step 7) is enabled.

---

6. **Allow anonymous client connection:** Allowing anonymous client connections is recommended ONLY for initial setup configurations and testing. Anonymous client connections should be disabled for production environments.

If anonymous client connections are allowed *and* OS security is enabled for the Galaxy, anonymous client connections will be READ-ONLY.

Refer to *Client Access Rules and Galaxy Security* on page 300 to see the effect that this option has on data access in different scenarios.

7. **Allow authenticated Galaxy Users to write attributes:** Enabling this setting will only take effect when encrypted communication is also enabled (see Step 5, above). The OPC UA client must match this configuration.
8. The **Assignments** section (below the right pane) represents the platform nodes where the service configuration can be deployed. Select one or more runtime nodes where you want to deploy the OPC Server service, and then click **Update**.
9. Right-click the OPC UA Server service instance name and select **Check-in** from the context menu.
10. On the left-pane, right-click the instance, and then select **Deploy** from the context menu, or press CTRL+D. A message appears indicating whether the service has been successfully deployed to the OPC UA client node. If deployment is successful, the icon next to the instance name changes to indicate that the instance has deployed.

### To add additional OPC UA services

Each OPC UA service is dedicated to a single OPC UA client node. To add additional OPC UA services:

1. Right-click **AVEVA.OPCService**, and then select **Create** from the context menu, or press CTRL+N. The new instance appears in the tree structure.

---

**Note:** Each instance must have a unique port number. Enter the port number in the **Base Address** field. The default port number is **48031**. See *Configuring Archedra Service TCP Ports* on page 419 for a list of port numbers used by ASB services.

---

2. Rename the OPC UA service as needed. Right-click on the service name and select **Rename** from the context menu, or press F2. Then, enter the new name.
3. Repeat the steps above for configuring and deploying each additional OPC UA service.

### To change a deployed OPC UA service

1. Check out the service instance.
2. Make any needed changes.
  - Port Number:** If you are creating multiple services, each service instance should have a unique port number. If more than one service has the same port number, an error is generated in the logger. Multiple instances of the service can be deployed, as long as each service has a unique port number. A new URI (uniform resource identifier) is automatically generated when a port number is changed.

---

**Note:** You may need to open the inbound port in the firewall to allow communication with the remote node.

---

- Security Configuration:** When enabled (default), communication between OPC UA clients and the OPC UA server is encrypted. This is the recommended setting. If this setting is unchecked (disabled), communication is not encrypted.
- Client Access Rules:**
  - When **Allow anonymous client connection** is enabled (default), an anonymous OPC UA client is allowed to connect to the OPC UA server. This is recommended only for testing and initial set up configurations. Once you have completed configuration and/or testing, be sure to disable this setting to provide protection against possible unwanted intrusions and to ensure that only authenticated users have access. Anonymous client connection should not be enabled in a production environment.

Galaxy Security settings do not have any affect on these behaviors. See *Configuring Security* on page 481 for more information.

- When **Allow authenticated Galaxy user to write to attributes** is enabled (default), an authenticated Galaxy user can change attribute values in run time, if their security role allows them to do so. See *About Roles* on page 485 for more information.

When **Allow authenticated Galaxy user to write to attributes** is unchecked (disabled), an authenticated Galaxy user is not permitted to change attribute values in run time, even if their security role allows them to do so.

- See *Client Access Rules and Galaxy Security* on page 300 for more information about user permissions for each setting combination.

3. Check in the service or services.
4. Undeploy and then redeploy the service or services.

## Client Access Rules and Galaxy Security

Client Access Rules configured for the OPC UA Service interact with the Galaxy security authentication mode to allow or deny different levels of access for authorized users.

There are two configurable Client Access Rules in the OPC UA Service dialog. By default, both rules are enabled:

- Allow anonymous client connection (no username/password)
- Allow authenticated Galaxy Users to write to attributes, depending on their security role

The following table defines the level of data access users are allowed under different combinations of Client Access Rule configurations, Galaxy security authentication mode, and the type of OPC UA credentials (anonymous or authenticated user with username/password).

If security for the Galaxy is enabled (Galaxy security = **Secured**, column 1), encrypted communication between the OPC UA clients and OPC UA service must also be enabled. See *Configuring and Deploying the OPC UA Service* on page 298.

Galaxy security authentication mode	OPC UA Client credentials	Client Access Rules		Level of Data Access		
		Allow anonymous connection	Allow authenticated Galaxy Users	Connect	Read	Write (see below)
Secured	Authenticated	Enabled	Enabled	YES	YES	YES
Secured	Authenticated	Enabled	Disabled	YES	YES	NO
Secured	Authenticated	Disabled	Enabled	YES	YES	YES
Secured	Authenticated	Disabled	Disabled	YES	YES	NO
Secured	Anonymous	Enabled	Enabled	YES	YES	NO
Secured	Anonymous	Enabled	Disabled	YES	YES	NO
Secured	Anonymous	Disabled	Enabled	NO	N/A	N/A
Secured	Anonymous	Disabled	Disabled	NO	N/A	N/A
None	Authenticated	Enabled	Enabled	NO	N/A	N/A
None	Authenticated	Enabled	Disabled	NO	N/A	N/A
None	Authenticated	Disabled	Enabled	NO	N/A	N/A
None	Authenticated	Disabled	Disabled	NO	N/A	N/A

Galaxy	OPC UA Client	Client Access Rules		Level of Data Access		
None	Anonymous	Enabled	Enabled	YES	YES	NO
None	Anonymous	Enabled	Disabled	YES	YES	NO
None	Anonymous	Disabled	Enabled	NO	N/A	N/A
None	Anonymous	Disabled	Disabled	NO	N/A	N/A

**Important:** Whenever Client Access Rules and Galaxy Security allow a user to write data, this permission is always conditioned by whether or not the user's configured security role also allows them to write data to a specific attribute. This means that when Galaxy security is enabled, the user's security role must explicitly allow them to write to attributes, regardless of the OPC UA client access rule setting. If their security role does not allow them to write to attributes, they cannot, even if the level of data access in the above table shows that they can.

## Configuring the Firewall for the OPC UA Service

OPC UA communications in Application Server require that the run-time node firewall allows a connection with an OPC UA client node. Before changing firewall settings, however, it is recommended that you perform a *Firewall Test* on page 304.

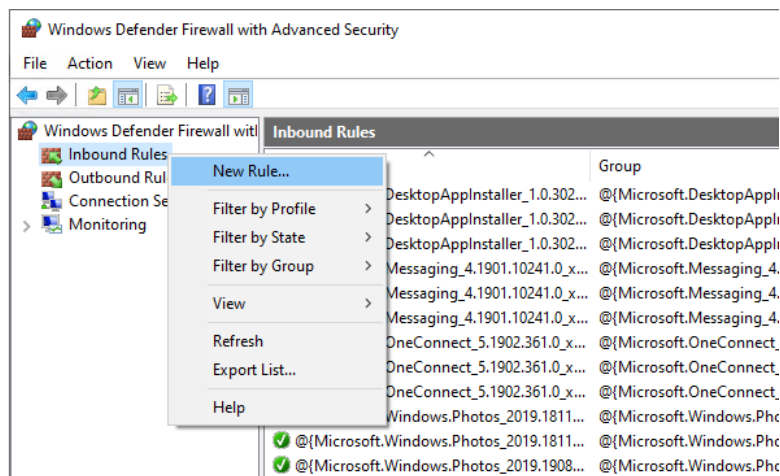
## Configure the Run-Time Node Firewall

**Important:** The firewall rules must be added to the node to which the OPC UA Server Service is deployed.

### To configure the run-time node firewall

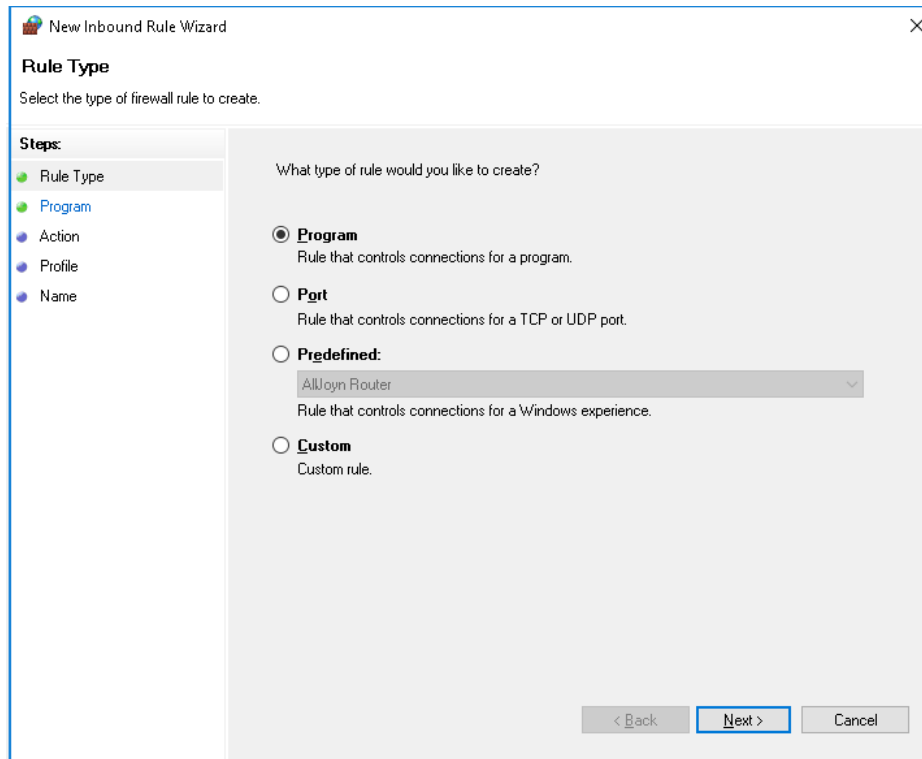
On the run-time node(s) where the OPC UA Server Service is deployed, open the Windows firewall and configure it as follows:

1. In the Windows Search bar, open **Windows Firewall**.
2. Select **Advanced Settings** and create an **Inbound Rule**.



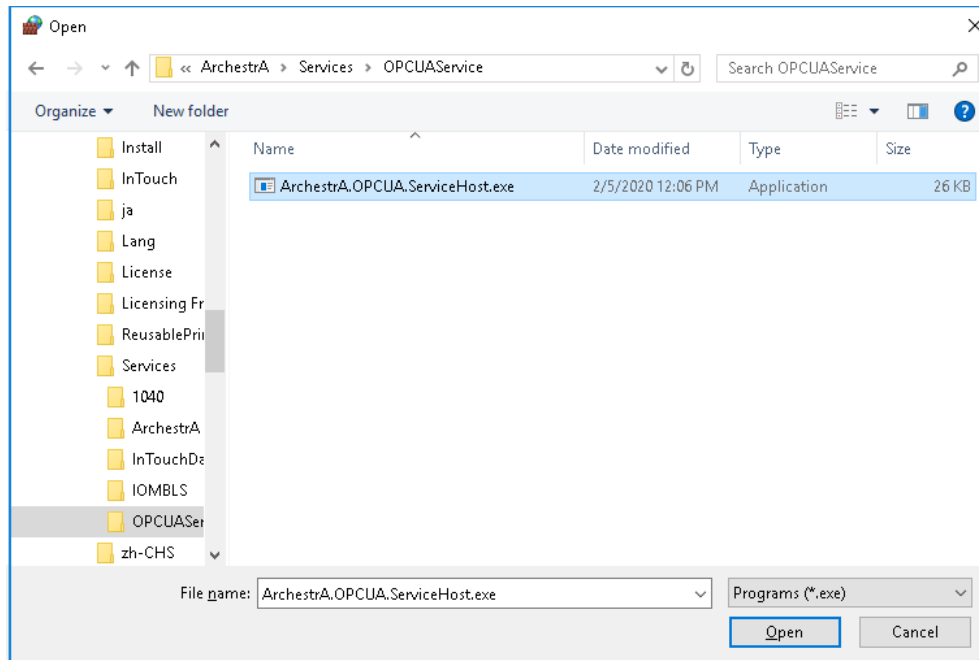
3. Click "New Rule." The Rule Wizard opens. .

4. Select **Program** for the Rule Type and click **Next**.



5. Browse to the OPC UA Server location. If System Platform was installed in the default location, the path should be:

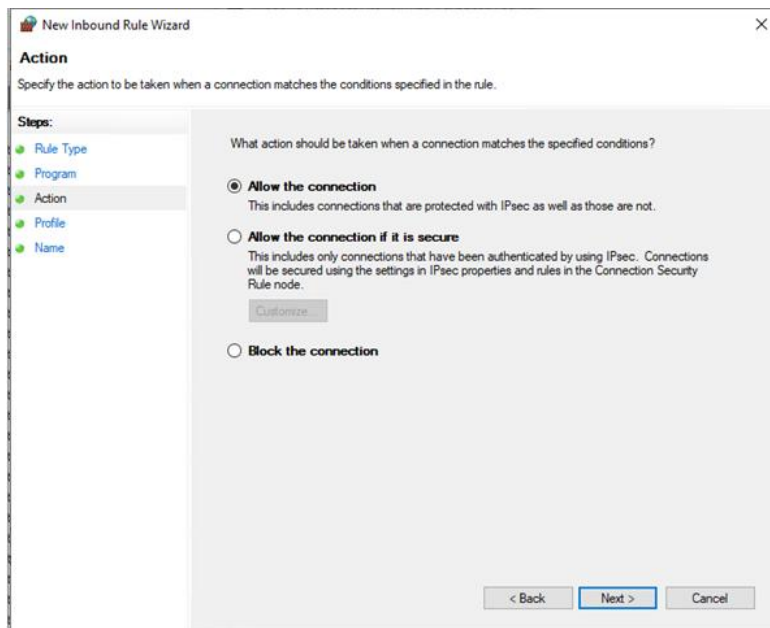
C:\Program Files (x86)\Common Files\ArchestrA\Services\OPCUAService



6. Select "**ArchestrA.OPCUA.ServiceHost.exe**" and then click **Open**.

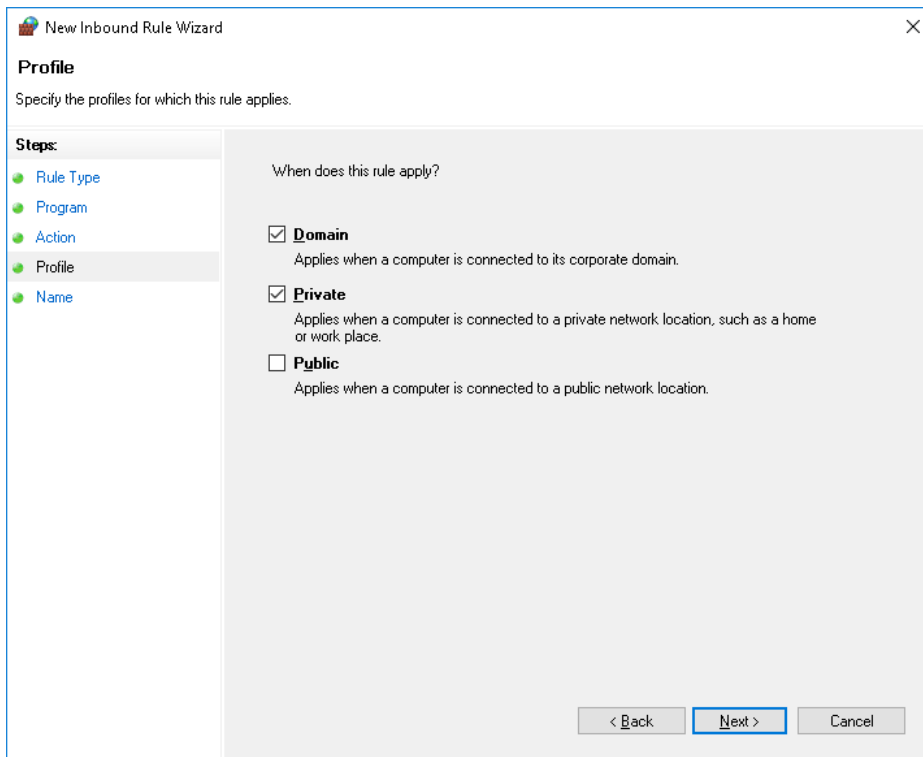
This adds the file path to the wizard. Click **Next**.

7. On the next screen, select the option "**Allow the connection.**" Click **Next**.

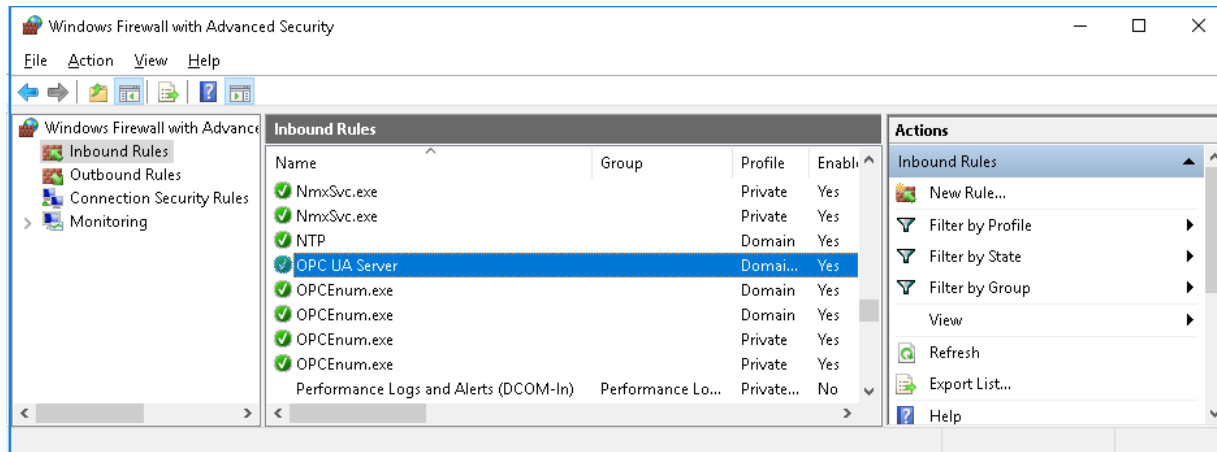


8. The wizard will ask when the rule applies.

- For **Domain** environments: Select **Domain** and **Private**. We recommend that you deselect **Public**.
- For **Workgroup** environments: Select **Public**. The Domain and Private settings have no affect in a Workgroup environment.



9. Finally, provide a name for this rule (for example, "OPC UA Server"). If you will be configuring multiple OPC UA services from Application Server, be sure to use names that differentiate each service from the others.
10. Now, check that the new rule has been added to the list of InBound Rules in the Windows Firewall and that it is enabled.



11. Verify that you can connect to the run-time node from the OPC UA client node by repeating the *Firewall Test* on page 304.

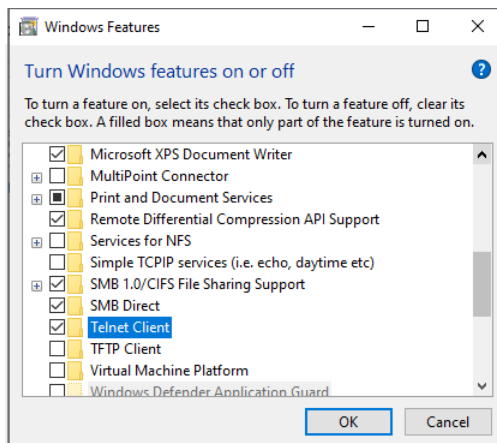
## Firewall Test

### To perform a firewall test with Telnet

1. From a separate node from where the OPC UA Server service is running, enable **Telnet** by turning on the **Telnet Client** Windows Feature.

**Note:** This test is ideally run from the OPC UA client node.

- a. Open the Windows **Control Panel**.
- b. Go to **Programs and Features**, then select **Turn Windows features on or off**.
- c. Scroll down the list of features to **Telnet Client** and enable it. Telnet is disabled by default.



2. Prior to running this test, verify that the Application Server OPC UA Service has been deployed from the run-time (GR) node to the OPC UA client node. See *Configuring and Deploying the OPC UA Service* on page 298 for details.



3. Run Telnet in a command window on the OPC UA client node by entering the following:

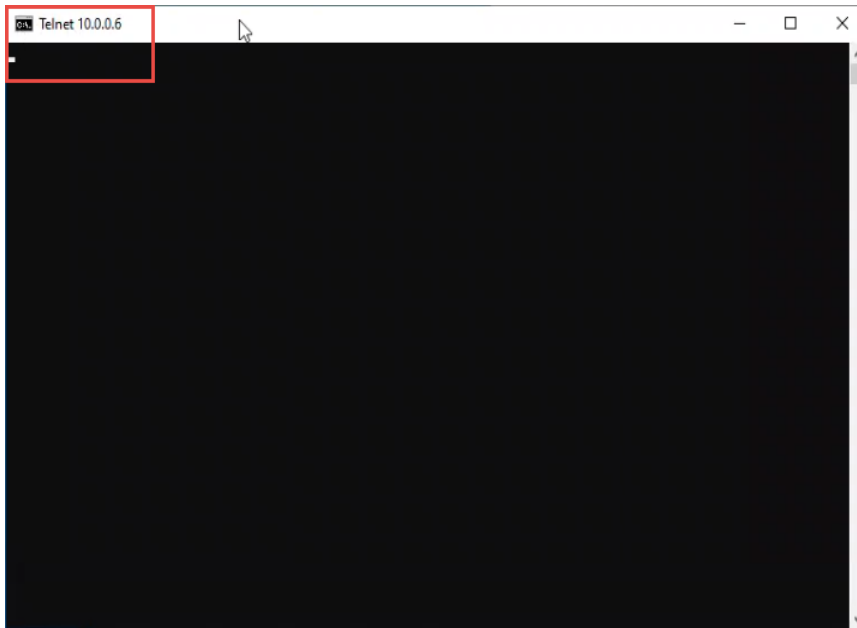
```
telnet <nodeName or ipAddress> <portNumber>
```

where

- **nodeName** is the machine name of the Application Server run-time node. Use nodeName or ipAddress, not both.
- **ipAddress** is the IP address of the Application Server run-time node. Use nodeName or ipAddress, not both.
- **portNumber** is the port number you configured in Application Server for the OPC UA Service. The default port number 48031.

Example: `telnet 10.10.10.06 48031`

- If the command is not successful, it will time out with a message stating that the connection failed. In this case, go to *Configure the Run-Time Node Firewall* on page 301.
- If the telnet command is successful, the command prompt changes to a Telnet prompt.



4. If the Firewall Test is successful, configure the OPC UA client and OPC UA server certificates. The next step in setting up your OPC UA connection depends on if you are using a third-party OPC UA client application, or the OPC UA connection available with OI Gateway. See either:
  - *Configuring Server and Client Certificates for Third-Party OPC UA Client Applications* on page 305, or
  - *Using OI Gateway to Configure the Client Security Certificate* on page 311

## Configuring Server and Client Certificates for Third-Party OPC UA Client Applications

---

**IMPORTANT!** These procedures apply ONLY if you are using a third party OPC UA client. If you are using OI Gateway as the OPC UA client, skip to *Using OI Gateway to Configure the Client Security Certificate* on page 311.

---

To configure encrypted communications between the System Platform OPC UA server and a third-party OPC UA client, both computers will need access the following certificates:

- <Computer Name> ASB OPC UA Server
- The client certificate from your OPC UA client.

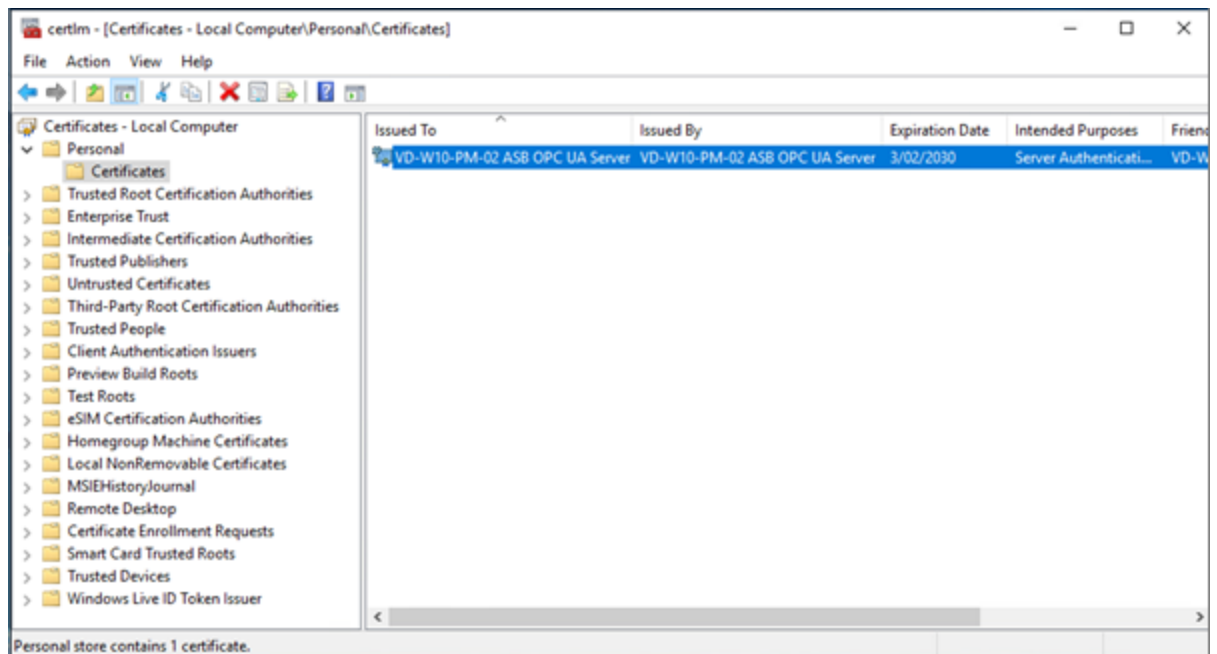
To complete this setup, you will need to perform three steps:

1. Copy the certificate from the OPC UA server node to the OPC UA client node. This step includes the following:
  - Exporting the OPC UA server certificate.
  - Installing the certificate on the client node.
2. Copy the certificate from the OPC UA client node to the OPC UA server node.
3. Make sure that the firewall has been configured to allow the Arcestra.OPCUA.ServiceHost.exe application.

## Export the OPC UA server certificate to the OPC UA client node

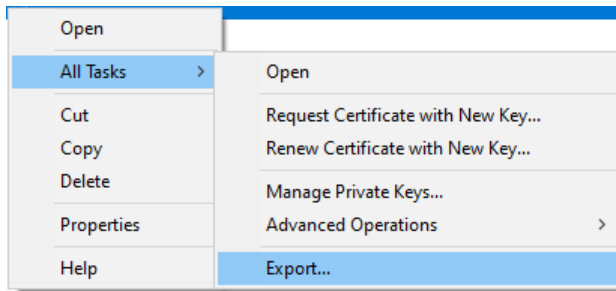
### To export the OPC UA server certificate

1. Open the Windows Certificate Manager on the OPC UA server node.  
To open the Certificate Manager, either type “Manage Computer Certificates” in the Windows search box and select, or open the command prompt and run "certlm.msc."
2. In the tree view, expand the **Personal** node, then click on **Certificates**.

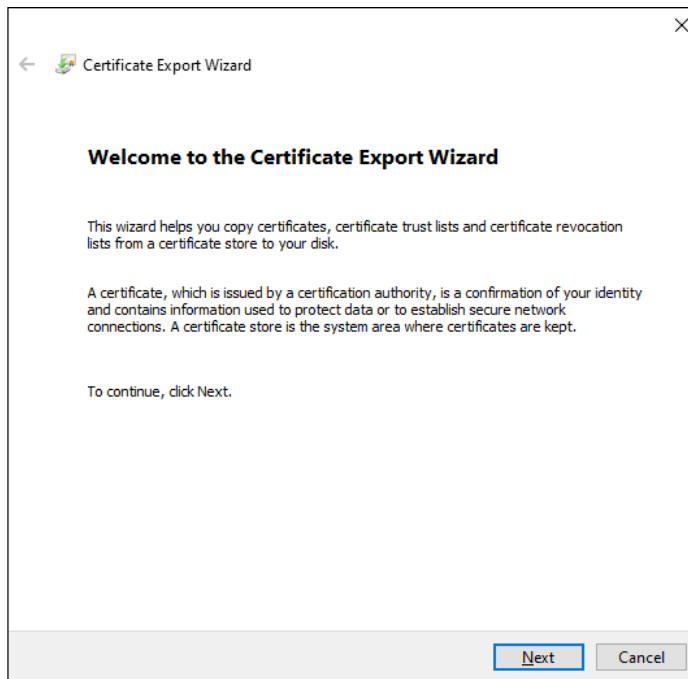


3. Locate the certificate “<computer name> ASB OPC UA Server”.

4. Right-click on the certificate and select 'All Tasks\Export...'



The **Certificate Export Wizard** opens.



5. Depending on type of certificates that your client application uses, you will need to export the certificate as one of two certificate file types:

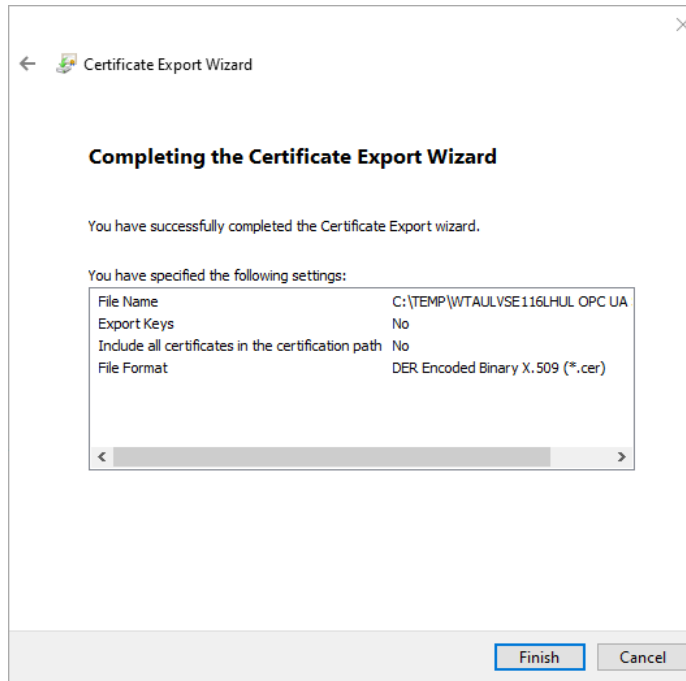
- .cer file**, if the client uses the Windows Certificate Store
- .der file**, if the client uses file-based certificates.

Even though the file extensions are different, the file formats are the same.

6. In the Certificate Export Wizard, select the following options.

- Export Private Key:** select "No, do not export the private key" (default), then click **Next**.
- Export File Format:** select either "DER encoded binary X.509 (.CER)" or "Base-64 encoded X.509 (.CER)," depending on the requirements of your client application. Make the selection, then click **Next**.
- File to Export:** enter a file name to export the Root CA, for example "c:\temp\Next.

7. When the **Export Wizard** finishes, you may need to change the file extension of the certificate from “.cer” to “.der,” depending on what your client application is expecting. You will need to do this if your OPC UA Client application stores certificates in a specific folder, rather than in the Windows Certificate Store.



## Import the OPC UA Server Certificate on the Client Computer

To complete the process of copying and installing the OPC UA server certificate to the OPC UA client node, you need to import the OPC UA Server certificate to the OPC UA client computer.

Each OPC UA client application has its own mechanism to manage certificates. Generally, an OPC UA client will use one of two mechanisms to manage certificates:

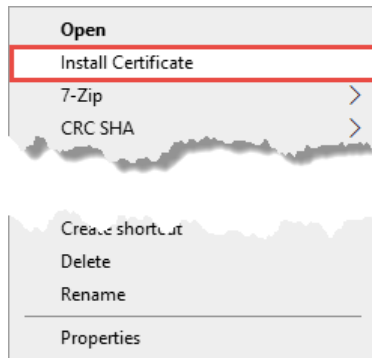
- Utilizing the Windows Certificate Store
- Storing certificates in a specified folder, defined by the OPC UA Client application.

Refer to the documentation for your OPC UA client applications for more details on how to import a server certificate.

### To import a certificate into the OPC UA client Windows Certificate Store

1. Copy the certificate file (<machine name> OPC UA Server.cer) to the OPC UA client node. Where you copy the file is not important.

2. Right click on the certificate file and select "Install Certificate" from the context menu.

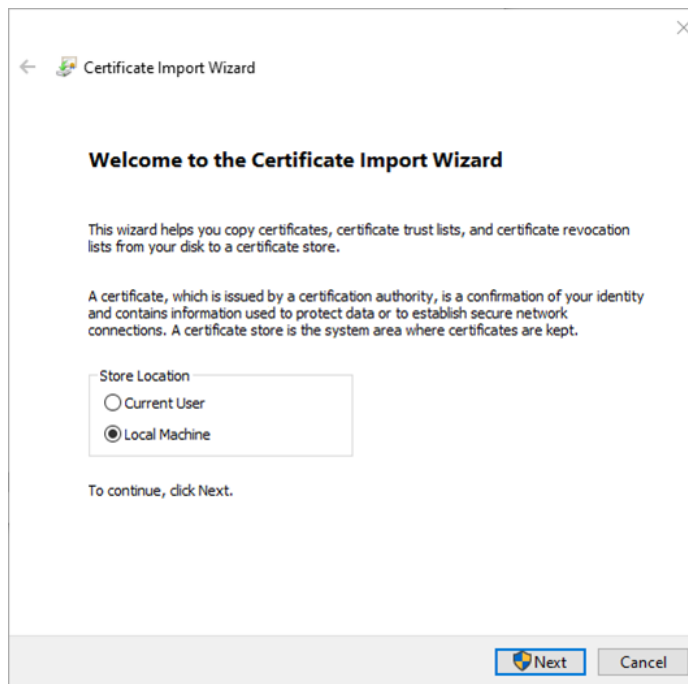


This opens the Certificate Import Wizard.

---

**Note:** Administrator privileges are required to import the certificate.

---



3. In the Certificate Import Wizard, select the following options.
  - Store location:** Select "Local Machine," then click **Next**.
  - Certificate store:** Browse to "Personal," then click **Next**.
  - Completing the Certificate Import Wizard:** Review the settings, then click **Finish**.

**To copy a certificate to specified location on the OPC UA client**

1. Copy the certificate file (<machine name> OPC UA Server.cer) to the folder that is specified by your OPC UA client applications. The following table shows the location of the certificate folder for a number of common OPC UA clients.

OPC UA Client	Manufacturer	Folder
Datafeed OPC UA Client	Softing	C:\ProgramData\Softing\OpcClient\pki\trusted\certs
UaExpert	UnifiedAutomation	C:\Users\Admin\AppData\Roaming\unifiedautomation\uaexpert\PKI\trusted\certs
UA Client Getting Started	UnifiedAutomation	C:\ProgramData\unifiedautomation\UaSdkNetBundleEval\pkiclient\trusted\certs
Matrikon	Matrikon	C:\Users\Admin\AppData\Local\Matrikon\OPCUAExplorer\pki\DefaultApplicationGroup\trusted\certs
KEP Server	Kepware	C:\ProgramData\Kepware\KEP Server EX\6\UA\Client Driver\cert
Top Server	Software Toolbox	C:\ProgramData\Software Toolbox\TOP Server\6\UA\Client Driver\cert

Refer to the documentation for the OPC UA client application for additional information about managing certificates.

2. Configure the OPC UA certificate, as described below.

## Configure OPC UA Client Certificates on the OPC UA Server

The next step in the of configuring OPC UA server and client certificates is to trust the OPC UA client certificate that has been installed on the OPC UA server node.

The easiest way is to attempt to connect an OPC UA client to the server. Since trust of the client certificate has not yet been established, the connection is expected to fail.

Once the connection fails, any OPC UA client certificates that are not installed on the OPC UA server are placed in the “Rejected Certificate” folder on the OPC UA Server.

By default, the folder location is:

C:\ProgramData\AVEVA\PCS\OPC UA Rejected Client Certificates\certs

---

**Note:** Access to this folder requires administrator rights, and the folder is hidden by default.

---

### To import certificates placed in the Rejected Certificate folder

1. To import the OPC UA Client certificates, browse to the rejected certificate folder.
2. Right click on the certificate for the OPC UA Client that you want to trust and select “Install Certificate”. This opens the **Import certificate Wizard**.
3. Select the following options in the wizard:
  - Store location:** Select "Local Machine," then click **Next**.
  - Certificate store:** Select “Trusted People,” then click **Next**.
  - Completing the Certificate Import Wizard: Review the settings, the click **Finish**.

## Port Usage

### Port usage

OPC UA communicates via a single TCP port. This is specified in the **Endpoint Connection** setting in the configuration of the OPC UA Server, and defaults to port 48031. For details, see *Configuring and Deploying the OPC UA Service* on page 298

If you will run multiple OPC UA Servers on the same computer, you must specify a different port number for each subsequent server.

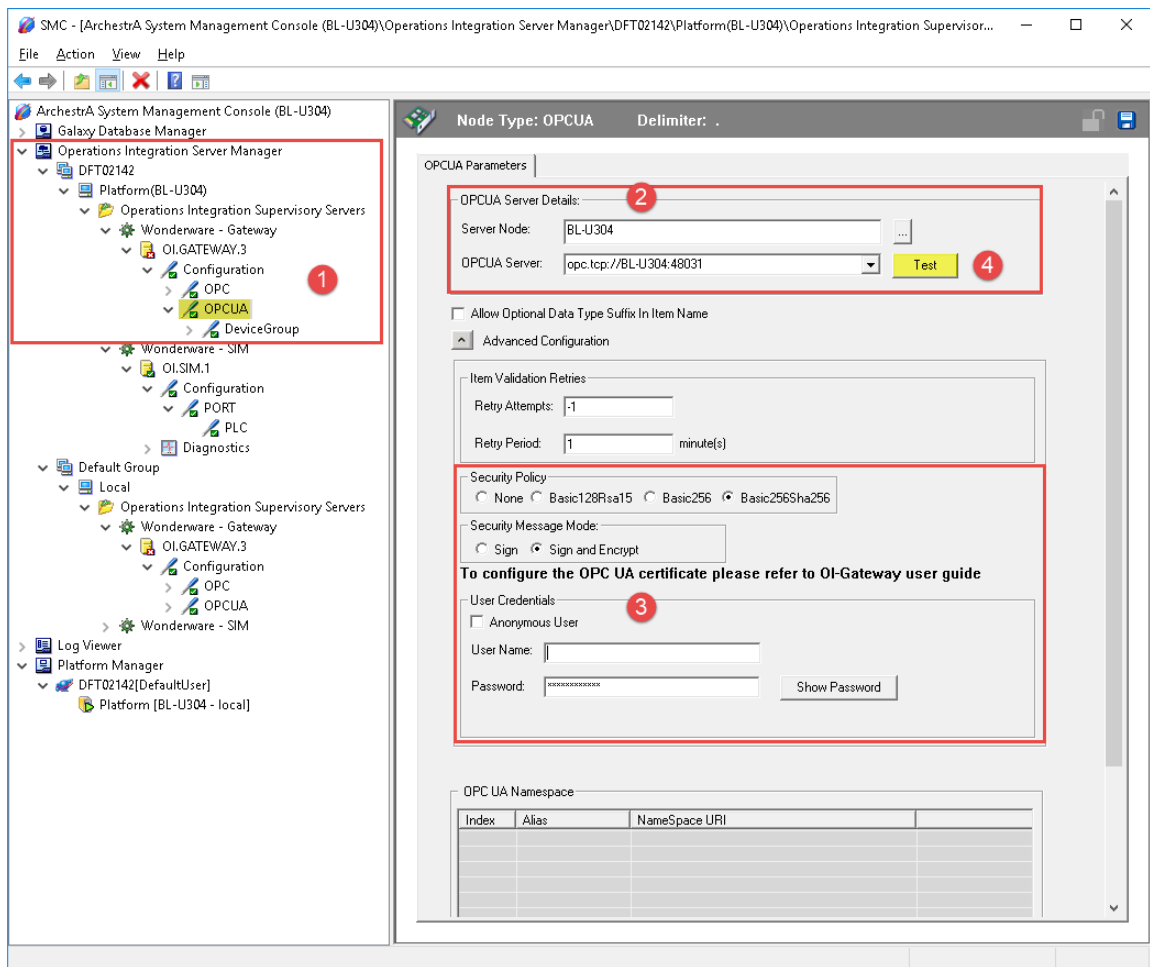
Confirm that this port is not blocked by any firewall software installed on your computer. For information about testing and configuring communications through firewall, see *Configuring the Firewall for the OPC UA Service* on page 301

## Using OI Gateway to Configure the Client Security Certificate

OI Gateway provides a convenient method for configuring security certification on the server and client OPC UA nodes.

### To configure the security certificate through OI Gateway

1. From the Start Menu on the run-time node, open the System Platform Management Console (SMC) (**Start > AVEVA > System Platform Management Console**).



2. In the console tree, navigate to the **OI.GATEWAY.3** node under Operations Integration Supervisory Servers (1).
3. Create an OPC UA connection.
4. Configure OPC UA Server Details (2).
  - **Server Node:** Enter the machine name of the run-time node.
  - **OPC UA Server:** This is the URI (uniform resource identifier) for the OPC UA server (the run-time node). The address must be entered manually because it is not currently discoverable. Enter it in the format `opc.tcp://<machine name>:<OPC UA port number>`

Use the OPC UA port number that you entered when configuring the AVEVA OPCUAService in the IDE (ArchestrA Services Configuration). The default port number is 48031.

5. Enter the authorization and authentication credentials (3).

You must match the authorization settings configured in the OPC UA server dialog. If the Require Security Authentication checkbox is checked, then you must select the following settings:

- Security Policy:** Basic256Sha256.
- Security Message Mode:** Sign and Encrypt.

Similarly, you must match the Client Access Rules from the OPC UA Server configuration dialog. If the **Allow authenticated Galaxy users** checkbox is checked, enter the user name and password of a valid Galaxy user. These will be used to configure the OPC UA security certificate.

---

**Note:** Galaxy security should already be configured.

---

6. Click the **Test** button (4). The test will fail, but it will download the OPC UA certificate.

---

**IMPORTANT! The reason for this initial test failure is because the certificates between the client and server applications must be trusted. Installing the certificates will fix this.**

---

7. Go to the next section, *Trusting the Certificate between the OPC UA Server and OPC UA Client* on page 312.

Once the certificates are trusted, the OPC UA client configuration will need to be validated.

## Trusting the Certificate between the OPC UA Server and OPC UA Client

In this release of the OPC UA Server service, the operation of creating the trust between the OPC UA Server and the OPC UA client must be done manually. In the example above, *Using OI Gateway to Configure the Client Security Certificate* on page 311, the **Test** operation causes the OI Gateway to submit its own certificate to the OPC UA Server node so it can be trusted. The following steps show how to then trust the client certificate from the OPC UA Server node. If you are not using OI Gateway, follow the procedures listed in *Configuring Server and Client Certificates for Third-Party OPC UA Client Applications* on page 305.

1. Access the folder **C:\ProgramData\AVEVA\PCS\OPC UA Rejected Client Certificates**.

This is the location where the certificate from the client is initially placed by default as an attempt to connect to the OPC UA Server.

---

**Note:** The ProgramData folder is hidden by default. You may need to enable the hidden items option in Windows Explorer in order to view it.

---

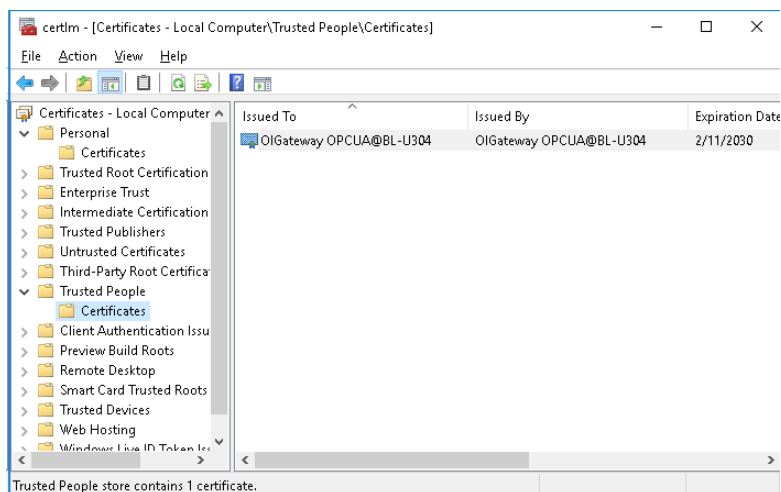
2. Right-click on the certificate name, for example, **OI Gateway OPC UA@OPCUA client node{long hex ID}.der**.
3. Select **Install Certificate** from the context menu. This opens the **Certificate Import Wizard**.
4. Select **Local Machine** for the Store Location, then click **Next**.
5. From the **Select Certificate Store** list, select **Trusted People** as the certificate store. This is the only choice that will work with the OPC UA certificate.
6. Close the wizard to complete installation.
7. Finally, delete the certificate from the **OPC UA Rejected Client Certificates** folder, since the certificate is now installed.



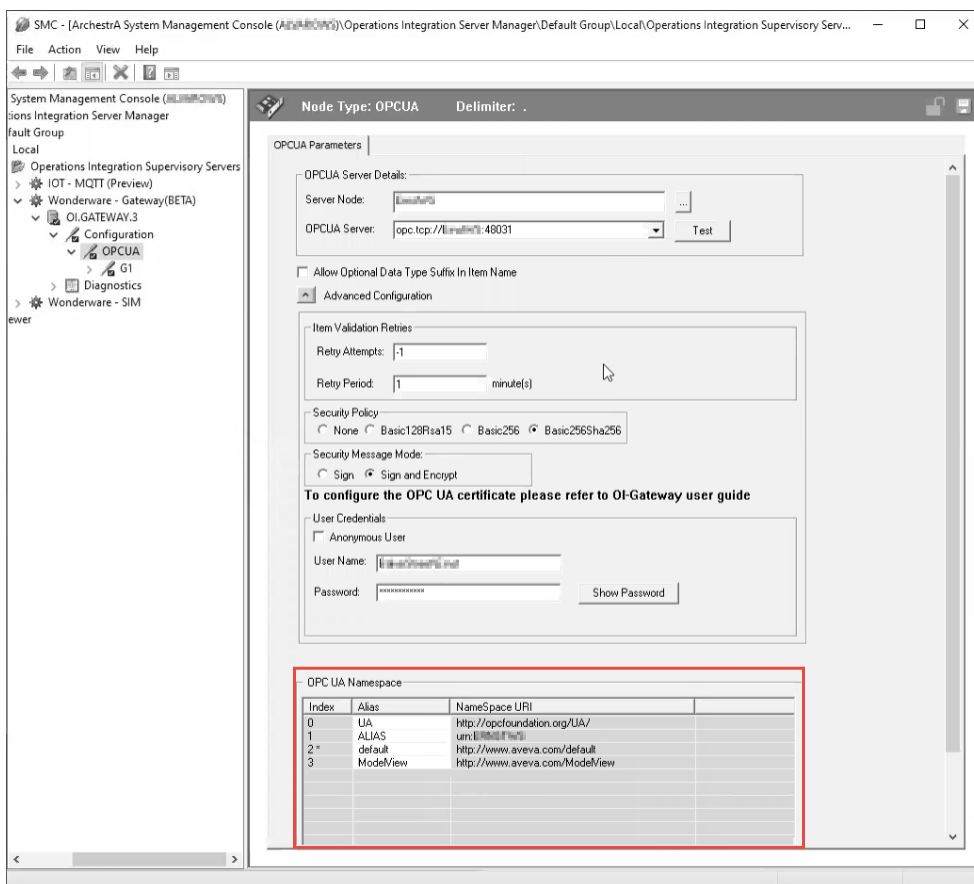
## Verify OPC UA Certificate Installation

### To verify certificate installation

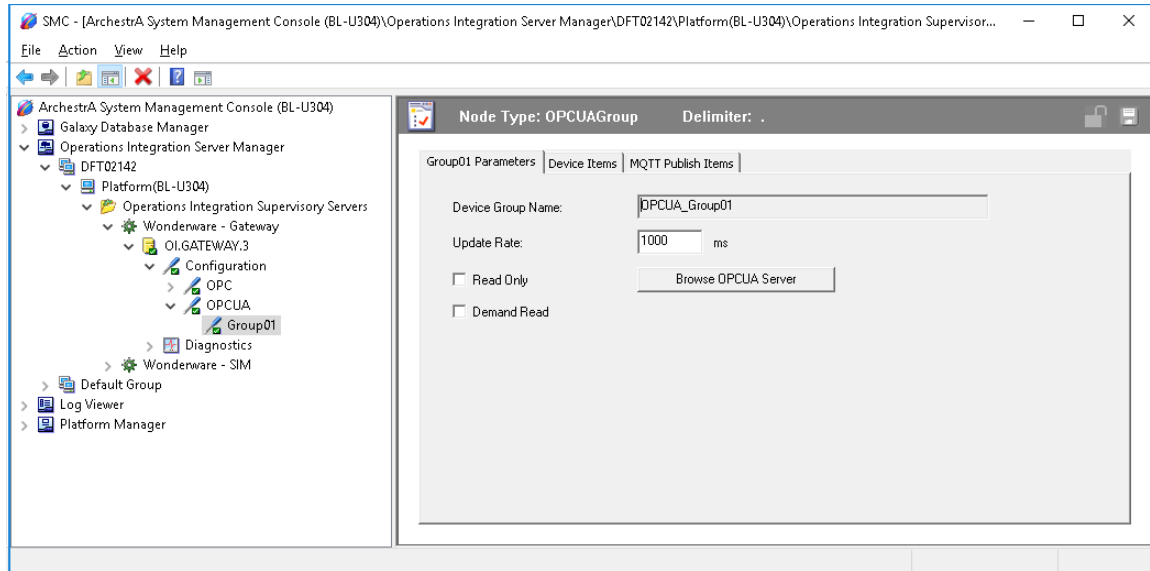
1. Open the certificate manager by typing **certificate** in the Windows search window, then select **Manage Computer Certificates** from the search results. The **Certificate Manager** opens.
2. Navigate the **Trusted People** folder and check that the OPC UA certificate has been installed.



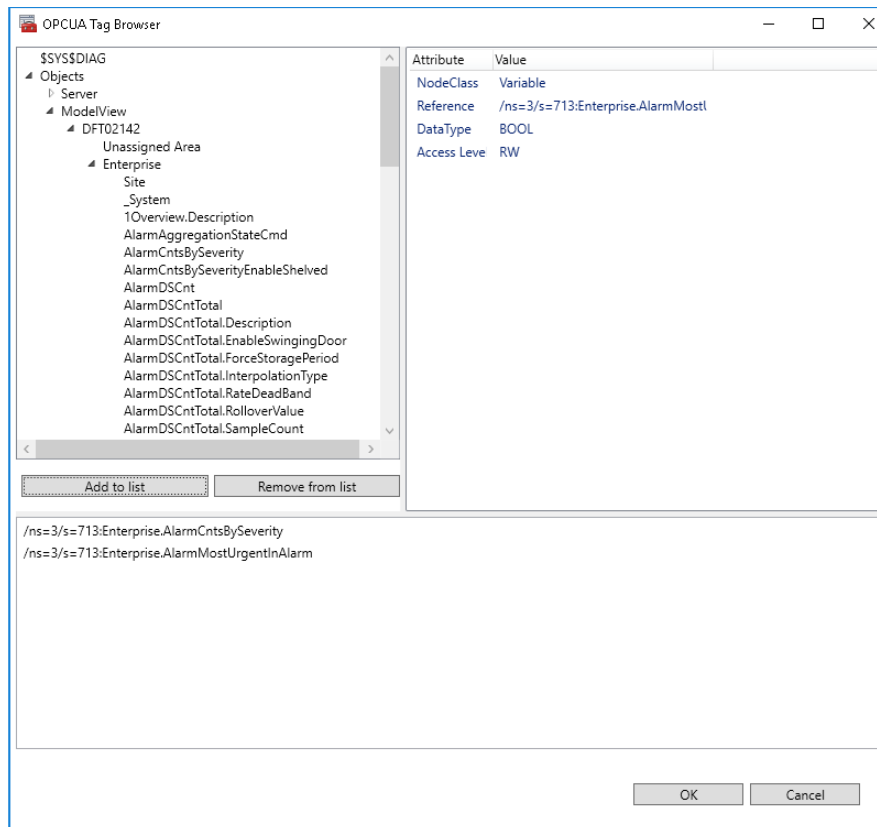
3. From the OPC UA client node, open the **System Platform Management Console** again, and click the **Test** button in OPC UA parameters window. At the bottom of the window, the **OPC UA Namespace** alias list will be automatically populated, indicating the connection was successful.



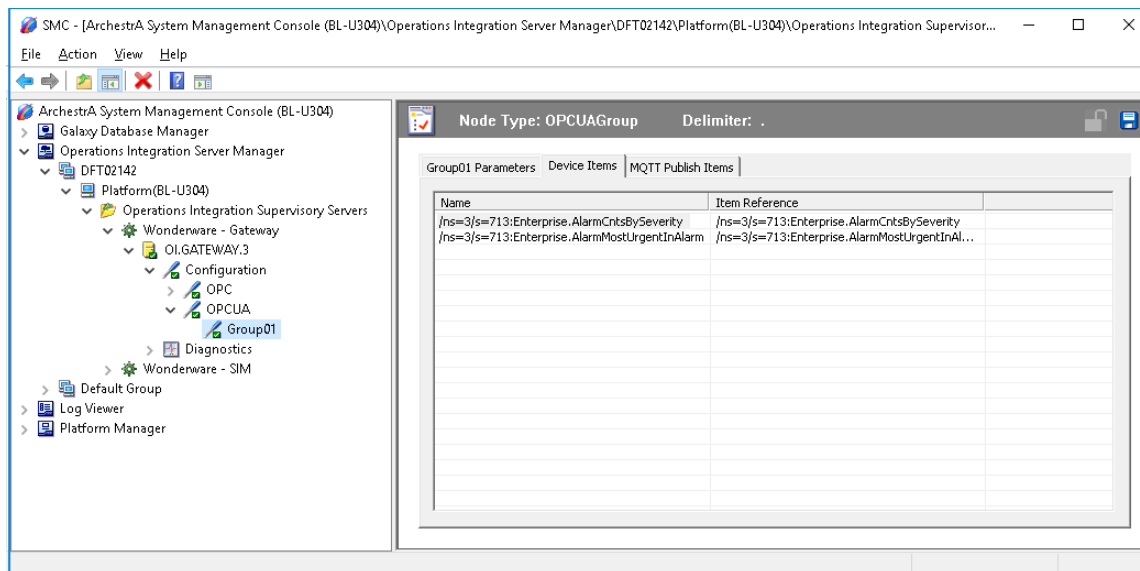
4. Add a group under the OPC UA Connection (named **Group01** in the following figure). The device group name prefixes **OPCUA\_** to the group name you enter.



5. Click the **Browse OPC UA Server** button to browse the OPC UA hierarchy.
6. Optional: Add OPC UA tags to monitor in the alias list to help ensure that you can browse the namespace.



- Switch back to OI Gateway configuration in the **System Platform Management Console**, and notice that the OPC UA tags are now listed in the **Device Items** window.



- By default, item names in the list duplicate the complete item reference path. Rename the items as needed to be more user-friendly.



## CHAPTER 8

# Deploying and Running an Application

You can deploy and test your objects at any time during development. When you are ready to test or run the application in production, you deploy the Galaxy.

You can see what your application looks like in the Deployment view or the Model view. Both views show you the structure of your application. For more information, see *IDE Application Views* on page 28.

A simulation server, derived from the \$OPCClient Device Integration system object, is included with Galaxies created from the Default.cab file. The simulator can be imported to or created for other Galaxies. Use the simulator to test a Galaxy prior to deployment.

When you create a Galaxy from the Default.cab file, the Galaxy's assets are automatically assigned to the simulator's Fast scan group. When you have completed testing and created new Device Integration objects, move the assets from the simulator to scan groups within the new DI objects.

## Deployment Overview

The deployment process copies a Project's assets from the development environment to the run-time environment. Once they are deployed, your assets are "live" and functional.

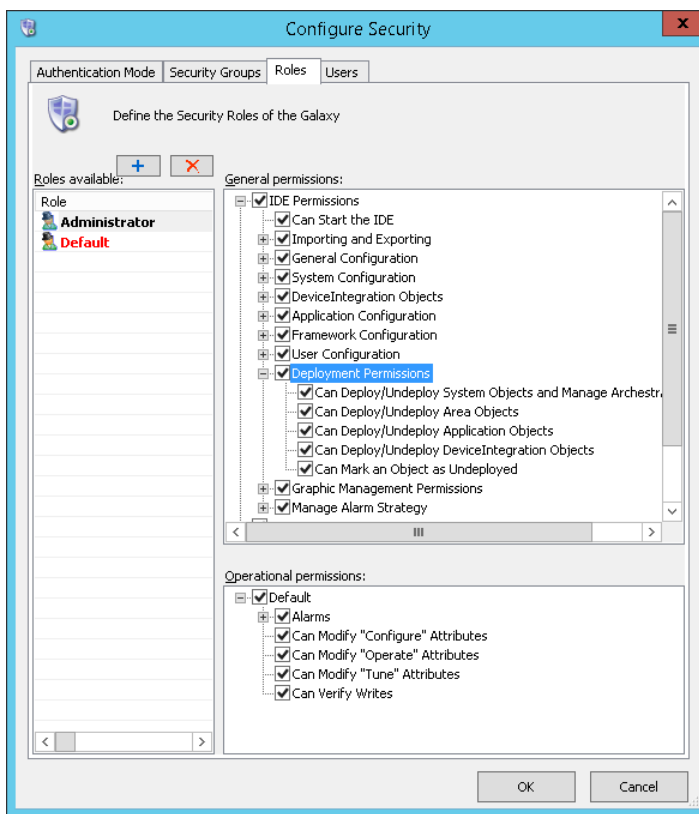
Configuration changes do not appear in the run-time environment until the changes are deployed.

You must have the appropriate level of permissions to deploy assets.

### To view your permissions

1. Select the **Galaxy** menu from the **System Platform IDE** toolbar.
2. Select **Configure > Security**.

3. Go to the **Roles** page. *Deployment permissions* are listed under *General permissions*.



**To deploy an asset from the System Platform IDE**

1. Highlight the asset or assets you want to deploy in the **Deployment** or **Model** view.
2. Select the **Object** menu from the toolbar
3. Select **Deploy...**

## Deploying Objects from the IDE to Run Time

Objects deploy from the configuration environment to the run-time environment as follows:

IDE Config Environment	→	Object Viewer Run-Time Environment
Galaxy database	→	[Does not exist in run-time environment]
Templates	→	[Does not exist in run-time environment]
Instance objects	→	Instance objects [Run-time configuration and behavior]
Security: General permissions	→	[Does not exist in run-time environment]
Security: Operational permissions	→	Run-time permissions to acknowledge alarms and modify attributes
Scripts configuration	→	Scripts execution
Alarms configuration	→	Alarms generate and acknowledge

IDE Config Environment		Object Viewer Run-Time Environment
History configuration	→	History Logs (Historian)

## Allocating Run-time Resources

Distribute components across platforms and engines as you develop your IDE configuration environment. This will help to balance the I/O, memory, and CPU burden across computing resources at run time.

The operating system determines engine assignment to individual CPU cores, and attempts to load balance across CPU and other computing resources. If you need to change how resources are allocated to improve load balancing, undeploy the objects and reassign them in the IDE configuration environment.

If you are using redundant AppEngines, you can control how objects are loaded at runtime. For more information, see *CPU Load Balancing* on page 547.

## Allocating Galaxy Repository Node Memory

The Galaxy database resides on SQL Server, which runs on the Galaxy Repository (GR) node. In some cases, particularly when a GR node contains less than the recommended RAM amount, system performance may be impacted as SQL Server will use more CPU to compensate for the lower amount of available memory. In some cases, high CPU usage can even occur on systems configured with the recommended RAM amount. A galaxy with many objects, for example, can use more memory and cause high CPU usage.

If you notice CPU usage increasing over time to the point that is affecting system performance, we recommend that you set the SQL Server minimum memory (min server memory) to 1/3 of total physical memory. Application Server automatically configures the max server memory setting, and there is no need to manually configure it.

Example minimum server memory settings:

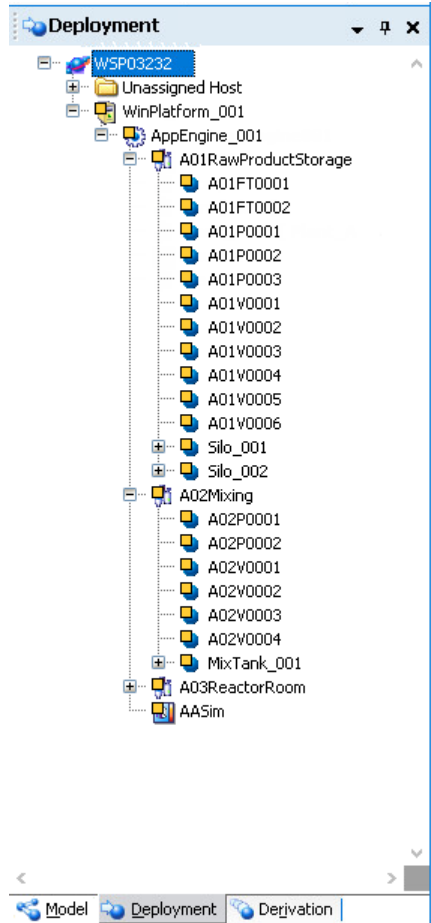
Application	Level	Installed RAM (GB)	min server mem (MB)
<b>Small</b> (1 - 25k I/O per Node)	Minimum	2 GB	682 MB
	Recommended	4 GB	1365 MB
<b>Medium</b> (25k - 50k I/O per Node)	Minimum	8 GB	2730 MB
	Recommended	12 GB	4096 MB
<b>Large</b> (> 50k I/O per Node)	Minimum	16 GB	5461 MB
	Recommended	24 GB	8192 MB

## Object, Area, Engine, and Platform Assignments

*IDE Overview* on page 19 shows the basic Galaxy structure. When you create an application object, such as a User Defined Object, it must be assigned to an area. When you look at your objects in the *Deployment View* on page 32, an area is assigned to an AppEngine, and an AppEngine is assigned to a WinPlatform.

The **Deployment** view uses the WinPlatform as its foundation, and all objects cascade from the WinPlatform object. The WinPlatform object can be thought of as the node (computer) that hosts the rest of the objects. The basic hierarchy of objects in the **Deployment** view is:

Galaxy > WinPlatform > AppEngine > Area > Application Object  
 DI Object



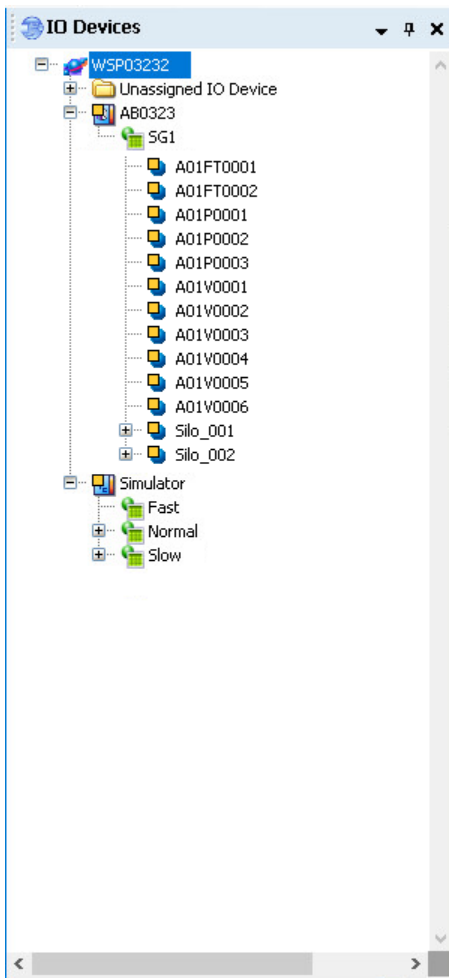
Data Integration (DI) Objects provide the interface between application objects and PLCs in the physical environment. DI Objects contain scan groups, and application objects are assigned to scan groups; this assignment can be done in the **IO Devices** view.

The relationship of other objects in the Galaxy to DI Objects is most clearly viewed in the **IO Devices** view. All IO between objects in the Galaxy and the PLC is through a DI Object. Scan groups are configured in the DI Object, and objects are assigned to a scan group.

Galaxy > DI Object > Scan Group > AppEngines  
 Areas  
 Application Objects



While objects can be deployed from any IDE view, the **Deployment** view provides the simplest way to understand the object hierarchy during deployment. The **IO Devices** view provides the simplest view of the relationship of each scan group and its assigned objects.



## Allocating AppEngines to Platforms

If you are using Flex licensing, each platform object can be configured for **No Engine**, for a **Single Engine**, or for **Unlimited Engines**. The **Deployment** view lets you configure Flex licensing for the WinPlatform objects in the galaxy. The engine configuration of the platform is only applicable if you are using Flex licensing, and is only configurable in the **Deployment** view.

---

**Note:** Although the Unlimited Engines option does not limit how many AppEngine objects can be assigned to a platform, there is actual limit that is determined by system performance and will vary due to system hardware, number of IO operations, historization, and other factors such as galaxy complexity and other running software. This limitation also applies when using perpetual licenses.

---

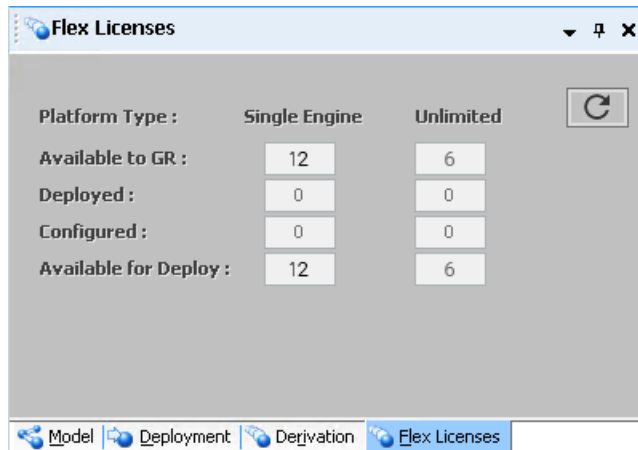
### To configure Flex licensing

---

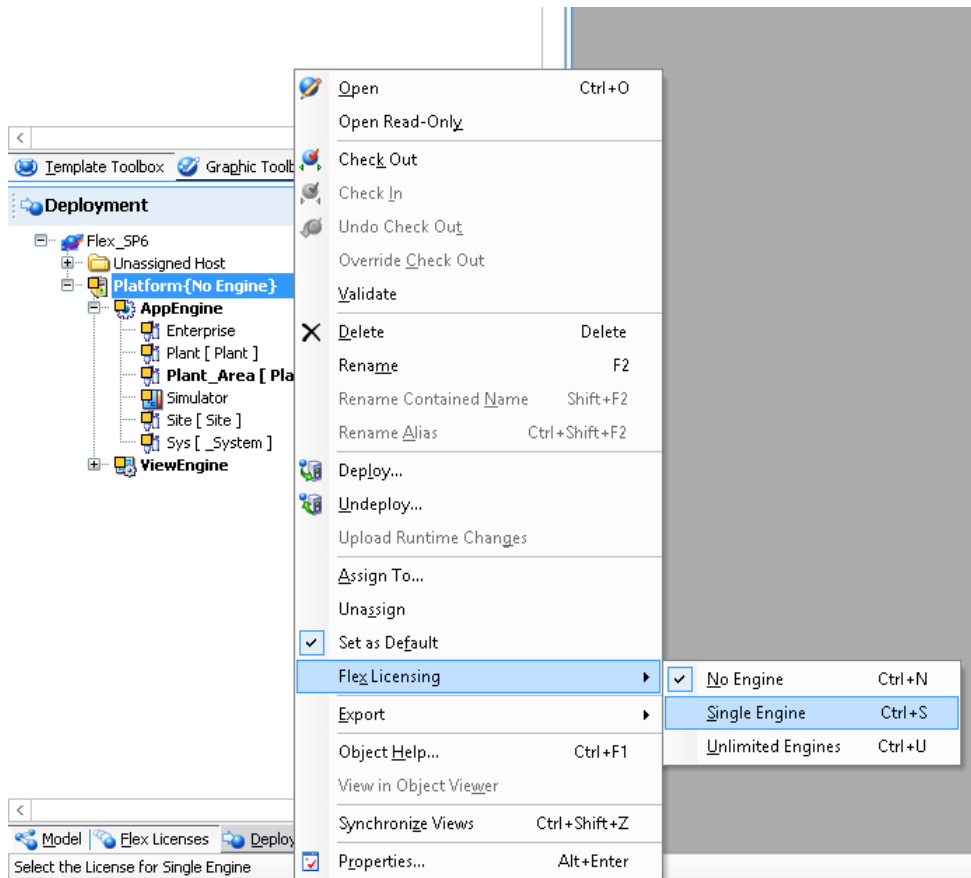
**Note:** You must undeploy a WinPlatform object before configuring or changing Flex license assignments.

---

1. Select the **Flex Licenses** view and check how many licenses are available.



2. Select the **Model** or **Deployment** view.
3. Select the WinPlatform object to be configured for licensing.
4. Right-click and select the applicable option from the context menu:



- When **None** is selected, the platform will not accept any Flex licenses. If you are using Flex licensing and assign an AppEngine to the platform, the platform will show a configuration error. You will not be able to deploy the platform, nor will you be able to deploy any AppEngine objects or any Application Objects in the deployment hierarchy of the platform object.

- When **Single Engine** is selected, you can assign one AppEngine object to the platform. If additional AppEngines are assigned, a configuration error will be displayed and the AppEngines and objects assigned to them will not be deployable. Note that when more than one engine is assigned, the license is assigned on a nondeterministic basis; that is, you will not know until deployment occurs which engine has been licensed. Unlicensed engines cannot be deployed.
  - When **Unlimited Engines** is selected, you can assign as many AppEngine objects to the platform as you like. There is actual limit that is determined by system performance and will vary due to system hardware, number of IO operations, historization, and other factors such as galaxy complexity and other running software.
5. Deploy the platform.
  6. To check Flex License availability, switch to the **Flex Licenses** view. License count is automatically updated upon deployment or undeployment of a platform, connection to the galaxy, or after a change to the platform license assignment. You can manually update the license count by clicking the refresh button in the **Flex Licenses** view.



## Data Integration Objects

Data Integration Objects (DI Objects) provide the interface between PLCs and Galaxy objects, including area objects and application objects. DI Objects contain scan groups (topics); application objects are assigned to individual scan groups to handle IO between the objects and the PLC.

## Auto-Build

Refer to **Using Auto-Build** in the *OI Core Communications Driver Help* for information about the Auto-Build process. Key points to remember include:

- Specify a new Galaxy as the target Galaxy until you are familiar with how Auto-Build works. This will help to avoid accidentally overwriting objects.
- Always specify a prefix. This will also help you avoid overwriting objects and lets you differentiate between similarly-named controllers.
- Use pre-check validation to catch illegal object names, such as names that exceed the maximum number of characters and that contain illegal characters. Note that Auto-Build skips names with errors .
- You can only run one Auto-Build process at a time.
- Auto-Build add two levels of derivation to accommodate different layers of standardization, for example, corporate standards (global) and local standards.

Auto-Build creates a DI Object for you. The DI Object is derived from the \$DDESuiteLinkClient template, and will include the two character prefix that you specify (first character must be alphabetic, the second character can be a letter or number (alphanumeric). The DI Object must include at least one scan group (topic).

The Auto-Build process adds the prefix that you specified to all objects it captures. After Auto-Build completes, you can view, test, and adjust object assignments as needed, using the different IDE views.

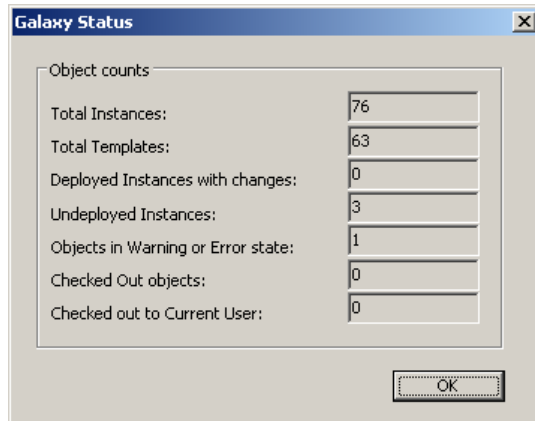
- Use the **Deployment** view to modify object assignments.
- Use the **IO Devices** view to modify scan group assignments.
- Use the **IO Device Mappings** view to verify and test IO references.

## Determining Galaxy Status

You can see an overview of the condition of your Galaxy before you deploy. This lets you know if you have objects that are in warning or error status.

### To determine the status of a Galaxy

1. Connect to the Galaxy.
2. On the **Galaxy** menu, click **Galaxy Status**. The **Galaxy Status** dialog box appears.



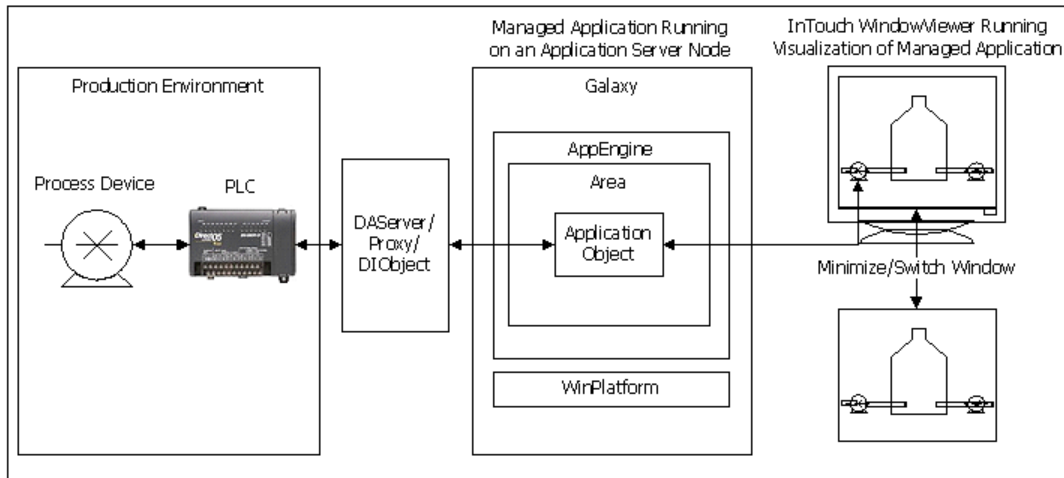
The **Galaxy Status** dialog box shows the counts of total instances, total templates, deployed instances with changes, undeployed instances with changes, objects that have an error or warning state, objects that are checked out, and object you have checked out.

3. Click **OK**.

## Configuring Advanced Communication Management

Advanced Communications Management minimizes network traffic and CPU usage of a DI Object and DAServer when a particular DI attribute is currently subscribed to, but its value is not currently shown. For example, scanning for updated values from a DI attribute representing pump RPM is suspended when an operator minimizes the application window containing a pump graphic containing attribute references that subscribe to the DI Object's pump RPM attributes.

**Note:** This dependence on the attribute being referenced by a running application is only the case when the attribute subscription is not already active due to configuration for history, alarming, or scripting. For example, if an attribute is configured for history, the subscription to the field device will always be active regardless of what window is active in InTouch, and regardless of whether InTouch is running at all.



When the items on the current window are no longer active because the window is minimized, a Suspend operation is performed for each item. Suspending scanning causes data updates to stop flowing to the application without deleting the subscriptions for each item. This reduces the overhead of unsubscribing and subscribing again to items when windows are minimized and restored.

In Advanced Communication Management, applications monitor the number of outstanding references to attributes that are updated with values from an external source object. When an attribute's external reference count reaches zero, the application sends a Suspend message to the source object requesting it to suspend updates. For example, a UserDefined object named PumpRPM has an attribute named RPM, which has an integer data type. The input source of RPM is plc1.n7:11, where plc1 is an ABPLC DI object.

When there are no active subscriptions to ud1.RPM, the subscription between input source and PumpRPM is suspended. When a client like ObjectViewer or a managed application running in WindowViewer subscribes to ud1.RPM, an Activate message is sent to the DI object to start sending data between the plc1 object and the PumpRPM object. This enables ObjectViewer or a managed client application running in WindowViewer to receive current plc1 data.

Unlike defined static attributes, some scan groups contain *dynamic* attributes. Dynamic attributes within a scan group provide the means for a running application's objects to dynamically subscribe to data from an external device. Dynamic attributes are created, activated and added to the scan group using Arcestra object attributes or InTouch indirect tags and scripts that include the IOSetRemoteReferences function.

When a dynamic attribute is poked, the time stamp is updated to the timestamp passed in with the value if available, or the current time provided by the hosting AppEngine is used. If the data provider passes in a Value, Time, Quality (VTQ) triplet of data in the callback, the time stamp is associated with the value and quality used to update the attribute.

To enable Advanced Communication Management, you must:

- Select Advanced Communication Management for your Galaxy.
- Set the scan mode for each scan group that belongs to your device integration objects within the Galaxy.

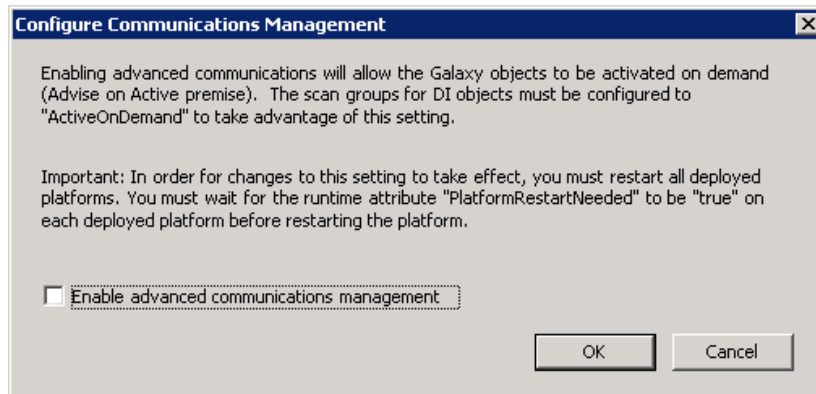
## Selecting Advanced Communication Management

Advanced Communication Management is a Galaxy-wide configuration setting that you set from the System Platform IDE. Advanced Communication Management is disabled by default.

When upgrading a Galaxy or restoring a Galaxy from a .cab file, the configuration setting that was previously selected is retained.

### To configure Advanced Communication Management

1. Connect to the Galaxy.
2. On the **Galaxy** menu, click **Configure** and then click **Communications Management**. The **Configure Communications Management** dialog box appears.



3. If necessary, select **Enable advanced communication management** and click **OK**.

---

**Note:** Any deployed platform and application engines must be restarted when the Advanced Communication Management state changes. Restarting should be done by means of the SMC.

---

## Configuring Scan Modes

A scan group is a collection of object attributes with associated data items. Scan group attributes reflect I/O points in the address space of Data Access Server whose values are polled from devices at a common update interval. The update interval for each scan group is inherited from the scan groups configured in the source DIObjects.

You can configure scan groups for OPCClient, InTouchProxy, DDESuiteLinkClient, and RedundantDIObject device integration objects. Within each scan group, you can specify data items to use as the object attributes. At run time, scan group items are updated with the latest values from the OPC DAServer according to the update interval.

The OPCClient, RedundantDI, DDESuiteLinkClient, and InTouchProxy objects contain a Scan Mode attribute that can be set for each scan group. The assigned scan mode value determines if objects are continuously updated with current data when an operator minimizes or closes the application window.

You can assign three different scan modes to a scan group:

- **Active**
  - Items are deleted and added to the scan group as requested (referenced) by the clients.
  - Items that exist when a scan mode change occurs are not deleted unless the previous mode was ActiveAll and the items are no longer referenced.
  - The Active scan mode polls all points that are referenced, whether active or inactive.
  - In Active scan mode an attribute is always in the active state. When the last reference to the attribute is unregistered/unadvised, the attribute is deleted.

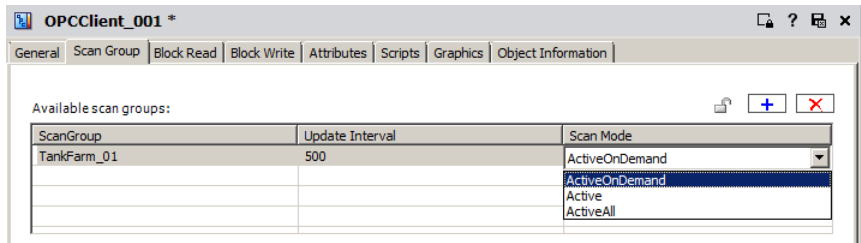
- **ActiveAll**
  - Scanning is continuous and dynamic attributes are never removed when unsubscribed.
  - Items are activated by client requests, but continue to exist even after the client are not interested in them anymore.
  - Items are not removed when the scan mode changes.
  - The scan group polls the field device for all points irrespective of whether they are currently active, inactive, or even subscribed to by items.
- **ActiveOnDemand**
  - When the scan mode is configured as ActiveOnDemand, attributes that are not actively being referenced by any client or object are made Inactive and are not scanned.
  - New Items are deleted and added to the scan group as requested (referenced) by clients.
  - Items that exist when a scan mode change occurs are not deleted unless the previous mode was ActiveAll and the items are no longer referenced.
  - ActiveOnDemand scan mode polls the field device for currently active items only. Inactive items are not scanned.

The following table shows scan mode scan states for dynamic attributes based on whether items are referenced or not.

Dynamic Attribute State	Scan Mode Active	Scan Mode ActiveAll	Scan Mode ActiveOnDemand
Reference/ Some active	Scan	Scan	Scan
Reference/All Inactive	Scan	Scan	No Scan
Not Referenced	Delete	Scan	Delete

**To set a scan mode for a scan group**

1. Edit the device integration object.
2. Show the editor page containing scan groups by completing one of the following:
  - Click the **Scan Group** tab if you are editing a RedundantDIOobject or OPCClient object.
  - Click the **Topic** tab if your are editing a DDESuiteLinkClient object.
  - Click the **Items Configuration** tab if your are editing an InTouchProxy object.



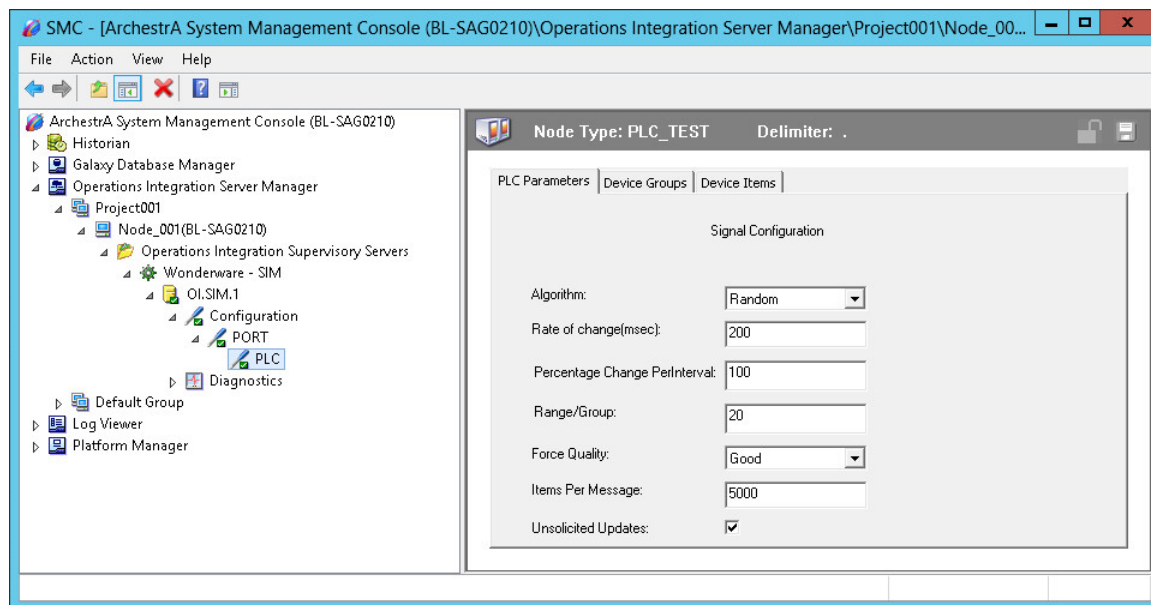
3. If necessary, add a scan group by clicking the plus sign box above the list of available scan groups.

4. If necessary, set the length of the scan group update interval in milliseconds. The default update interval is 500.
5. Double-click in the **Scan Mode** box of the scan group to show a drop-down list of available scan modes. ActiveOnDemand is the default scan mode.
6. Select a scan mode from the list.
7. Save your changes to the device integration object and exit from the editor.

## Simulation Server

System Platform includes a simulated data access server that lets you develop and test a Galaxy before deploying it to your production environment.

The simulation server's hierarchy can be viewed in the **Operations Integration Server Manager**, within the **System Platform Management Console (SMC)**. To run the SMC, from the **Start** menu, point to **AVEVA Utilities** and select **System Platform Management Console**.



The simulation server is automatically installed on the Galaxy Repository node for Galaxies derived from the Default.cab file, and is always available. It mimics the data output from a PLC. The simulation server sends data to attributes of instances configured for I/O Automatic Assignment in your Galaxy, the same way that a PLC would.

The simulation server contains all the system objects needed to create a functional test environment for testing I/O references that use I/O Automatic Assignment. However, some manual configuration is required before the simulation server can be used. The simulation server includes an Operations Integration server and a PLC simulator.

The simulation server is based on an OPC client device integration (DI) object. You can add it to Galaxies that do not contain it by default. If you need to add the simulation server, you will configure it the same way as you would an OPC client in a normal production environment. Required configuration steps include choosing the server node and server name, and adding scan groups. This process is described in more detail below.

In its default configuration, the simulation server consists of the following elements:

- A channel for OPC communications between your Galaxy and the simulation server. The communications channel has the default name "Port."



- A PLC simulator to generate simulated data for your Galaxy. The PLC simulator has the default name "PLC."
- A device integration (DI) object that is used to link the PLC simulator to the instances in your Galaxy (OI.Sim.1).

## Simulation Server and the OPCClient Object

The simulation server uses the OPCClient object as its basis. If your Galaxy does not include the simulation server object, you can configure one from the \$OPCClient template. The OPCClient object can be used with the I/O Auto Assignment feature. In simulated environments, application objects in your Galaxy that contain I/O references are linked with the simulation server. In a production environment, these references would be linked to actual PLCs.

To use the simulation server, you must link instances to one of the simulation server's scan groups in the **IO Devices view** (the default scan group is **Fast**). The simulation server will return integer values for all supported data types. See *View Simulated Data* on page 331 for the list of supported data types.

When you view simulated run-time data, a snap shot value is provided for each attribute that has I/O Auto Assignment. Since the simulation server tests I/O Auto Assignment references, references verified as good with the simulation server will automatically reconfigure when you associate the instances with the appropriate DI object and deploy the Galaxy in your actual environment.

## I/O Auto Assignment in the Simulation Server

When using the I/O Auto Assignment feature, the simulation server uses the normal namespace used by I/O Auto Assignment, that is:

*<IODeviceName>.<ScanGroupName>.<ObjectName>.<AttributeName>*

You can view I/O assignments for objects in the **I/O Device Mapping View**. For objects configured with the simulation server and assigned to a Scan Group, the device and scan group for the I/O resulting reference will map as follows:

*<IODeviceName> = **Simulator***

<ScanGroupName> = **Fast**, **Medium**, or **Slow**

The screenshot shows the 'IO Device Mapping' window with a table containing the following data:

Device	ScanGroup	Object	Attribute	I/O	Resulting Reference	De	Object.Attribute Overri	Runtime Value
Simulator	Fast	UserDefined_001	Attribute001	I	<b>Simulator.Fast.Tank001.Outlet</b>		Tank001.Outlet	
Simulator	Fast	UserDefined_001	Attribute002	I	Simulator.Fast.UserDefined_001.Attribute002			
Simulator	Fast	UserDefined_001	Attribute003	I	Simulator.Fast.UserDefined_001.Attribute003			
Simulator	Fast	UserDefined_001	Attribute004	I	Simulator.Fast.UserDefined_001.Attribute004			
Simulator	Fast	UserDefined_001	Attribute005	I	Simulator.Fast.UserDefined_001.Attribute005			
Simulator	Fast	UserDefined_001	Attribute006	I	<b>Simulator.Fast.Tank001.Inlet</b>		Tank001.Inlet	
Simulator	Fast	UserDefined_002	Attribute001	I	Simulator.Fast.UserDefined_002.Attribute001			
Simulator	Fast	UserDefined_002	Attribute002	I	Simulator.Fast.UserDefined_002.Attribute002			
Simulator	Fast	UserDefined_002	Attribute003	I	Simulator.Fast.UserDefined_002.Attribute003			
Simulator	Fast	UserDefined_002	Attribute004	I	Simulator.Fast.UserDefined_002.Attribute004			
Simulator	Normal	UserDefined_003	Attribute001	I	Simulator.Normal.UserDefined_003.Attribute001			
Simulator	Normal	UserDefined_003	Attribute002	I	Simulator.Normal.UserDefined_003.Attribute002			
Simulator	Normal	UserDefined_003	Attribute003	I	Simulator.Normal.UserDefined_003.Attribute003			
Simulator	Slow	UserDefined_004	Attribute001	I	Simulator.Slow.UserDefined_004.Attribute001			
Simulator	Slow	UserDefined_004	Attribute002	I	Simulator.Slow.UserDefined_004.Attribute002			
Simulator	Slow	UserDefined_004	Attribute003	I	Simulator.Slow.UserDefined_004.Attribute003			
Simulator	Slow	UserDefined_004	Attribute004	I	Simulator.Slow.UserDefined_004.Attribute004			
Simulator	Slow	UserDefined_004	Attribute005	I	Simulator.Slow.UserDefined_004.Attribute005			
Simulator	Slow	UserDefined_004	Attribute006	I	Simulator.Slow.UserDefined_004.Attribute006			
Simulator	Slow	UserDefined_004	Attribute007	I	Simulator.Slow.UserDefined_004.Attribute007			
Simulator	Slow	UserDefined_004	Attribute008	I	Simulator.Slow.UserDefined_004.Attribute008			

The IO Device name for the simulation server must not be changed from its default name, **Simulator**.

You assign application objects (instances) to the simulator server's scan groups the same way as you would assign instances to an I/O device's scan groups. The default scan group is **Fast**.

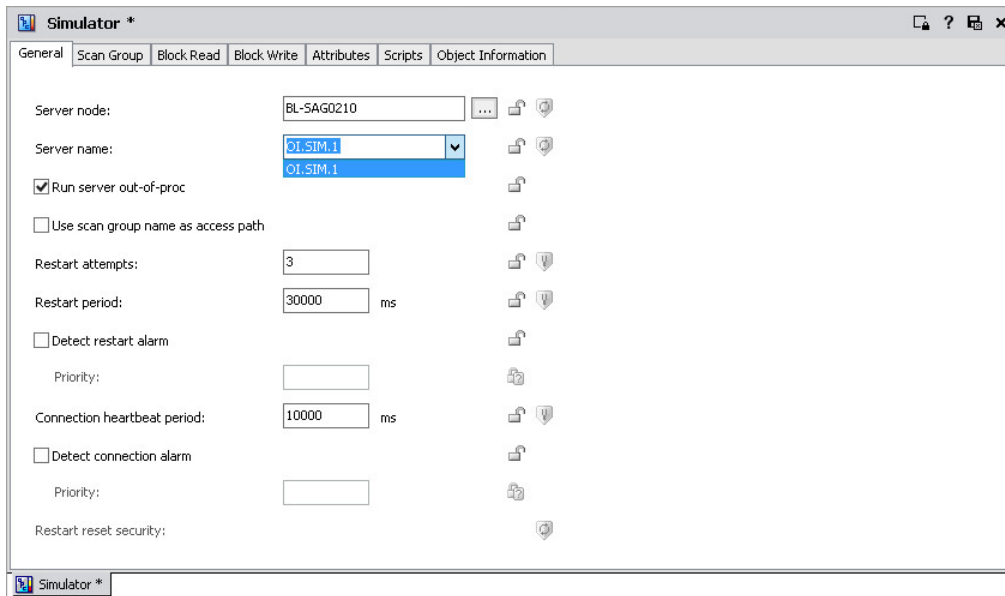
## Configure the Simulation Server

The Simulation Server is automatically installed and configured for Galaxies derived from the Default.cab file. You can create a simulator server for Galaxies derived from other cab files that contain the \$OPCCClient base template, such as the Default\_EMPTY.cab, or you can import the \$OPCCClient object after exporting it from another Galaxy.

### To configure a new Simulation Server

1. Derive an instance from the **\$OPCCClient** object in the **Template Toolbox**.
2. Rename the new OPCClient instance **Simulator**.
3. Open **Simulator** in Object Editor and configure it as follows:
  - **General** page:
    - a. Enter the **Server node** name.

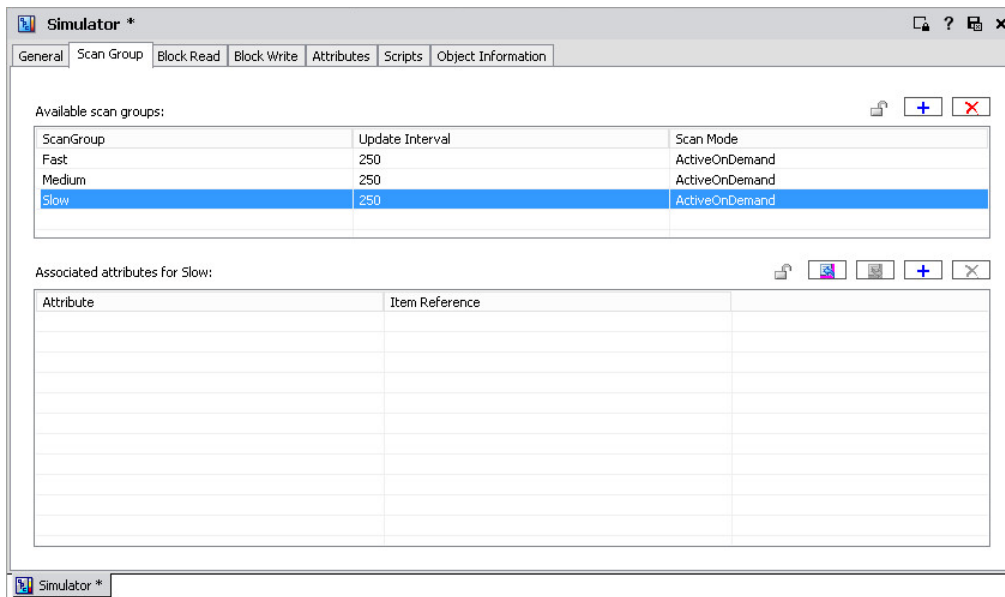
b. Select *OI.Sim.1* for **Server name**.



**Scan Group** page:

c. Add a scan group called Fast.

d. If desired, you can also add Medium and Slow scan groups. You can leave *Update Interval* and *Scan Mode* at their default settings.



4. Save and close **Simulator**.

## View Simulated Data

Before you can view simulated run-time data, your Galaxy:

- Must contain attributes that are configured to use I/O Auto Assignment.
- Must be deployed along with the Simulator and any instances with I/O Auto Assignment. If necessary, you can link instances to the simulator server in the **IO Devices** view.

**Note:** When instances are created, they are assigned to the Simulator Fast scan group by default. To change the scan group assignment of an asset, you must undeploy the instance first.

### To connect to the Simulation Server

1. In the **IO Device Mapping view**, select the **Validate Reference(s)** button. The simulation server generates simulated numeric data and displays a randomly generated number for attributes that contain valid input references. Validated references are shown with a green background. References that did not validate are shown with a red background.
2. To refresh the simulated run-time value, select the **Validate Reference(s)** button again.

Device	ScanGroup	Object	Attribute	I/O	Resulting Reference	Runtime Value
Simulator	Fast	UserDefined_001	Attribute001	I	Simulator.Fast.Tank001.Outlet	8
Simulator	Fast	UserDefined_001	Attribute002	I	Simulator.Fast.UserDefined_001.Attribute002	3
Simulator	Fast	UserDefined_001	Attribute003	I	Simulator.Fast.UserDefined_001.Attribute003	0
Simulator	Fast	UserDefined_001	Attribute004	I	Simulator.Fast.UserDefined_001.Attribute004	0
Simulator	Fast	UserDefined_001	Attribute005	I	Simulator.Fast.UserDefined_001.Attribute005	0
Simulator	Fast	UserDefined_001	Attribute006	I	Simulator.Fast.Tank001.Inlet	11
Simulator	Fast	UserDefined_002	Attribute001	I	Simulator.Fast.UserDefined_002.Attribute001	7
Simulator	Fast	UserDefined_002	Attribute002	I	Simulator.Fast.UserDefined_002.Attribute002	2
Simulator	Fast	UserDefined_002	Attribute003	I	Simulator.Fast.UserDefined_002.Attribute003	12
Simulator	Fast	UserDefined_002	Attribute004	I	Simulator.Fast.UserDefined_002.Attribute004	16
Simulator	Normal	UserDefined_003	Attribute001	I	Simulator.Normal.UserDefined_003.Attribute001	6
Simulator	Normal	UserDefined_003	Attribute002	I	Simulator.Normal.UserDefined_003.Attribute002	4
Simulator	Normal	UserDefined_003	Attribute003	I	Simulator.Normal.UserDefined_003.Attribute003	15
Simulator	Slow	UserDefined_004	Attribute001	I	Simulator.Slow.UserDefined_004.Attribute001	13
Simulator	Slow	UserDefined_004	Attribute002	I	Simulator.Slow.UserDefined_004.Attribute002	20
Simulator	Slow	UserDefined_004	Attribute003	I	Simulator.Slow.UserDefined_004.Attribute003	15
Simulator	Slow	UserDefined_004	Attribute004	I	Simulator.Slow.UserDefined_004.Attribute004	4
Simulator	Slow	UserDefined_004	Attribute005	I	Simulator.Slow.UserDefined_004.Attribute005	11
Simulator	Slow	UserDefined_004	Attribute006	I	Simulator.Slow.UserDefined_004.Attribute006	11
Simulator	Slow	UserDefined_004	Attribute007	I	Simulator.Slow.UserDefined_004.Attribute007	8
Simulator	Slow	UserDefined_004	Attribute008	I	Simulator.Slow.UserDefined_004.Attribute008	19

### Supported Data Types

The simulation server supports the following data types:

- Boolean
- Integer
- Float
- Double
- String
- Time
- ElapsedTime
- InternationalizedString

All supported data types are treated as integer and return integer data for validated references.

### Unknown Data Types

The simulation server treats attributes with unknown types as integer.

## Deploying Objects

You deploy object instances for two reasons:

- Testing.
- Placing the application into production to process field data.

When you are ready to deploy, make sure the following conditions are met:









- Bootstrap software is installed on the target computers.
- The objects being deployed are not in an error state in the Galaxy database.
- You created, configured, and checked in objects to the Galaxy.
- Objects are assigned to a host.
- The object's host is already deployed. A cascade deploy operation, which deploys a hierarchy of objects, deploys all objects in the correct order. This deploys an object's host before the object is deployed.
- Any associated script libraries are ready for use on the target computer. For more information, see *Importing Script Function Libraries* on page 109.

---

**Note:** DInetwork objects have specific configuration limits. For example, whether more than one object can be deployed to a single WinPlatform. The IDE does not check for these limits. For more information, see the help file for the DInetwork object for specifics on configuration limits.

---

You can tell if you have objects that need to be deployed by looking at the icons next to the objects. Deployment status icons include:

-  Not deployed
-  Deployed
-  Deployed, but pending configuration changes exist that have not been deployed.
-  Deployed, but software modifications exist that have not been deployed.
-  Applies only to redundant AppEngines. An AppEngine is undeployed, but its redundant pair is deployed.
-  Applies only to redundant AppEngines. An AppEngine is deployed, but its redundant pair is not deployed.
-  Applies only to redundant AppEngines. An AppEngine is deployed, its redundant pair is not deployed, and pending configuration changes exist that have not been deployed.
-  Applies only to redundant AppEngines. An AppEngine is deployed, its redundant pair is not deployed, and software modifications exist that have not been deployed.
- [No icon] Good. No additional icon is displayed for deployed objects running on-scan or off-scan with a good status.



Warning



Error. The object is in an Error state and cannot be deployed.

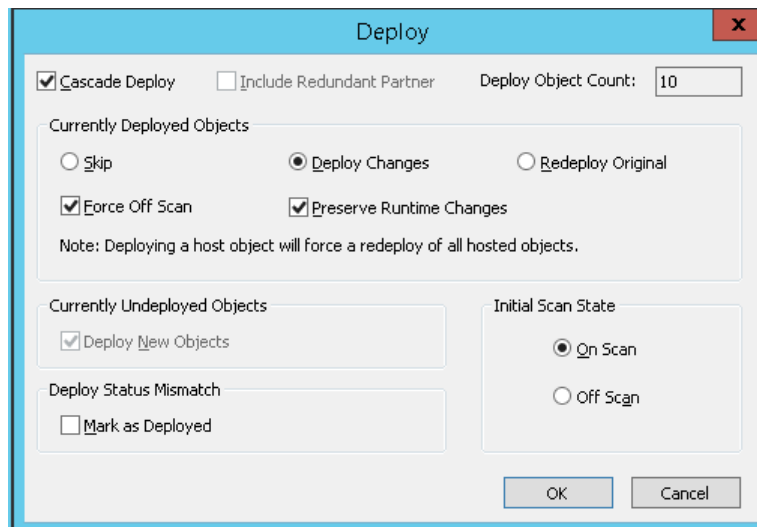


InTouchViewApp application files are being asynchronously transferred to the target node. This icon is normally visible for only a few moments at the end of an InTouchViewApp deployment operation, unless the object is deployed on a slow network.

This icon is larger than the other icons and completely replaces the original while it is being shown.

## To deploy objects

1. Select the objects you want to deploy in an **Application view**.
2. On the **Object** menu, click **Deploy**. The **Deploy** dialog box appears.



3. Select one or more of the following:
  - Cascade Deploy:** Select this check box to deploy the object selected for deployment as well as any objects it hosts. This option is selected by default if the object is a host. If you are deploying an individual host object, clear the check box. Objects being deployed across multiple platforms are deployed in parallel.
  - Include Redundant Partner:** Select this check box to also deploy an AppEngine's redundancy partner object. This option is selected and unavailable when the redundant engine has pending configuration changes or software updates.
4. In the **Currently deployed objects** area, select one or more of the following options. These options are not available if the selected object has not been deployed before.
  - Skip:** If one of the objects you are deploying is currently deployed, selecting **Skip** makes no changes to the already-deployed object.
  - Deploy Changes:** If one of the objects you are deploying is currently deployed, this option updates the object in question new configuration data. If you are redeploying an object, it is updated with new run-time data or new configuration data, depending on whether a change was made to a configured value, to a run-time value, and whether or not the **Preserve Runtime Changes** option is selected.

- If both **Deploy Changes** and the **Preserve Runtime Changes** option are selected, values changed during run time are loaded as the object is redeployed. The new run-time value is used, even if you configure a new value in the IDE.
  - If **Deploy Changes** is selected but the **Preserve Runtime Changes** option is NOT selected, new configured values set in the IDE are used, and will overwrite any changes made during run time.
  - **Redeploy Original:** If one of the objects you are deploying is currently deployed, this option deploys the same version as previously deployed. For example, use this option to redeploy an object that is corrupted on the target computer. If the **Preserve Runtime Changes** option is selected, the object is deployed with the last run time value, even if you change the configured value in the IDE.
  - **Force Off Scan:** If one of the objects you are deploying is currently deployed, this option sets the target object to off scan before deployment occurs.
  - **Preserve Runtime Changes:** Selecting this option will produce different results, depending on whether you have chosen **Deploy Changes** or **Deploy Original**, if values were changed at run time, and if configuration values were changed in the IDE.
    - If a value was changed in run time and **Deploy Changes** and this option are checked, the object is deployed with the last run time value, even if you change the configured value in the IDE.
    - If a value was not changed during run time and **Deploy Changes** and this option are checked, the object is deployed with the last configured value from the IDE.
    - If a value was changed in run time and **Redeploy Original** and this option are checked, the object is deployed with the last run time value, even if you change the configured value in the IDE.
    - If a value was not changed during run time and **Redeploy Original** and this option are checked, the object is deployed with the original configured value from the IDE. If you configured a new value in the IDE, the prior configured value is used.
5. In the **Currently undeployed objects** area, select the **Deploy New Objects** check box to start a normal deployment.
6. In the **Deploy Status Mismatch** area, select the **Mark as Deployed** check box to mark the object as deployed in the Galaxy. A mismatch happens when the object is previously deployed to a target node, but the Galaxy shows the object is undeployed. Clear this option to redeploy the object to the target node.
7. In the **Initial Scan State** area, select one of the following:
- **On Scan:** Sets the initial scan state to on scan for the objects you are deploying. If the host of the object you are deploying is currently off scan, this setting is ignored and the object is automatically deployed off scan. When you are deploying multiple objects, the deploy operation deploys all of the selected objects "off-scan." After all of the objects are deployed, the system sets the scan-state to "on-scan."
- Objects can run only when both the host/engine is "on scan" and the object is "on scan." If either the host/engine or the object is "off scan," the object cannot run.
- Always deploy Areas to their host AppEngines on scan. Because Areas are the primary providers to alarm clients, deploying Areas off scan results in alarms and events not being reported until they are placed on scan.
- **Off Scan:** Sets the initial scan state to off scan for the objects you are deploying. If you deploy objects off scan, you must use the ArcestrA System Management Console Platform Manager utility to put those objects on scan and to function properly in the run-time environment.

---

**Note:** The System Management Console controls the state of the host/engine. The ObjectViewer controls the state of the objects.

---

The default scan setting is set in the **User Default** settings in the **Configure User Information** dialog box. For more information, see *Configure User Information* on page 38.

8. Click **OK** to deploy the objects. The **Deploy** progress box appears. If you see error messages, see *Deployment Error Messages* on page 336. When the deploy is complete, click **Close**.

## Deployment Error Messages

If the object being deployed has configuration problems that were not known during configuration, the **Deploy** progress box shows messages.

The **Progress** dialog box also shows the affected instances and any error and warning messages. The target object can take any actions necessary to achieve a valid state, including changing attribute values provided during deployment.

WinPlatforms are the only objects whose configuration designates its deployment location. Deployment problems unique to WinPlatforms are the following:

- The target computer could not be found on the network. Ensure the WinPlatform was configured properly and the target computer is properly connected to your network.
- Another WinPlatform is deployed already to the target computer. Resolve this problem by undeploying the existing WinPlatform before deploying the new one.
- The target platform is running on the old version of the product. To resolve this problem, the user must upgrade the remote platform.
- If a WinPlatform object is deployed on a slow network and it does not respond to the IDE before the 30-second message time-out, a communication error occurs and the object is shown as not deployed. You may need to adjust the message time-out for the WinPlatform object to accommodate the slow network speed.

## Publishing Managed InTouch Applications

You can publish a managed InTouch application using the System Platform IDE. As part of this process, you can create a compressed, self-extracting package file that contains all relevant files and set-up procedures to install an InTouch application on another computer. For more information, see the *InTouch HMI Application Management and Extension Guide*.

## Redeploying Objects

Updating assets that are already deployed requires a redeployment of the asset. Redeploying an asset or a group of hierarchical assets can be done almost seamlessly, with very little downtime. When you redeploy an asset, a maximum of two scan cycles are needed between the time the existing asset configuration is undeployed and the asset with new configuration information is deployed in its place (typically, less than one second). This redeployment workflow provides two important benefits:

- Minimized downtime. While an asset is down, it does not provide status updates and does not respond to scripts or commands.
- Reduced points of failure. The existing configuration remains in effect until the new configuration is available.



This minimal downtime needed to redeploy assets ensures that uptime is maximized and the impact on operations is minimized. To ensure against a deployment failure due to communications or other error, the existing asset configuration remains deployed and onscan until the new configuration has been successfully deployed. Therefore, if a redeploy attempt fails for any reason, operations are not affected, as the existing configuration remains available within the production environment. The asset is not stopped or taken offscan until the new configuration replaces it. Since less than two scan cycles are needed for a redeploy operation, and this occurs only at the final stage of pushing out the new configuration, there is little risk of adverse effect on operations.

- Redeploying assets follows the same workflow as deploying assets. See *Deployment Overview* on page 317 for additional information. Redeployment occurs when you select and deploy the asset.

When assets with retentive attributes are redeployed, the Application Engine for the assets restores the retentive attributes to the values previously set at run time, if the **Preserve Runtime Changes** option is selected. See *Deploying Objects* on page 333 and *Retentive Attribute Properties and Behavior* on page 338 for additional information.

You may have an object with a Pending Update deployment state. The Pending Update state means the object changed since its last deployment. When you deploy those changes, the new object is marked as the last deployed version in the Galaxy.

### To redeploy

1. On the **Object** menu, click **Deploy**.
2. Follow the procedure for *Deploying Objects* on page 333.

## Redeploying the WinPlatform Object

You can redeploy a WinPlatform object without undeploying and redeploying the objects hosted on WinPlatform. When you make changes to the WinPlatform object, you can propagate the changes by redeploying WinPlatform alone, without taking AppEngines and other objects hosted on the WinPlatform off-scan.

### To redeploy a WinPlatform without redeploying hosted objects

1. Select the WinPlatform object on the **Object** menu that has updates pending, then select **Deploy**. The **Deploy** dialog box appears.
2. Clear the **Cascade Deploy** check box to prevent other hosted objects from being redeployed. By default, the check box is selected.

---

**Note:** It is not necessary to redeploy hosted objects if you are uploading changes only to the WinPlatform object.

---

3. In the **Currently Deployed Objects** area:
  - Select the **Deploy Changes** option. This sends any configuration changes made in the IDE to the objects being deployed.
  - Select the **Preserve Runtime Changes** option if you want to save overrides to configured settings that may have been made during run time. See *Retentive Attribute Properties and Behavior* on page 338 for more information on preserving run-time changes.

For more information on deploying objects, see *Deploying Objects* on page 333.

4. Select **OK**. The **Deploy** progress box appears. If an error occurs during deployment, an error message appears. For more information on error messages during deployment, see *Deployment Error Messages* on page 336.

---

**Important:** The WinPlatform object will briefly shut down during the redeploy operation as checkpoints are synchronized. However, AppEngines and other objects remain deployed.

---

5. Select **Close** when deployment is complete.

---

**Note:** If the WinPlatform object hosts redundant AppEngines, the AppEngine may automatically fail over if the failover timeout is detected by the standby engine.

---

## Retentive Attribute Properties and Behavior

Some changes that are made to attributes at run time are automatically tracked and saved. These attributes are called "retentive attributes." Changes made at run time to retentive attributes are automatically retained. If an object with a retentive attribute is undeployed and subsequently redeployed with the **Preserve Runtime Changes** option selected in the **Deploy** dialog, the last run time value is loaded into the retentive attribute. If the **Preserver Runtime Changes** option is NOT selected, the object is redeployed with its configured setting, not the value set during run time. See *Deploying Objects* on page 333 for additional information.

Changes to retentive attributes are retained as long as the new attribute value is different than the previous saved value, and data quality of the attribute is good. You do not have to do anything to mark an attribute as a retentive attribute. Attributes are automatically set to be retentive if all of the following are true:

- The attribute is checkpointed.
- The attribute is User- or Object-Writeable.
- The attribute does not contain an input extension or input/output extension.

At every checkpoint period, the AppEngine looks for changes to retentive attributes on objects that it hosts. A change made at run time to a retentive attribute is saved on the node where the AppEngine is running. When any of the retentive attributes associated with the AppEngine have been changed, the changed values are loaded into memory. After several checkpoint periods have passed, the changed values in memory are written to disk, and memory is cleared. In this way, only values that have changed are retained. However, retentive attribute run time changes are not uploaded to the IDE. Therefore, if an object that contains a retentive attribute is undeployed and moved to a different node or even to a different AppEngine, the run time changes are not preserved when the object is redeployed.

For redundant pairs, changes will be preserved, even if the redundant partner is on a different engine or node. If the value of a retentive attribute changes and it is part of a redundant pair, the changed value will be sent to the redundant partner during checkpoint synchronization, if the following are both true:

- Attribute data quality is good
- The new attribute value is different than the previous value

## Undeploying Objects

You may need to undeploy one or more objects. Undeploying removes one or more objects from the run-time environment.

Before you start, you need to select the object or objects you want to undeploy in the IDE.

Before you delete or restore a Galaxy, undeploy all objects in the Galaxy.

Undeploying can fail if the target object has objects assigned to it. Make sure you select **Cascade Undeploy** in the **Undeploy** dialog box.

### To undeploy

1. On the **Object** menu, click **Undeploy**. The **Undeploy** dialog box appears.

In the upper right of this dialog box, the **Undeploy Object Count** box shows the number of objects being undeployed. You can select a single object in **Application view** and, if you selected **Cascade Undeploy** and other objects are assigned to the selected object, the total number of objects appears in this box.

2. Select one or more of the following. Some of these options might not be available, depending on the kinds of object you select.

- Cascade Undeploy:** Select to undeploy the selected object as well as any objects it hosts.
- Include Redundant Partner:** Select to also undeploy an AppEngine's redundancy partner object.

---

**Note:** The AppEngine in a redundant pair that was configured as the Primary can be undeployed alone because objects hosted by it run on the deployed Backup AppEngine, which becomes Active.

---

- Force Off Scan:** If one of the objects you are undeploying is currently on scan, selecting **Force Off Scan** sets the target object to off scan before undeployment. If you do not select **Force Off Scan** and the target object is on scan, the undeployment operation fails.
- On Failure Mark as Undeployed:** Marks the object as undeployed in the Galaxy when the object targeted for undeployment is not found.

## Uploading Run-time Configuration

You can upload run-time configuration changes to the Galaxy database. This lets you keep any attribute values that changed during run time.

---

**Note:** A node cannot upload run-time changes to the Galaxy if the version of Application Server it is running is older than the version running on the GR node.

---

Certain attribute values can be set in the configuration environment, but they can also change at run time. As a result, the values of these attributes can differ between the run-time and configuration environments. These types of attribute writeability are:

- Calculated
- Object Writeable
- Calculated Retentive
- User Writeable

For example, you create an object with a User Writeable attribute called `myInteger` and specify an initial value of 30 in the Object Editor.

Then you deploy the object. At run time, you write a new value to `myInteger` of 31. If you redeploy this object, the original value of 30 overwrites any value assigned during run time. To avoid losing changes made during run time, upload changes before redeploying an object.

Run-time configuration changes to auto assigned I/O references are uploaded as overrides to the original assignments. The I/O reference will continue to be auto assigned and the run-time changes are retained when the object is redeployed. See *Uploading Run-Time Configuration Changes for Auto Assigned Objects* on page 157 for additional information.

Objects whose configuration are successfully uploaded have a new version number and a change log entry for the upload operation. The run-time object's version number also has a new version number. That version number matches the version in the configuration database.

### To upload run-time changes to the Galaxy

1. Select one or more objects in the **Model** view or **Deployment** view. For example, you could select an entire hierarchy from AppEngine down.
2. On the **Object** menu, click **Upload Runtime Changes**. The run-time attributes of the selected objects are copied over those in the Galaxy database.

## Uploading Restrictions

Certain run-time changes to the Galaxy cannot be uploaded. Before uploading changes, make sure the selected objects are:

- Not checked out
- Not edited and checked in since last deployment or upload
- Not in Pending Update state

If you select an object that is currently checked out to you, a warning appears during run-time upload. If you continue, you lose all configuration changes you made to the checked out object. The Galaxy performs an Undo Check Out operation on it before the run-time attributes are copied to the Galaxy database.

---

**Note:** You cannot upload run-time changes for objects checked out to other users.

---

## Undeployment Situations

The following situations occur in these specified undeployment scenarios:

- After undeploying a WinPlatform, only the Bootstrap software remains on the target computer. All other WinPlatforms are notified of the undeployment of the WinPlatform and stop trying to communicate with it over the network.
- Alarm Clients know immediately that an area is undeployed and is no longer available. They remove the area from selection lists. Alarms associated directly with the area (not as a result of containment) are immediately removed from current alarm views.
- Undeploying an object that has Pending Updates status removes that status. It is now marked as undeployed.
- If cascade undeploy fails on one object, then the undeploy continues to the extent possible on other objects. The entire operation is not terminated because the undeploy fails for one object. However, a host might not be undeployable if one of its assigned objects cannot be undeployed.

Assume an ApplicationObject is hosted by the Active AppEngine in a redundant pair and a number of subscriptions is configured in that ApplicationObject that refers to items in a DIOject. If you undeploy the ApplicationObject in question, the items are not removed immediately from the item count of the DIOject. How fast those items are removed depends upon the value of the **Maximum time to maintain good quality after failure** option (Redundancy.StandbyActivateTimeout attribute) on the **Redundancy** page in the AppEngine's editor. This behavior does not apply to the undeployment of ApplicationObjects hosted by non-redundant AppEngines.

## Associating All Galaxy Graphics with an InTouchViewApp

Associating all Galaxy graphics with an InTouchViewApp template enables deployed and published InTouch applications to execute "show graphic" requests made of any graphic in the Galaxy without having to embed them in the application.

- The ShowGraphic() function uses the graphic as a parameter. Associating all Galaxy graphics ensures that the graphic is deployed and available at run time whether or not it is referenced by InTouchViewApp.
- Associating all Galaxy graphics ensures that template symbols referenced by the ShowGraphic() function are deployed and available at run time.

---

**Note:** The term "graphic" includes any symbol or client control present in the Graphic Toolbox, and any symbols owned or inherited by templates and instances.

---

## About Associating All Galaxy Graphics with an InTouchViewApp

Associating all Galaxy graphics with an InTouchViewApp (ITVA) exhibits the following deployment and publishing behavior:

- The associate all Galaxy graphics option—"Include all Galaxy graphics"—changes only the ITVA template. The option is inherited by the template instances, and changes the template only if none of its instances are deployed.
- A deployed ITVA with the "Include all Galaxy graphics" option enabled will be marked "pending change" under the following scenarios

Scenario	Operations Performed
Creating graphics	Create new graphics in the graphics toolbox Check in new symbol(s) in an automation object Import automation object with symbol(s) Import graphics toolbox graphics Create a new derived template or instance object that inherits symbols GR Load new symbols into the galaxy
Modifying graphics	Edit and check-in symbol(s) Modifying symbols through an Import operation Modifying symbol localization through an Import operation
Deleting graphics	Delete graphics from the graphics toolbox Check in an application object from which symbols have been deleted Delete template or Instance object with symbols
Renaming graphics	Rename graphics in the graphics toolbox Check in an application object, any of whose symbols have been renamed
Rename object	Rename Instance or Template which has owned or inherited symbols
Object containment	Containment of automation objects that own or inherit graphics

- Deploying an ITVA with the "Include all Galaxy graphics" option enabled for the first time will deploy all checked-in graphics in the Galaxy. Client controls will be deployed whether or not they are embedded in a symbol. Undeploy behavior does not change.
- Deploying changes with an ITVA that has the "Include all galaxy graphics" option enabled will deploy the Galaxy graphics changes since the previous ITVA deployment.
- Redeploying the original ITVA that has the "Include all Galaxy graphics" option enabled will not account for graphics that were added, deleted, renamed or modified since the original deployment.
- Publishing an ITVA with the "Include all galaxy graphics" option enabled will copy all the checked-in galaxy graphics to the published application folder. The ITVA will retain its "pending change" status as this is not a deploy operation.

- The "Include all Galaxy graphics" enabled setting will persist through a Galaxy backup and restore operation, but will not persist through an ITVA import or export, or through GR dump and GR load operations.

## Configuring the Include All Galaxy Graphics Option

Access the option through the InTouchViewApp right-click context menu.

### To include all Galaxy graphics with an InTouchViewApp

1. Right-click the InTouchViewApp template you wish to configure. The InTouchViewApp context menu appears.
2. Select the **Associate Galaxy Graphics** menu item. The **Associate Galaxy Graphics** dialog box appears.
  - a. When the **Associate Galaxy Graphics** dialog appears, the \$InTouchViewApp template will be checked out.
  - b. If the \$InTouchViewApp template is not yet initialized or checked out, the **Include all Galaxy graphics with this application** option will be disabled.
3. Click the check box and click **OK**. A status box displays check in progress.

If an InTouchViewApp instance is deployed when you attempt to modify the **Include all Galaxy graphics** setting, you will see an information message to undeploy all deployed instances.

# CHAPTER 9

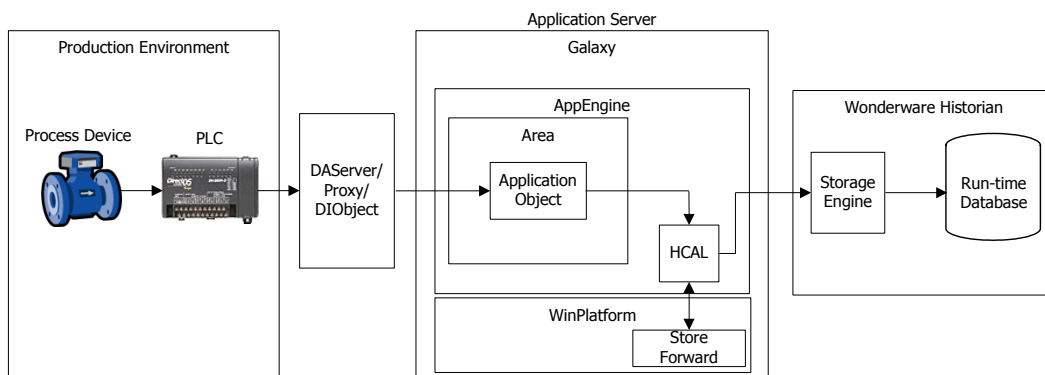
## Working with History

You can configure application objects to store process data in the Historian (formerly called IndustrialSQL Server). Historical data from Application Server can be retrieved and viewed using standard Historian database utilities. Also, you can use historical data to produce reports shown from your client applications. InTouch includes a set of ActiveX controls designed to show historical data from the Historian in trends or graphs that can be embedded in client application windows.

### Application Server History Components

To save your process data to a historical database, you must install the Historian Server. A Historian Server database can be installed on any computer outside the Galaxy, but on the same network. In a production environment, the Historian should be installed on a dedicated computer without other ArchestrA products running on it.

The following figure shows the major ArchestrA components to save process data from a production device to the Historian.



A single Historian installation can receive historical data from a single Galaxy. A push model is used to send and save new historical updates to Historian. Each system object Engine (Platform, AppEngine, ViewEngine) includes a historian feature that sends all history updates for all hosted objects to the historian. The historian feature receives the history updates from objects on the same engine only. All Engine objects include an attribute to specify the node name of the computer hosting Historian.

The figure shows a single Historian. This may be a common configuration, but other Application Server configurations support multiple Historian databases for a Galaxy. However, each Engine object only sends its historical data to one Historian.

There is a one-to-one relationship between a historical object attribute and a tag in Historian.

### Sending Historical Data Between Application Server and the Historian

Application Server communicates with the Historian through an interface called the Historian Client Access Layer (HCAL).

HCAL can establish and maintain a connection to one or more historians either synchronously or asynchronously. If a disconnection from the historian occurs, HCAL attempts to restore the connection.

If HCAL cannot communicate with the historian, all data currently being processed can be stored locally on the computer running HCAL. This hard drive location is called the store-and-forward path. Historical data is stored until the threshold capacity of the path is reached or communication to the historian is restored. In the event that all store forward disk capacity is used to store historical data, no more data is stored. An error message is logged. Remote store-and-forward paths are not supported.

HCAL can go into store-and-forward mode even if it has never been connected to the historian. After the tags are configured successfully on the historian, current data will start to be sent, along with the locally-stored data.

You can specify how disconnects between Application Server and the Historian should be reflected in the data until the disconnect period can be backfilled with store-and-forward data.

- If you enable the **Reconnect as soon as possible & do not mark disconnects** option for an AppEngine, client-side trends during the disconnect interval are filled in with the last-received value before the disconnect, and trends do not show a gap.
- If you disable the option, a gap appears in the client-side trends during the disconnect interval. NULL values are injected during the disconnect interval to create the gap.

For more information about HCAL, see the Historian documentation. For more information about the **Reconnect as soon as possible & do not mark disconnects** AppEngine option, see *AppEngine Reconnect Configuration for Undeploying and Redeploying Objects Linked to the Telemetry Server* on page 448.

## Saving Object Attribute Data to the Historian

Some attribute values can be saved as historical data when values change. Data quality and time are also associated with attribute values. When available, Application Server uses the attribute's value, the timestamp when the value changed, and the calculated quality of the data to create a Value, Time, Quality (VTQ) packet sent to the Historian. If there is no timestamp associated with the attribute value, the current scan time from the AppEngine hosting the object is used instead.

## Saving Process Values as Historical Data

For attribute data to be stored in the historian, a host AppEngine must be configured to send history data to a Historian node. For each object, you can configure attributes of the following data types to be saved to the Historian.

- Float (numerical).
- Double (numerical) maps to a Historian float data type. If the value of the double exceeds the range of a float, its value is clamped to the maximum value for the float and the quality is set to Uncertain.
- Integer (numerical)
- Boolean (non-numerical)
- String – Unicode (non-numerical). Limited to 512 characters. Characters that exceed the 512 maximum length are truncated. If truncation occurs, the quality value of the packet is set to Uncertain.
- CustomEnum (non-numerical) maps to a Historian integer.
- ElapsedTime (numerical) maps to a Historian float and is converted to seconds.
- Arrays or parts of arrays are not supported.
- Enum type attributes are saved as Historian integer ordinal values.
- IEEE NaN values for float and double data types are converted to null values prior to being sent to the historian. NaN values are associated with a Bad OPC data quality.



- All numerical attributes sent to the Historian are in the engineering units specified for the attribute. The Historian does not scale numerical values.

## Saving Data Quality as Historical Data

The Historian supports the OPC Quality definition. A 16-bit value for OPC data quality is sent to the Historian in the VTQ packet. Within the 16-bits, the low-order byte is the standard OPC portion. The high order byte is reserved and is currently unused. The Good, Bad, Initializing (which is a form of Bad) and Uncertain quality states are specified in the low-order byte.

Any change in the mapped Historian Quality Detail saves a new record to the Historian, regardless of whether the attribute value has changed. If an attribute value remains constant over a period of time with varying changes in quality, multiple records are stored in the Historian. The OPC quality stored in the Historian is the actual quality of the attribute in Application Server without modification.

The Historian can create and insert additional (non-ArchestrA generated) VTQ records that modify quality in the database. These additional VTQ records provide additional information about network outages or other events to client applications that use historical data from the Historian.

## Additional Quality Data Saved to the Historian

The Historian also has two additional data quality fields (quality and quality detail) that are non-OPC compliant. Both fields are stored with each record in addition to the OPC quality. The Historian maps the value of these fields in a manner consistent with its quality definition for those fields. The values of those fields are driven by the OPC quality sent by Application Server. For example, an OPC Good quality results in the Historian quality detail of Good.

Historian Quality is a 1-byte flag that summarizes the quality of the associated data value in the packet sent by Application Server. Historian Quality is an enumerated type with the following values:

Hex	Decimal	Name	Description
0x00	0	Good	Good value.
0x01	1	Bad	Value was marked as invalid.
0x10	16	Doubtful	Value is uncertain. Only fabricated at retrieval time.
0x85	133	Initial Value	(Good) Initial value for a delta request.

OPC Qualities from Application Server result in storage of a Historian Quality as follows:

OPC Quality (substatus form)	Historian Quality Decimal (1-byte)
Good (any)	0 - Good
Bad (any)	1 - Bad
Bad (Initializing form)	1 - Bad
Uncertain (any)	0 - Good

Historian QualityDetail contains the detail of data quality as a 32-bit value. QualityDetail is derived from OPC quality. Each source is allocated a byte position within the QualityDetail 32-bit word, which is formatted as:

0xXXRRDDDD

Only the low-order two bytes (DDDD) are mapped to OPC quality for data sent by Application Server and saved to the historian:

<b>OPC Quality (any substatus form)</b>	<b>Results in any one of these Historian Quality Details</b>
Good (any)	192 - Good value 150 - Initial value (good) 151 - Initial store/forward value (good) 252 - First value received (good) 44 - First good value received by storage after the connection to HCAL has been restored. 248 - First value in the second stream during a store-and-forward.
Bad (any)	24 (DAServer disconnect)
Bad (Initializing form)	24 (DAServer disconnect)
Uncertain (any)	192

## Saving the Data Timestamp as Historical Data

Application Server includes a timestamp in the VTQ packet sent to the historian for each attribute value/quality update that is saved as a historical record. Application Server propagates the timestamp associated with an attribute value. The timestamp is propagated throughout all Application Server components as UTC in FILETIME format.

If an object attribute has a Value DeadBand specified, a quality change or value change from previous scan cycle greater than Value DeadBand causes the attribute's Value, Time and Quality (VTQ) to be saved as historical data in the Historian. When a Value DeadBand is not specified, any change to the attribute's value, timestamp, or quality cause the attribute to be saved as historical data.

When an attribute is configured to periodically save a historical record, the attribute's current time is used as the timestamp. If the attribute does not support a timestamp, the hosting AppEngine's current scan time is used as timestamp included in the packet.

If slow updates for real-time data values are received from an I/O source and an intermittent network disconnect occurs, the timestamp of the first data value of a historized attribute uses the AppEngine's timestamp instead of the source's timestamp. In the following example, in row number 4, the value "30" is sent again to the historian, but the timestamp is the AppEngine timestamp and not the actual timestamp of "2008-11-24 18:50:41.061" when the value of 30 was generated from the I/O source.

<b>DateTime</b>	<b>TagName</b>	<b>Value</b>	<b>Quality</b>	<b>QualityDetail</b>	<b>OPCQuality</b>
2008-11-24 18:48:30.081	Realtime_ Client.ival	20	133	44	192
2008-11-24 18:50:41.061	Realtime_ Client.ival	30	0	192	192
2008-11-24 18:51:38.666	Realtime_ Client.ival	NULL	1	24	0
2008-11-24 18:52:06.670	Realtime_ Client.ival	30	0	192	192 → Row number 4
2008-11-24 18:53:22.988	Realtime_ Client.ival	40	0	192	192

If slow updates for late data values are received from an I/O source and an intermittent network disconnect occurs, the timestamp of the first data value of a historized attribute is modified to maintain the time sequence. In the following example, in row number 4, the value "30" is sent again to the historian, but the timestamp is modified by adding 5 milliseconds to previous timestamp of the NULL data value. Note that a QualityDetail of 704 is stored for the data value in row number 4.

<b>DateTime</b>	<b>TagName</b>	<b>Value</b>	<b>Quality</b>	<b>QualityDetail</b>	<b>OPCQuality</b>
2008-11-24 18:48:30.081	Latedata_ Client.ival	20	133	44	192
2008-11-24 18:50:53.624	Latedata_ Client.ival	30	0	192	192
2008-11-24 18:50:53.625	Latedata_ Client.ival	NULL	1	24	0
2008-11-24 18:50:563.630	Latedata_ Client.ival	30	0	704	192 → Row number 4
2008-11-24 18:53:26.863	Latedata_ Client.ival	40	0	192	192

## Saving Alarms and Events as Historical Data

Alarms and events detected by the Application Server at run time are saved as historical data to the Historian for the host engine. Alarms are a type of event that can be historized. In addition, alarm -related events such as Acknowledge are also saved as historical alarm data.

You can save alarm counts by severity level as historical data. For more information about mapping alarm severities to priority ranges and saving alarm counts aggregated by severity level to the Historian, see Mapping Alarm Severity to Priority.

## Deploying and Undeploying Attributes

When an object is deployed with historical attributes, the history feature causes dynamic reconfiguration of the Historian. Each history feature causes a new tag to be created and configured automatically in the Historian at deployment time. The Historian storage system that will be used is determined by the configuration of the host engine object.

This dynamic configuration causes appropriate row/column configuration in the Historian schema.

If the connection to the Historian is down at deploy time, the attempt to dynamically reconfigure Historian is achieved when the connection is restored. In other words, automatic retry is built in. However, any data generated by the object during the time between deployment and the recovery is lost and cannot be recovered by store forward.

When an object configured for history is redeployed, changes to an object's attributes makes the historian reconfigure storage. For example, if the engineering units string of an object changes from "Deg F" to "Deg C" when the object is redeployed, the Historian configuration database shows the change.

When an object configured for history is undeployed, all history remains in the Historian. The history data can be viewed in the future even if the object is no longer deployed.

## Saving Historical Data During Run Time

While an application is running, data is saved to the historian as follows:

- If no previous historical attribute value exists in the historian, the first value is always saved to the historian.
- Numerical attributes (double, float or integer): If the value for the attribute changes and that change is more than the value deadband, or the value's quality changes (for example, from Good to Bad), the data is saved to the historian.
- If the attribute type is qualitative (enumeration, string, or Boolean) and the attribute's value or quality changes.
- If the current time exceeds the last Forced Store period occurred for the attribute by the time period that was set as the Force Storage Period. If no last Forced Store occurred since starting the object, a new store occurs immediately.

---

**Note:** If enabled, the Value Deadband mechanism resets itself based on this new stored value.

---

- All new attributes value, timestamp, and quality are sent to the Historian for storage.
- The Historian stores the historical records to the database.

## Advanced Communication Management

To ensure that attribute data saved to the historian is always current and continuous, Application Server registers a reference to the attribute. By registering a reference to the attribute, it prevents Message Exchange from suspending historical updates during the period when the client application has minimized the window containing the object. Historical data continues to be saved to the Historian even during the period when the object is in an Advanced Communication Management Suspended state.

## Store-and-Forward Mode

If the Historian shuts down or the network connection to it is lost while an application is running, historical data continues to be stored locally on the computer hosting the WinPlatform object.

When the Historian node recovers, data is forwarded from the local node to the Historian node at a low priority.

If an AppEngine loses connectivity to the Historian node, the Historian reports bad data quality to clients. When you undeploy an object with attributes configured for history, the Historian stores the final data points with Bad quality.

## Buffered Behavior

The buffered data feature enables efficient accumulation and propagation of VTQ (Value, Time, and Quality) data updates, without folding and data loss, to data consumers such as objects, alarms, the Historian, and scripts from field devices that support buffering.

Buffered data is defined as data captured and stored locally on a remote device for later transfer to a supervisory system for processing, analysis, and long-term storage. The Buffer property is input-only.

When an object is associated with an attribute that supports buffered data (HasBuffer property is true), the object monitors and processes the Buffer property of the attribute. Each VTQ entry in the buffer that meets value deadband criteria is pushed to the Historian.

Attributes that are buffer-enabled are registered to Historian in the same manner as non-buffered attributes are registered. The Historian does not differentiate a buffered from a non-buffered registered attribute. An attribute is considered registered after a Historian key is returned from HCAL to the object.

All the buffered VTQs that are received by the parent attribute while Historian registration is pending are ignored and discarded.

For more information about buffered data, see *Working with Buffered Data* on page 439. For further reading on historizing events and attributes that are buffering-enabled, see *About Buffered Data and History* on page 447.

## Configuring Common Historical Attributes

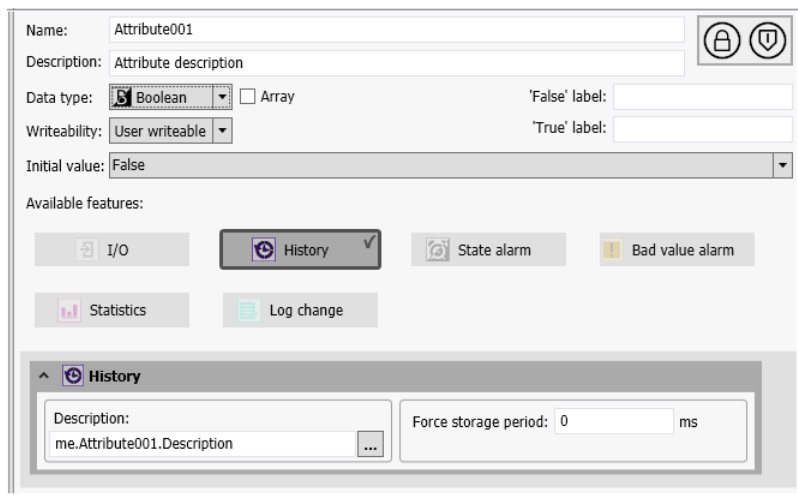
Each application or system object that you select from the Template Toolset include a set of common attributes you configure to save data to the Historian. These are configured after activating the **History** feature in the **Attributes** page.

The historical attributes you configure are based on the data type of the object. For example, the following figure shows the common historical attributes for numerical attributes (integer, float, or double).

The screenshot displays the configuration interface for an attribute named 'Attribute001'. The 'Data type' is set to 'Integer' and 'Writeability' is 'User writeable'. The 'Initial value' is 0. The 'Available features' section includes 'I/O', 'History' (checked), 'Limit alarms', 'ROC alarms', 'Deviation alarms', 'Bad value alarm', 'Statistics', and 'Log change'. The expanded 'History' section contains the following settings:

- Description: me.Attribute001.Description
- Force storage period: 0 ms
- Value deadband: 0.0 EU
- Trend high: 10.0 EU
- Trend low: 0.0 EU
- Interpolation type: Linear
- Rollover value: 0.0
- Enable swinging door:
- Rate deadband: 0.0 %

If you are configuring a boolean or string attribute, you see a smaller set of historical attributes. Similarly, a discrete field attribute will also have this same, smaller set of historical attributes.



This section describes the common historical attributes that you can configure for your system and application objects. Refer to this section as you complete the procedures to configure your objects.

- **Force Storage Period**

Interval in milliseconds in which an attribute value is saved to the Historian, regardless of whether the value exceeds its value deadband setting or not. An attribute value is saved to the Historian at every Force Storage interval. The default value of zero (0) disables the Force Storage period.

- **Value Deadband**

Threshold value in engineering units, which is the absolute difference between the current and most recent saved historical values. The current value must exceed the absolute deadband to be saved as historical data.

The Value Deadband filters out small, momentary value changes from being saved to the Historian. A new value that is within the range of the deadband is not saved to the Historian. The default value of 0.0 disables the Value Deadband and any change to a value is saved as historical data.

- **Trend Hi**

Initial maximum trend value in engineering units for clients. The default is 100.

- **Trend Lo**

Initial minimum trend value in engineering units for clients. The default is 0.0.

- **Enable Swinging Door**

A flag that indicates whether the swinging door rate deadband is enabled or disabled. The default is disabled.

Enable Swinging Door is disabled if the historical attribute data type is Boolean or a string.

- **Interpolation Type**

List of methods used by the Historian to interpolate analog historical data. The interpolation type determines which analog value is selected during a Historian data retrieval cycle. Interpolation types include System Default, Stairstep, or Linear. The default interpolation type is System Default.

- **Stairstep**

No interpolation occurs. The last known value is returned with the given cycle time. If no valid value can be found, a NULL is returned to the Historian.

- Linear
 

The Historian calculates a new value at the given cycle time by interpolating between the last known value prior to the cycle time and the first value after the cycle time.
- System Default
 

The Historian system-wide interpolation setting. The system-wide setting must be either stairstep or linear interpolated.
- **Rollover Value**

Positive integer value that represents a tag's reset limit when the Historian operates in counter retrieval mode. In counter retrieval mode the Historian uses a tag's rollover value to calculate and return the delta change between consecutive retrieval cycles. The default value is 0.0.

The Rollover value applies only to numeric attributes. The Rollover value is disabled if the historical attribute data type is Boolean or a string.
- **Rate Deadband**

Percentage rate of change deadband based on the change in the slope of incoming data values to the Historian. For example, specifying a swinging door rate deadband of 10 percent means that data is saved to the Historian if the percentage change in slope of consecutive data values exceeds 10 percent.

The default is 0.0, which indicates a swinging door rate deadband is not applied. Any percentage greater than 0.0 can be assigned to the rate deadband.

## Configuring System Objects to Store Historical Data

The following table shows the historical attributes for Application Server system objects.

History Attributes	Application Server System Objects				
	WinPlatform	AppEngine	Area	ViewEngine	InTouchViewApp
Description	◆		◆	◆	
Enable compression	◆	◆		◆	
Reconnect as soon as possible & do not mark disconnects		◆			
Enable Storage to Historian	◆	◆		◆	
Enable Swinging Door	◆	◆	◆	◆	
Enable Tag Hierarchy	◆	◆		◆	
Engineering units	◆	◆	◆	◆	

History Attributes	Application Server System Objects				
	WinPlatform	AppEngine	Area	ViewEngine	InTouch/ViewApp
Force Storage Period	◆	◆	◆	◆	
Historian	◆	◆		◆	
History store forward directory	◆	◆		◆	
Interpolation Type	◆	◆	◆	◆	
Pre-processing buffer size	◆	◆		◆	
Rate Deadband	◆	◆	◆	◆	
Rollover Value	◆	◆	◆	◆	
Store forward threshold	◆	◆		◆	
Store forward minimum duration	◆	◆		◆	
TCP Port	◆	◆		◆	
Throttling network bandwidth	◆	◆		◆	
Trend High	◆	◆	◆	◆	
Trend Low	◆	◆	◆	◆	
Value Deadband	◆	◆	◆	◆	
Wait to send incomplete packets	◆	◆		◆	

## Configuring the WinPlatform Object to Store Historical Data

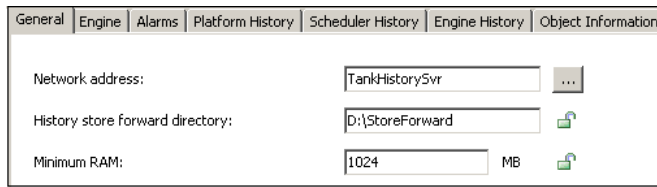
A WinPlatform object contains attributes to configure how historical store-and-forward data is cached locally and then pushed to the Historian. Also, you can select the WinPlatform object’s platform, scheduler, and engine data to be saved to the Historian. Finally, you can select the WinPlatform’s attribute values to be saved to the Historian.

### To configure a WinPlatform Object to store historical data

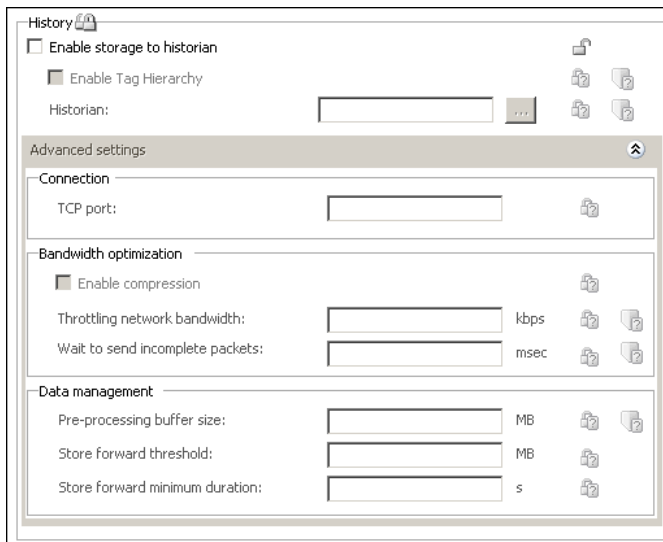
1. Double-click on an WinPlatform derived template or instance to open the Object Editor.



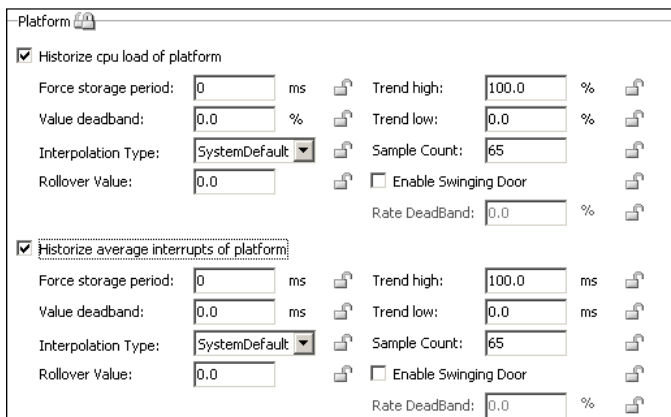
2. Click the **General** tab to show history attributes.



3. In the **Network address** box, type or select the node name of the computer assigned to the WinPlatform object.
4. In the **History store forward directory** box, enter the path to the folder to save store and forward historical data. The folder must exist on the computer specified in the **Network address** box.
5. Click the **Engine** tab to show history attributes.



6. In the **History** area of the **Engine** page, configure the attributes. For more information on each attribute, see the WinPlatform object Help.
7. Click the **Platform History** tab to show the attributes to save system data from the computer hosting the WinPlatform object to the historian.



8. Select the check box next to each attribute whose data you want saved in the historian. For more information about these historical attributes, see *Configuring Common Historical Attributes* on page 349.
9. Click the **Scheduler History** tab to show the attributes to save data to the historian.

10. Select the check box next to each scheduler attribute whose data you want saved in the historian.
11. Click the **Engine History** tab to show the attributes to save data from AppEngines hosted by the WinPlatform object.
12. Select the check box next to each engine attribute whose data you want saved in the historian.
13. Save your changes and check in the WinPlatform object.
14. Deploy the object to its target computer in an on scan state.

## Configuring an AppEngine Object to Store Historical Data

If an AppEngine is deployed before Historian is started, history data can be stored locally by HCAL until the objects successfully register with the Historian.

---

**Note:** Except for Late Data, the ViewEngine object contains the same historical attributes as the AppEngine object.

---

### To configure an AppEngine object to store historical data

1. Double-click on the AppEngine instance to open it within the Object Editor.
2. Click on the **General** tab to show history attributes.

The screenshot shows the 'General' tab of the Object Editor with the following configuration options:

- Historize active alarm counter:**
  - Force storage period: [ ] ms
  - Value deadband: [ ] EU
  - Interpolation Type: [ ]
  - Rollover Value: [ ]
  - Trend High: [ ] EU
  - Trend Low: [ ] EU
  - Rate DeadBand: [ ] %
  - Enable Swinging Door:
- Historize unacknowledged alarm counter:**
  - Force storage period: [ ] ms
  - Value deadband: [ ] EU
  - Interpolation Type: [ ]
  - Rollover Value: [ ]
  - Trend High: [ ] EU
  - Trend Low: [ ] EU
  - Rate DeadBand: [ ] %
  - Enable Swinging Door:
- Historize disabled (or silenced) alarm counter:**
  - Force storage period: [ ] ms
  - Value deadband: [ ] EU
  - Interpolation Type: [ ]
  - Rollover Value: [ ]
  - Trend High: [ ] EU
  - Trend Low: [ ] EU
  - Rate DeadBand: [ ] %
  - Enable Swinging Door:

3. In the **History** area of the **Engine** page, configure the attributes. For more information on each attribute, see the AppEngine object Help.
4. Click the **Scheduler History** tab to show the attributes to save data to the historian.
5. Select the check box next to each scheduler attribute whose data you want saved in the historian.
6. Click the **Engine History** tab to show the attributes to save data from AppEngines hosted by the WinPlatform object.
7. Select the check box next to each engine attribute whose data you want saved in the historian.
8. Save your changes to the AppEngine object.

## Configuring an Area Object to Save Alarm Counts as Historical Data

An Area object represents a plant area to group objects for modelling and report alarms. You can configure a set of attributes to save the counts of an area's alarm states to the historian. An Area object contains a set of attributes to save the counts of the following alarm states to the historian:

- Active alarms
- Unacknowledged alarms
- Disabled or silenced alarms

### To configure an Area object to store historical data

1. Double-click on the Area instance to open it within the Object Editor.
2. Click on the **General** tab to show attributes for area alarm counts that can be saved as historical data.

The screenshot shows the 'General' tab of the Object Editor. It contains three sections, each for a different alarm state:

- Historize active alarm counter:** Includes checkboxes for 'Historize active alarm counter', 'Trend High', and 'Trend Low'. Attributes include Force storage period (ms), Value deadband (EU), Interpolation Type (dropdown), Rollover Value, Enable Swinging Door (checkbox), and Rate DeadBand (%).
- Historize unacknowledged alarm counter:** Includes checkboxes for 'Historize unacknowledged alarm counter', 'Trend High', and 'Trend Low'. Attributes include Force storage period (ms), Value deadband (EU), Interpolation Type (dropdown), Rollover Value, Enable Swinging Door (checkbox), and Rate DeadBand (%).
- Historize disabled (or silenced) alarm counter:** Includes checkboxes for 'Historize disabled (or silenced) alarm counter', 'Trend High', and 'Trend Low'. Attributes include Force storage period (ms), Value deadband (EU), Interpolation Type (dropdown), Rollover Value, Enable Swinging Door (checkbox), and Rate DeadBand (%).

3. Select the check box next to each alarm state counter that you want to save to the historian.
4. Set the attributes for each alarm counter that you select. For more information about assigning values to historical attributes, see *Configuring Common Historical Attributes* on page 349.
5. Save your changes and close the Object Editor.

## Configuring Application Objects to Save Historical Data

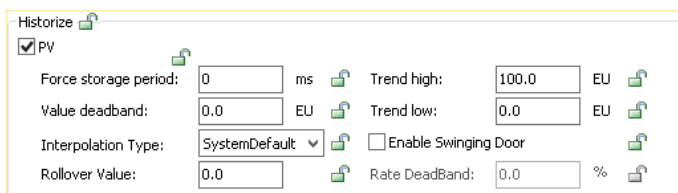
The following table shows the historical attributes for application objects. These history attributes are described in *Configuring Common Historical Attributes* on page 349.

History Attributes	Application Server Application Objects										
	AnalogDevice	Boolean	DiscreteDevice	Double	FieldReference	Float	Integer	Sequencer	String	Switch	UserDefined
Enable Swinging Door	◆	◆		◆	◆	◆	◆				
Force Storage Period	◆	◆	◆	◆	◆	◆	◆		◆	◆	
Interpolation Type	◆	◆		◆	◆	◆	◆				
Rate Deadband	◆	◆		◆	◆	◆	◆				
Rollover Value	◆	◆		◆	◆	◆	◆				
Trend High	◆	◆		◆	◆	◆	◆				
Trend Low	◆	◆		◆	◆	◆	◆				
Value Deadband	◆	◆		◆	◆	◆	◆				

**Note:** The historical attributes available from the UserDefined object are based on the data type associated with the selected attribute.

### To configure an application object to store history

- Open the application object in the Object Editor.
  - If you selected a UserDefined object, click the **Attributes** tab.
  - If you selected an AnalogDevice object, click the **History** tab. Then, click the General tab to show the history attributes for the object’s PV value.



- Enable historization.
  - For UserDefined objects, create an attribute and then activate the **History** feature. Assign values to the history parameters. For more information about assigning values to the history feature, see *Configuring Common Historical Attributes* on page 349.
  - For AnalogDevice objects, select the **Historize** feature to enable the common history attributes shown in the **History** area of the page.

The History page includes check boxes to save **PV**, **SP**, **PV mode**, **Control mode**, and **Control Track Flag** data to the historian.

Assign values to the PV history attributes that appear in the **History** area of the page. For more information about assigning values to the common history attributes, see *Configuring Common Historical Attributes* on page 349.

- Save your changes and close the Object Editor.





# CHAPTER 10

## Working with Alarms and Events

You can create ArcestrA applications that generate alarms and events to provide the status of a running application.

ArcestrA objects include built-in event and alarming reporting capabilities. You must configure alarms for each object in the IDE to use the event and alarm functions.

### Understanding Events

Events represent normal system, application, or user occurrences that produce status messages. A typical event occurs when an operator logs on to an application at the beginning of a work shift. Application Server can detect events, store them as historical data, and report them to client applications.

An event indicates a significant occurrence that is detected, reported, and saved as historical data. Events can be detected by any Application Server component including automation objects and the SMC.

Events provide a means for any software component to log information about a system, application, or operator action. Application Server components use an event API to send event messages to Alarm/Event distributors and the Historian. Also, events are sent to client applications like AVEVA OMI to be shown in real-time trends or reports.

### Types of Events

Application Server creates several types of run-time event messages, which are saved as historical data.

- System events

Event messages are logged when a Platform, Engine, Area, DI Network or DI Device start or stop. Any system action that affects a large number of objects is logged as an event.

A System event message contains the following information:

- Event type, which is System.
- Tagname, which is the name of the object generating the event.
- Tag description, which is a short description of the object generating the event.
- Area, which is the name of the area that contains the object generating the event.
- Event description, which describes the system action and can be either Started or Stopped.
- Timestamp, which is the current system time.

- Security-audit (operator change) events

Event messages are logged to the Alarms and Events subsystem when an operator logs on to or logs off from an application. Also, security-audit events are logged when an operator changes actions by means of User sets.

A Security-audit event message contains the following information:

- Event type, which is operator change.
- Timestamp, which is the date and time when the operator change event occurred. The timestamp of the event is the current AppEngine scan time.

- Tagname, which is the name of object generating the event.
- Prim.attr, which is the reference string of the attribute being changed.
- Tag description, which is a short description of the object. For Secured and Verified writes, this will contain the following:
  - Type of write (Secured or Verified)
  - Comment from the user, if any
  - Reason description, if any
- FieldAttribute description, if the attribute is a FieldAttribute and has a description; otherwise, this is the Object description.
- Area, which is the name of the area that contains the object generating the event.
- UserEngine, which is the name of the view engine or other user engine requesting the operator change.
- Operator1, which is the full name of the primary operator requesting a change. The full name is an attribute of the UserProfile.
- Operator2, which is the full name of the secondary operator validating the change, if any.
- Old value, which is the previous value of an attribute.
- New Value, which is the new value of an attribute.
- Application (or process) related events

Application event messages are generated by application objects in response to process actions. For example, application event messages are created when a process pump starts or stops.

An Application event contains the following information:

- Event type, which is data change.
- Timestamp, which is the date and time when the application event occurred. The timestamp assigned to the event is the timestamp of the attribute associated with the event, if available. Otherwise, the event timestamp is the current AppEngine scan time.
- Tagname, which is the name of the object generating the event.
- Prim.attr, which is the reference string of the attribute being changed.
- Tag description, which is a short description of the object.
- Area, which is the name of the area that contains the object generating the event.
- Old value, which is the previous value of an attribute.
- New Value, which is the new value of an attribute.

## Understanding Alarms

Alarms warn about process conditions that can potentially cause problems. Typically, you set up an alarm to become active when a process value exceeds a defined limit. For example, you can set an alarm for a pump that warns when no fluid pressure is detected.

An alarm is an abnormal condition that requires immediate attention. An operator usually acknowledges an alarm. ArchedrA handles the real-time reporting of alarms and provides special clients for viewing them.

Application and system objects can detect and generate an alarm. To detect an alarm, a system or application object sets a Boolean Attribute flag to indicate whether the object's alarm condition is currently true or false.



To report an alarm, the object must contain an alarm feature. The alarm feature makes a reference to the object's Boolean flag to determine whether the alarm condition is true. It then combines this information with the current alarm mode to determine whether to report a this as an active or inactive alarm state. An alarm feature is dedicated to reporting a single alarm condition's state. The alarm feature send alarm notification messages to ArchestrA alarm and event distributors.

Every alarm notification includes a set of fields containing data that describes the alarm. Some alarm notification data is saved as historical data. The following list describes all fields sent with an alarm notification.

- **TagName**, which is the name of the object generating the alarm. Saved as historical data.
- **Name**, which is the name of the alarm. Saved as historical data.
- **InAlarm**, which is a Boolean value that indicates whether the object's alarm state is currently active or inactive. Saved as historical data.
- **Quality**, which is the current quality of the data upon which the alarm is based. Saved as historical data.
- **OnTimeStamp**, which is the time when the attribute value transitioned into an alarm state. The attribute's value timestamp is used, if available. Otherwise, the timestamp is the AppEngine scan time. Saved as historical data.
- **OffTimeStamp**, which is the time when the alarmed attribute value returns to normal. The attribute's value timestamp is used, if available. Otherwise, the OffTimeStamp is the AppEngine scan time. If an active alarm is disabled, it forces a return to normal and the timestamp is the current time. Saved as historical data.
- **Category**, which is an integer between 1 and 14 that identifies the type of alarm and source of the alarm. These values are associated with Internationalized category labels.

Alarm category labels can be localized to other languages. Application Server uses the default Galaxy language to retrieve these strings and send them to InTouch. The alarm category labels appear in InTouch and InTouch history as the default Galaxy language strings. Saved as historical data.

- **Priority**, which is an integer value from 1 to 999 indicating the severity of the alarm. An alarm priority of 1 is most urgent and 999 least urgent.
- **TargetValueReference**, which is an optional field that makes a reference to the target value of the alarm. Not saved as historical data.
- **ActualValueReference**, which is an optional field that makes a reference to the actual attribute value for the alarm condition. Not saved as historical data.
- **TargetValue Snapshot**, which is an optional field containing the attribute's target value at the time when the alarm became active. Saved as historical data.
- **ActualValueSnapshot**, which is an optional field containing the attribute's actual value at the time when the alarm became active. Saved as historical data.
- **EngUnitsReference**, which is the reference to the engineering units string for the condition. Saved as historical data.
- **AcknowledgedFlag**, which indicates whether the alarm is acknowledged or not. If this flag is FALSE, the alarm is still unacknowledged. Saved as historical event data.
- **AcknowledgeTime**, which indicates the time when the alarm was acknowledged if the AcknowledgedFlag is TRUE. Saved as historical data at the time of acknowledgement.
- **AcknowledgeUserId**, which is the string containing the name of the user who acknowledged the alarm. Saved as historical data at the time of acknowledgement.

- AlarmMode, which indicates whether the alarm mode is Enabled, Silenced or Disabled. Saved as historical data at the time when the alarm mode changes.
- Message text describing the alarm, which can be statically or dynamically constructed. The message typically contains the alarm description, the exceeded limit value, and possibly the variable value. For an alarm feature, this message is retrieved by means of a reference to a string/international string attribute in the object. The reference is setup at ObjectDesigner time for alarms. If none is specified, then the common feature short description attribute is utilized. This field is also provided in the alarm feature section and can be dynamically generated and scripted. Saved as historical data.
- Area, which is the name of the area that contains the object generating the alarm. Saved as historical data.

## Types of Alarms

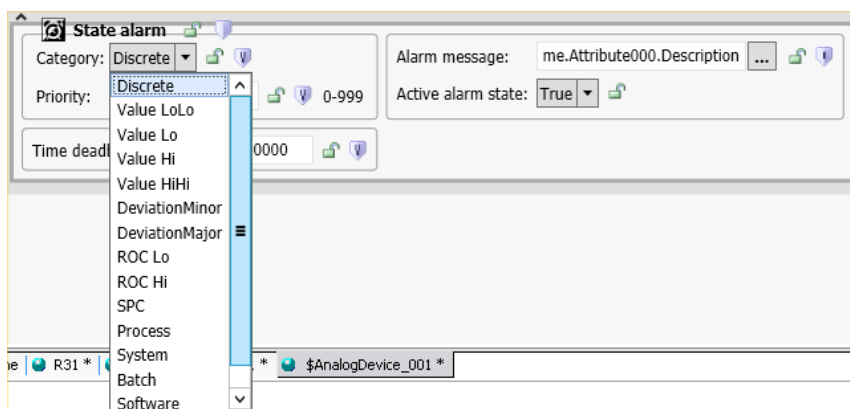
Application Server supports the following types of alarms:

- State alarms, which are also known as Boolean alarms
- Limit alarms
- Target deviation alarms
- Rate of change alarms

The type of alarm that you can configure is based on the data type of the attribute's value.

## State Alarms

A state alarm set on an attribute of Boolean data type corresponds to a discrete tag with two possible states. When you create a state alarm, you configure whether the active alarm state corresponds to the TRUE or FALSE state of the attribute.



You can set an alarm category. Valid categories are:

Category	Description
Discrete	A discrete value event or alarm, such as a change of state.
Value LoLo	A continuous value is significantly below the acceptable range.
Value Lo	A continuous value is below or is approaching the low acceptable range.
Value Hi	A continuous value is above or approaching above the high acceptable range.
Value HiHi	A continuous value is significantly above the acceptable range.
DeviationMinor	A value has a minor deviation (plus or minus) from the target or setpoint.

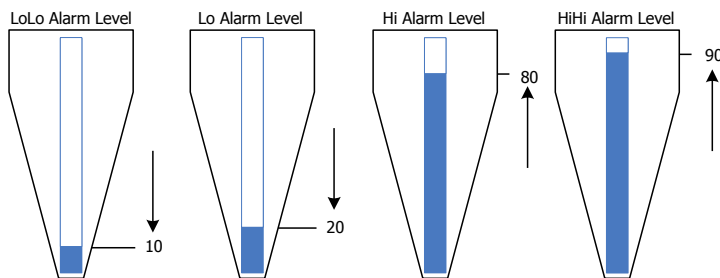
DeviationMajor	A value has a major deviation (plus or minus) from the target or setpoint.
ROC Lo	The rate of change for a value is too slow.
ROC Hi	The rate of change for a value is too fast.
SPC	A value deviates from the SPC target/range.
Process	An alarm or event associated with the physical process/plant has occurred.
System	An alarm or event associated with the automation system has occurred.
Batch	An alarm or event associated with a batch process has occurred.
Software	An alarm or event associated with a software operation or logic (such as divide by zero in script) has occurred.

You can set an alarm message and Priority for a state alarm. The time deadband sets the length of time that an attribute value must continuously remain in an alarm or unalarmed state. The time deadband filters out rapid, transitory value spikes.

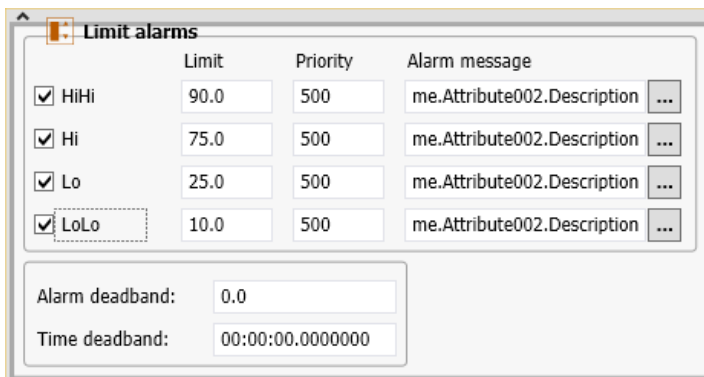
The timestamp when a state alarm becomes active or inactive is the most current timestamp of the corresponding input value. If there is no timestamp associated with the alarmed value, the AppEngine timestamp is used instead.

### Limit Alarms

A limit alarm compares the current value to one or more predetermined alarm limits within the attribute's full range of values. If the value exceeds a limit, an alarm occurs.



You can individually select and configure values and priorities for the LoLo, Lo, Hi, and HiHi alarm limits. You can set individual messages for each alarm limit.



You can also configure alarm and time deadbands for limit alarms. The alarm deadband is expressed as a percentage of the attribute's full value range. The deadband value sets the percentage of the total range that the attribute value must change to reset a limit alarm to the inactive state.

For example:

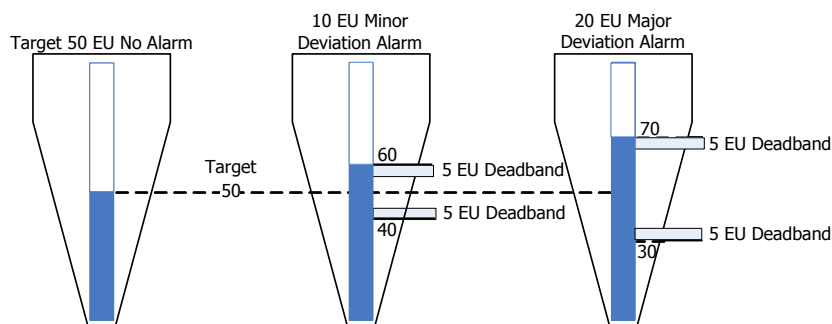
- An attribute's full value range is 0 to 50, and the HiHi alarm limit is 40.
- The alarm deadband is set to 5. Since the attribute's range is 50, the actual alarm deadband value is 2.5 (50 \* 5%).
- The attribute value reaches the HiHi alarm limit (40).
- To reset the HiHi alarm to an inactive state, the attribute value must fall below 37.5 (40 - 2.5).

The time deadband sets the length of time that an attribute value must continuously remain in an alarm or unalarmed condition. The process variable must remain above or below the indicated limit for at least the indicated deadband time before the application object updates the status of the alarm CONDITION Boolean. Then, standard Alarm feature logic determines whether to take that updated alarm condition and report changes to the alarm state or not.

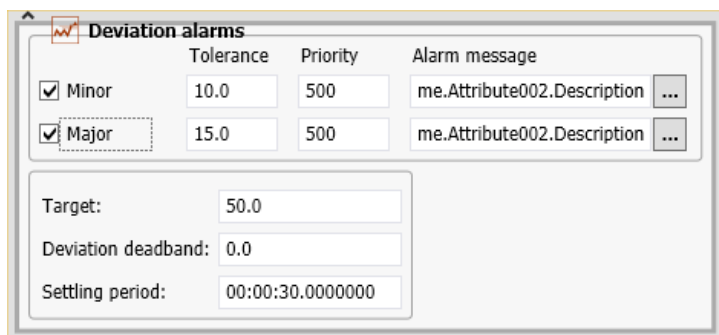
The timestamp when a limit alarm becomes active or inactive is the most current timestamp of the corresponding input value. If there is no timestamp associated with the alarmed value, the AppEngine timestamp is used instead.

### Target Deviation Alarms

A target deviation alarm compares the current attribute value to a target Engineering Units value. Then, the absolute value of the difference is compared to one or more alarm deviation limits expressed in EngineeringUnits.



You can individually select and configure values and priorities for the minor deviation limit and the major deviation limit. You can set individual messages for the minor and major deviation alarm limits.



The deviation alarm's settling period is the time allowed for the attribute value to reach an expected target value after a device starts. No alarm can occur during the settling period.

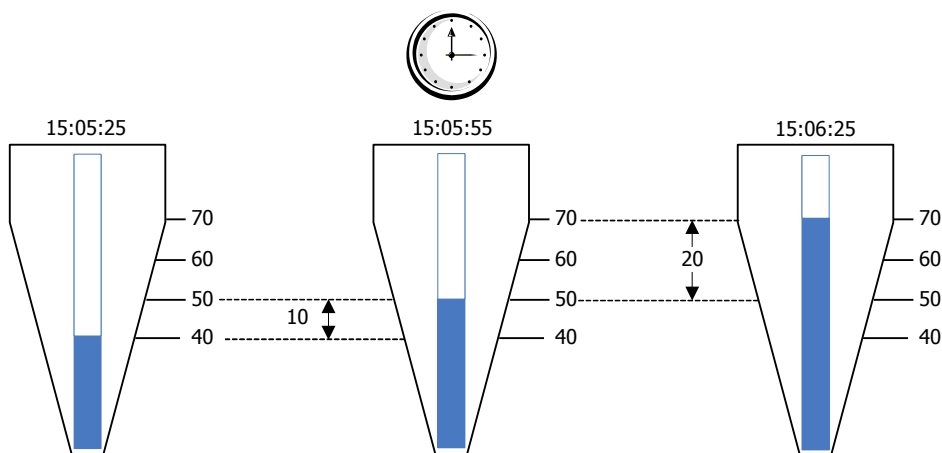
You can also configure a value for a deviation deadband, which is expressed in Engineering Units. The deadband range sets a threshold that an attribute value must change from a deviation limit to reset the alarm to the inactive state.

The timestamp when a deviation alarm becomes active or inactive is the most current timestamp of the corresponding input value. If there is no timestamp associated with the alarmed value, the AppEngine timestamp is used instead.

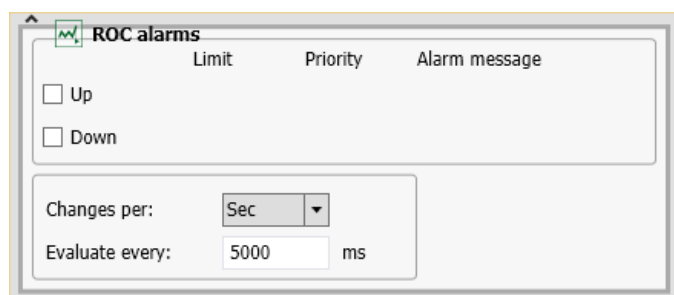
## Rate of Change Alarms

A rate of change alarm identifies when an attribute value is changing too quickly over time. For example, you can set a rate of change alarm for a tank level that indicates when the pump inlet pressure is too high.

Rate of change is the calculated slope, which is the absolute difference between the current and previous attribute values divided by a specified interval. When the slope (positive or negative) exceeds a specified value, a rate of change alarm occurs. For example, if a tank volume increases from 17 to 45 liters over a 5 minute interval, the calculated slope is 5.6 liters per minute. If you set your rate of change alarm limit to 5.0 liters per minute, a rate of change alarm condition exists.



Alarm limits are expressed in the Engineering Units of the attribute's value over an interval, which can be per second, minute, hour, or day.



You can select and configure the value and priority for the upward and downward ROC limits. You can set individual messages for ROC alarms that exceed the upward or downward limits.

The timestamp when a rate of change alarm becomes active or inactive is the most current timestamp of the corresponding input value. If there is no timestamp associated with the alarmed value, the AppEngine timestamp is used instead.

## Statistical Alarms

A statistical alarm is one in which a statistic is calculated, based upon an attribute. If the statistic exceeds some pre-set limit, the object flags the alarm condition as TRUE.

## Setting Alarm State with Object Attributes

The Application Server alarm enable/disable mechanism includes four attributes to set an object alarm mode and report alarm status:

- *AlarmModeCmd Attribute* on page 366
- *AlarmInhibit Attribute* on page 366
- *AlarmMode Attribute* on page 366
- *\_AlarmModeEnum Attribute* on page 367

### AlarmModeCmd Attribute

AlarmModeCmd is a writable attribute that sets the current commanded alarm mode for the object. You set the AlarmModeCmd to enabled, silenced, or disabled with a script, user input, or from an attribute configured to read from an input source.

### AlarmInhibit Attribute

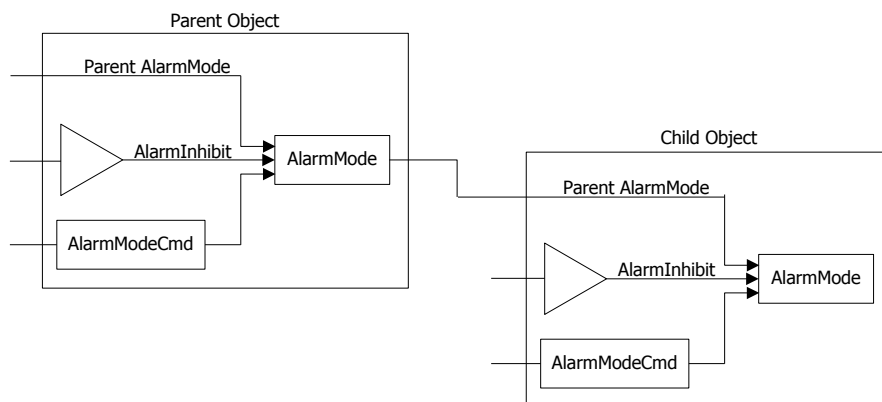
The AlarmInhibit attribute disables one or more alarms when set to TRUE. The value of the AlarmInhibit attribute is typically set by a script, manually by the user, or from an attribute configured with an input feature. If the AlarmInhibit attribute is set TRUE, all alarms of the object and of any contained objects are disabled.

When the AlarmInhibit attribute is set to FALSE, alarms are not inhibited and the object AlarmMode and parent object AlarmMode determine whether alarming is enabled, silenced, or disabled.

### AlarmMode Attribute

The AlarmMode is a calculated attribute that identifies the object alarm mode and is based upon the current values of an object's:

- AlarmModeCmd attribute
- AlarmInhibit attribute
- Parent object AlarmMode attribute



Application Server checks the AlarmModeCmd and AlarmInhibit attributes of an object and the AlarmMode status of the parent object. Application Server then updates the object's AlarmMode attribute to reflect the most restrictive setting.

All individual alarms use the object's AlarmMode status to determine whether they are enabled, silenced, or disabled.

## \_AlarmModeEnum Attribute

The `_AlarmModeEnum` attribute can be used by any object to obtain an enumeration of permissible settings for the `AlarmModeCmd` attribute.

## Alarms and Buffered Data

An object with alarms enabled detects the alarm condition and sets an associated alarm condition attribute to true. If the object detects the alarm condition based on an attribute with buffered data (`HasBuffer` property set to true), it enables buffer support for the condition attribute and sets the condition attribute's `Buffer` property to a VTQ buffer of true/false values representing alarm conditions.

### Alarm Detection

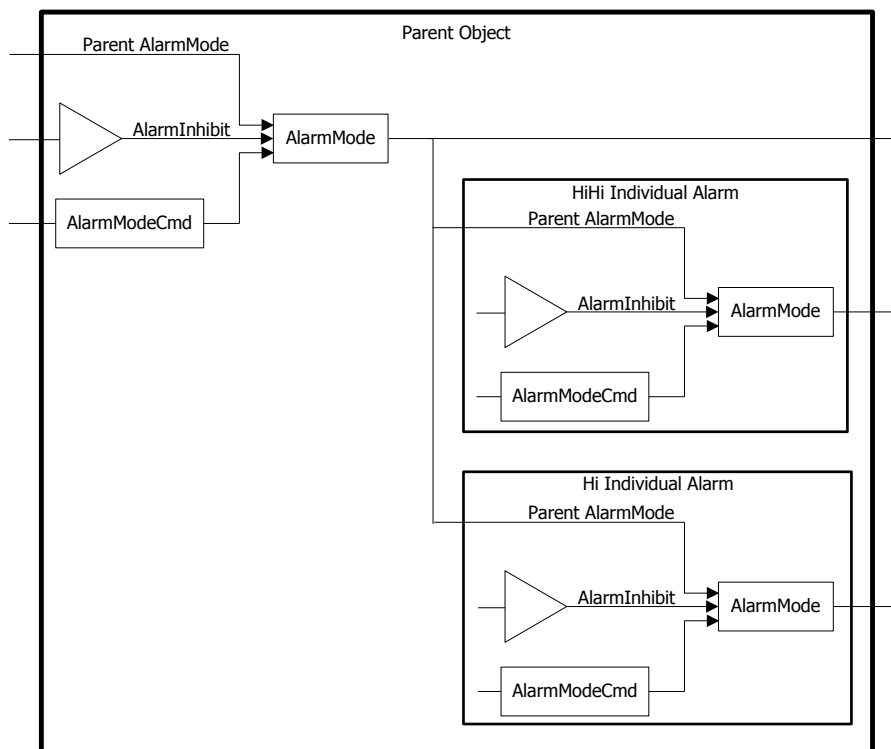
The Buffered Data feature will pass buffered alarm VTQs to the alarm subsystem in the same manner as with buffer data not enabled. When enabled, the Buffered Data feature can generate multiple alarm messages in the same scan.

For more information on alarm detection, see [About Buffered Data and Alarms and Events](#).

## Setting Alarm State for Individual Alarms

You can set individual alarms within an object for each type of alarm. For example, you can set alarms for each of the limits of a level alarm.

The following figure shows an object's individual alarms for the HiHi and Hi alarm limits.



The calculated `AlarmMode` attribute value of an individual alarm uses the same inputs as an object alarm. The parent `AlarmMode` attribute is from the object itself. As with object alarms, the individual alarm mode is set to the most restrictive input state. For example, if the object's `AlarmMode` state is disabled and the individual alarm's `AlarmInhibit` is FALSE, the individual alarm is disabled.

Each individual alarm is autonomous from other individual alarms in an object. The AlarmMode of an individual alarm is not propagated to other alarms. Unlike inhibit for the entire object, inhibit of an individual alarm does not affect the alarms of any contained objects. You can selectively enable, silence, or disable an individual alarm and not set other alarms to the same value within the object hierarchy.

## Enabling, Silencing, and Disabling Alarms

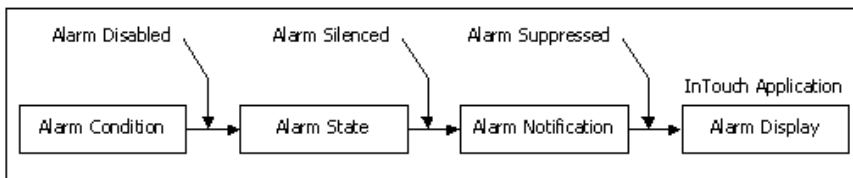
Alarms can be enabled, disabled, or silenced while an application is running. An object’s alarm state can be set at the Area level, at the container object level, or at the individual object. In addition, individual alarms within a single object can be enabled, silenced, or disabled.

- **Enabled:** All alarms for an object are reported to client applications and saved as historical data. The enabled state is less restrictive than the silenced or disabled alarm states.
- **Silenced:** All alarms for an object are detected and logged in the Historian alarm and event history database. Silenced alarms are not logged to the InTouch alarm database, and are not shown in alarm clients displaying current alarms or recent history alarms. The silenced alarm state is more restrictive than the enabled state, but less restrictive than the disabled state.
- **Disabled:** No alarms for the object are detected. The alarm is return-to-normal until the alarm is re-enabled. The disabled state is more restrictive than the silenced and enabled alarm states. A disabled alarm does not require acknowledgement.

When an alarm state changes from silenced to enabled, the following applies:

- Only the last occurrence of the alarm appears in alarm clients.
- Only the last occurrence of the alarm is logged into the alarm database.
- Only the last occurrence of the alarm is saved to history.

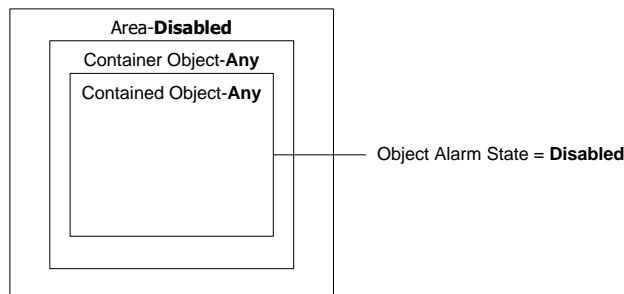
The following figure shows how the different alarm modes affect the different phases of an alarm. In the case of Alarm Suppressed, selected alarms can be filtered out of the InTouch AlarmViewer display by an operator, or programmatically by a script.



To ensure that alarmed attributes always have current data, the alarm feature always registers a reference to the alarmed attribute. This guarantees that Message Exchange never suspends updates for this attribute. Even if alarms are disabled for a particular attribute, the alarmed attribute cannot enter an Advanced Communication Management Suspended state.

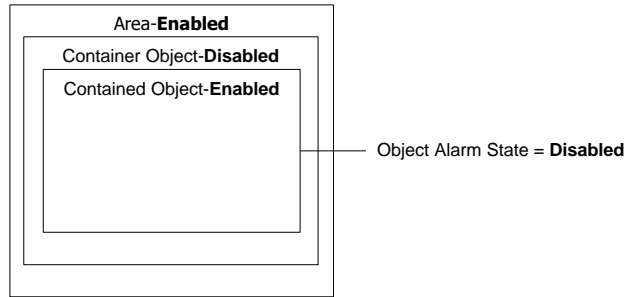
The object hierarchy and alarm states determine the final alarm condition of an object.

- An Area object’s alarm state determines the alarm state for all alarms of objects that belong to the Area.

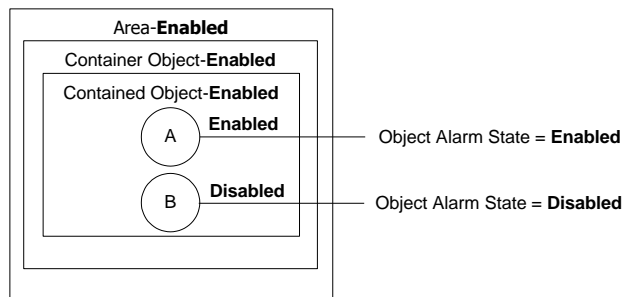




- The most restrictive setting within an object hierarchy determines the object's alarm state.



- When an individual object alarm is silenced or disabled, it applies only to that alarm and not to other alarms belonging to the object.



The alarms on any contained object are not affected. The disabled or silenced state of an individual alarm does not propagate downward through the object hierarchy to the alarms of any contained or assigned object.

## Enabling Alarms

To enable an object's alarms, you must ensure that the AlarmModeCmd and AlarmInhibit attributes are enabled for the object, its container, and its area. An event, including the user's name, is generated indicating the object's alarms are enabled.

When object alarms are enabled, you can enable, silence, or disable an individual alarm.

## Silencing Alarms

When object alarms are silenced, an individual alarm that is enabled or silenced is forced to be silenced. When object alarms are silenced, an individual alarm can be disabled.

## Disabling Alarms

When object alarms are disabled, an individual alarm that is enabled or silenced is forced to be disabled.

When object alarms are enabled and an individual alarm is enabled or silenced, the individual alarm can be inhibited. This forces the individual alarm to be disabled.

When object alarms are silenced and an individual alarm is enabled or silenced, the individual alarm can be inhibited. This forces the individual alarm to be disabled.

When object alarms are inhibited, an individual alarm that is enabled or silenced is forced to be disabled.

## Shelving Alarms

You can shelve alarms to temporarily hide them from displays for a fixed period. Alarms continue to be historized, even when they are shelved.

Shelving typically is used to reduce alarm "noise", or to temporarily suppress alarms that might be triggered during system modifications or repairs, allowing you to focus on correcting other more urgent alarms.

Shelving is similar to silencing an alarm, but shelved alarms differ from silenced alarms in the following ways:

- Shelved alarms have a built-in associated time-out period. Shelved alarms are automatically unshelved when the configured shelving period expires. You can also manually unshelve alarms and return them to an active, displayed state.
- Alarm shelving must be enabled at an area level, but shelving applies only to individual alarms. You cannot shelve a hierarchy of alarms for an entire area or for an entire object. You cannot propagate alarm shelving through the Model View hierarchy.
- The system enforces role-based limitations on permission to shelve alarms, alarm severity levels that can be shelved, and the total number of alarms a user can shelve.

The system tracks who shelved the alarm, from what workstation, the reason for shelving the alarm, when shelving began, and when shelving will expire. Shelved alarms aggregate in similar fashion to silenced alarms. For information about alarm aggregation, see *Using Aggregated Alarm Information* on page 393.

A set of seven attributes provide run-time alarm shelving information and control:

AlarmShelveCmd	User writeable. Use this attribute to shelve or unshelve an alarm. Default values: Duration = 0, Reason = ""
AlarmShelved	Read-only, Boolean value. Shows True if alarm is shelved, False if alarm is unshelved. Default value: False
AlarmShelveStartTime	A read-only date/time stamp indicating when alarm shelving began, based on the engine time when the shelving request was received, Default value: Blank
AlarmShelveStopTime	A read-only date/time stamp equal to the AlarmShelveStartTime plus the duration for which the alarm is to be shelved. Default value: Blank
AlarmShelveReason	A read-only string value providing the reason for which the alarm was shelved or un-shelved by the Alarm Shelve command. Default value: Blank (See AlarmShelveCmd attribute.)
AlarmShelveUser	Read-only, the name of the user who most recently shelved or un-shelved the alarm with the Alarm Shelve command. Default value: Blank

**AlarmShelveNode** Read-only, the name of the computer node from which the user most recently shelved or un-shelved the alarm with the Alarm Shelve command.

If the node is hosted in a Terminal Server client session, the node and the TSE ID are both identified.

Default value: Blank

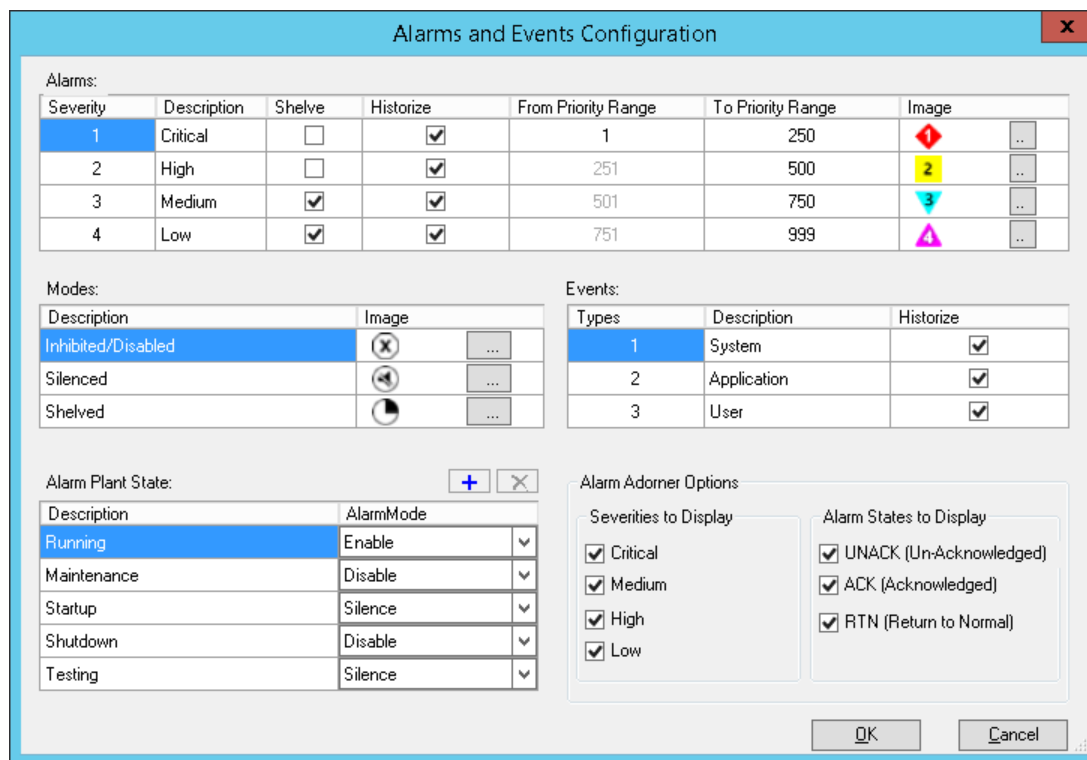
## Enabling Alarm Shelving

Alarm shelving is configured from the IDE **Configuration** menu, and is enabled on the Area object.

### To enable alarm shelving

1. Enable at least one security role to configure alarm shelving. Associate the relevant user with that role, if not already associated, before proceeding to step 2.
2. On the IDE main menu, click **Galaxy**, then click **Configure**, then click **Alarms and Events Configuration**.

The **Alarms and Events Configuration** dialog appears.



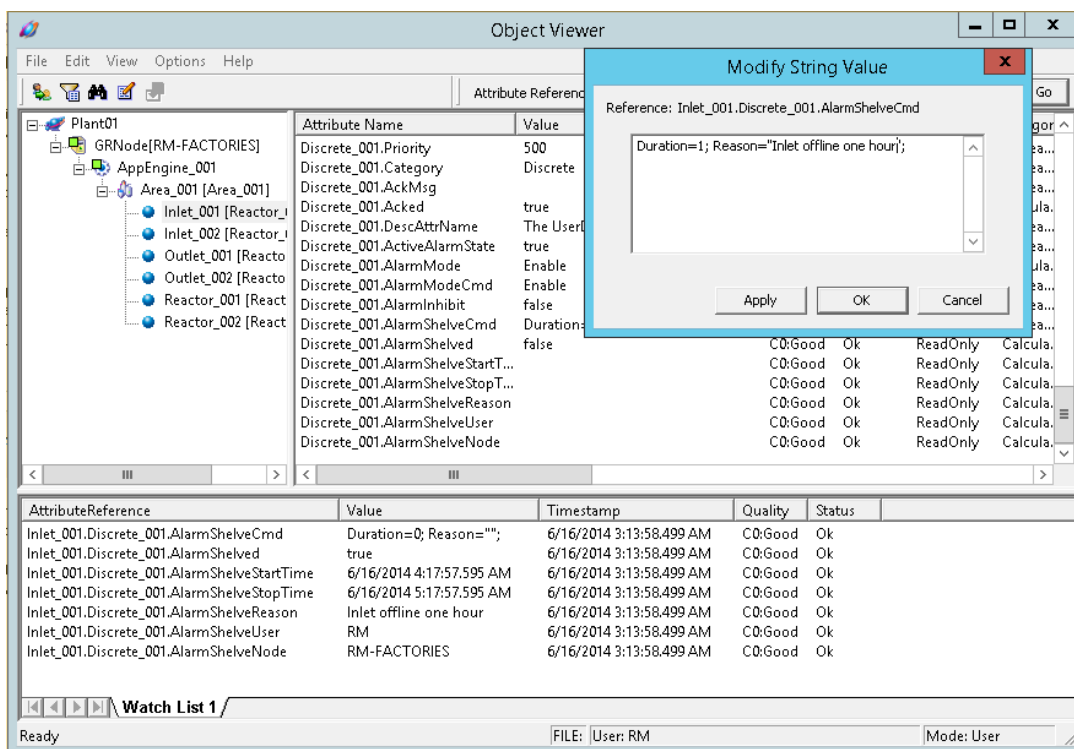
3. Select the severity level in the **Shelve** column. By default, severity levels 3 (medium) and 4 (low) are enabled.
4. In the IDE, open the relevant Area object editor. Click the **Shelve Alarms** checkbox to enable shelving for that area. Shelving is enabled by default.

## Shelving Alarms at Run Time

Use a run-time client to shelve and unshelve alarms. From Application Server, you can use Object Viewer to monitor and control shelved alarm attributes. Shelved alarms can also be monitored and controlled from an embedded Alarm Control and at least one animation to write to the AlarmShelveCmd attribute. For information about using the Alarm Control to shelve alarms, see Shelve and Unshelve Alarms at Run Time.

### To shelve or unshelve an alarm in Object Viewer

1. In Object Viewer, select the object in the **Console Tree** (left pane) which contains the alarm you want to shelve. The object attributes display in the **Details Pane** (right pane).
2. In the Attributes list, find the attribute configured with the alarm you want to shelve.  
For example, "Inlet\_001.Discrete\_001".
3. In the Attributes list, below the attribute in step 2, find the **AlarmShelveCmd** attribute.  
For example, "Inlet\_001.Discrete.001.AlarmShelveCmd".
4. Double-click the .AlarmShelveCmd attribute. The **Modify String Value** dialog opens.



- a. To shelve an alarm, enter a duration in hours or in decimal fractions if the duration is to be less than one hour. The duration cannot be blank.
- b. Enter a reason between the quotation marks for shelving or unshelving. The reason cannot be blank. Click **Apply**.
- c. To unshelve a shelved alarm, enter a value of "0".

The six read-only alarm shelving attributes will display the current shelving status information.

## Throttling Alarms

Alarm throttling prevents network and message queues from flooding during periods of high alarm activity. Throttling sets a maximum number of transitions into and out of an alarm state within a defined period. Alarm transitions that exceed the throttling limit are not reported.

The WinPlatform object includes tuning parameters that specify the maximum alarm rate per second on an engine. The Alarm throttle limit attribute must be used in conjunction with the scan period to determine the maximum number of alarms that can occur in a scan cycle. You can set the alarm throttle limit when you configure the WinPlatform object. For more information about setting the scan period and alarm throttle limit, see *Configuring WinPlatform Object Alarms* on page 376.

Alarm messages can be throttled for alarms going into alarm and out of alarm state. Alarm acknowledgement messages are not throttled. Users can still acknowledge alarms even when alarm throttling is active. Users can still disable alarms for objects or areas when the alarm rate causes throttling to occur. Also, the alarm inhibit and the disable/enable/silence messages are not throttled.

If an active alarm is disabled, the going out of alarm and disabled messages are sent to the notification distributor. If alarms are being throttled, the going out of alarm message can be throttled. The disabled message is accepted regardless of the throttling status. The going out of alarm message is not raised again. This can result in never logging a message for that alarm indicating it went out of alarm. The final alarm list in the Notification Distributor still shows the correct alarm state.

## Propagating Timestamps with Alarms and Events

An alarm feature always registers a reference to the alarmed attribute. This registered reference guarantees that Message Exchange never suspends updates for the alarmed attribute. Even if alarms are disabled for a particular attribute, the Advanced Communication Management feature cannot suspend the attribute.

Be aware that the time stamp propagates to all masked bits of an integer attribute, even if only one of the bits changes.

For example, you have an Integer address in a PLC that represents 16 different alarm states. You assign ObjA.Attr\_Integer to point to the PLC address. You then split the bits to different alarm attributes, adding one attribute for each alarm and naming them ObjA.Attr\_Alarm00 to ObjA.Attr\_Alarm15. Each attribute has an input source that refers to a different bit of ObjA.Attr\_Integer. For example, ObjA.Attr\_00.InputSource -> me.Attr\_Int.00, and so on. At run time, when bit 00 is changing in the PLC, all of the attributes (ObjA.Attr\_Alarm00 to ObjA.Attr\_Alarm15) get a new time stamp, as all the bits changed. This can cause incorrect time stamps for alarms.

## Alarms or Events Become Active

For alarms or events, if an attribute has no timestamp, the current AppEngine scan time is used instead. If an attribute has a timestamp, the timestamp of the value that caused the alarm to occur is used as the value for the object's AlarmOnTime attribute.

When an alarm is silenced or disabled, and an alarm condition becomes TRUE, when the alarm is later enabled, the timestamp for AlarmOnTime is the timestamp of the most recent attribute change, not the time at which the alarm was enabled.

For an event, if an attribute has a timestamp, the timestamp of the value that caused the event to be reported is used for the EventTime. A System Event timestamp is the current system time.

## Alarms Become Disabled

If an active alarm becomes disabled, the alarm is forced to return to normal. The timestamp corresponds to the current time when the alarm became disabled, not the timestamp of the attribute nor of the alarm condition. Otherwise, the assigned timestamp is the AppEngine scan time.

## Alarms Revert to Normal

If an attribute value has a timestamp, the timestamp of the value that caused the alarm to revert to normal is used for the AlarmOffTime.

If alarm is based upon several attributes or upon several values of a single attribute, the most recent timestamp is used when assigning values to the AlarmOnTime or AlarmOffTime.

## Alarm Acknowledgement

Alarms are acknowledged by users who view unacknowledged alarms from their client applications like InTouch HMI. Only alarms of certain priority levels need to be acknowledged.

The basic workflow to acknowledge an alarm consists of the following general steps:

- The alarm is detected and reported to any subscribed alarm clients. The alarm is unacknowledged unless it is of a priority level that does not require acknowledgement.
- An authorized user attempts to acknowledge the alarm from the client. The user can type an optional comment when acknowledging the alarm.
- The user's acknowledge request is sent to the detecting object's alarm feature. The acknowledgment must pass through standard security checks first. The acknowledge request contains the user's name and any alarm comment.

For the alarm utility, the alarm is acknowledged within the alarm feature immediately. The user name, comment, and acknowledged time are also saved. Alarm comments can be localized into any supported language.

The acknowledge is considered an alarm state change, which is sent to all subscribed clients. When an alarm is acknowledged, the current AppEngine timestamp is used as the acknowledgement time.

After an alarm is acknowledged for the first time, any additional (extra) acknowledgement attempts for the same alarm are rejected and an error is returned.

## Acknowledging Alarms with Signature Required

You can use the SignedAlarmAck( ) script function in Industrial Graphics to require a signature to acknowledge alarms. Add the script function to a symbol as described in the *Creating and Managing Industrial Graphics User Guide*, and then embed the symbol in the WindowMaker window.

When you acknowledge the alarms at run time, the script function checks for a required signature. If a signature is required, it checks Galaxy security settings and alarm priority. If any alarm in the list falls within the configured conditions, you must provide your signature to acknowledge the alarms. You must sign in to acknowledge the alarms if no user is logged on to the InTouch application.

If any alarm in the list requires a signature, then a signature is required for the entire list of alarms.

---

**Note:** The SignedAlarmAck() script function supports only ArcestraA alarms.

---

By inspecting the Alarm Viewer or the Alarm Control client, you can determine if the alarm acknowledgements were successful.

## Configuring Alarms

Alarming capabilities are a part of object templates, but they are not implemented until you configure the object in the IDE. After alarms are configured, you can view Application Server alarms in a client visualization application like InTouch HMI.

Configuring an object to be an alarm provider includes the following general sequence of steps:

- Decide whether alarm notification is needed for each possible alarm condition of an object. For example, a command time-out alarm for a valve if the output command fails to move the valve.

- Edit the object and set an attribute that specifies alarming.
- Edit the object from the IDE and assign values to alarm attributes.
- Configure the alarm properties. Typically, the fields that require configuration are Category, Priority and Description.
- Configuring any limit fields to set an alarm. For example, the feedback time-out time limit.

You can add alarm detection and reporting capabilities to objects that were not originally developed to detect alarms. You do this by configuring alarm features for the object's attributes.

## Configuring Alarming for System Objects

To enable alarming for your application, you need to configure your Galaxy's WinPlatform and AppEngine objects as alarm providers. Both system objects report their own alarms.

Client applications subscribe to application object alarms by the area containing the objects. Client applications can also subscribe to WinPlatforms and AppEngines directly. These are called "pseudo-areas." They do not need to be assigned to an area for a client to see the alarms, although the user may want to assign them to an area, such as for simplifying the alarm query in the InTouch Alarm Viewer or Alarm DB Logger.

The following list shows the alarms for each system object.

- WinPlatform
  - Excess CPU load alarm
  - Low disk space alarm
  - Excessive page faults alarm
  - Low memory alarm
  - Engine failure alarm
  - Engine checkpoint failure alarm
  - Object quarantined condition
  - Subscription folding condition
  - Scheduler scan overrun condition

A WinPlatform object includes a general communication alarm when it loses contact with the areas to which it is subscribed.

- AppEngine
  - Checkpoint failure alarm
  - Object quarantined condition
  - Subscription folding condition
  - Scheduler scan overrun condition
  - Redundancy failover alarm
  - Redundancy Standby unavailable
  - Redundancy Standby not ready
- ViewEngine
  - Checkpoint failure alarm
  - Object quarantined condition

- Subscription folding condition
- Scheduler scan overrun condition

The Area and InTouchViewApp system objects do not include any alarms.

## Configuring WinPlatform Object Alarms

You select the areas of your Galaxy to monitor for alarms from the WinPlatform object. Also, you can select specific alarms for the status of the WinPlatform itself.

You must specify that the WinPlatform object is an InTouch alarm provider to subscribe to alarms from the various areas (and pseudo-areas) of the Galaxy and report them to the InTouch Alarm Manager.

## Configuring Communication Failure Alarm Priority

InTouch Alarm Provider reports any communication outage to the InTouch Alarm Manager. Communication alarms are displayed and must be acknowledged. By default, the priority of a communication failure is "1", the highest priority.

There can be instances when a communication failure is not a high-priority alarm, and should not require the attention of a high-priority alarm. Shutting down or undeploying a platform as a normal operation can result in communication failure alarms, for example.

To indicate whether the alarm is of high or low importance, the communication failure alarm priority is configurable at design time as an integer of range 1–999. InTouch Alarm Provider must be enabled in the WinPlatform object editor, **General** tab.

## Selecting the Register using "Galaxy\_<GalaxyName>" instead of "Galaxy" option

When you select the **Register using "Galaxy\_<Galaxy name>"** instead of **"Galaxy" option**, you enable ITAlarmProvider contained in WinPlatform to register the ITAlarmSubsystem using a provider name of Galaxy\_GalaxyName.

For example, if your galaxy is named Andromeda, the ITAlarmSubsystem registers as Galaxy\_Andromeda.

This configuration can be useful if you are using an Alarm Client to display alarms from multiple galaxies.

- This means that all InTouch applications which query alarms from this platform should be modified to use Galaxy\_GalaxyName!AreaName.
- Select this option when the ArchestrA Alarm Control, the embedded alarm client or EAC, or the AlarmViewControl in InTouch queries alarms from different galaxies.

### To configure a WinPlatform object to be an alarm provider

1. Open the WinPlatform object in the Object Editor.
2. Click the **General** tab.

InTouch alarm provider

Enable InTouch alarm provider

Register using "Galaxy\_<Galaxy name>" instead of "Galaxy"

Alarm areas (blank for all):

Communication Failure Alarm Priority: 1

3. Select the **InTouch alarm provider** check box.



4. Select the **Register using "Galaxy\_<GalaxyName>"** instead of **"Galaxy"** check box to enable Galaxy\_<GalaxyName> registration for alarm comment language switching. An information box appears. Click **OK** on the information box to continue.
5. In the **Alarm Areas** box, type the names of areas to subscribe to for alarms.  
If you leave the **Alarm Areas** box blank, the WinPlatform subscribes to all areas in the Galaxy.  
If you want to subscribe to only selected areas within the Galaxy, insert a space between each area name. For example:  
Area1 Area2 Area3
6. Enter a number from 1 to 999 to configure the **Communication Failure Alarm Priority**.
7. Click the **Engine** tab to show the **Alarm throttle limit** box. Either accept the default throttle limit of 2000 alarms per second, or enter another value. A value of 0 disables alarm throttling. For more information about alarm throttling, see *Throttling Alarms* on page 373.
8. Click the **Alarms** tab to show platform, engine, and scheduler alarms that can be set for the WinPlatform object.

The screenshot shows the configuration window for 'TankFarm8WinPlatform' with the 'Alarms' tab selected. The window is divided into three sections: Platform, Engine, and Scheduler.

**Platform Section:**

- Report excess CPU load alarm
  - Alarm limit: 90.0
  - Value deadband: 10.0
  - Priority: 800
- Report low disk space alarm
  - Alarm limit: 750.0 MB
  - Priority: 800
- Report excessive page faults alarm
  - Alarm limit: 5.0
  - Value deadband: 1.0
  - Priority: 800
- Report low memory alarm
  - Alarm limit: 250.0 MB
  - Value deadband: 50.0
  - Priority: 900
- Report engine failure alarm
  - Priority: 900

**Engine Section:**

- Report checkpoint failure alarm
  - Priority: 500
- Report object quarantined condition
  - Priority: 500
- Report subscription folding condition
  - Priority: (empty)

**Scheduler Section:**

- Report scan overrun condition
  - Consecutive scan overrun limit: -1
  - Priority: 500

9. Select the check box next to each alarm that you want to enable for the WinPlatform object.
10. Set the limit, value deadband, and priority for each alarm you selected.

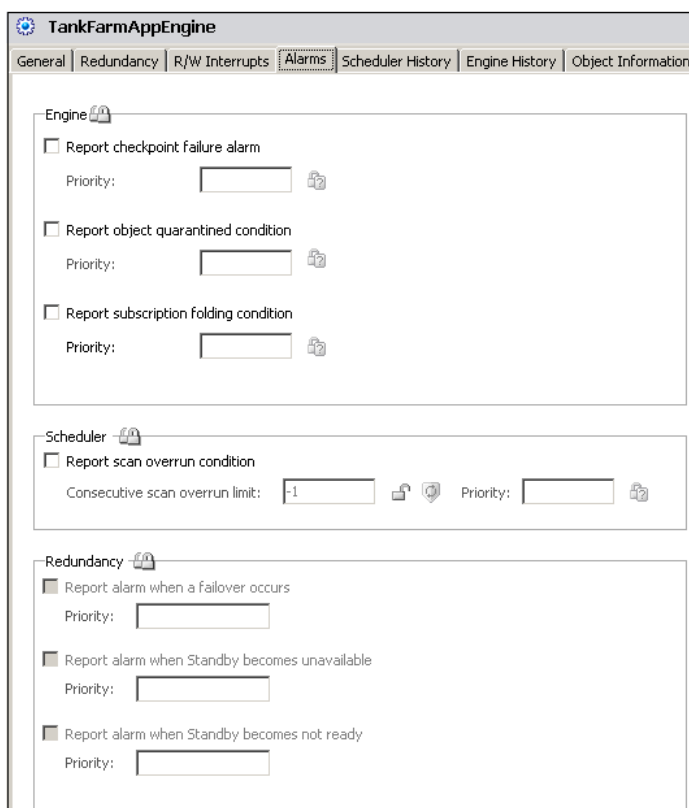
11. Save and close the Object Editor.
12. Check the object in to the Galaxy.
13. Deploy the object in an on scan state.

## Configuring Alarms for an AppEngine Object

You can set AppEngine attributes that determine whether alarms are enabled for an engine, scheduler, and redundant failover engine.

### To configure AppEngine object alarms

1. Open the AppEngine object in the Object Editor.
2. Click the **Alarms** tab to show engine, scheduler, and redundancy alarms that can be set for the AppEngine object.



3. Select the check box next to each alarm that you want to enable for the AppEngine object.
4. Set the priority for each alarm you selected.
5. Save and close the Object Editor.
6. Check the object in to the Galaxy.
7. Deploy the object in an on scan state.

## Configuring Alarms and Events for Application Objects

The following table shows the different types of alarms that can be specified for application objects. The table shows the application objects containing native alarm attributes.

You can also set bad value and state alarms for an object's attributes. For more information about setting alarms for attributes, see *Setting Alarms on the Attributes Page* on page 381.

### Alarm Types

Application Objects	State	Limit	Target Deviation	Rate of Change	Statistical
AnalogDevice	◆	◆	◆	◆	
Boolean					
DiscreteDevice	◆				◆
Double					
FieldReference					
Float					
Integer					
Sequencer	◆				
String					
Switch	◆				
UserDefined	◆	◆	◆	◆	

The following list shows the types of alarms for each application object in more detail.

- AnalogDevice
  - Level alarms (HiHi, Hi, Lo, LoLo) [limit alarms]
  - Rate of Change alarms (Up, Down)
  - Target Deviation alarms (Minor, Major)
  - PV Bad Value alarm [state alarm]
- DiscreteDevice
  - Uncommanded change alarm [state alarm]
  - Command time-out alarm [state alarm]
  - Active1 state alarm [state alarm]
  - Active2 state alarm [state alarm]
  - Fault state alarm [state alarm]
  - Active1 state duration alarm [statistical alarm]
  - Active2 state duration alarm [statistical alarm]
- Sequencer
  - Execution halted [state alarm]
  - Condition trigger failure [state alarm]
  - OnEntry output failure [state alarm]

- OnExit output failure [state alarm]
- Switch
  - PV State alarm [state alarm]
- UserDefined (Attributes can be alarmed)
  - PV State alarm [state alarm]
  - PV Bad Value alarm (that is, bad quality) [state alarm]
  - Attribute alarm features
    - State alarms
    - Limit alarms (HiHi, Hi, Lo, LoLo)
    - Rate of Change alarms (Up, Down)
    - Deviation alarms (Minor, Major)
    - Bad Value alarm (that is, bad quality)

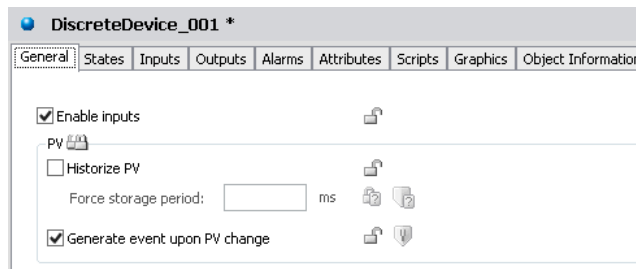
There are no built-in alarms for these application objects:

- FieldReference
- Boolean
- Double
- Float
- Integer
- String

You can also configure your application objects to generate an event each time the object’s PV value changes. In addition, you can configure an alarm feature on any object for any Boolean attribute.

**To configure alarming and events for application objects**

1. Open the application object with the Object Editor.
2. For a DiscreteDevice, click the **General** tab, enable inputs, and then enable **Generate event upon PV**.



For objects that do not include this check box, use the **Attributes** page.

---

**Note:** If you are using field attributes instead of Attributes, use the **Field Attributes** page.

---

- a. Enable the **Log change** feature.

b. Enable **Generate event upon change**.

The screenshot shows the Object Editor for an attribute named 'Attribute001'. The 'Available features' section includes buttons for I/O, History, Limit alarms, ROC alarms, Deviation alarms, Bad value alarm, Statistics, and Log change. The 'Log change' button is checked. Below this, a sub-panel for 'Log change' is expanded, showing a checked checkbox for 'Log system events in addition to user events'.

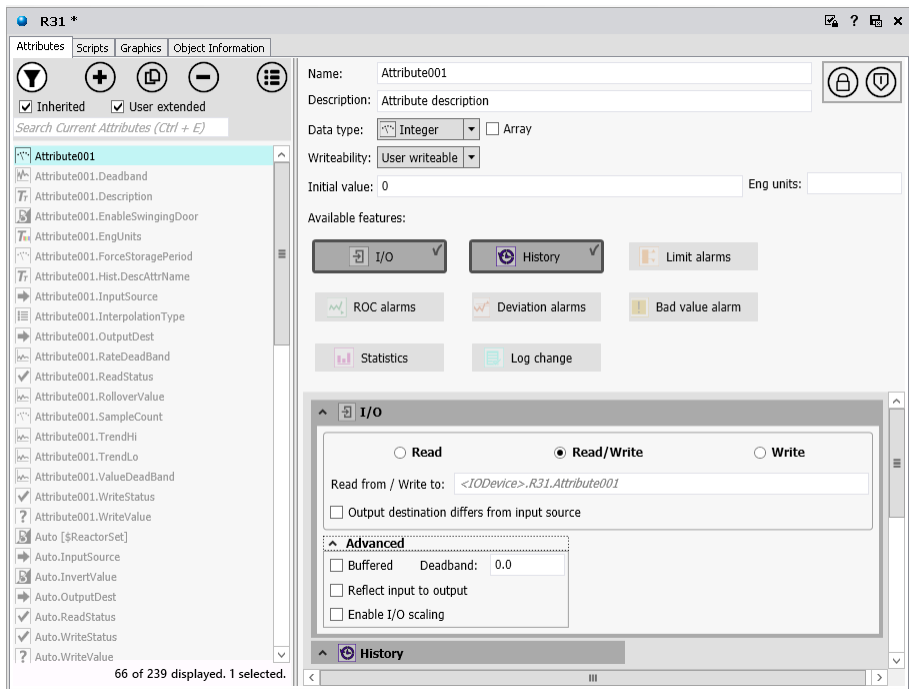
3. Select or clear the check box based on whether you want to generate an event each time the object's PV value changes.
4. Click the tab that lists alarm attributes.
  - For the AnalogDevice object, click **Alarms**.
  - For the DiscreteDevice object, click **Alarms**.
  - For the Sequencer object, click **Settings**.
  - For the Switch object, click **General**.
  - For the UserDefined object, click **Attributes** (or **Field Attributes**, if you are using field attributes). See *Setting Alarms on the Attributes Page* on page 381 for more information.
5. Select the check box that enables alarming for the object.
6. Assign values to the attributes for the type of alarm you selected by completing the following steps:
  - a. Assign values to the alarm limits based on the type of alarm.
  - b. Assign an alarm priority (1-999) for each limit you set.
  - c. Accept the default alarm message or include another message for each alarm limit.
  - d. Assign values to the remaining attributes based on the type of alarm you selected. For more information about other alarm attributes, see *Types of Alarms* on page 362.
7. Save your object changes and close the Object Editor.

## Setting Alarms on the Attributes Page

You set alarms for all Application Server objects in the Attributes page in the Object Editor.

### To specify alarms for an object's attributes

1. On the **Attributes** page of the Object Editor, select an attribute from the **Attributes List**.



2. Select the alarm features you want to configure for the attribute. For Boolean data types, you can select:
  - State alarm**
  - Bad value alarm**

For Integer, Float and Double data types, you can select:

  - Limit alarms**
  - ROC alarms**
  - Deviation alarms**
  - Bad value alarms**
3. For each alarm feature that you activate, enter the settings for each alarm.
  - a. Assign values to the alarm limits based on the type of alarm.
  - b. Assign an alarm priority (1-999) for each limit you set.
  - c. Accept the default alarm message or include another message for each alarm limit.
  - d. Assign values to the remaining attributes based on the type of alarm you selected. For more information about other alarm attributes, see *Types of Alarms* on page 362.
4. To configure event logging for an object, select the **Log change** feature and click the checkbox "Generate event upon change."
5. Save your object changes and close the Object Editor.

## Distributing Alarms and Events

After you configure object instances for alarm detection, deploy the instances and put them On scan. The instances begin checking for alarm conditions.

When an alarm is detected, or an event occurs, a notification is reported to its alarm and event distributor, which is running on the same AppEngine.

These alarm and event distributors include:

- Area objects                      Area objects report detected alarms through the Area, which distributes them to alarm and event clients.
- WinPlatform objects              Report their own alarms and events.
- AppEngine objects                Report their own alarms and events.
- Device Integration objects       Report their own alarms and events.

The Area object plays a key role in alarm and event distribution. All objects belong to an Area. Areas can contain sub-Areas. Alarm and event clients are configured to subscribe to a set of Areas.

Areas provide a key organizational role in grouping alarm information and assigning it to users who use alarm and event clients to monitor their Areas of responsibility.

WinPlatforms, AppEngines and Device Integration objects do not report their alarms and events to Area objects even though they belong to Areas. This allows alarm clients to receive alarm notifications without any dependencies on Area objects. For example, a deployed and running WinPlatform can report alarms even though its Area is not deployed and running.

Alarm-event distributor objects maintain a list of all currently active alarms and inactive but unacknowledged alarms. They do not maintain a list of events, which are routed to clients that are currently subscribed at the time of the event.

You can configure a WinPlatform to act as an InTouch Alarm Provider in the run-time environment.

The WinPlatform sends an alarm through the InTouch Distributed Alarm System to InTouch clients when the WinPlatform loses communication with an Area that it subscribes to. This condition typically occurs during a network outage with computers hosting those Areas.

In a network outage, the WinPlatform InTouch Alarm Provider sends an alarm for each disconnected Area that it subscribes to, including all of its alarm distribution hierarchy. Each of these alarms is a high priority alarm that contains the name of the Area to which communication is lost. These communication problem alarms must be acknowledged.

Although they still appear in the historical record, any current alarms from the disconnected Area drop from the InTouch client's summary list. They can no longer be acknowledged.

When communication to the disconnected Areas is restored, any unacknowledged alarms generated in those Areas are sent to the alarm client.

## Subscribing to Alarms and Events from a Client

Clients indicate interest in alarms and events by subscribing to an Area. When subscribing to an Area, the subscription is actually to all notification distributors within that Area.

For example, if an Area contains sub-Areas, those sub-Areas are subscribed to. If WinPlatforms, AppEngines or Device Integration objects belong to an Area, those objects are also directly subscribed to.

When a notification distributor receives an alarm and event subscription from a client, the notification distributor provides the client with the following:

- A list of all current alarm conditions, including unacknowledged return-to-normal conditions.
- An alarm condition state change. A state change includes transitions into or out of alarm (return to normal) and change in acknowledged flag.
- An event occurrence.

Alarm and event subscription requests do not include filters, for example, only show alarms greater than a specific priority value. All alarm and event messages received by the notification distributor are sent to all subscribed clients. Filtering is provided as a display option by clients.

## Using InTouch HMI as the Alarm and Event Client

InTouch run-time clients subscribe to event reports from a Galaxy. Application Server reports alarms to the InTouch Distributed Alarm System, which subscribe to alarm and event reports from a Galaxy.

An InTouch client application can visualize Application Server components. An InTouch alarm client can show alarm information for new, unacknowledged alarms, including all required fields.

The new alarm is in the unacknowledged state. An operator can view alarms, acknowledge alarms, disable alarms, and enable alarms from the client application running in InTouch WindowViewer.

## Understanding the Syntax of Alarm Queries

InTouch alarm queries subscribe to alarm and event information from objects within a Galaxy. The alarm and event queries can be in the form of user input or a script.

Alarm query syntax must be in one of the following forms:

```
\Provider!Area
```

or

```
\\Node\Provider!Area
```

You can have one or more references in a query separated by spaces.

You can also optionally append a tagname filter at the end, separated by another exclamation mark:

```
\Provider!Area!Filter
```

```
\\Node\Provider!Area!Filter
```

The filter can have a wildcard \* character at the beginning or at the end, but not both.

The \\Node at the beginning is only important if you want to query for alarms from a provider on another computer. Otherwise, you can leave it off and the reference is assumed to be a provider on the local computer. The provider name Galaxy refers to alarms and events that get reported by the WinPlatform configured as an InTouch alarm provider on that computer node.

## Alarm Query Syntax when Register Using Galaxy\_<GalaxyName> is Enabled

In the WinPlatform object, when you enable InTouch alarm provider, you can enable **Register using Galaxy\_<GalaxyName> instead of Galaxy**. This option will register the platform to the alarm subsystem using the Galaxy name prefixed by "Galaxy\_" instead of just the word "Galaxy". This allows an InTouch application to monitor alarms from multiple Galaxies and avoid name conflicts.

Syntax changes slightly when Galaxy\_GalaxyName is enabled:

- Use \\ for computer name.
- Use \ for Galaxy or Galaxy\_<GalaxyName>.
- Use ! for Area.

For example: \\Galaxy\MyGalaxy!Area001.

If Galaxy\_GalaxyName is not enabled in WinPlatform, then the default behavior described in the previous section applies.

You can determine if Galaxy\_<GalaxyName> has been enabled by monitoring the run-time attribute of the platform ITAlarmProvider.ProviderNameAsGalaxyNameEnabled.



## Examples of Alarm Queries

- You can submit a query to get all alarms from Area1 and all other alarms within Area1, as reported by the WinPlatform object on the local computer.

```
\Galaxy!Area1
```

The query returns all alarms and events from all objects directly contained in Area1 and any sub-areas contained by Area1. This hierarchy is determined by what is configured in the Model View in the IDE.

- If Area1 and Area2 are two separate mutually exclusive areas, you can submit a query for alarms from both areas.

```
\Galaxy!Area1 \Galaxy!Area2
```

- If you're on NodeA and the WinPlatform is on NodeB, you can submit a query for the alarms from the remote computer.

```
\\NodeB\Galaxy!Area1
```

- You can submit a query for all alarms from objects whose name begins with "Tank" in the TankFarm1 area.

```
\Galaxy!TankFarm1!Tank*
```

The trailing wildcard character matches alarms from all objects with names that begin with "Tank" like Tank001, Tank002, TankUpper, or TankLower.

- You can submit a query for specific alarm types. For example, you can submit a query for all HiHi alarms in the TankFarm1 area.

```
\Galaxy!TankFarm1!* .HiHi
```

- You can submit a query for all types of alarms from a specific object within an area.

```
\Galaxy!TankFarm1!Tank752.*
```

The trailing wildcard character matches all alarm types for Tank752.

## Alarm Requirements for InTouch Client Applications

For Application Server alarming to function, the following conditions must be met:

In Application Server:

- One or more Area objects are deployed and running.
- The source object is on scan.
- The source object's Area is on scan.
- Alarming must be enabled for the target object.
- An InTouch alarm provider on any WinPlatform in the Galaxy.

In InTouch:

- The InTouch client application is running in WindowViewer.
- An InTouch alarm ActiveX control is placed in a window and configured as an alarm consumer for the Galaxy.

- The user is logged into InTouch using Arcestra security and is authorized to acknowledge alarms for the object that is in the alarm state. If the user only wants to view alarms, security authorization is not required.

Application Server validates the user has sufficient security privileges to acknowledge the alarm.

If the user does not have privileges to acknowledge alarms, the user can attempt to acknowledge the alarm, but the Galaxy rejects the acknowledgment request. The alarm remains unacknowledged in the InTouch Alarm display.

The rejected alarm acknowledge event is recorded in InTouch Event History if the user attempting the acknowledgement has a valid Galaxy user account. Otherwise, the rejected acknowledgement is not recorded as an event.

You can acknowledge multiple alarms by providing a valid signature. At run time, the Industrial Graphics SignedAlarmAck() script function checks a set of alarms and conditions to find out if any alarm from the list requires a signature in order to acknowledge it. If so, you must provide your user name, password and domain or your Smart Card details and PIN to authenticate yourself and acknowledge the alarms. To be able to provide the Smart Card details, your computer must be configured for Smart Card authentication.

---

**Note:** Smart Card authentication is not supported in multi-galaxy environments for read/write operations to remote galaxies.

---

If even one alarm in the list requires a signature, you must provide a signature to acknowledge the alarms. The SignedAlarmAck() function will acknowledge ALL the alarms in the list.

With the SignedAlarmAck( ) function, you can provide credentials and acknowledge an alarm or group of alarms even if you are not the logged-on operator.

## Alarms and Events in InTouch HMI and in Application Server

Both InTouch HMI and Application Server implement alarms and events. The following table shows the similarities and differences between the two products.

Item	InTouch	Application Server
Alarm configured or detected by	Within a tag	Within an object
Alarm Classes (client column)	Only certain classes of alarms are supported or detected: DSC, VALUE, DEV, ROC, SPC.	No system-wide distinction for classes. Alarms are tied to a Boolean that can be triggered from any logic.
Alarm Type (Sub-class) (client column)	Discrete, LoLo, Lo, Hi, HiHi, MinorDev, MajorDev, ROC, SPC. Client column.	No sub-class. The Alarm feature name is the closest concept. For example, ".PVHiAlarm". Mapped from Category.
Priority (client column)	1-999 (1 most urgent)	Priority 0-999. 0 most urgent. 0 is mapped to 1 in InTouch.
Name (client column)	Alarm name = Tag name.	Object.attribute

Item	InTouch	Application Server
Comment (client comment)	Separate alarm comment, which is different from the tag comment.	Object short description or alarm message where available.
Group	Alarm group allows client-side filtering. Sub-groups must be on same InTouch.	No alarm group. But Area provides mappable concept. Sub-Areas can be on different nodes.
State	<p>Four states, which are combinations of ACK/UNACK and ALARM/RTN</p> <ul style="list-style-type: none"> <li>• UNACK/ALARM (usually displayed as UNACK)</li> <li>• ACK/ALARM (usually displayed as ACK)</li> <li>• UNACK/RTN (usually displayed as UNACK_RTN)</li> <li>• ACK/RTN (usually displayed as ACK_RTN)</li> </ul> <p>These states have a 1:1 correspondence with states of the Alarm feature, which keeps track of whether the alarm is InAlarm and IsAcked.</p> <p>Alarms in the state ACK/RTN are not shown in the SUMMARY alarm display because they do not need any further action from the operator. But, all four states appear in the HISTORY display, and in the Alarm Database.</p>	Alarm state provides equivalent concept and can be mapped.
Value, CheckValue	Only static values sent with alarm message.	Static values and dynamic references are provided.
Ack	All alarms sent to client and require acknowledgement regardless of priority.	All alarms sent to client and require acknowledgement regardless of priority.
History	Alarm state changes are logged to event history and shown on historical client.	Alarm state changes are logged to event history and shown on historical client.

## Configuring Plant State-Based Alarms

You can map alarm modes on a per-Galaxy basis to different plant operational states to control how alarms are reported. Five plant states are pre-defined, and have default alarm states associated with them:

Plant State	Default Alarm State	Available Alarm States
Running	Enable	Enable
Maintenance	Disable	Enable / Silence / Disable
Startup	Silence	Enable / Silence / Disable
Shutdown	Disable	Enable / Silence / Disable
Testing	Silence	Enable / Silence / Disable

You can define new plant states, rename plant states, or remove existing plant states, except the "Running" state (you can, however, rename "Running"). The alarm state for Running is read-only and cannot be changed from Enable.

After you have defined plant state-based alarm configurations for the Galaxy, you can assign plant state-based alarming to area objects in the Galaxy. This is done by setting the PlantState attribute for each area object that will use plant state-based alarming. The area object will automatically update its PlantAlarmMode attribute to match the alarm state that is set for the PlantState currently assigned to it.

Attribute	Definition
PlantState	The name of the currently assigned plant state (Running, Maintenance, etc.).
PlantStateAlarmMode	The alarm state of the assigned plant state (enable, silence, or disable). This attribute is read-only at run time.

## Mapping Alarm Modes to Plant States

Define plant states and map them to alarm modes in the IDE. To access the configuration dialog:

1. Click **Galaxy** on the IDE menu.
2. Select **Configure**, then select **Alarms and Events Configuration**. The **Alarms and Events Configuration** dialog opens.

**Note:** Settings in this dialog are not shared across Galaxies in a multi-Galaxy environment. Each Galaxy in the environment will have its own **Alarms and Events Configuration**.

To use the default values, you do not have to set anything in this dialog. Simply enable plant state-based alarms for each area object that is to utilize this feature. See *Configuring State-Based Alarming on an Area Object* on page 390 for additional information.

The dialog box is titled "Alarms and Events Configuration" and contains several sections:

- Alarms:** A table with columns: Severity, Description, Shelve, Historize, From Priority Range, To Priority Range, and Image.
 

Severity	Description	Shelve	Historize	From Priority Range	To Priority Range	Image
1	Critical	<input type="checkbox"/>	<input checked="" type="checkbox"/>	1	250	
2	High	<input type="checkbox"/>	<input checked="" type="checkbox"/>	251	500	
3	Medium	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	501	750	
4	Low	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	751	999	
- Modes:** A table with columns: Description, Image.
 

Description	Image
Inhibited/Disabled	
Silenced	
Shelved	
- Events:** A table with columns: Types, Description, Historize.
 

Types	Description	Historize
1	System	<input checked="" type="checkbox"/>
2	Application	<input checked="" type="checkbox"/>
3	User	<input checked="" type="checkbox"/>
- Alarm Plant State:** A table with columns: Description, AlarmMode.
 

Description	AlarmMode
Running	Enable
Maintenance	Disable
Startup	Silence
Shutdown	Disable
Testing	Silence
- Alarm Adorners Options:** Two groups of checkboxes.
  - Severities to Display:**
    - Critical
    - Medium
    - High
    - Low
  - Alarm States to Display:**
    - UNACK (Un-Acknowledged)
    - ACK (Acknowledged)
    - RTN (Return to Normal)

Buttons for OK and Cancel are at the bottom right.

## Change Alarm Modes and Modify Plant States

- To change the AlarmMode value, click the AlarmMode associated with the Plant State that you want to change. Then, select the new value (enable, silence, or disable).

---

**Note:** AlarmMode for Running is read-only and cannot be changed.

---

- To add a new Plant State, Click the **add** button. A new row is created at the bottom of the table. Enter the name of the new plant state. The maximum name length is 64 characters. Select a value for the AlarmMode (enable, silence, or disable).

- To delete an existing Plant State, click the **remove** button. The selected plant state and its associated value are removed from the table.

---

**Note:** The first row of the Alarm Plant State table is read-only and cannot be deleted. Running is the default PlantState name in the first row.

---

- To rename an existing PlantState, double click the plant state you want to rename, then enter the new name for the plant state. The maximum name length is 64 characters.

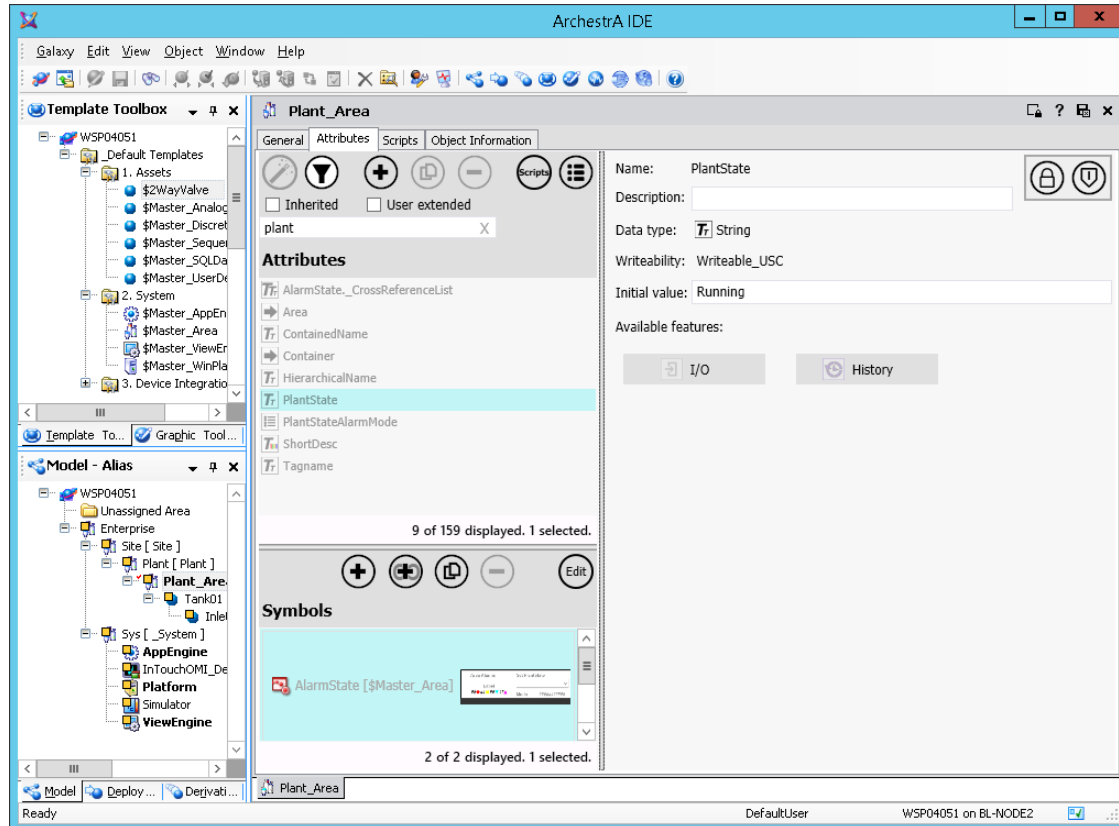
---

**Note:** All Plant States, including Running, can be renamed.

---

## Configuring State-Based Alarming on an Area Object

State-based alarm modes apply to String attributes, with writeability set to Writeable\_USC (these cannot be changed). To use state-based alarming for one or more areas, you must configure the PlantState attribute for each area that will use plant state-based alarming.



### Configure plant state alarming

1. Open the area object and click the Attributes page.
2. Select the PlantState attribute.
3. Enable the **I/O** feature and select Read, Read/Write, or Write.
4. Enable the **History** feature and configure it.
5. Select the PlantStateAlarmMode.
6. Repeat steps 3 and 4 to configure the **I/O** and **History** features.

## Viewing Plant State-Based Alarms at Run Time

You can view plant state-based alarming at runtime through clients such as Object Viewer and InTouch applications in WindowViewer.

## User Access and Security During Run Time

Operators with the appropriate permissions can change the PlantState of an area through a run time client. An operator may have to change PlantState from Startup to Running, for example. Scripts can also be used to implement modifications.

To change a PlantState, the operator enters a string to match one of the defined PlantState values (for example, "Running"). Before a change to PlantState can be implemented, the system checks that the user belongs to a role that has the permission "Can Modify Plant State of an Area" and is therefore authorized to make the change. If the operator enters a string that does not match a defined PlantState, the change is rejected.

At run time, the AlarmMode attribute is read-only. Therefore, the AlarmMode for a PlantState cannot be changed through a run time client (for example, changing from enable to silence). It can only be changed through the IDE.

If an area is assigned to a PlantState and the PlantState is deleted through the IDE, the area will remain in that PlantState until it is changed. If AlarmMode for the deleted PlantState is anything other than enable, the AlarmMode will change to enable.

---

**Note:** Operators with appropriate permissions can use the AlarmModeCmd attribute to change AlarmMode of an area at run time. However, the AlarmModeCmd can only be used to set a more restrictive condition than the AlarmMode of the area's corresponding PlantState. For more information about changing alarm modes, see *Enabling, Silencing, and Disabling Alarms* on page 368.

---

## Configuring Alarm Historization, Mapping, and Shelving Priority Ranges

The **Alarms and Events Configuration** dialog lets you:

- Map alarm severities to priority ranges.
- Enable alarm shelving by priority range.
- Enable or disable historization of both alarm severities and event types. All alarms are historized by default.

---

### Default Alarm Mapping and Historization Values

---

Severity	Description	Shelve	Historize	From Priority Range	To Priority Range
1	Critical	N	Y	1	250
2	High	N	Y	251	500
3	Medium	Y	Y	501	750
4	Low	Y	Y	751	999

Shelving, historization and priority ranges are configurable.

### Mapping Alarm Severity to Priority

- Severity 1 starts at priority 1 by default. Severities can be mapped to priorities in ascending or descending order. For example, severity 1 can map to priority range 1–250 or it can map to priority range 999-751.
- Severity 4 ends at 999 by default, but 999 is not required to be the end-of-range number. For example, severity 4 can end at 900, leaving a priority range above 900 unmapped to any severity level.
- A priority outside the configured priority ranges will return severity 0, not mapped. Since the priority is unmapped, it cannot be historized.

**Note:** Mapped alarm severities are not shared across galaxies in a multi-galaxy environment. Each galaxy in the environment has its own priority to severity mapping.

## Configuring Historization for Alarms and Events

Historization is configurable for both events and alarms. Event types and the default historization setting for each type are:

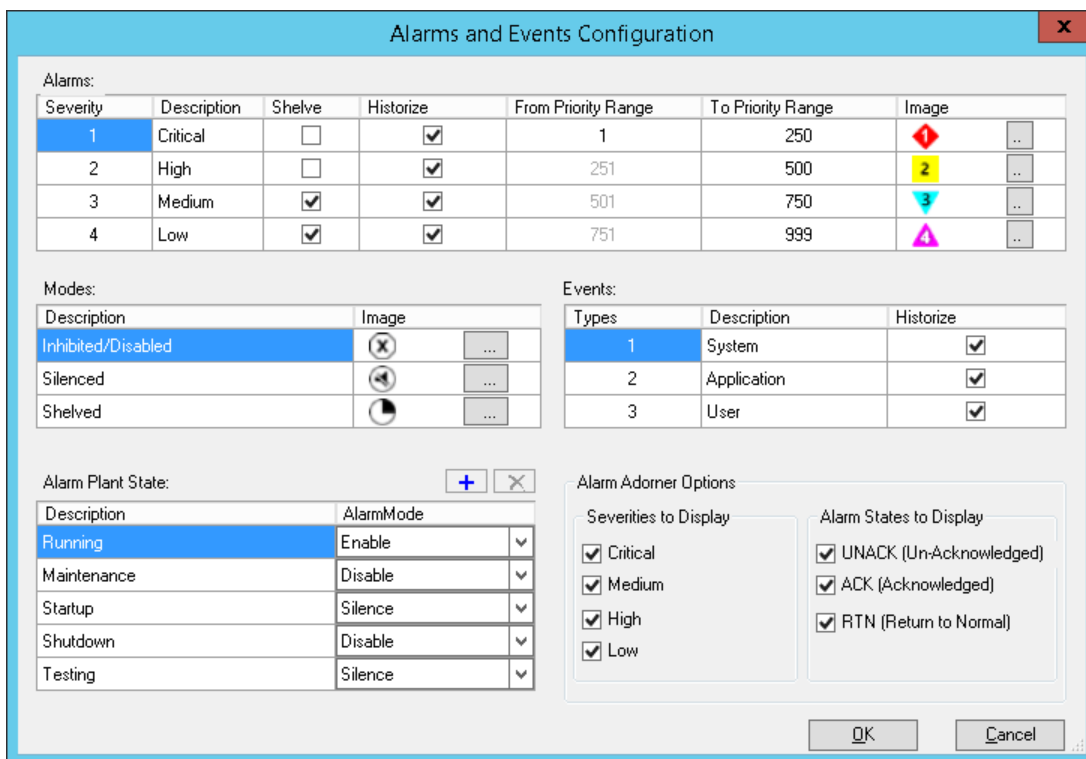
### Default Event Type Historization Values

Type s	Description	Historize
1	System	Y
2	Application	Y
3	User	N

**Note:** In prior versions of Application Server, historization of alarms and events was configured through the Alarm DB Logger Manager (an InTouch component) and utilized the SQL database "WWALMDB." The method described here replaces the Alarm DB Logger method.

### To configure alarm and event mapping

1. On the IDE main menu, click **Galaxy**, then click **Configure**, then click **Alarms and Events Configuration**. The **Alarms and Events Configuration** dialog appears.



2. Configure alarms:
  - a. Edit the priority range for each severity level, or accept the default values.



- b. Accept the default (enabled) historization or click to clear the check box and disable historization for each severity level.
3. Configure events: Accept the default historization values, or click to clear the check box and disable historization for each event type.

If you want to change the default Alarm Border indicator images shown in the **Alarms and Events Configuration** dialog, see "Changing Alarm Border Indicator Icons" in the *Creating and Managing Industrial Graphics User Guide*.

## Monitoring Alarm Severities at Run Time

Monitor alarms by severity, expressed as integers 1–4, using clients such as Object Viewer and AVEVA OMI ViewApps. You can monitor individual alarms and you can monitor alarms aggregated by containment level. For more information about aggregating alarm states, see *Aggregating Alarm State Information* on page 394.

Typically, alarm priorities and priority to severity mapping are not changed during run time, but it is possible to change alarm priority configuration and severity mapping without closing the client application.

## Understanding How Alarms are Ranked at Run Time

The most important alarm has the lowest severity number, but other criteria are taken into account when ranking alarms by urgency at run time.

<b>Alarm Mode</b>	Enabled is more urgent than silenced.
<b>Alarm Shelved</b>	FALSE (not shelved) is more urgent than TRUE (shelved).
<b>InAlarm</b>	TRUE (InAlarm) is more urgent than FALSE (normal).
<b>Acked</b>	FALSE (unacked) is more urgent than TRUE (acked).
<b>Severity Level</b>	1 is most urgent; 4 is least urgent.

## Using Aggregated Alarm Information

Alarm aggregation lets you see all the active alarms on a containing object, such as an area object, as well as all the active alarms on its contained objects. This provides an efficient way at run time to identify if there are any active alarms (in the InAlarm state), and the location and status of alarms, for example, unacknowledged and unacknowledged - return to normal. This makes it possible to follow a trail from one level to the next to find the underlying cause of a complex object's alarms.

To obtain aggregated alarm severity status information:

1. Map alarm severity levels to priority ranges. For more information, see *Configuring Alarm Historization, Mapping, and Shelving Priority Ranges* on page 391.
2. Configure alarms on objects. For more information, see *Configuring Alarms* on page 374.
3. Enable alarm aggregation by Area. For more information, see *Configuring Alarm State Aggregation* on page 394.

- View aggregated alarm status information by means of run-time clients and applications. For more information, see *Monitoring Alarm State Information at Run Time* on page 400.

## Configuring Alarm State Aggregation

Configuring alarm state aggregation consists of normal alarm configuration procedures plus an added step of enabling the aggregation feature on each relevant Area object.

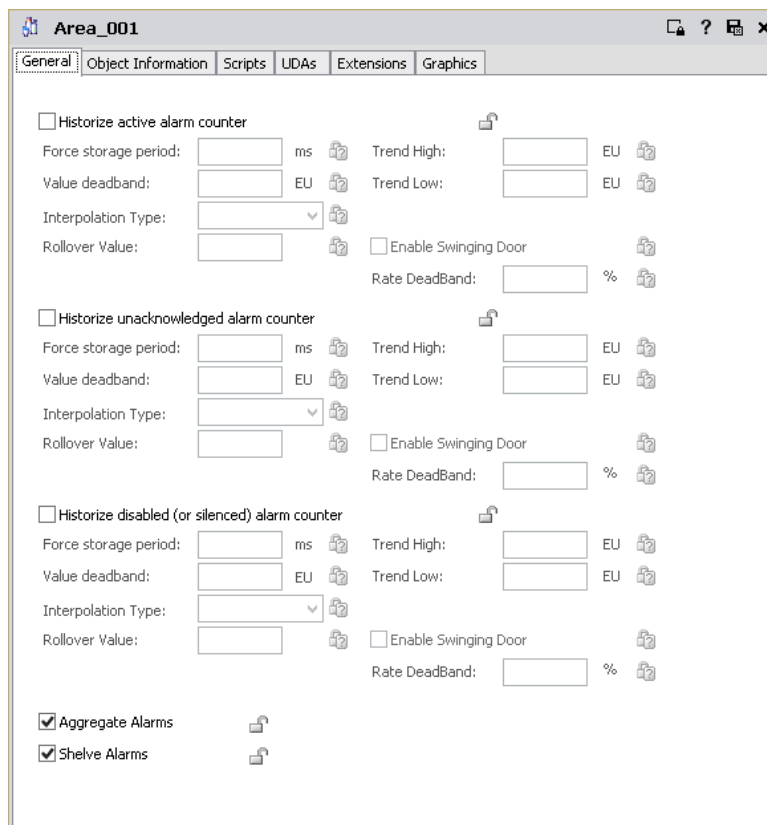
---

**Note:** Alarm aggregation is enabled by default on Area objects. Alarm aggregation cannot be disabled on application objects, such as User Defined objects.

---

### To configure alarm state aggregation

- Accept the default settings or configure alarm severities for each of the four severity levels you want to aggregate. This is a global configuration.
- Configure an object with one or more alarms. All alarms configured on all objects will be aggregated.
- Set the AlarmModeCmd of the object to enabled or silenced, but not to disabled.
- Set the AlarmModeCmd of at least one alarm on the object to enabled or silenced, but not to disabled.
- Set the AlarmModeCmd of the Area object to enabled or silenced, but not to disabled.
- Check the **Enable Alarm Aggregation** checkbox on the Area object editor to enable alarm aggregation. This sets the value of the AlarmAggregationStateCmd attribute to True.



## Aggregating Alarm State Information

Run-time clients and applications, such as Object Viewer, AVEVA OMI ViewApps, and InTouch HMI applications, can show alarm aggregation statuses.

You can use animations such as the alarm border animation with Situational Awareness Library symbols or with Industrial Graphics in both Managed InTouch applications and AVEVA OMI ViewApps, and you can add alarm aggregation displays to navigation items in AVEVA OMI ViewApps

Alarm aggregation functionality can be described for an object, for an area, and for an attribute.

---

Object	<p>Aggregation represents all alarms on the object, on all contained objects, and on their descendents down to the lowest level of the model view.</p> <p>Alarm aggregation values on child objects are added to the values of the parent object or objects.</p> <p>All objects have this alarm aggregation functionality.</p>
--------	--

---

Area	<p>Aggregation represents the alarms on the Area object itself, on all objects assigned to the area, and on all sub-Areas, down to the lowest level of the model view.</p> <p>If the Area's AlarmMode is silenced, all alarms on all Objects in that Area will be silenced. For more information about setting alarm modes, see the following topics:</p> <ul style="list-style-type: none"> <li>• <i>Understanding Alarms</i> on page 360</li> <li>• <i>Setting Alarm State with Object Attributes</i> on page 366</li> <li>• <i>Enabling, Silencing, and Disabling Alarms</i> on page 368.</li> </ul>
------	---

---

Attribute	<p>Aggregation represents all the alarms on the attribute itself. This is the lowest level of aggregation.</p>
-----------	--

---

Alarms are aggregated if they are in one of three states:

- UNACK\_ALM
- ACK\_ALM
- UNACK\_RTN

Alarms in the ACK\_RTN state are not aggregated.

Alarms in silenced mode are aggregated, even though they do not appear in a run-time client.

A set of five attributes provide run-time aggregated alarm status information:

---

AlarmMostUrgentSeverity	<p>Displays the severity as an integer 1–4 of the most important current alarms on an object and its descendents. If no alarms are in the InAlarm state or waiting to be acknowledged, the value is 0.</p>
-------------------------	--

---

AlarmMostUrgentMode	<p>Displays the mode (enabled or silenced) of the most important current alarm. Alarms in disabled mode are not aggregated.</p>
---------------------	---

---

AlarmMostUrgentInAlarm	<p>Displays the InAlarm status (true or false) of the most important current alarm.</p>
------------------------	---

---

AlarmMostUrgentAcked	<p>Displays the acknowledgement status (true or false) of the most important current alarm.</p>
----------------------	---

---

AlarmCntsBySeverity	<p>In AVEVA OMI and InTouch HMI run-time client applications, this attribute is used to show the number of alarms by status and severity. In the Object Viewer, this attribute displays an array that shows:</p>
---------------------	--

---

- The total number of active alarms (UNACK\_ALM + UNACK\_RTN + ACK\_ALM) at each severity level.
- The UNACK\_ALM alarm count.
- The UNACK\_RTN alarm count.
- A decimal representation if any active alarm bits are set on the local object.

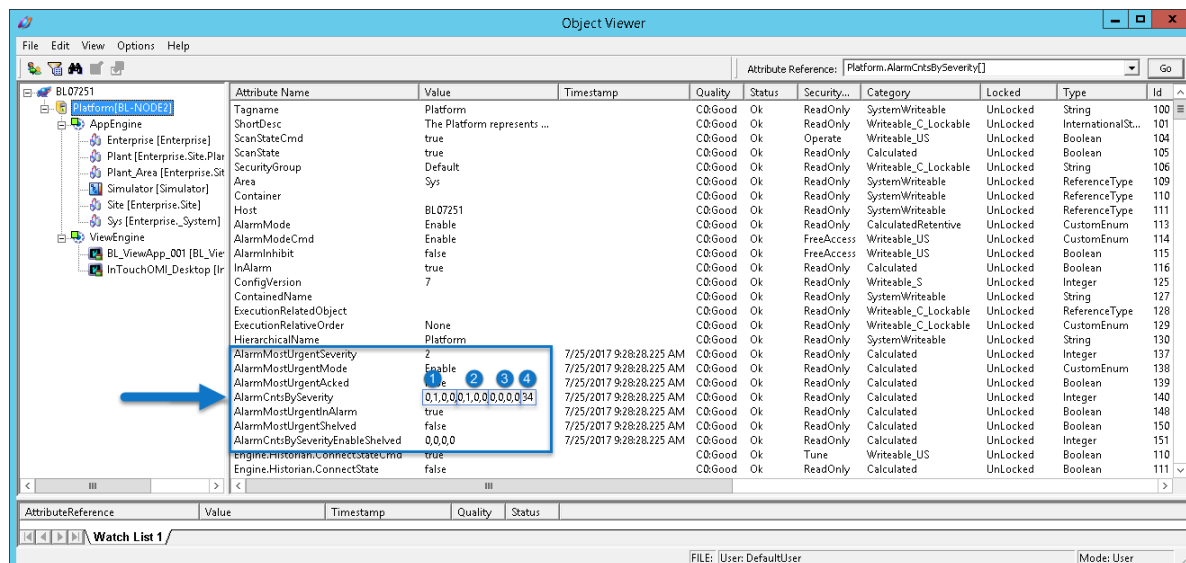
## About the AlarmCntsBySeverity Attribute

**AlarmCntsBySeverity** is a 13 element array, divided into three groups of four, plus one additional element to indicate local alarm severity and status. The first group of four elements shows the number of aggregated active alarms by severity (S1 through S4), the next group shows the aggregated number of unacknowledged alarms (UNACK\_ALM) by severity, and the third group shows the aggregated number of return to normal (UNACK\_RTN) alarms by severity. The final element of the array is a bit value that defines which, if any, of the alarms are on the selected (local) object. See *Local Alarm Display Display in the Object Viewer* on page 396 for additional information.

You can determine the number of active, acknowledged alarms (ACK\_ALM) by subtracting the number of unacknowledged alarms shown in the second group of elements from the active alarm count shown in the first group of elements.

## Local Alarm Display Display in the Object Viewer

The Object Viewer shows thirteen values, separated by commas, for the **AlarmCntsBySeverity** attribute. The first twelve values (logically divided into groups of four each) show how many alarms of each severity and type have been aggregated for the selected object or area:



1. Number of active alarms, by severity.
2. Number of unacknowledged active alarms, by severity.
3. Number of unacknowledged alarms that have returned to normal, by severity.
4. Bit value that shows which alarm severity and status applies to the local object.

This value is a sum of the bit values shown in the following table:

	← (LSB) Active (1)				UnAck_ALM (2)				UnAck_RTN (3) (MSB) →			
	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4
Bit value	1	2	4	8	16	32	64	128	256	512	1024	2048

In the Object Viewer example shown above, the local object bit value (4) is 34. This is the sum of the active S2 bit value plus the UnAck\_ALM S2 bit value. Other examples:

- If the local object has one or more active S1 alarms that are unacknowledged, the bit value is 17 (S1 Active Alarm = 1; S1 UnAck\_ALM = 16).
- If the local object has one or more active S4 alarms that are UnAck\_RTN, the bit value is 2056 (S4 Active Alarm = 8; S4 UnAck\_RTN = 2048).
- If there are no local alarms, the bit value is 0.
- If there are multiple local alarms with different severities and unack statuses, the bit value is the sum of each bit that is set. For example, if there are two unacknowledged S1 alarms, but one has returned to normal, the bit value is 273 (two S1 Active Alarms = 1; one S1 UnAck\_ALM = 16; one S1 UnAck\_RTN = 256). Note that each bit that is set is only counted once, even if more than one alarm exists at that severity or status.

When alarm aggregation is shown on a navigation item in an AVEVA OMI ViewApp, the bit value is used by the navigation item to determine the alarm location (selected object or a contained object). See *Aggregated Alarm Display in AVEVA OMI Navigation* on page 397 for additional information.

## Aggregated Alarm Display in AVEVA OMI Navigation

The **Alarms and Events Configuration** dialog is used to configure which aggregated alarms are displayed on a Galaxy-wide basis. You can set whether or not display alarm information in individual AVEVA OMI ViewApps by enabling/disabling the **ShowAlarms** attribute. AVEVA OMI ViewApps display alarm indicators and alarm counts for aggregated alarms on NavigationApp controls (NavigationTree and BreadcrumbControl), if the **ShowAlarms** attribute is enabled in the ViewApp. To enable this attribute, select the **Show Alarms** option when you configure the control in the **Layout Editor**.

---

**Note:** Alarm indicators ("adorners") used in AVEVA OMI navigation are FrameworkElement Adorners, part of the Microsoft .NET Framework used in creating and configuring apps for AVEVA OMI ViewApps.

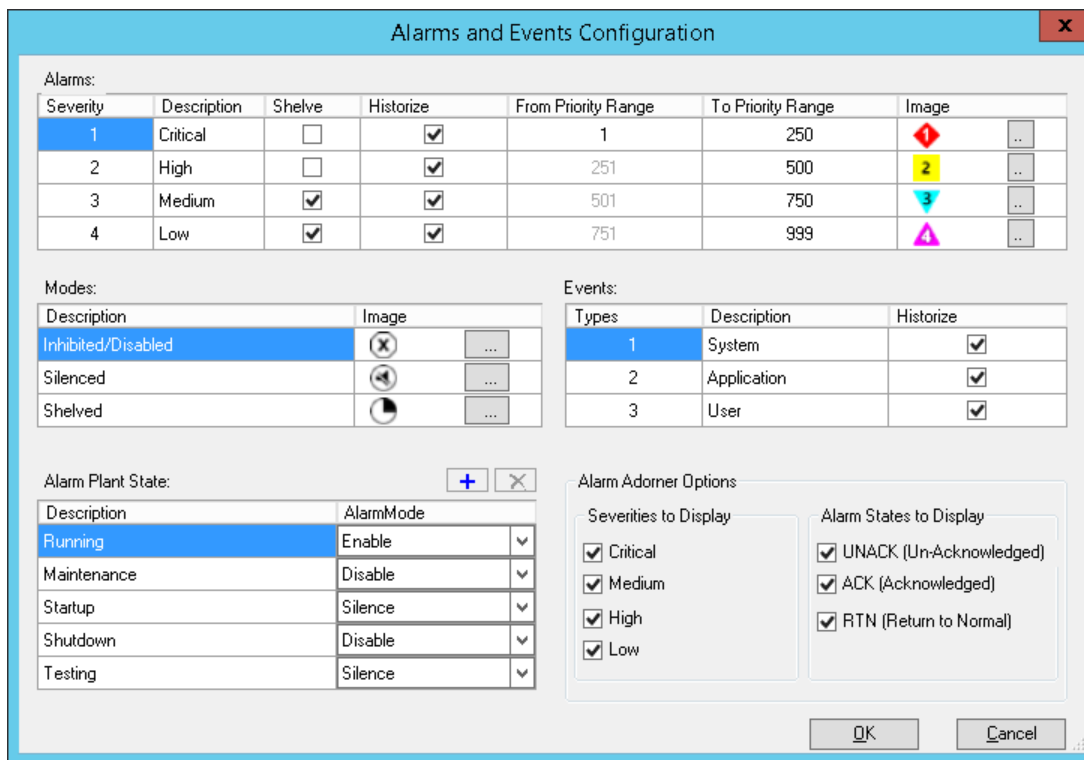
---

The opacity of the alarm indicator shows if the alarm is on the selected navigation node or on a contained object. When enabled, the following alarm severities and alarm statuses are shown:

- If the color of the alarm indicator is opaque (solid), the alarm is on the selected (local) object. If enabled, the severity (S1 to S4) and status of the alarm (UNACK\_ALM, UNACK\_RTN, ACK\_ALM) is shown graphically, along with a count for each alarm severity that has been enabled.
- If the color of the alarm indicator has some transparency, the alarm is on a contained object.
- The color of the blinking line under the object name shows the highest-severity (most urgent) alarm.
- An indicator on the NavigationTree or BreadCrumbControl shows if the data quality of the object is bad. Bad quality may result if communication has been lost with the object.

You can also determine which navigation node or asset contains the alarm by selecting the alarm icon. This displays a tooltip that indicates the node(s) or asset(s) sending the alarm(s).

Set **Alarm Adorner Options** in the **Alarms and Events Configuration** dialog (Galaxy > Alarms and Events Configuration) to configure which aggregated alarms are displayed on NavigationApp controls and any custom apps that subscribe to the aggregated alarm count during run time.



The following **Alarm Adorner Options** are set through the dialog. The corresponding attribute settings are propagated to run-time nodes (all attribute names are prefixed by MyViewApp.Alarms). For additional information about these attributes, see *About Alarm Adorner Attributes* on page 399.

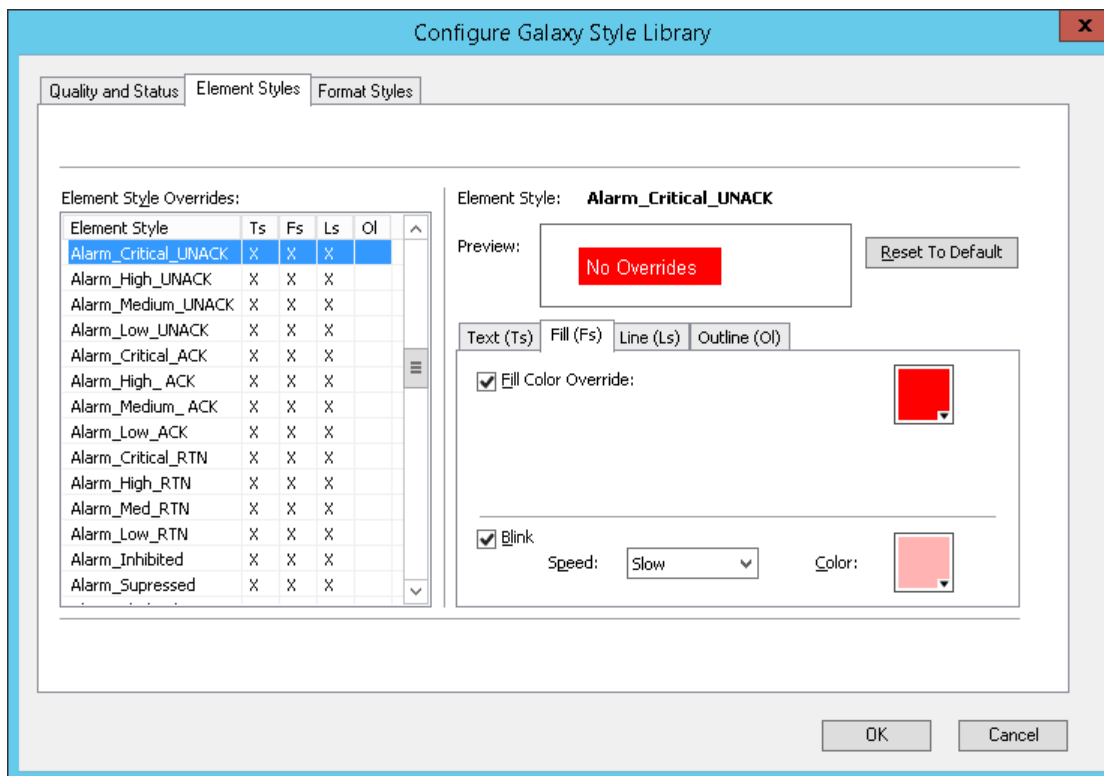
Alarm Severities to Display	Attribute Name (Alarms Namespace)	Description
Critical	ShowCriticalAlarms	When enabled, the ViewApp displays critical alarms
High	ShowHighAlarms	When enabled, the ViewApp displays high alarms
Medium	ShowMediumAlarms	When enabled, the ViewApp displays medium alarms
Low	ShowLowAlarms	When enabled, the ViewApp displays low alarms

Alarm States to Display	Attribute Name (Alarms Namespace)	Description
UNACK (Unacknowledged)	ShowUnacknowledgedAlarm	When enabled, the ViewApp displays UNACK alarms
ACK (Acknowledged)	ShowAcknowledgedAlarms	When enabled, the ViewApp displays ACK alarms

RTN (Return to Normal)	ShowReturnToNormalAlarms	When enabled, the ViewApp displays RTN alarms
------------------------	--------------------------	---

### Alarm Indicator Appearance

The appearance of alarm indicators is configured through the **Element Styles** tab of the **Galaxy Style Library** dialog, and uses the symbols that are shown in the **Alarms and Events Configuration** dialog.



### About Alarm Adorner Attributes

Alarm Adorner attributes are available only for use in AVEVA OMI ViewApps. All Alarm Adorner attributes operate in the MyViewApp.Alarms namespace. Attributes for the Alarm Adorner are specified with the MyViewApp.Alarms prefix in the form MyViewApp.Alarms.attribute\_name. Alarm Adorner attributes cannot be used in an InTouch application because the MyViewApp.Alarms prefix is regarded as a configuration error.

Attribute Name	Data Type	Attribute Type	Default Value	Description
ShowCriticalAlarms	Boolean	Read/ Write	True	If True, the ViewApp displays critical alarms
ShowHighAlarms	Boolean	Read/ Write	True	If True, the ViewApp displays high alarms
ShowMediumAlarms	Boolean	Read/ Write	True	If True, the ViewApp displays medium alarms
ShowLowAlarms	Boolean	Read/ Write	True	If True, the ViewApp displays low alarms

Attribute Name	Data Type	Attribute Type	Default Value	Description
ShowUnackedAlarms	Boolean	Read/ Write	True	If True, the ViewApp displays UNACK alarms
ShowAckedAlarms	Boolean	Read/ Write	True	If True, the ViewApp displays ACK alarms
ShowReturnToNormal Alarms	Boolean	Read/ Write	True	If True, the ViewApp displays RTN alarms

## Monitoring Alarm State Information at Run Time

You can obtain run-time alarm state information, including aggregated alarm information, using the Object Viewer or InTouch Tag Viewer. You can visualize the objects in an AVEVA OMI ViewApp or InTouch HMI application using the AlarmApp for AVEVA OMI or Alarm Client for InTouch HMI, along with the alarm border animation to represent the aggregated alarms at each level of containment. You can also enable aggregated alarm information for the NavTree

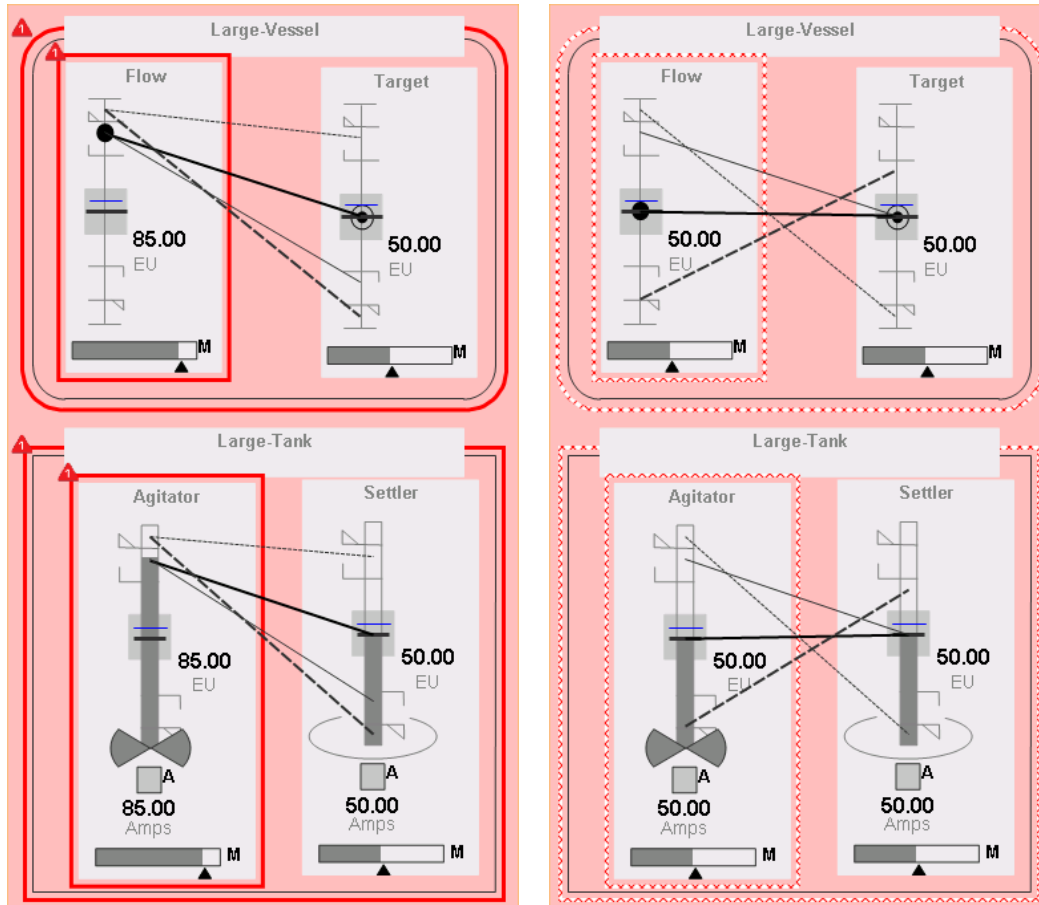
---

**Note:** AlarmApp is not supported in InTouch HMI applications. Similarly, AlarmClient is not supported in AVEVA OMI ViewApps. Use AlarmApp for AVEVA OMI ViewApps to allow alarm aggregation to follow selected objects and areas.

---

An alarm border animation applied to a group of symbols can indicate a priority 1 alarm with a red border, flashing or solid for different alarms. The alarm border animation can also indicate a return to normal (RTN) for the same symbols.





Alarm clients configured in InTouch applications will display the severity of each alarm in the **User1** field. You can change "User1" to a more descriptive heading, such as "Severity".

For information about configuring alarm animations, see "Configuring an Alarm Border Animation" in the *Creating and Managing Industrial Graphics User Guide*.



# CHAPTER 11

## Working with Multiple GR Nodes and Galaxies

This section describes how to pair Galaxy Repositories to configure multi-galaxy communication. This lets you connect to and configure remote galaxies, and use standard run-time tools such as the Galaxy Browser, Object Viewer, and ViewApps to browse and subscribe to attributes across multiple galaxies deployed across multiple GR nodes.

---

**Important!** If you are using an IDE node that includes a local GR to configure galaxies on remote GR nodes, System Platform enhanced security will not let you connect to a remote GR unless you pair to the GR first. You must configure multi-galaxy communication to allow the connection. See *Connecting to a Remote GR from the IDE* on page 435 for additional information.

---

### Understanding the Multi-Galaxy Environment

Multi-Galaxy communication lets you configure a scalable, distributed automation system that spans multiple galaxies. It also allows you use the IDE to work with galaxies on remote Galaxy Repository (GR) nodes. You can configure connectivity for multiple GRs, browse attributes in multiple Archestra namespaces, and subscribe to attributes at run time across remote GRs.

A basic multi-galaxy environment consists of an IDE-only node (no GR), and two or more GR-only nodes. The GR nodes should not have the IDE installed.

As of System Platform 2017, communication from a GR with a local IDE to a remote GR is blocked by default. See *Connecting to a Remote GR from the IDE* on page 435 to configure multi-galaxy communication between a GR with a local IDE and another GR node, with or without its own local IDE.

### Setting up Multi-Galaxy Communication

Multi-Galaxy communication (MGC) lets multiple GR nodes to talk to each other at runtime. MGC is also required for configuring remote GR nodes.

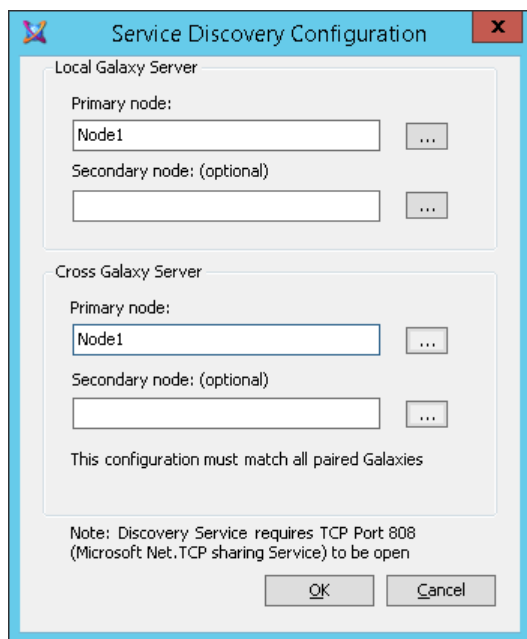
The functionality of tools such as Object Viewer, Galaxy Browser, ViewApps, and Managed InTouch applications is very similar in a multi-Galaxy environment as it is when working within a single galaxy. However, setting up the multi-galaxy environment requires some basic configuration.

The configuration workflow, using a two-galaxy pair as an example, is as follows:

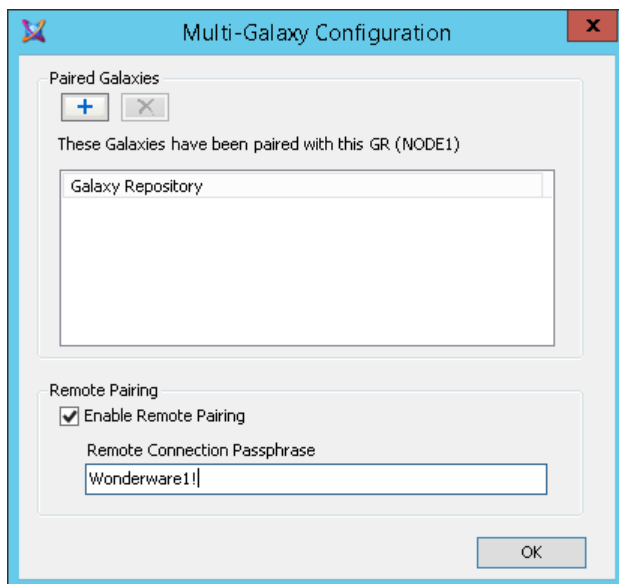
1. Configure a three node system.
  - IDE node 0 (no GR)
  - GR node 1 (no IDE): contains Galaxy 1
  - GR node 2 (no IDE): contains Galaxy 2
2. Prepare the multi-galaxy environment by adjusting galaxy names as necessary. Each galaxy in a multi-galaxy environment must have a unique name.
3. From the IDE, connect to Galaxy1 on Node1.
4. Configure Service Discovery for Galaxy 1 (**Galaxy > Configure > Service Discovery**).

Set **Node 1** as both the **Local Galaxy Server** and the **Cross Galaxy Server**. See *Defining Service Discovery* (see "*Defining Service Discovery*" on page 417) for additional information about the Local and Cross Galaxy Servers. You can configure secondary nodes for redundancy if needed.

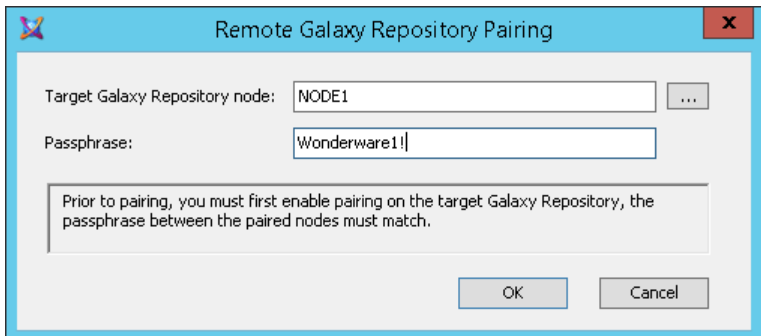
**Important!** There can be only one Cross Galaxy Server in a multi-galaxy environment. All GRs in the environment must point to the same Cross Galaxy Server node. You can specify nodes by name or by IP address.



5. Configure Multi-Galaxy for each Galaxy (**Galaxy > Configure > Multi Galaxy**).
  - a. Connect to **Galaxy 1** (on GR **Node 1**) and select the checkbox to enable Remote Pairing. Then, enter a passphrase. The passphrase must be the same for each galaxy to be paired, and must meet a minimum complexity standard:
    - Minimum length is nine characters, including at least eight alphanumeric characters (with at least one upper case and one lower case letter and at least one number)
    - At least one symbol: ! @ # \$ % ^ & \* ( )



- b. Click OK. A message that pairing has been enabled is displayed.
- c. Connect to **Galaxy 2** (on **GR Node 2**) and select Multi Galaxy (**Galaxy > Configure > Multi Galaxy**). Then, click the + sign under Paired Galaxies. The Remote Galaxy Repository Pairing dialog appears.



- d. Enter **Node 1** as the Target Galaxy Repository, and enter the same Passphrase as you entered for Galaxy 1.

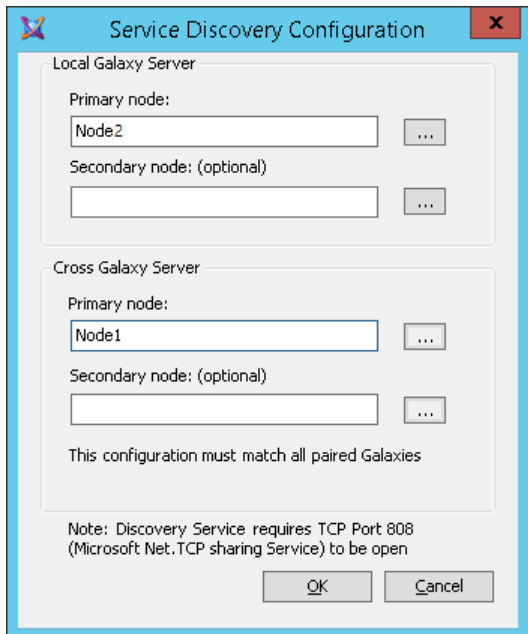
Node 1 is now listed as a paired Galaxy Repository in the **Multi-Galaxy Configuration** dialog.

### To change the Cross Galaxy Server

Use this procedure to change the Cross Galaxy Server for a previously paired Galaxy:

- 1. From the IDE, connect to the Galaxy on a remote node.
- 2. Configure Service Discover for the Galaxy (**Galaxy > Configure > Service Discovery**).

Set the node as the **Local Galaxy Server**. Set the new node as the **Cross Galaxy Server**. You can configure secondary nodes for redundancy if needed.



When pairing, the GR node initiating the pairing gets a copy of the primary Cross Galaxy Server node if none has been configured.

## Naming Galaxies in a Multi-Galaxy Environment

Setting up a multi-galaxy environment requires a unique name for each galaxy in the environment. This may require you to rename existing galaxies if you plan to include galaxies in your multi-galaxy environment that currently have the same name.

We recommend performing any necessary renaming operations prior to implementing the multi-galaxy environment.

For more information about creating and backing up galaxies, see *IDE Overview* on page 19 and *Managing Galaxies* on page 529.

For more information about renaming galaxies, see *Renaming Galaxies* on page 406.

---

**Important:** You must create a new galaxy with a new, unique name, from a backup .cab file rather than by creating a galaxy and performing a restore of the backup .cab file.

---

## Unpairing Galaxies

You can modify your multi-galaxy configuration by unpairing individual galaxies.

### To unpair galaxies

1. Click **Galaxy** on the IDE menu, then click **Configure**, then click **Multi Galaxy**.  
The **Multi-Galaxy Configuration** dialog box appears.
2. In the **Paired Galaxies** area, select the Galaxy Repository you want to unpair, then click the **Delete (x)** button.

Under certain conditions, the unpairing operation can fail. For more information, see *Troubleshooting a Remote Galaxy Connection* on page 429.

## Renaming Paired Nodes

Renaming a node that is paired with other nodes will disable communication with all other nodes to which the renamed node has been paired.

If a paired node is renamed, particularly if it is the designated Cross Galaxy Server, you must unpair all paired nodes and re-pair them using the new node name.

If you need to rename a node that is already paired, you should unpair the node first, then re-pair it after renaming it.

For information about naming Galaxies for use in a multi-galaxy environment, see *Naming Galaxies in a Multi-Galaxy Environment* on page 406.

## Renaming Galaxies

### To rename a galaxy for use in a multi-galaxy environment

1. Select a galaxy you want to rename, undeploy it and back it up to create a .cab file.
2. Use the .cab file as a "template" by placing it in one of the following directories, according to your operating system:
  - (32-bit)  
  \Program Files\ArcheStrA\Framework\Bin\BackupGalaxies
  - (64-bit)  
  \Program Files (x86)\ArcheStrA\Framework\Bin\ BackupGalaxies
3. Create a new galaxy based on the backup .cab file, with a new name. The name must be unique, not in use anywhere else in the multi-galaxy environment.

4. Repeat the preceding steps for each galaxy to be renamed with a unique name.
  5. Redeploy each newly created galaxy.
  6. Delete the original galaxy from the Galaxy Repository node.
  7. Upgrade to ASB-enabled Application Server using the latest System Platform installation program.
- Your galaxies can now be configured for use in a multi-galaxy environment.

## Accessing Multiple Galaxies

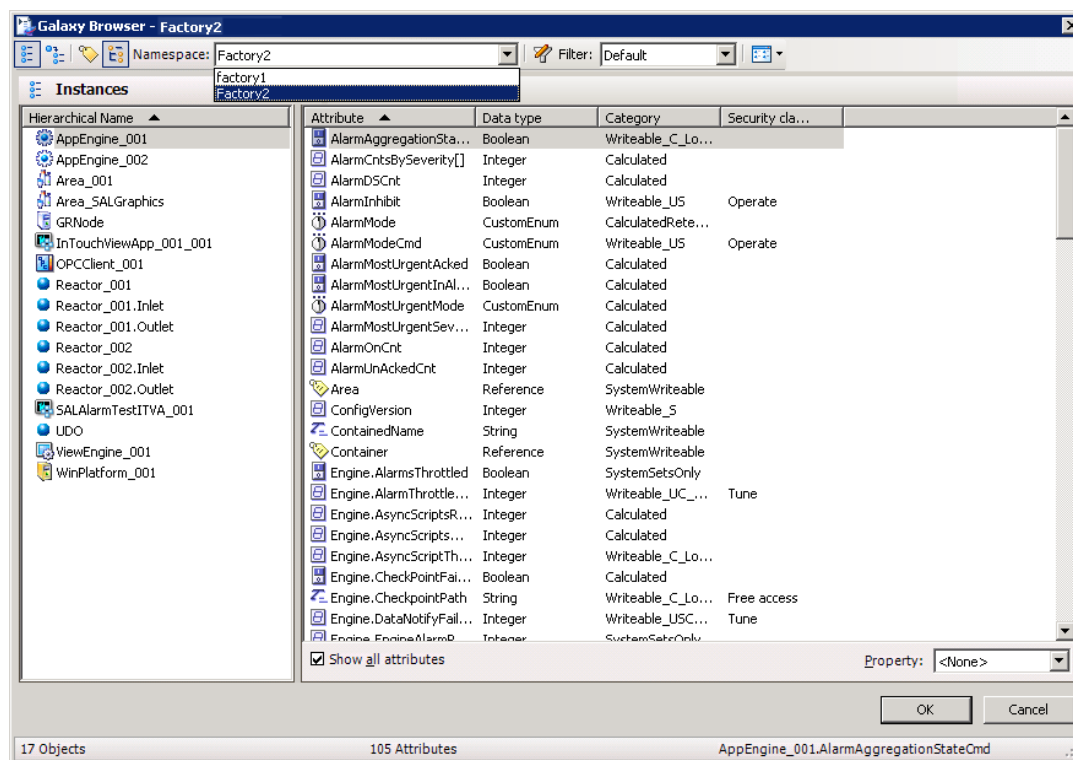
You can browse from one galaxy to another in a multi-galaxy environment through several methods.

- Use the Galaxy Browser, as described in *Reference Objects Using the Galaxy Browser* on page 88.
- Use the Object Viewer, as described in the *Object Viewer User's Guide*.
- Use the Archedra OPC UA Client Service, as described in *Defining User Configurable ASB Services*, and in *Browse with an Archedra OPC UA Client* on page 90.
- Browse galaxies while configuring and running Managed InTouch applications, as described in the *InTouch HMI Application Management and Extension Guide*, the *InTouch HMI Data Management Guide*, and other InTouch HMI user guides and online help.
- Browse galaxies while configuring and running AVEVA OMI View Applications, as described in the *AVEVA OMI Online Help*.

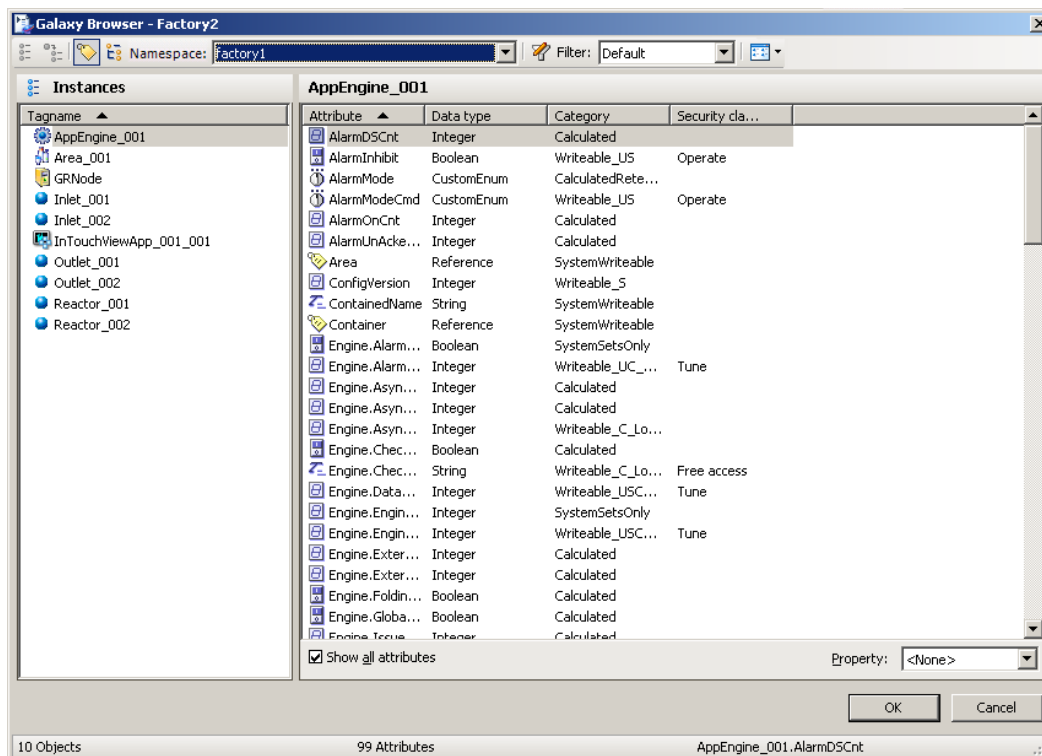
## Using the Galaxy Browser with Multiple Galaxies

Open the Galaxy Browser as usual to display the objects and attributes in your galaxy. For illustration purposes, this galaxy's name is Factory2 and you have paired it with Factory1.

The names of all paired galaxies will appear in the **Namespace** selection box. Select Factory2 to view the Tagnames and Attributes in Factory2.



Using the same instance of the Galaxy Browser, you can select Factory1 from the **Namespace** selection box. The Factory1 Tagnames and Attributes will appear.



You can work with the Factory1 Tagnames and Attributes in the same way as with those of Factory2.

For example, you are working in the Object Editor scripts tab, and you want to add a reference from Factory1 in a script. You can open the Galaxy Browser, select Factory1 in the **Galaxy Selection** box, select the Tagname "Outlet\_001", select the Attribute "udInt3", then click OK. The Attribute reference is inserted into the script as

```
Factory1:Outlet_001.udInt3
```

at your cursor location. This functionality is unchanged except for the addition of the GR node prefix to the Tagname and Attribute.

Using an instance of the ArchestrA OPC UA Client Service to browse a Galaxy namespace requires the use of the specific service instance name as a prefix.

For example, using an instance of the UA Client Service named "OPCUAClient\_010" to browse the preceding example would be addressed as follows:

```
OPCUAClient_010:Factory1:Outlet_001.udInt3
```

## Using Object Viewer with Multiple Galaxies

Open Object Viewer as usual to view data types, quality, values, timestamps, and status of the Application Objects, to perform diagnostic testing, and to modify selected attributes.

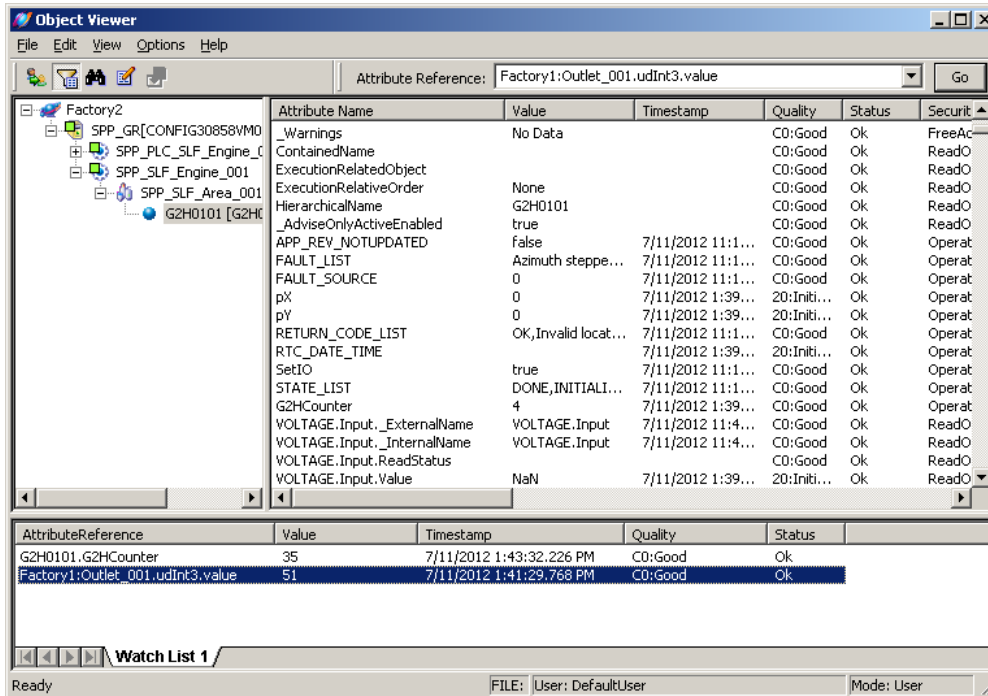
Object Viewer will display the console tree hierarchy of the GR node on which you have opened it. In this example, open Object Viewer from the Factory2 GR node. The attribute names, values, timestamps, and other data will display for a selected object. Add and manage attribute references to the Watch Window from Factory2 as usual.

In this example, if you want to add the attribute reference "Outlet\_001.udInt3".value to the Watch Window, you must prefix it with the paired galaxy name as follows:



```
Factory1:Outlet_001.udInt3.value
```

Everything following the prefix is addressed as usual. Adding the attribute reference to the Watch Window is the same as usual. The attribute reference will appear in the Watch Window with the GR node name prefix.



Using an instance of the ArchestrA OPC UA Client Service in Object Viewer requires the use of the specific service instance name as a prefix.

For example, using an instance of the UA Client Service named "OPCUAClient\_010" in the preceding example would be addressed as follows:

```
OPCUAClient_010:Factory1:Outlet_001.udInt3.value
```

## Using InTouch with Multiple Galaxies

You can use an InTouch application as a client to access data in a multi-galaxy environment in the same way you would access data within a single, local galaxy.

You can use the InTouch Tag Browser to select an Application Server object as a tag source and browse the Galaxy database in a multi-galaxy environment in an almost identical fashion to the traditional browsing workflow described in the *InTouch HMI Data Management Guide*.

Application Server object attributes or attribute properties can be used in remote references or as items for InTouch I/O tags in a multi-galaxy environment.

The address syntax changes for use in a multi-galaxy environment. The following is an example of a Galaxy reference used within a single galaxy:

```
Galaxy:Pump1.PV
```

The following is the same reference with the same access name, but used in a multi-galaxy environment to target a remote galaxy named "Factory2", containing object Pump1 with attribute PV:

```
Galaxy:"Factory2:Pump1.PV"
```

The entire expression, enclosed in quotation marks, becomes an item name. This syntax can be used in Industrial Graphics .NET scripting, and in InTouch scripting.

Using an instance of the ArchedrA OPC UA Client Service in this scenario requires the use of the specific service name as a prefix.

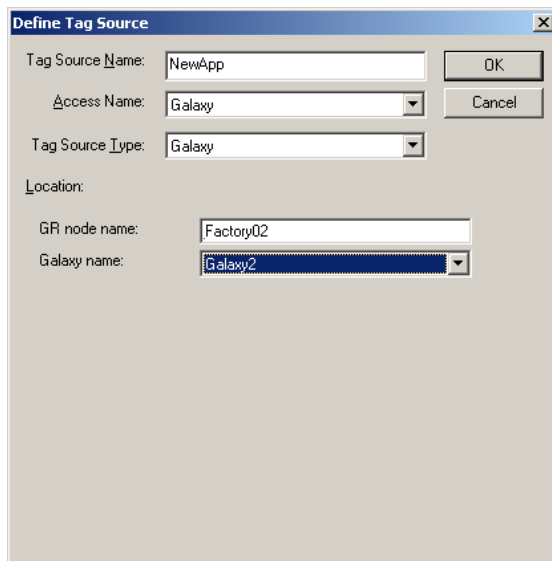
Scope Name/Service Name is specified in the ArchedrA UA Client service editor. This can be the identifier of the OPC UA server to which it is connecting.

For example, using an instance of the UA Client Service named "TestServer" would be addressed as follows:

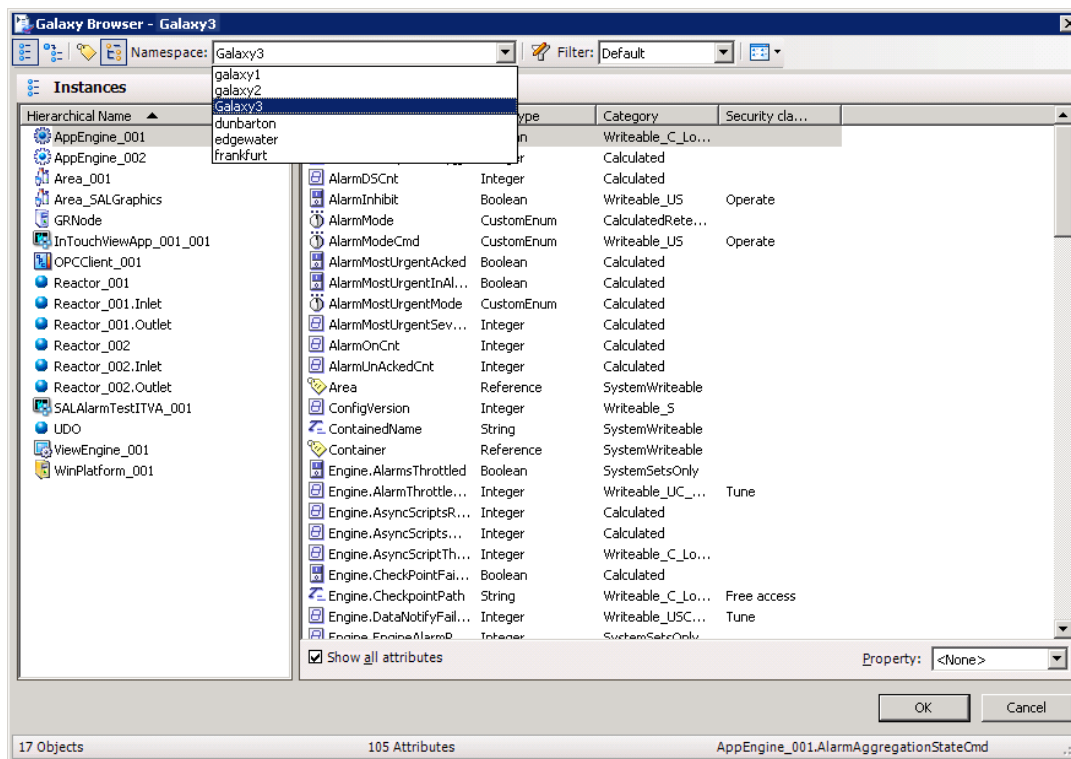
```
TestServer:Item1
```

**Important:** If you manually enter custom property overrides for an Industrial Graphic, do not enter the double quotation marks shown in the example. Browsing automatically provides the correct syntax, without the quotation marks, for custom property overrides. This syntax variation applies to custom property overrides only.

You can define a paired galaxy as a tag source, with an InTouchViewApp leveraging the galaxy pairing already configured in Application Server.



You can then use the Galaxy Browser to access the Galaxy namespaces of the paired galaxies.



## Guidelines for Accessing Multiple Galaxies

Working with multiple Galaxies is basically the same as working with a single Galaxy, but there are some important variations.

The general guidelines outlined in the sections that follow can help you to understand and work with those variations.

- *Working with Security in a Multi-Galaxy Environment* on page 411
- *Pairing Galaxies with Multiple Network Interface Cards* on page 412
- *Working with IDE Extensions on a GR Node* on page 412
- *Working with Alarms* on page 413

## Working with Security in a Multi-Galaxy Environment

The multi-Galaxy environment supports the OS User and OS Group ArchestrA Security models.

The Galaxy ArchestrA Security model is not supported:

In order for writes to succeed in a multi-galaxy environment:

- The logged-on user must be a member of the domain, either OS User or OS Group.
- Both galaxies in the pair must be configured to use the same security model.

A user with domain security credentials in a galaxy from which a write is initiated can perform writes, secured writes, and secured and verified writes, depending on configured privilege level—to a remote galaxy without logging on to the remote galaxy.

- Use InTouch as a client to write to attributes in a remote galaxy whether or not the user has logged on to InTouch.

- Use Object Viewer to write to attributes in a remote galaxy.
- Use the MxAccess client to write to attributes in a remote galaxy.

For more information about security, see *Configuring Security* on page 481.

## Pairing Galaxies with Multiple Network Interface Cards

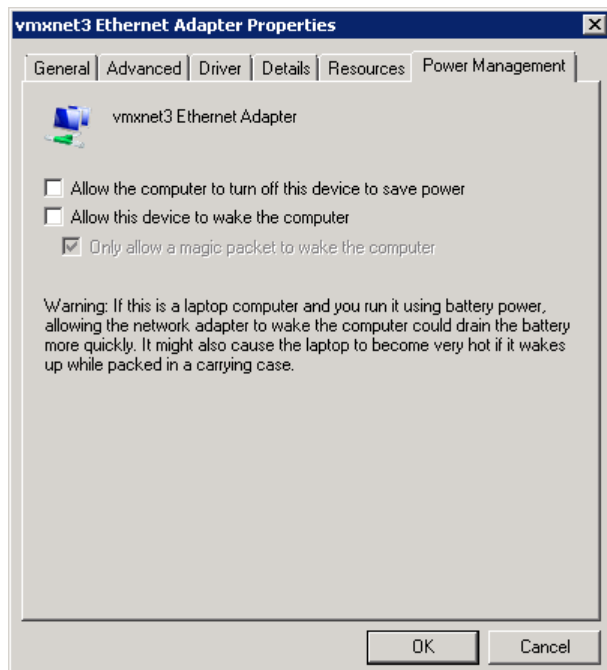
When using more than one Network Interface Card (NIC), galaxy pairing can fail if the NICs are not properly configured. For more information about configuring multiple NICs, see *Using Multiple Network Interface Cards* on page 539.

### Preventing ASB Services Disruptions

Windows includes a power management option to place a NIC into sleep mode to save power during periods of low activity. When a NIC is placed into sleep mode, ASB services are disrupted and multi-Galaxy communication is no longer possible.

#### To prevent ASB Services disruptions

1. On the computer hosting a multi-Galaxy environment, click **Start** and select **Control Panel**.
2. Select **Device Manager** to show the list of installed hardware devices.
3. Click **Network Adapters** to show a list of installed NICs on the computer.
4. Right-click on a listed NIC and select **Properties** from the shortcut menu.
5. Click the **Power Management** tab.
6. Clear the **Allow the computer to turn off this device to save power** option.



7. Click **OK**.
8. Clear this option for all nodes in your multi-galaxy environment.

## Working with IDE Extensions on a GR Node

Multi-Galaxy configuration options are either limited or are not available in the following scenarios:

- **IDE running on the GR node**

When you run the IDE on the GR node, you can only configure multi-galaxy communication for the GR on the same node. From the GR node, you can connect the IDE to a different GR to perform other configuration operations, but any attempts to perform multi-galaxy configuration will result in an informational message that the functionality is not allowed.

- **IDE running on a deployed platform node**

When you run the IDE on a deployed platform node, you can only configure multi-galaxy communication for the GR the platform is deployed from. From the platform node, you can connect the IDE to a different GR to perform other configuration operations, but the multi-galaxy configuration dialogs will be disabled.

- **Multiple instances of the IDE on a stand-alone node**

On a stand-alone node, if you or other users run more than one instance of the IDE, multi-galaxy can only be configured for the GR to which the IDE last connected.

The IDE node automatically pairs with the GR it connects to. While an IDE is connected to a GR node—and paired with it—if another IDE connects to a different GR on the same computer, the IDE node pairs with the new GR. The previously running IDE will continue to function, but multi-galaxy configuration dialogs will be disabled.

By default, installing Application Server deploys the ASBGRBrowsing service. Other user-defined services are not deployed during installation. When you deploy the platform, the ASBMxDataProvider and the ASBAuthentication services automatically deploy. These are default instances which you can modify. You can configure and deploy new instances of each of these services.

---

**Important:** The multi-galaxy environment does not support buffered data.

---

## Working with Alarms

All of the available alarm controls are unchanged, and function in a multi-galaxy environment as described in their individual user guides and online help. See the user guides for information on how to configure the various controls, how to query alarms, and how to retrieve alarm information.

- *ArchestrA Alarm Control Guide*
- InTouch HMI Alarm Viewer Control online help
- InTouch HMI Alarm DB View Control online help.

The alarm controls offer varied results and functional limits in a multi-galaxy environment.

### ArchestrA Alarm Control Function and Results

- Can take several seconds to display the **Alarm Statistics** dialog box with tens of alarm providers. Can take longer with 30–60 alarm providers.
- Can subscribe (query and retrieve) up to 20,000 alarms.
- Has an historical alarms limit for the maximum number of records of 32,766. Can show more than 32,766 in historical mode.
- Consumed high CPU resources at 200 alarms-per-second from 4 providers. During that time, might not update in a timely manner.

### Alarm Viewer Control Function and Results

- Typically requires one second to display the Alarm Statistics dialog box under the same conditions as the ArchestrA Alarm Control.
- Can subscribe (query and retrieve) more than 20,000 alarms.
- Can refresh the alarm list every second.

## Alarm DB View Control Function and Results

- Alarm queries are limited to 1,958 characters. Larger queries are truncated.
- Can generate alarms from 4 providers at 20 alarms per second, then will start to buffer alarms.
- Logger can store 20,000 entries, then the cache is full.

We recommend using the ArchestrA Alarm Control under the limits specified. In excess of 20,000 alarms, 20 alarm providers, and 100 alarms per second from multiple providers, we recommend using the Alarm Viewer Control.

## Enhancing Multi-Galaxy Performance

Multi-galaxy communication can compound the volume of run-time subscriptions and data changes, potentially impacting system resources. You can manage potential performance degradation by creating and deploying additional ArchestrA Service instances, particularly the ASBMxDataProvider service.

Browsing operations typically place less stress on system resources. Operations tend to be fewer in number than run-time subscriptions and are performed sequentially rather than in volume, in parallel. The ASBGRBrowsing service can be instantiated and deployed to remedy any browsing performance issues.

Operations requiring authentication, such as secured writes, also tend to place less stress on system resources than run-time subscriptions. Performance could become a factor in some scenarios, such as acknowledging a large volume of alarms where the alarm acks require secured writes. The ASBAuthentication service can be instantiated and deployed to remedy any performance issues with secured writes or other operations requiring authentication.

You can take one of two general approaches to multi-galaxy performance: remedy performance issues after the fact, or plan the multi-galaxy environment before the fact. We recommend scaling the multi-galaxy environment before the fact, particularly in large systems.

## Remedying Performance Issues After the Fact

Remedying performance issues as they occur involves monitoring subscriptions for data change rates. If your system begins to slow down or starts to show communication failures or bad quality data on subscribed attributes, do the following:

- Create a new ASBMxDataProvider service instance on the publishing side
- Configure the new service instance
- Assign the new service instance to a node and deploy it

Perform the operation on any node in the multi-galaxy environment exhibiting performance issues.

In a smaller multi-galaxy environment, or when you are just starting to implement multi-galaxy communication, this approach can have the advantage of saving up front configuration time, and can also serve as a trial period to gather metrics about your system.

The disadvantages of this approach are that you can lose time dealing with a system that is slowing down at run time.

## Scaling the Multi-Galaxy Environment for Optimum Performance

We recommend at least general advance planning of service and node deployment.

As a general guideline, a single instance of the ASBMxDataProvider service can subscribe up to 20,000 data changes per second. Beyond that number, you should create and deploy a new ASBMxDataProvider service on the publishing side.

An object running under an AppEngine subscribing to remote attributes or an InTouchViewApp, as a general guideline, can support up to 20,000 data updates per second from remote galaxies. Beyond that number, you should create and deploy a new ASBMxDataProvider service on the publishing side.

You can assign the new service instances to the same node as existing service instances provided each instance has a unique port number.

You can assign new nodes and deploy the new service instances to those nodes. In that scenario, a new service instance can have the same port number as an existing service instance on a different node.

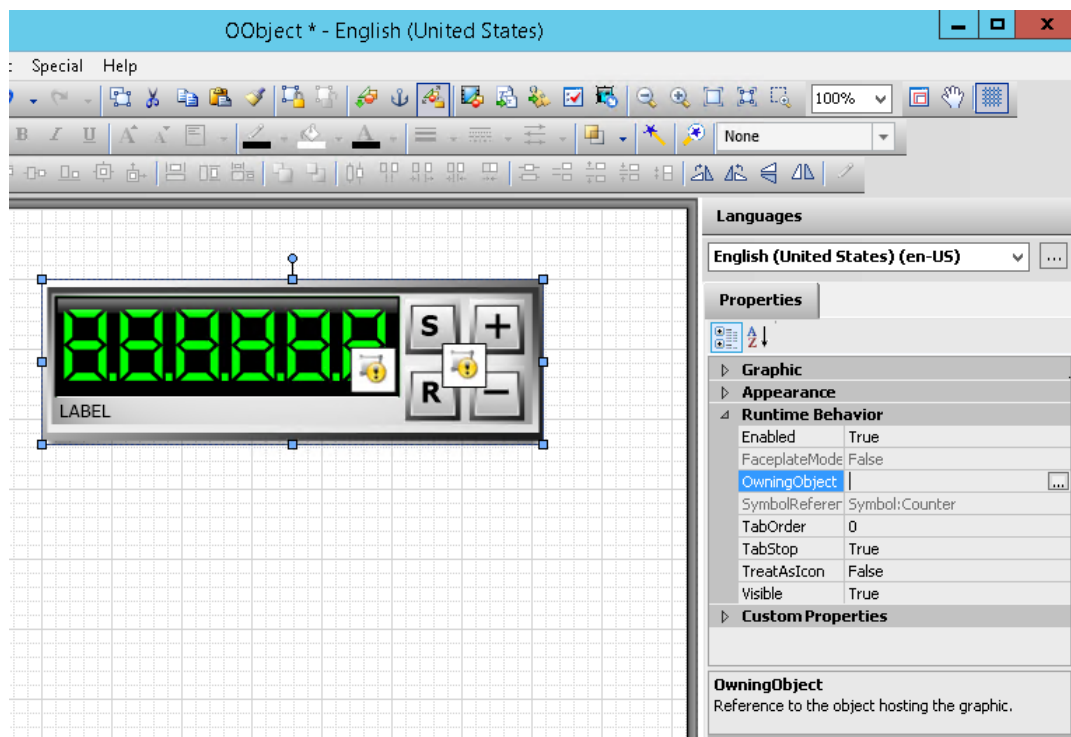
## Setting Remote Galaxy References in the Owing Object Property

You can assign a remote galaxy reference to the owning object property of an embedded graphic during configuration through the Graphic Editor.

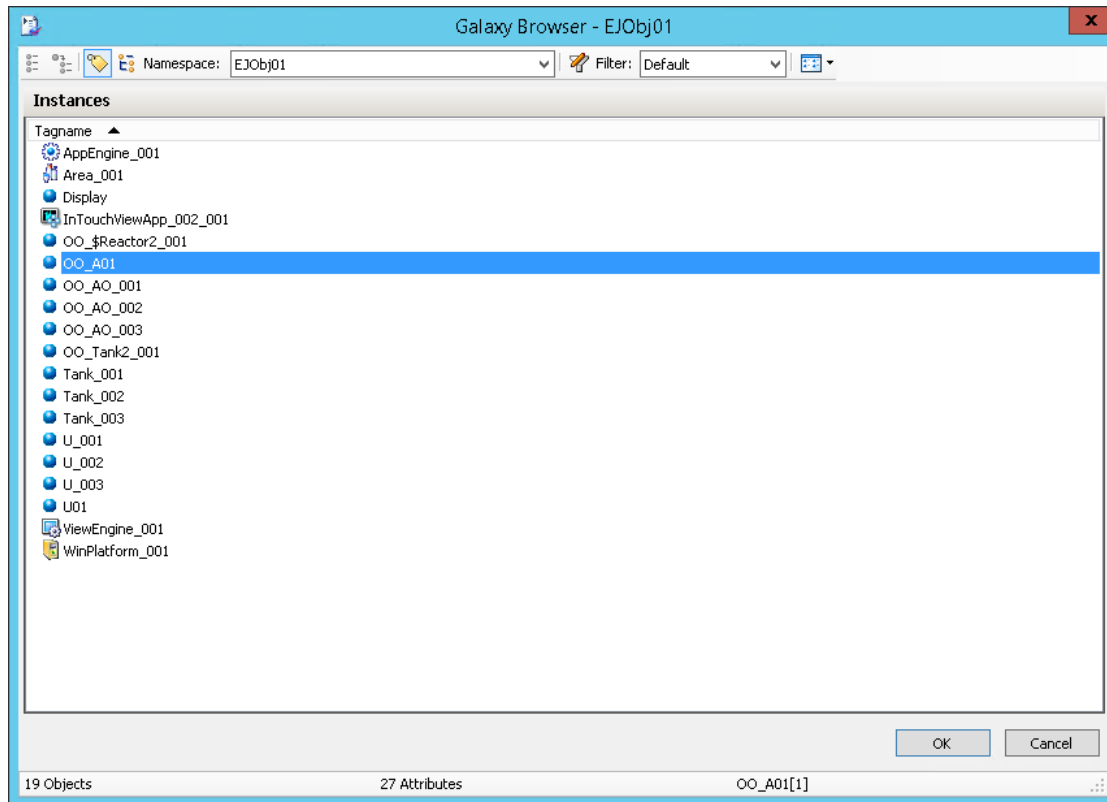
If needed, you can use scripting to change the owning object property during run time.

### To configure the owning object property in the Graphic Editor

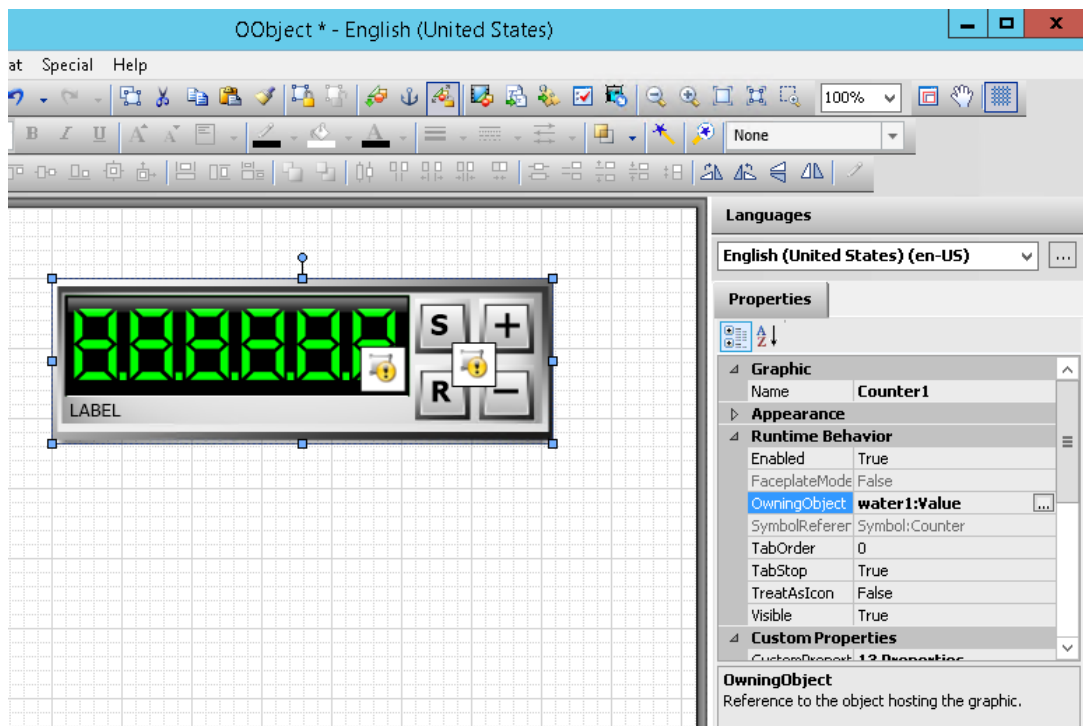
1. Launch the Graphic editor and embed a graphic. Select the graphic.
2. In the **Properties** pane, under **Runtime Behavior**, select the **Owning Object** property. You can set the owning object property through the Galaxy browser, or by typing the object reference in the field. Object references use the following format: *RemoteGalaxyName:ObjectName*.



- To use the Galaxy Browser to set the owning object (instead of typing the reference), click the **ellipsis** button. This launches the browser.



- From the Galaxy Browser, select the remote galaxy from the **Namespace** list of galaxy names. The objects contained in the galaxy are listed by tagname.
- Select the object you want to set as the owing object and click **OK**.





## Working with ArcestrA Services

The ArcestrA Services provide tools for multi-galaxy communication scalability and performance. For example, you can create additional instances of the ASBMxDataProvider Service if your run-time attribute subscription volume increases to a point where it begins to impact performance.

One default instance of each service is provided during the System Platform installation. You can rename and modify these instances and you can create additional instances.

## About the ArcestrA Service Bus

The ArcestrA Service Bus (ASB) is a framework that hosts services based on general principles of a service-oriented architecture (SOA).

- SOA provides a mesh of interoperable services that can be used within a single system or extended to external systems based on an established trust relationship between the systems.
- SOA also generally provides a way for services to discover other available SOA-based services.

ASB components are installed with Application Server as part of the System Platform installation. ASB functionality appears as menu items in the IDE. For information about the ASB components in the IDE, see *Setting Up a Multi-Galaxy Environment*.

ASB components include a set of core run-time services, a catalog of user-configurable services, and a scheme that enables connectivity, interoperability and exchange of data among the services as well as with internal and external applications.

## Defining Service Discovery

The discovery servers provide an infrastructure for ASB services to find and communicate with one another, either on the same local node or on a remote node. Multi-Galaxy communication uses three tiers of discovery servers.

- **Local Discovery Server**

The Local Discovery Server provides discovery for a single, local node. All services register with the Local Discovery Server, and all clients query it. It is not user configurable.

- **Local Galaxy Server**

The scope of the Local Galaxy Server encompasses the services in an entire galaxy. You can designate primary and secondary servers running on different nodes for redundancy.

At least one Local Galaxy Server must be configured and online to enable multi-galaxy communication.

- **Cross Galaxy Server**

The Cross Galaxy Server encompasses the services available across multiple galaxies. You can designate primary and secondary servers running on different nodes for redundancy. In any multi-galaxy environment, there can be only one designated primary and one designated secondary Cross Galaxy Server.

The Cross Galaxy Server can reside on any node in the multi-galaxy environment, but all galaxies in the environment must configure the Cross Galaxy Server to run on the same node.

## Defining User-Configurable ASB Services

The following user-configurable services are installed with Application Server as part of the ArcestrA Service Bus (ASB). You can begin configuring your ASB services with the default instances of each service provided as part of the installation.

You must configure at least one instance of the OPC UA Service to establish OPC UA client/server machine to machine communication within Application Server.

- **ASBMxDataProvider Service**

The ASBMxDataProvider service connects to ArchestrA run-time components and provides access to run-time data.

- **ASBGalaxyBrowsing Service**

The ASBGalaxyBrowsing service connects to the ArchestrA namespace to enable browsing of ArchestrA objects and attributes.

- **ASBAuthentication Service**

Users are authenticated against Active Directory user accounts. You can configure more than one ASBAuthentication service instance to point to different domain servers as needed.

- **OPCUAService**

The AVEVA OPC UA Service is an OPC UA server service. When you deploy this service and complete additional, required OPC UA configuration tasks, you can use a third-party OPC UA server to browse objects and attributes, and to connect, subscribe to, and retrieve data natively, that is, via an OPC UA Server, without the need for a gateway or other protocol translation mechanism. See *Configuring and Using the OPC UA Server* on page 297 for more information.

The following user-configurable services are not included with Application Server but can be installed separately. Installing these services adds default instances of each service, which you can then configure as needed.

- **Event Service**

The Event service is installed separately and is used to raise, monitor, and route events. The Event Service is not required to enable or configure multi-galaxy communication.

- **EventHistorization Service**

The EventHistorization service is installed separately and is used to store events in a configured database. It is not required to enable or configure multi-galaxy communication.

The EventHistorization service stores events in a configured database. It is not required to enable or configure multi-galaxy communication.

- **ASBOPCUAClient Service**

The ArchestrA OPC UA Client Service is supplied as a separately installed service in the ASB infrastructure, and provides OPC UA data access connectivity to OPC UA servers. Multiple instances of the OPC UA Client service are supported within the same ArchestrA Galaxy.

---

**Note:** Close the IDE before uninstalling the OPC UA Client Service. If you uninstall the OPC UA Client Service with the IDE running, the service will not function properly when you re-install it.

---

## Configuring and Deploying ArchestrA Services

The configuration tool and workflow are the same for all user-configurable ArchestrA Services. To configure an ArchestrA Service, you will need to perform the following tasks:

1. Create an instance of an ASB service
2. Assign the service instance to a node
3. Deploy the instance

You can make multiple instances of a single service, and deploy them to multiple nodes. You can also deploy a single instance to multiple nodes. This approach helps reduce service management. On the same machine, you can have more than one service simply by creating new instances.

## Creating an Instance of an ASB Service

You must create an instance of an ArcestrA Service (for example, ASBGRBrowsing Service) to configure the service.

## Assigning the Service Instance to a Node

After you create an instance of the ArcestrA Service, you must assign the instance to the desired node. You must check out the instance before assigning it to a node. You must check in the instance after assigning it to a node.

---

**Note:** For the ASBGRBrowsingService, you can also specify the name of the Galaxy to assign an instance of the service.

---

## Deploying the Instance

After assigning a service instance to a node, you must deploy the instance. A deployment confirmation message appears.

For information about configuring specific ArcestrA Services, see the following topics:

- *Configuring the ASBGRBrowsing Service* on page 421
- *Configuring the ASBMxDataProvider Service* on page 421
- *Configuring the ASBAuthentication Service* on page 422

## Configuring ArcestrA Service TCP Ports

---

**Important:** As of Windows System Platform 2014 R2, there is no need to configure any TCP ports. All ArcestrA Service Bus communication is based on the Microsoft WCF shared port 808. This information is retained for existing applications built with a version of Windows System Platform prior to the 2014 R2 release that have been assigned custom port numbers.

---

To configure ports for OPC UA clients and servers, the following requirements apply:

- On the node running the ASBOPCUAClient service, you need to open two ports, one for IData and one for IBrowse.
- On a node running an OPC UA server, you need to open two ports, one for the OPC UA server and one for the OPC UA discovery service, if configured.

See the following table for more information about OPC UA port configuration.

The ASB Core Services, installed with the ASB and started by the Watchdog Service, also use specific ports. The following table lists the default TCP ports assigned to the use r-configurable ASB Services and the ports in use for the ASB Core Services.

- The ASB Core Services port numbers should not be used to configure instances of user services.
- The firewall must allow incoming connections to the ports in the following table for their corresponding services to function.

<b>Arche strA Service</b>	<b>Port Numbers in Use</b>
<b>User-configurable ASB Services</b>	
ASBGRBrowsing Service	7500, 7501 (default) See note following this table.
ASBMxDataProvider Service	3572 (default)
ASBAuthentication Service	7779 (default)
Event Service	3575 (default, configurable)
EventHistorization Service	3586 (default, configurable)
ASBOPCUAClient Service	8666, 4600 (default, configurable) IData and IBrowse ports can be auto-assigned or user-assigned.
OPC UA Service	48031 (default, configurable) Endpoint for Application Server OPC UA Server instance
OPC UA Discovery Service	4840 (OPC Foundation service - not an ASB service)
<b>ASB Core Services-Not configurable</b>	
Local Discovery Server	9111
Primary Local Galaxy Server	9110
Secondary Local Galaxy Server	9210
Primary Cross Galaxy Server	9310
Secondary Cross Galaxy Server	9410
Galaxy Pairing	7085
Configuration Service	6332
Content Provider Service	6011
Deploy Agent Service	6533, 6633
Service Manager Service	6111, 6113
System Authentication Service	9876
aaServiceASBSoftwareUpdate	7587

**Important:** If a Galaxy Repository (GR) has more than one Galaxy, two additional ports must be opened to enable a remote GR to browse to each additional galaxy. For example, two galaxies would require ports 7500, 7501, 7502, and 7503 to be open. Three galaxies would require ports 7500-7505 to be open.

## Configuring the ASBGRBrowsing Service

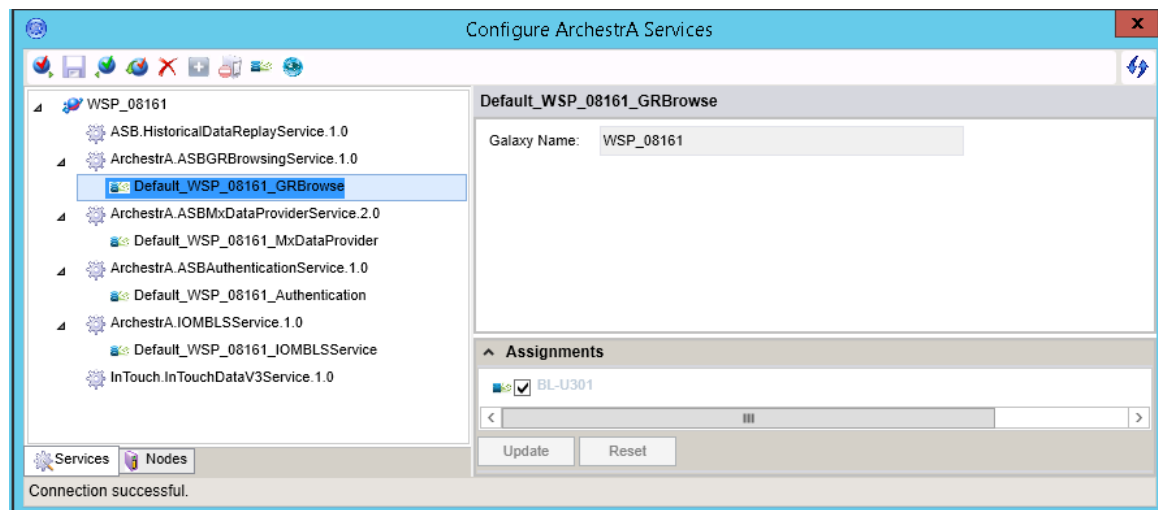
The ASBGRBrowsingService enables you to browse Application Server namespaces.

**Important:** The ASBGRBrowsing service is configured with one default service instance named "Default\_<GalaxyName>\_GRBrowse". Do not change the name of this default service instance or delete this service instance. The GRBrowsing service will stop working when you undeploy the GR Platform. If the default service instance has been renamed, revert the service instance name back to the default format. If the default service instance has been deleted, create a new instance with the correctly formatted default name.

### To configure the ASBGRBrowsing Service

1. In the IDE, click **Galaxy** on the menu, then click **Configure**, then click **ArchestrA Services**.

The **Configure ArchestrA Services** utility opens.



2. Right-click **ASBGRBrowsingService**, and then click **Create**. You can also type CTRL+N. The new instance appears in the tree structure.
3. Right-click the instance name and click **Check-out**
4. Enter the Galaxy name in the **Galaxy Name** box.
5. In the **Assignments** area, select the node you want to assign to the instance, and then click **Update**.

**Note:** The **Update** button is enabled only when you select a node. You cannot delete a node that is already assigned to a service instance. To delete a node, you must first unassign the node from the service instance and then delete it.

6. On the left-pane, right-click the instance, and then click **Check-in**.
7. On the left-pane, right-click the instance, and then select **Deploy**. A message appears indicating whether the node is successfully deployed.

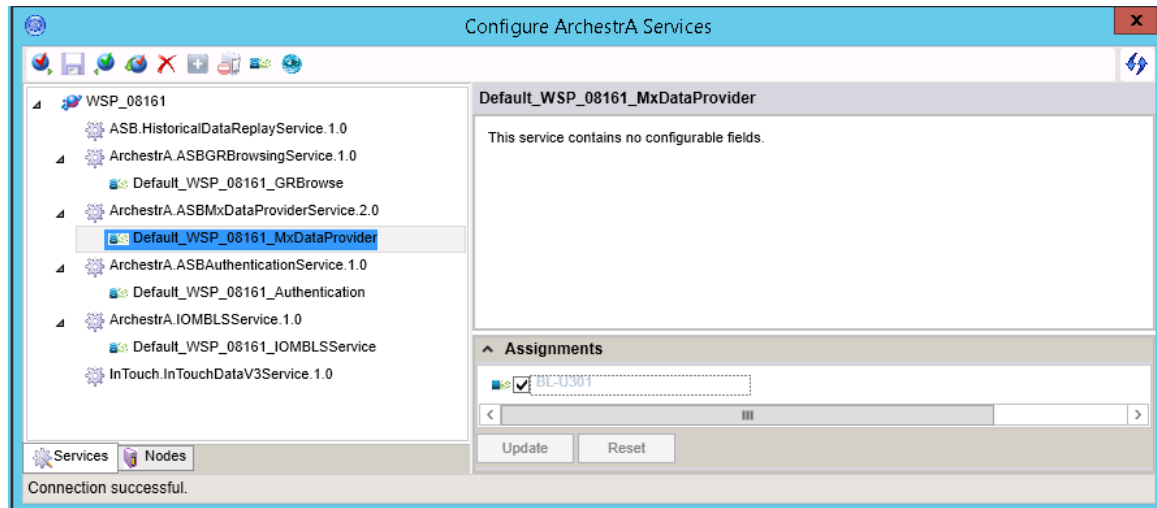
## Configuring the ASBMxDataProvider Service

The ASBMxDataProvider Service acts as a gateway providing access to Application Server data.

### To configure the ASBMxDataProvider Service

1. In the IDE, click **Galaxy** on the menu, then click **Configure**, then click **ArchestrA Services**.

The **Configure ArcestrA Services** utility opens.



2. Right-click **ASBMxDataProviderService**, and then click **Create**. You can also press CTRL+N. The new instance appears in the tree structure.
3. Right-click the instance name and click **Check-out**.
4. In the **Assignments** area, select the node you want to assign to the instance, and then click **Update**.
5. On the left-pane, right-click the instance, and then click **Check-in**.
6. On the left-pane, right-click the instance, and then select **Deploy**. A message appears indicating whether the node is successfully deployed.

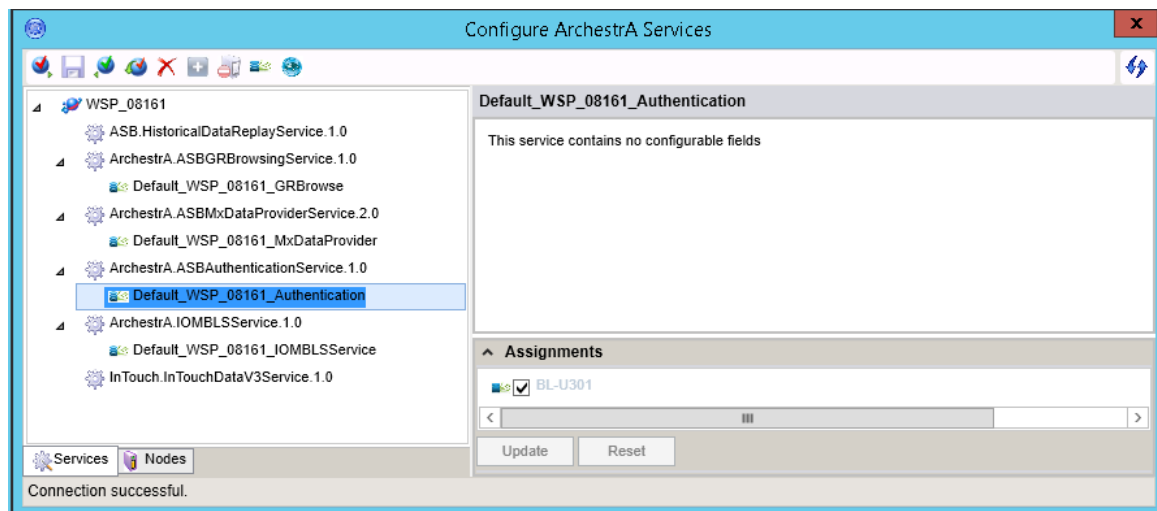
## Configuring the ASBAuthentication Service

The ASBAuthenticationService provides user authentication.

### To configure the ASBAuthentication Service

1. In the IDE, click **Galaxy** on the menu, then click **Configure**, then click **ArcestrA Services**.

The **Configure ArcestrA Services** utility opens.



2. Right-click **ASBAuthenticationService**, and then click **Create**. You can also press CTRL+N. The new instance appears in the tree structure.

3. Right-click the instance name and click **Check-out**
4. In the **Assignments** area, select the node you want to assign to the instance, and then click **Update**.
5. On the left-pane, right-click the instance, and then click **Check-in**.
6. On the left-pane, right-click the instance, and then select **Deploy**. A message at the bottom of the window appears indicating whether the node is successfully deployed.

**Note:** A run-time node will be offline if its hosted WinPlatform object is undeployed or if there is a pending software upgrade (SUP state). To change the status of the node to online and deploy new services, apply any required software upgrades and redeploy the WinPlatform object.

## IOMBLSService

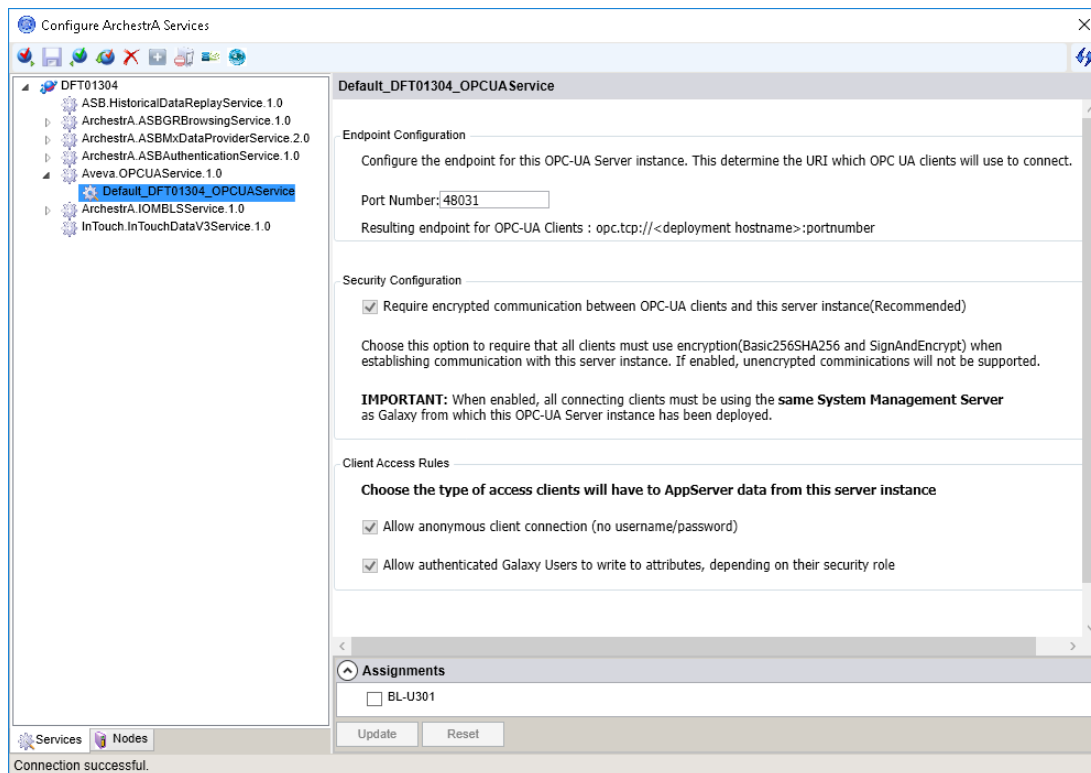
The IOMBLSService provides services for browsing, configuring, and deploying objects and visual elements. All the services provided by IOMBLSService are hosted in a single executable (aaSvcHost32.exe). IOMBLService is not user-configurable, and runs in the background. Do not delete, duplicate, close, or modify this service in any way.

## Configuring and Deploying the OPC UA Service

The Archestra OPC UA Service provides access from an OPC UA client to Application Server data, without the need for the Galaxy Browser, a gateway, or other protocol translation mechanism.

### To configure and deploy the OPC UA Server Service

1. In the IDE, click **Galaxy** on the menu bar, select **Configure**, then select **Archestra Services**.
2. The **Configure Archestra Services** utility opens.



3. Right-click the instance name and select **Check-out** from the context menu.
4. Edit the **port number** for the OPC UA server instance. The default port is **48031**.

5. **Security Configuration (require encrypted communication):** It is strongly recommended that you enable this option, as this will encrypt the payloads across the connection. Note that the client must match this configuration.

---

**Important!** An OPC UA connection cannot be established if you do not enable this option while OS security is enabled, even if the option "Allow authenticated Galaxy Users to write attributes" (Step 7) is enabled.

---

6. **Allow anonymous client connection:** Allowing anonymous client connections is recommended ONLY for initial setup configurations and testing. Anonymous client connections should be disabled for production environments.

If anonymous client connections are allowed *and* OS security is enabled for the Galaxy, anonymous client connections will be READ-ONLY.

Refer to *Client Access Rules and Galaxy Security* on page 300 to see the effect that this option has on data access in different scenarios.

7. **Allow authenticated Galaxy Users to write attributes:** Enabling this setting will only take effect when encrypted communication is also enabled (see Step 5, above). The OPC UA client must match this configuration.
8. The **Assignments** section (below the right pane) represents the platform nodes where the service configuration can be deployed. Select one or more runtime nodes where you want to deploy the OPC Server service, and then click **Update**.
9. Right-click the OPC UA Server service instance name and select **Check-in** from the context menu.
10. On the left-pane, right-click the instance, and then select **Deploy** from the context menu, or press CTRL+D. A message appears indicating whether the service has been successfully deployed to the OPC UA client node. If deployment is successful, the icon next to the instance name changes to indicate that the instance has deployed.

## To add additional OPC UA services

Each OPC UA service is dedicated to a single OPC UA client node. To add additional OPC UA services:

1. Right-click **AVEVA.OPCService**, and then select **Create** from the context menu, or press CTRL+N. The new instance appears in the tree structure.

---

**Note:** Each instance must have a unique port number. Enter the port number in the **Base Address** field. The default port number is **48031**. See *Configuring ArchestrA Service TCP Ports* on page 419 for a list of port numbers used by ASB services.

---

2. Rename the OPC UA service as needed. Right-click on the service name and select **Rename** from the context menu, or press F2. Then, enter the new name.
3. Repeat the steps above for configuring and deploying each additional OPC UA service.

## To change a deployed OPC UA service

1. Check out the service instance.
2. Make any needed changes.
  - **Port Number:** If you are creating multiple services, each service instance should have a unique port number. If more than one service has the same port number, an error is generated in the logger. Multiple instances of the service can be deployed, as long as each service has a unique port number. A new URI (uniform resource identifier) is automatically generated when a port number is changed.



**Note:** You may need to open the inbound port in the firewall to allow communication with the remote node.

- **Security Configuration:** When enabled (default), communication between OPC UA clients and the OPC UA server is encrypted. This is the recommended setting. If this setting is unchecked (disabled), communication is not encrypted.
- **Client Access Rules:**
  - When **Allow anonymous client connection** is enabled (default), an anonymous OPC UA client is allowed to connect to the OPC UA server. This is recommended only for testing and initial set up configurations. Once you have completed configuration and/or testing, be sure to disable this setting to provide protection against possible unwanted intrusions and to ensure that only authenticated users have access. Anonymous client connection should not be enabled in a production environment.

Galaxy Security settings do not have any affect on these behaviors. See *Configuring Security* on page 481 for more information.

- When **Allow authenticated Galaxy user to write to attributes** is enabled (default), an authenticated Galaxy user can change attribute values in run time, if their security role allows them to do so. See *About Roles* on page 485 for more information.

When **Allow authenticated Galaxy user to write to attributes** is unchecked (disabled), an authenticated Galaxy user is not permitted to change attribute values in run time, even if their security role allows them to do so.

- See *Client Access Rules and Galaxy Security* on page 300 for more information about user permissions for each setting combination.

3. Check in the service or services.
4. Undeploy and then redeploy the service or services.

## Client Access Rules and Galaxy Security

Client Access Rules configured for the OPC UA Service interact with the Galaxy security authentication mode to allow or deny different levels of access for authorized users.

There are two configurable Client Access Rules in the OPC UA Service dialog. By default, both rules are enabled:

- Allow anonymous client connection (no username/password)
- Allow authenticated Galaxy Users to write to attributes, depending on their security role

The following table defines the level of data access users are allowed under different combinations of Client Access Rule configurations, Galaxy security authentication mode, and the type of OPC UA credentials (anonymous or authenticated user with username/password).

If security for the Galaxy is enabled (Galaxy security = **Secured**, column 1), encrypted communication between the OPC UA clients and OPC UA service must also be enabled. See *Configuring and Deploying the OPC UA Service* on page 298.

Galaxy security authentication mode	OPC UA Client credentials	Client Access Rules		Level of Data Access		
		Allow anonymous connection	Allow authenticated Galaxy Users	Connect	Read	Write (see below)
Secured	Authenticated	Enabled	Enabled	YES	YES	YES

Galaxy	OPC UA Client	Client Access Rules		Level of Data Access		
Secured	Authenticated	Enabled	Disabled	YES	YES	NO
Secured	Authenticated	Disabled	Enabled	YES	YES	YES
Secured	Authenticated	Disabled	Disabled	YES	YES	NO
Secured	Anonymous	Enabled	Enabled	YES	YES	NO
Secured	Anonymous	Enabled	Disabled	YES	YES	NO
Secured	Anonymous	Disabled	Enabled	NO	N/A	N/A
Secured	Anonymous	Disabled	Disabled	NO	N/A	N/A
None	Authenticated	Enabled	Enabled	NO	N/A	N/A
None	Authenticated	Enabled	Disabled	NO	N/A	N/A
None	Authenticated	Disabled	Enabled	NO	N/A	N/A
None	Authenticated	Disabled	Disabled	NO	N/A	N/A
None	Anonymous	Enabled	Enabled	YES	YES	NO
None	Anonymous	Enabled	Disabled	YES	YES	NO
None	Anonymous	Disabled	Enabled	NO	N/A	N/A
None	Anonymous	Disabled	Disabled	NO	N/A	N/A

**Important:** Whenever Client Access Rules and Galaxy Security allow a user to write data, this permission is always conditioned by whether or not the user's configured security role also allows them to write data to a specific attribute. This means that when Galaxy security is enabled, the user's security role must explicitly allow them to write to attributes, regardless of the OPC UA client access rule setting. If their security role does not allow them to write to attributes, they cannot, even if the level of data access in the above table shows that they can.

## Managing Galaxies in a Multi-Galaxy Environment

The workflow for backing up and restoring a galaxy in a multi-galaxy environment is unchanged from those operations in a single-galaxy environment.

There are some important differences in behavior in galaxy management operations performed in a multi-galaxy environment:

- *Uninstalling and Reinstalling Application Server* on page 427
- *Backing Up a Galaxy* on page 427
- *Deleting a Galaxy* on page 427
- *Restoring a Galaxy* on page 427
- *Upgrading a Galaxy* on page 428
- *Migrating a Galaxy* on page 428
- *Upgrading SQL Server* on page 428

## Uninstalling and Reinstalling Application Server

When you uninstall Application Server, the ArcestrA Services database is not deleted from the system. This means that any existing galaxy pairings will persist when you reinstall Application Server on the same system. The persistent pairing information is automatically available after the reinstallation.

This pairing information, however, can be out of date and multi-galaxy operations not working as the Discovery Service information is lost in the transition from uninstall to install.

To resolve out-of-date pairing information, you must unpair and re-pair to the required galaxies after performing uninstall and re-install operations on the same node.

For example:

1. Install the latest Application Server software on two nodes, Node1 and Node2.
2. Create a galaxy with a platform, and pair the nodes.
3. On Node1, uninstall Application Server and restart the system.
4. On Node1, reinstall Application Server and connect to the same galaxy.
5. Open the **Multi-Galaxy Configuration** dialog. Notice that all the pairing information has persisted through the uninstall-reinstall operation. Also, notice that multi-galaxy operations are not functioning. You can't browse to or subscribe from Node2 even though the nodes appear to be paired.
6. Unpair and pair Node1 to get the new Discovery Service and pairing information from Node2.

## Backing Up a Galaxy

Use the Galaxy Database Manager in the SMC to perform a galaxy backup. The backup workflow is unchanged. There are minor differences in some details:

- The .cab file XML (\_GalaxyInfo.txt) has changed to include the backup node name.
- The file repository folder of the .cab file now contains a file, <gr node name><galaxy name>\_SR.csv, that lists all nodes and services configured for the galaxy whose backup the .cab file represents. The galaxy restore process uses this .csv file to recreate ASB nodes and services in the target galaxy.

You can back up and restore to a local node. You can back up and restore to a remote node.

---

**Note:** Galaxy pairing information is not included in the galaxy backup.

---

## Deleting a Galaxy

Deleting and restoring a galaxy requires that there be no clients connected to it. In a multi-galaxy environment, any deployed browsing service is considered a connection.

The following preparatory steps occur prior to initiating the galaxy delete operation:

1. Use the **Configure ArcestrA Services** editor to undeploy and delete all service instances. The ASB Service instances created for a galaxy are not deleted when the galaxy is deleted.
2. Use the **Configure ArcestrA Services** editor to undeploy all browsing services associated with the galaxy being deleted. If clients are connected, the delete operation is aborted and all browsing services that were previously undeployed are redeployed.

If the delete operation can proceed, all services, including the browsing services, are deleted.

## Restoring a Galaxy

The galaxy restore workflow remains unchanged. You can do the following restore operations:

- *Restoring to an Existing Galaxy* on page 428

- *Restoring to a New Galaxy* on page 428

## Restoring to an Existing Galaxy

If the existing galaxy to which you are restoring already has deployed browsing services, the browsing services are automatically undeployed as a first step in the restore process.

These browsing services are redeployed if the galaxy restore operation cannot proceed for any reason. Reasons for a galaxy restore operation being aborted:

- There are clients connected to the galaxy.
- The galaxy has a deployed platform.

At the end of the restore process, the service nodes and instances in the existing galaxy are deleted and replaced by new ones defined in the .cab file.

The node corresponding to the GR node in the backup replaces the GR node being restored.

The workflow for restoring Galaxy2 to Galaxy1 remains unchanged. Use the SMC to restore the galaxy, and target the existing galaxy, Galaxy1, to be replaced. The destination galaxy name will remain unchanged, even though the nodes will be replaced with what is defined in the .cab file.

## Restoring to a New Galaxy

Typically, you use the Galaxy Database Manager in the SMC to perform a restore operation. Specify the backup .cab file to be restored, and provide a galaxy name. You are restoring a backup, but the result is a new galaxy.

The exception to this workflow is when you must rename a galaxy to ensure that all galaxies in the multi-galaxy environment have unique names. For information about the renaming scenario and procedure, see *Naming Galaxies in a Multi-Galaxy Environment* on page 406.

## Upgrading a Galaxy

Back up your galaxies before performing an Application Server software upgrade.

When you restore a non-ASB galaxy from a backup to a later version of the software that includes the ArchastrA Services and ASB functionality, the restore process will set up the default services and extensions.

When you back up the upgraded galaxy, the backup .cab file XML will have the node name. The .csv file will contain the list of services and instances as they exist in the upgraded galaxy.

## Migrating a Galaxy

The migration workflow remains unchanged, but has an additional operation at the end of the migration process specifically for ASB.

Migrating from a non-ASB galaxy to an ASB-enabled galaxy, by default, does not create service instances and nodes during the migration process.

Creation of ASB service instances and nodes is deferred until the end of the migration process.

## Upgrading SQL Server

We recommend undeploying ASB services before you upgrade SQL Server on your computer.

### To upgrade SQL Server in an ASB services environment

1. Undeploy the galaxy.
2. On the menu bar, click **Galaxy**, then **Configure**, then **ArchastrA Services**. The **Configure ArchastrA Services** window appears.

3. Select the galaxy name in the left pane. Right-click and select **Undeploy** from the context menu, or click the **Undeploy** icon on the toolbar.
4. Verify that each service instance is undeployed, then close the configuration window.
5. Follow the relevant Microsoft instructions to install the new version of SQL Server.
6. When complete, you can deploy the ASB services, then deploy the galaxy.

## Troubleshooting a Remote Galaxy Connection

Informational and error messages remain the same in a multi-galaxy environment as they are for single-galaxies. The diagnostic tools already available, such as Object Viewer and the SMC Logger, continue to function in a multi-galaxy environment as they do in a single-galaxy environment.

Object Viewer displays communication or configuration errors, and displays data quality. The SMC Logger displays informational messages about system events.

---

**Important:** Watchdog and other ASB core services must be running whether or not you are operating in a multi-galaxy environment. Galaxy deletion, configuration, or run-time operations can fail if ASB services are not running, even when not operating in a multi-galaxy environment. See Local services in the following troubleshooting table for more information.

---

Indications of communication or configuration errors can include:

- Galaxies in a multi-galaxy environment do not appear in the **Galaxy:** list box in the **Galaxy Browser**.
- **Attribute References** in an Object Viewer watch list show Bad quality.
- **Attribute References** in an Object Viewer watch list show communication errors.
- Specifying an **Attribute Reference** in the Object Viewer text box returns a configuration error.
- Specifying an attribute reference in an InTouch application returns a configuration error or fails to return run-time data.
- A galaxy in the multi-galaxy environment does not appear in the **Browse Node** dialog box when attempting to select a Galaxy Repository with which to pair.
- You cannot connect to a galaxy in the multi-galaxy environment.
- You cannot undeploy or deploy an instance of an ArcestrA Service.
- Writes from InTouch WindowViewer fail when security is enabled.

When communication or configuration issues arise in a multi-galaxy environment, you can perform the following series of quick checks and remedies to determine if the issues have been caused by a multi-galaxy configuration error.

Check ...	Description	Procedure
<b>Local services</b>	<p>The ArchestrA Watchdog service is a utility to start all the ArchestrA Core Services. The Watchdog service starts automatically when you start your operating system.</p> <p>The Watchdog service and other ASB services are tightly coupled with Application Server configuration and run-time operations.</p> <ul style="list-style-type: none"> <li>• The Watchdog service must be running in order to create service instances and nodes, to deploy, undeploy, or delete user-defined ArchestrA services.</li> <li>• The Watchdog service must be running whether or not you are using a multi-galaxy environment.</li> </ul>	<p>Do not stop the Watchdog service even if you do not plan to use a multi-galaxy environment.</p> <p>These steps use the Windows 2008 R2 operating system for illustration.</p> <ol style="list-style-type: none"> <li>1. Click <b>Start</b>, then <b>Administrative Tools</b>, then <b>Component Services</b>. The <b>Component Services</b> utility opens.</li> <li>2. Click <b>Services (Local)</b> to populate the <b>Services (Local)</b> pane.</li> <li>3. Click <b>ArchestrA Watchdog Service</b> to view its state.</li> <li>4. If the Watchdog service is not running, click <b>Start the service</b>.</li> <li>5. Exit the <b>Component Services</b> utility.</li> </ol>
<b>Service Discovery configuration</b>	<p>By default, the Local Galaxy Server primary node is not configured (blank), and must be designated before you can enable galaxy pairing.</p> <p>By default, the Cross Galaxy Server primary node is not configured (blank), and must be set to the designated Cross Galaxy Server before you can enable galaxy pairing.</p> <p>There can be only one Cross Galaxy Server in a multi-galaxy environment. All galaxies in the environment must designate the same Cross Galaxy Server primary node.</p>	<p>Do these steps for each galaxy in the multi-galaxy environment for which you are having connectivity issues.</p> <ol style="list-style-type: none"> <li>1. In the IDE main menu, click <b>Galaxy</b>, then click <b>Configure</b>, then click <b>Service Discovery</b> to open the <b>Service Discovery Configuration</b> dialog box.</li> <li>2. Verify or configure the <b>Local Galaxy Server Primary node</b> as the local galaxy GR node.</li> <li>3. Verify or configure the Cross Galaxy Server Primary node as the node you have defined as the Cross Galaxy Server for all paired galaxies.</li> </ol>

---

Check ...	Description	Procedure
<b>Multi-Galaxy configuration</b>	Multi-Galaxy configuration establishes and maintains the trust relationship between paired galaxies for the duration of the pairing session.  Remote pairing must be enabled on each GR to be paired, each of which requires the same unique passphrase.	<ol style="list-style-type: none"><li>1. In the IDE main menu, click <b>Galaxy</b>, then click <b>Configure</b>, then click <b>Multi-Galaxy</b> to open the <b>Multi-Galaxy Configuration</b> dialog box.</li><li>2. Verify that the galaxy or galaxies to be paired are listed in the <b>Galaxy Repository</b> list box.<ol style="list-style-type: none"><li>a. If a particular galaxy is not listed, click the <b>Add</b> button to open the <b>Select Galaxy Repository to pair with</b> dialog box.</li><li>b. Enter the <b>Target Galaxy Repository node</b> in the text box.</li><li>c. Enter the <b>Passphrase</b> in the text box.</li></ol></li><li>3. If pairing fails, ensure that remote pairing is enabled on the target GR, and that you have the correct passphrase entered.</li></ol>

---

Check ...	Description	Procedure
<b>Arche strA Services configuration</b>	<p>The user-configurable Arche strA Services provide communication channels for attribute browsing and run-time data access.</p> <p>By default one instance of each service is configured and locally deployed.</p> <p>At least one instance of each service must be deployed on each galaxy in the multi-galaxy environment for multi-galaxy communication to function.</p>	<p>For more information about configuring Arche strA Services, see <i>Configuring and Deploying Arche strA Services</i> on page 418.</p> <p>For browsing issues, verify the ASBGRBrowsing service instance.</p> <p>For run-time data subscription issues, verify the ASBMxDataP rovider service instance on the publisher side, and ensure that the galaxy is deployed.</p> <p>For secured write issues, verify the ASBAuthentication service instance.</p>
	<p><b>Note:</b> When you set or change the Local Galaxy Server or the Cross Galaxy Server, the core ASB services automatically restart. This can create a timing issue if you immediately deploy a platform. The ASBMxDataP rovider and the ASBAuthentication services can fail to deploy. Run-time connectivity would then fail in the multi-galaxy environment.</p>	<ol style="list-style-type: none"> <li>1. In the IDE main menu, click <b>Galaxy</b>, then click <b>Configure</b>, then click <b>Arche strA Services</b> to open the <b>Configure Arche strA Services</b> dialog box.</li> <li>2. Expand the GR node in the hierarchy pane to view the installed services. Expand each service in the hierarchy to view the service instances.             <ol style="list-style-type: none"> <li>a. Select the instance to view its configuration in the configuration pane.</li> <li>b. Verify that the <b>Galaxy Name</b> is correct (ASBGRBrowsing service only).</li> <li>c. Verify that the service instance is assigned to a node, visible in the <b>Assignments</b> pane, and that the node is correct.</li> <li>d. Verify that the service instance is deployed.</li> </ol> </li> <li>3. Edit the service instance configurations as necessary.</li> </ol>



Check ...	Description	Procedure
<b>Client configuration</b>	<p>The Galaxy Browser, Object Viewer, and Managed InTouch applications can function as clients to browse attributes in the Arcestra namespace, or to subscribe to run-time data.</p> <p>Using these clients in a multi-galaxy environment is described in <i>Accessing Multiple Galaxies</i> on page 407.</p>	<p>For Object Viewer communication errors, review the Attribute Reference syntax. In particular, ensure that the correct &lt;GalaxyName&gt;: prefix is present and points to the correct galaxy.</p> <p>For Galaxy Browser errors, ensure that you have selected the correct galaxy in the <b>Galaxy:</b> list box.</p> <p>For Managed InTouch errors:</p> <ul style="list-style-type: none"> <li>• Ensure that you have selected the correct galaxy when accessing tagnames and attributes from the Galaxy Browser.</li> <li>• Ensure that syntax is correct, including the "Galaxy:" access name prefix, and the full attribute reference in quotes, with the remote galaxy prefix as part of the expression. Correct any incorrect syntax.</li> </ul>
		<hr/> <p><b>Note:</b> If you selected InTouch HMI <b>Development and Run Time</b> during installation, you cannot create a multi-galaxy environment. This installation does not include a GR, required for multi-galaxy functionality.</p> <hr/>

Check ...	Description	Procedure
<b>Security configuration</b>	<p>OS User with domain users only and OS Group (supports only domain users) are the only ArcestrA Security models supported in a multi-galaxy environment.</p> <p>Security issues, evident with failed writes, can occur for any of the following possible causes:</p> <ul style="list-style-type: none"> <li>• Security mode of galaxy is set to Galaxy Security</li> <li>• Security mode of InTouch is not set to ArcestrA</li> <li>• User has not logged into the remote Galaxy at least once</li> <li>• Security modes of local and remote galaxies don't match</li> <li>• User doesn't have sufficient permissions to perform the write</li> <li>• Default User Authentication service is not deployed on the GR node</li> </ul>	<p>For information about security configuration, see <i>Configuring Security</i> on page 481.</p> <p>In the IDE, set the security mode to <b>Galaxy Security</b>.</p> <p>In InTouch HMI, set the security mode to <b>ArcestrA</b>.</p> <p>In the multi-galaxy environment, set local and remote galaxy security to the same mode for each galaxy.</p> <p>In the multi-galaxy environment, log into each remote galaxy at least one time (for each user who requires access to those galaxies and has sufficient write permissions).</p> <p>In the Configure ArcestrA Services dialog, verify the ASBAuthentication service instance.</p> <ol style="list-style-type: none"> <li>1. Open the <b>Configure ArcestrA Services</b> dialog box.</li> <li>2. Expand the GR node in the hierarchy pane to view the installed services. Expand the each service in the hierarchy to view the service instances.             <ol style="list-style-type: none"> <li>a. Select the instance to view its configuration in the configuration pane.</li> <li>b. Verify that the service instance is assigned to a node, visible in the <b>Assignments</b> pane, and that the node is correct.</li> <li>c. Verify that the service instance is deployed.</li> </ol> </li> <li>3. Edit the service instance configurations as necessary.</li> </ol>

Check ...	Description	Procedure
<b>Unpairing galaxies</b>	<p>Unpairing galaxies requires that communication is established with the remote node to be unpaired. If communication with a remote galaxy is broken, unpairing with that node will fail.</p> <p>This condition can occur in different scenarios, including:</p> <ul style="list-style-type: none"> <li>• The remote node is down or not on the network.</li> <li>• The System Authentication service is not running.</li> <li>• The Watchdog service is not running</li> <li>• A mismatch or corruption has occurred in a security key.</li> </ul>	<p>When an unpair operation fails, you will see the <b>Un-pair operation with remote Galaxy Repository failed</b> error message and dialog box.</p> <p>This dialog box offers you a choice to force the unpair, or to wait until the remote GR is back online.</p> <ul style="list-style-type: none"> <li>• Click <b>Yes</b> to force the unpair.</li> </ul> <p>The forced unpairing occurs only on the local GR. When the remote GR is back online, any client running on the remote GR will be unaware that an unpairing has occurred, and will still be able to discover and connect to the service on the local GR from which you performed the failed unpairing operation.</p> <ul style="list-style-type: none"> <li>• Click <b>No</b> to end the unpairing operation and wait until the remote GR is back online.</li> </ul> <p>When you click <b>No</b>, the system takes no action, and pairing remains as configured.</p> <p>You can now troubleshoot why the remote GR is offline, then retry the unpairing operation.</p>

## Connecting to a Remote GR from the IDE

As of System Platform 2017, communication from a GR with a local IDE (single node with both IDE and GR) is blocked by default to a remote GR. This applies to using the IDE to configure a Galaxy on a remote GR, as well as runtime.

If the IDE you are running has a GR on the same node and you try to pair with a Galaxy on a remote GR, you will get an error message that communication with the GR node cannot be established.

### Scenario 1

The nodes have the following configuration:

- Node 1: IDE + GR (Galaxy 1)
- Node 2: GR only (Galaxy 2)
- Node 3: IDE only

### To establish communication with the blocked GR in Scenario 1

1. Open the IDE on Node 1 and connect to Galaxy 1.

2. Configure Service Discovery for Node 1, and set Node 1 as the Local Galaxy Server and Cross Galaxy Server (**Galaxy > Configure > Service Discovery**).
3. Open the IDE on Node 3 and connect to Galaxy 2.
4. Configure Service Discovery for Node 2, and set Node 2 as the Local Galaxy Server. Set Node 1 as the Cross Galaxy Server (**Galaxy > Configure > Service Discovery**).
5. Configure Multi-Galaxy for each Galaxy. Use the Node 1 IDE to configure Multi-Galaxy for Node 1, and the Node 3 IDE to configure Multi-Galaxy for Node 3.

Make sure that you use the same Passphrase for both nodes.

## Scenario 2

The nodes have the following configuration:

- Node 1: IDE + GR (Galaxy 1)
- Node 2: IDE + GR (Galaxy 2)

### To establish communication with the blocked GR in Scenario 2

1. Open the IDE on Node 1 and connect to Galaxy 1.
2. Configure Service Discovery for Node 1, and set Node 1 as the Local Galaxy Server and Cross Galaxy Server (**Galaxy > Configure > Service Discovery**).
3. Open the IDE on Node 2 and connect to Galaxy 2.
4. Configure Service Discovery for Node 2, and set Node 2 as the Local Galaxy Server. Set Node 1 as the Cross Galaxy Server (**Galaxy > Configure > Service Discovery**).
5. Configure Multi-Galaxy for each Galaxy. Use the Node 1 IDE to configure Multi-Galaxy for Node 1, and the Node 2 IDE to configure Multi-Galaxy for Node 2.

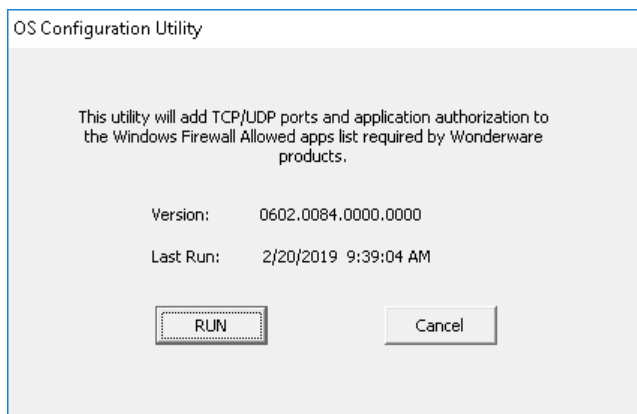
Make sure that you use the same Passphrase for both nodes.

## Fixing Communication Issues with the OS Configuration Utility

The **OS Configuration Utility** runs automatically as part of the System Platform installation process. In some cases, you may need to run the utility manually to correct communication issues within Application Server. Running the OS Configuration Utility may be necessary if you have problems running the IDE, connecting to a galaxy, or deploying a node. The full pathname of the OS Configuration Utility is (there is no start menu shortcut to the utility):

**<Root drive>\Program Files (x86)\Common Files\Arche strA\OSConfigurationUtility.exe**

The utility will open ports and change registry values to fix any communication issues it finds. Other than starting the utility, all updates are automatic. No user interaction is required or allowed, other than restarting the system after the utility has finished running.





# CHAPTER 12

## Working with Buffered Data

### About Buffered Data

Buffered data is data captured and stored locally on a remote device for later transfer to a supervisory system for processing, analysis, and long-term storage. The Buffer property is input-only.

Enabling buffered data alters the run-time behavior of some Application Server features such as History and Alarms. Buffering is essential if the Telemetry Server is installed.

The buffered data feature enables efficient accumulation and propagation of VTQ (Value, Time, and Quality) data updates, without folding and data loss, to data consumers such as objects, alarms, the Historian, and scripts from field devices that support buffering. If the Telemetry Server is used with Application Server, buffering must be enabled to ensure that all data from the Telemetry Server is received by Application Server, and then sent to the Historian in its entirety.

Enabling the buffered data feature preserves all intermediate data values. Thus, if an attribute receives different values during a single scan cycle, the data is not folded or compressed. Data folding occurs when an attribute value changes multiple times within a single scan, and only the latest value is stored. With buffered data enabled, the entire subset of values is propagated to data subscribers such as the alarm subsystem and the Historian, if the attribute is configured to be alarmed or historized. Scripts can also take advantage of processing buffered data. Since the Telemetry Server only sends a data request periodically, and often only once per day, enabling buffered data ensures that Application Server preserves all intermediate data values.

---

**Note:** If multiple attributes with I/O enabled are configured for buffered data from the same data point in the Telemetry Server, only one attribute will receive the buffered data at run time.

---

### Components that Use Buffered Data

The following table summarizes component functions related to buffered data subscription and processing.

---

#### Subscribing to the Buffer property

---

<b>Attribute</b>	User defined attributes with I/O configured to read or read/write data can support buffered data processing.
------------------	--

When buffered data is enabled, the attribute subscribes to the buffer property and receives the buffered data via Message Exchange. See *Using Message Exchange and Attributes* on page 453 for more information about Message Exchange.

<b>Scripts</b>	A script subscribes to an attribute and receives buffered data from it. A specific syntax is used to properly process buffered data in scripting. Scripts can also read the buffer from an attribute or directly from the DIOject.
----------------	--

For more information, see *Using ArchestrA Scripts to Process Buffered Data* on page 443.

---

---

### Processing buffered data

---

<b>Operations Integration (OI) Server</b>	Operations Integration (OI) Servers, and legacy Data Access (DA) Servers, collect and store VTQ data from field devices. OI Server and DAServer functionality is unchanged by the buffered data feature. Protocol interfaces within the servers allow buffered data from the field devices to be processed. The OI Servers and DAServers, in turn, can pass the buffered data as packets to the Application Engine.
<b>Telemetry Server</b>	The Telemetry Server collects and stores VTQ data from RTUs (remote terminal units) connected to field devices. The Telemetry Server typically only sends data to Application Server when polled. Since polling may be performed only periodically, there may be a large dataset of intermediate values that would be lost if buffering is not enabled.
<b>DIOBJECT</b>	For attributes with a buffer property that is being subscribed, the DIOBJECT collects VTQ values from the OI Server, and legacy DAServer, and packs the VTQ values into buffer collections, based on scan cycle completion or on end of data received.
<b>Application Engine</b>	The Application Engine hosts and executes the ApplicationObjects' logic.
<b>Data Consumers</b>	When buffered data is enabled, data consumers such as the Historian and alarms subsystems subscribe to the Buffer property of the configured attribute to receive published buffered data. The buffered data, a stream of VTQs, is assembled and appended in caches to avoid data folding, and is then published through Message Exchange to consumers such as the Historian, alarms and scripts. See <i>Using Message Exchange and Attributes</i> on page 453 for more information.

---

**Note:** The buffered data feature does not support out-of-sequence data. Out-of-sequence data occurs when the time stamps of the VTQ sequence for a specific reference is not arranged in incremental order from older to newer.

---

## Configuring Buffered Data for an Attribute

To configure buffered data for an attribute:

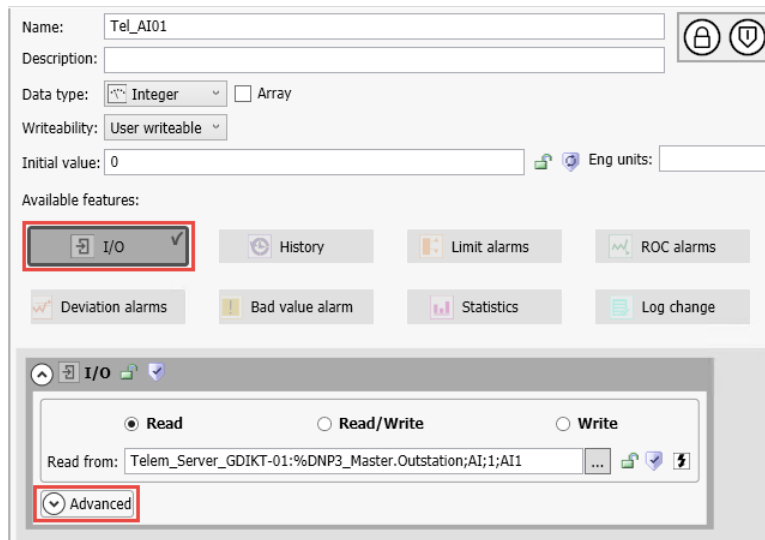
1. Open an object for editing and select the **Attributes** page.
2. Select an existing user-defined attribute or add one.

Buffered data can be configured for all attribute data types except InternationalizedString.

3. Enable the I/O feature for the attribute and select Read or Read/Write, as appropriate.



- Click the arrow to expand the **Advanced** I/O configuration options.



Name: Tel\_AI01

Description:

Data type: Integer  Array

Writeability: User writeable

Initial value: 0 Eng units:

Available features:

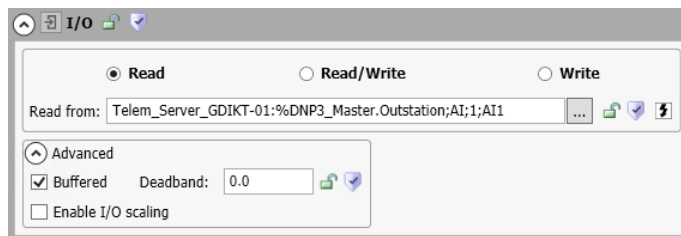
- I/O
- History
- Limit alarms
- ROC alarms
- Deviation alarms
- Bad value alarm
- Statistics
- Log change

Read  Read/Write  Write

Read from: Telem\_Server\_GDIKT-01:%DNP3\_Master.Outstation;AI;1;AI1

Advanced

- Select the checkbox next to **Buffered** to enable data buffering.



Read  Read/Write  Write

Read from: Telem\_Server\_GDIKT-01:%DNP3\_Master.Outstation;AI;1;AI1

Buffered Deadband: 0.0

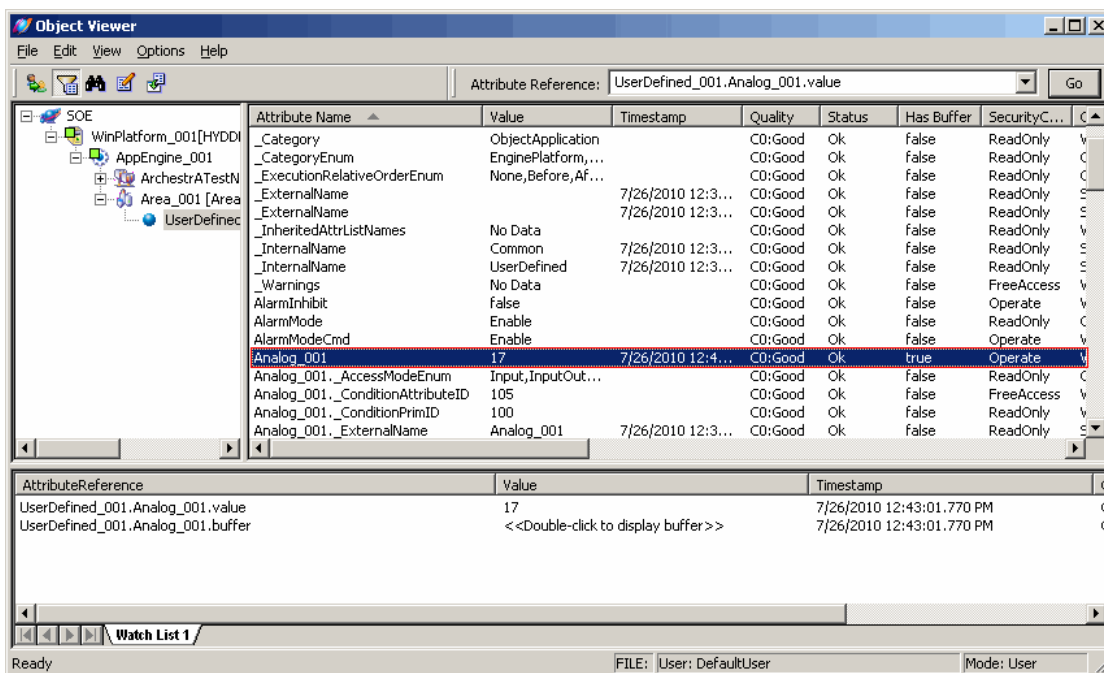
Enable I/O scaling

Select and modify other advanced options as needed. See *Using the I/O Feature* on page 127 for more information.

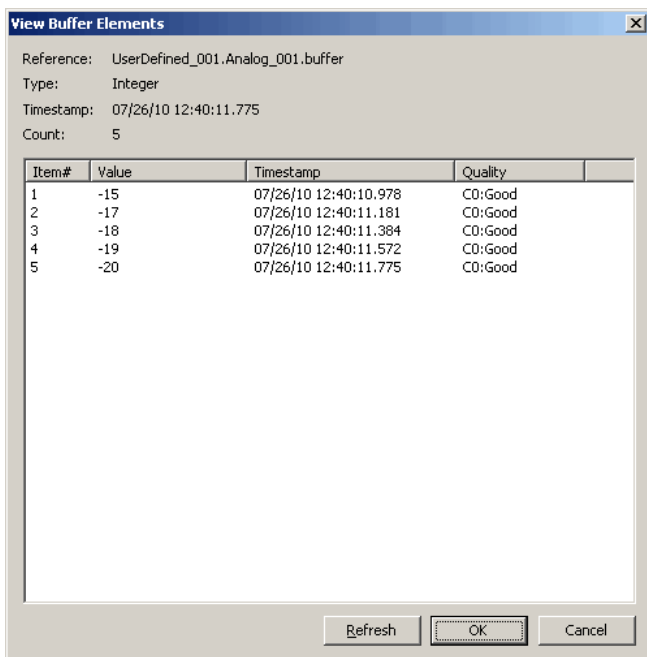
## Using Object Viewer with Buffered Data

You can use the Object Viewer utility to view the attribute values of a selected object at run time.

The **Object Viewer Attributes List** displays a **Has Buffer** column to reflect whether an attribute has a buffer. The value displayed in the column is True if the attribute has a buffer and False if the attribute has no buffer, as shown in the following illustration.



Add the `<attributename>.buffer` property to the **Object Viewer Watch Window**. The **Value** column displays the message `<<Double-click to display buffer>>`. Double-click the attribute to view the buffer elements. The **View Buffer Elements** dialog box appears.



The **View Buffer Elements** dialog box header displays the Attribute Reference, Type, Timestamp, and Count. A details window displays Value, Timestamp, and Quality (VTQ) columns for each item. The data is displayed as a snapshot. Refresh the window to retrieve another snapshot.

The source attribute must also have buffered data enabled. If buffered data is not enabled on the source attribute, the data quality will return as bad, and a configuration error message is displayed.

See the *Object Viewer User's Guide* for more information.

## Using ArcestrA Scripts to Process Buffered Data

ArcestrA scripts can be synchronous or asynchronous. The synchronous mode is the default choice and represents serial script execution by the ApplicationEngine in the course of calling the Execute method of all ApplicationObjects that are on-scan in the ApplicationEngine.

Synchronous mode requires that all scripts execute deterministically and quickly enough to prevent an ApplicationEngine over-scan condition.

---

**Important:** For scripts that process buffered data, it is strongly recommended that you configure scripts as asynchronous execution type. Processing buffered data may be time consuming and exceed the script execution time limit. See *Scripting Tips and Best Practices* on page 444 for more information.

---

## Scripting Basic Functions

The following is a collection of basic functions used to read buffer data from the script accompanied by script snippets.

### Declaration

```
dim vtq as ValueTimeQuality;
```

### To access samples in each buffer

```
FOR EACH vtq in me.U1.buffer
```

---

**Note:** The only syntax that can be used to process buffers in scripts is FOR EACH.

---

### To write the buffer to text file

Declaration:

```
dim strWriter as System.IO.StreamWriter;
```

Usage:

```
strWriter = System.IO.File.AppendText("C:\\sample.txt");
```

### To write the buffer to Logger (or SMC)

```
LogMessage();
```

## Sample Script and Output

### Read Buffered Data from the VTQ Buffer

```
dim vtq as ValueTimeQuality;

dim strWriter as System.IO.StreamWriter;

LogMessage("Script Start");
strWriter = System.IO.File.AppendText("C:\\sample.txt");
FOR EACH vtq in me.U1.buffer
strWriter.WriteLine("DAS value is "+vtq.value);
strWriter.WriteLine("Time stamp of the value is "+vtq.time);
```

```
strWriter.WriteLine("Quality of the value is "+vtq.quality);
strWriter.WriteLine("-----");
`Below messages will be logged to the Logviewer (or SMC)
LogMessage("DAS value is "+vtq.value);
LogMessage("Time stamp of the value is "+vtq.time);
LogMessage("Quality of the value is "+vtq.quality);

next;
strWriter.Flush();
strWriter.Close();

LogMessage("Script End");
```

For each VTQ in this script sample, Value, Timestamp, and Quality are written to the sample.txt file.

For purposes of illustration, UD1 in the script sample is a buffer-enabled Integer attribute with input extended to a DDE item.

### Sample Output Text Written to Text File

```
DAS value is 2

Time stamp of the value is 7/15/2010 11:28:54.880 AM

Quality of the value is 192
```

### Sample Output Text Written to Logger

```
Info      ScriptRuntime  UD1.S1: Script Start

Info      ScriptRuntime  UD1.S1: DAS value is 2

Info      ScriptRuntime  UD1.S1: Time stamp of the value is 7/15/2010 11:28:54.880
AM

Info      ScriptRuntime  UD1.S1: Quality of the value is 192

Info      ScriptRuntime  UD1.S1: Script End
```

## Scripting Tips and Best Practices

### Use Asynchronous Scripts for Buffered Data

Synchronous scripts may read buffered data. However, if the buffer is expected to take too much time for processing it would be better to mark the script as asynchronous.

### Buffered Data Script Syntax and Configuration

Scripts can process buffered data by using the FOR EACH syntax to iterate over an attribute's Buffer property.

To process all the buffers without any loss, asynchronous scripts need to have the following configuration:

- Execution type: Execute
- Trigger type: While True
- Expression: True
- Time period: 00:00:00.0000000

## Using BindTo for Buffered Data

When working with indirect scripts to read the `.buffer` property of an attribute, the `BindTo` call needs to be used in startup instead of using it in `execute`. Using `BindTo` in `execute` can lead to losing the data in alternate buffers. The following snippets show an example of using an indirect to process a buffer. In this example, the `BindTo` call is in `Startup` and `FOR EACH` is in `Execute`.

When binding to an on-engine attribute, the whole script could be put in `Execute`. In this example, the `BindTo` call is in `Startup` and `FOR EACH` is in `Execute`.

```
In Declarations:

dim x as indirect;

In Startup:

x.BindTo("me.attribute1.buffer");

In Execute:

dim vtq as ValueTimeQuality;

for each vtq in x
    LogMessage(vtq.value);
next;
```

When binding to an off-engine attribute, it may take more than one scan cycle to bind and the quality of the Indirect variable must be inspected before attempting its use. When binding to an off-engine attribute, declare the reference as `indirect`, and test the off-engine reference quality in the body of the script. You can use a `WHILE`-triggered script to ensure execution, illustrated in the following example. For more information, see [Binding to Off-engine Attributes](#) in the *AVEVA Application Server Scripting Guide*.

```
In Declarations:

dim x as indirect;
dim y as boolean;

In Execute (while true: me.z):

if not y then
    'This could also be done in the startup script
    x.BindTo("Object1.attribute1");
    y = true;
endif;

if IsGood(x) then
    LogMessage(x);
    me.z = false;
endif;
```

## Buffered Data Run-time Behavior

This following table describes how enabling buffered data affects the run-time behavior of the Scan Group, Inputs, and Redundancy for all phases:

- On Startup
- On Scan
- Execute
- Off Scan
- Shutdown

History run-time behavior and Data and Alarms run-time behavior also change when buffered data is enabled. The run-time behavior of those features is described in *About Buffered Data and History* on page 447 and *About Buffered Data and Alarms and Events*.

---

**Important:** Buffered data is not supported in a multi-galaxy environment.

---

Feature	Phase	Behavior
<b>Scan Group</b>		When the buffer property of a scan group's dynamic attribute is subscribed to at run time, the buffer data support is enabled. The scan group accumulates all of that item's updates into a buffer of VTQs.  The scan group supports buffering up to 10,000 VTQs per item per scan. Higher data throughput rates are not supported
	On Scan	When the object is placed On Scan, the system enables all available items and resets item error count.
	Off Scan	When the Scan Group goes Off Scan, it will clear all buffers associated with each item in the Scan Group.  When buffered data is enabled and the host object goes Off Scan, a VTQ with Bad Quality is added to the buffer. On a subsequent scan, the buffer property of the <b>Value</b> attribute is cleared.
<b>Inputs</b>	On Startup	When buffer is enabled on the attribute, buffer support for the input is enabled. No data is processed during this phase.
	Execute	The buffered attribute is registered, but no data is processed or stored until the buffer reference finishes initializing.
	Shutdown	Interfaces are unregistered. (No change in behavior for buffered data-enabled.)
<b>Redundancy</b>		<b>Application Engine</b>  Buffered attributes are not synchronized with a redundant partner. When a failover occurs, unprocessed buffered data is lost.

Feature	Phase	Behavior
		<p><b>Redundant DIOject</b></p> <p>When buffering is enabled, the RedundantDIOject (RDI) subscribes to the buffer property of its DI source items.</p> <p>On failover, all advised items are moved to the backup node, whether or not buffered data is enabled.</p> <p>Buffered attributes are not synchronized with a redundant partner. When a failover occurs, unprocessed buffered data is lost.</p>

## About Buffered Data and History

When an ApplicationObject starts up, if configured to save historical data, it registers its configuration data with the Historian using a Historian supplied interface.

The following table describes buffered data-enabled history behavior.

History Function	Behavior
Buffer Monitoring	The ApplicationObject monitors and historizes the attribute value. If the attribute being historized has a buffer property, the ApplicationObject monitors the attribute's buffer instead of its value, and historizes each VTQ in the buffer.
Attribute Registration with the Historian	Attributes that are buffer-enabled are registered with the Historian in the same manner as non-buffered attributes are registered.
Push Attribute Changes to the Historian	<p>After successful registration of an attribute with the Historian, the buffer-enabled attribute pushes the data changes to the Historian.</p> <p>If the buffer-enabled attribute reads an empty buffer, and if this is the first scan cycle after the attribute was registered, it gets the attribute's Value, Time, and Quality properties, and if Quality is Good, sends this VTQ to the Historian.</p> <p>If the buffer-enabled attribute reads an empty buffer, and if this is not the first scan cycle after the attribute was registered, it does nothing.</p> <p>For more information, see <i>Working with History</i> on page 343.</p>

## About Buffered Data and Alarms and Events

Application Server's alarm and event capabilities automate the detection, notification, historization, and viewing of either application alarms and events or system alarms and events.

An attribute, when configured for alarming and buffer-enabled, can receive a packet of VTQs in a single scan. This packet can contain values that will trigger an alarm condition. Alarm conditions will be generated based on the VTQs in the buffer.

**Important:** Alarms are not supported in this release of Telemetry Server.

## AppEngine Reconnect Configuration for Undeploying and Redeploying Objects Linked to the Telemetry Server

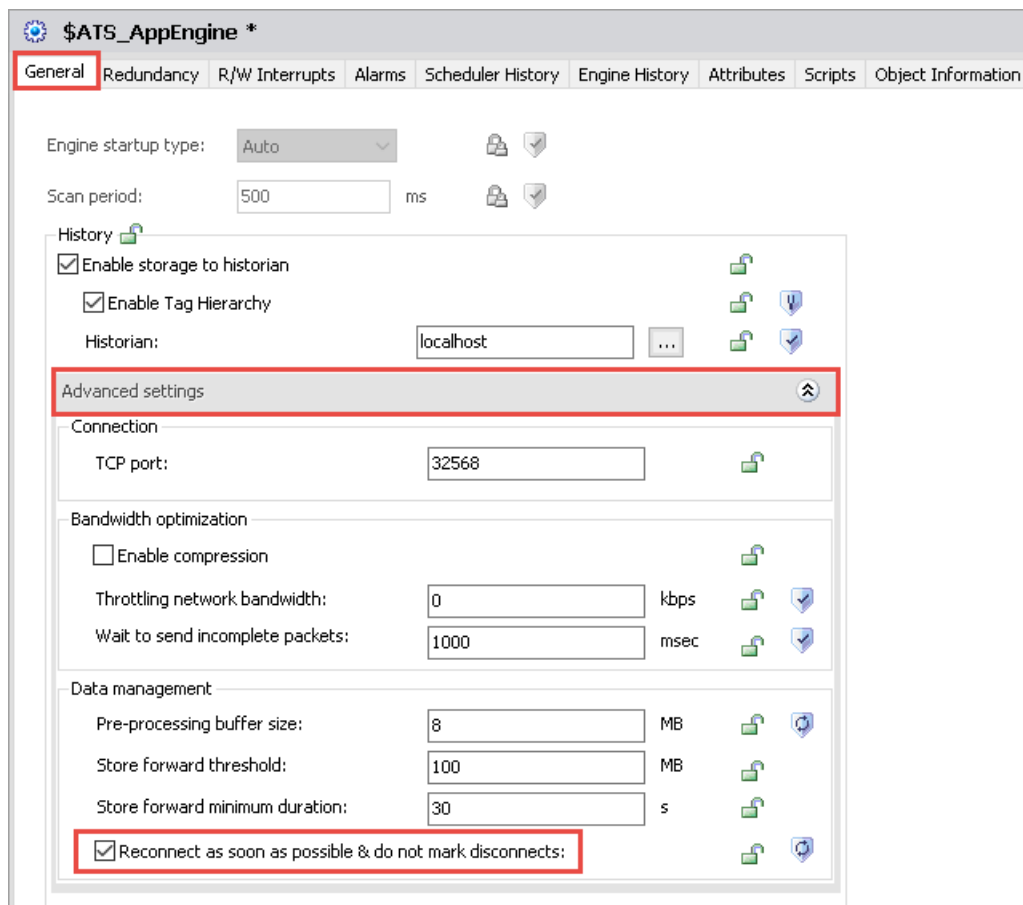
If an object with an attribute linked to a Telemetry Server is undeployed and then redeployed, you may see a gap in the timestamps that are historized by the Historian for the data collected between the undeployment and redeployment times. This can happen, even when buffering is enabled. Note that all data values are accurately preserved, but the timestamps for the values will not be accurate. This is a function of the behavior of the Historian and the default configuration of the AppEngine.

When an AppEngine or objects connected to it are undeployed and then redeployed, all values from the Telemetry Server are collected and sent to the Historian upon redeployment. The Historian gives the current timestamp to the first value received after redeployment. This leaves a gap between undeployment and redeployment times. The Historian adds 5 ms from the time of redeployment to all timestamps for values (after the first value) that were collected during the undeployed time. Thus, subsequent values have timestamps of redeploy time + 5 ms, redeploy time + 10 ms, etc.

To eliminate this gap in timestamp values for data collected during undeployment, configure the **Reconnect** setting under **Data management** in the AppEngine that contains the object. When the Reconnect setting is enabled, all values will be historized with the correct timestamp.

### To configure the AppEngine Reconnect setting

1. Locate the AppEngine that contains the object with the Telemetry Server-connected attribute.
2. Open the AppEngine for editing.
3. In the **General** tab, expand **Advanced Settings**.





4. Under **Data management**, enable the setting, "Reconnect as soon as possible & do not mark disconnects."
5. Save and close the AppEngine object.

## Alarm Functions and Buffered Data

The attribute, if configured for alarms, detects the alarm condition and sets an associated alarm condition attribute according to the base attribute being monitored. If the base attribute is configured with buffered data enabled, the condition attribute will also be enabled with buffered data support.

At run time, the condition attribute's buffer property will contain a VTQ buffer of true or false values representing alarm conditions corresponding to the base attribute buffer contents.

Only the most current alarm condition can be acknowledged.

For more information, see *Working with Alarms and Events* on page 359.

The following table describes buffer-enabled behavior for alarm functions.

Alarm Function	Behavior
Detection	<p>With buffered data enabled for the condition attribute, the alarm condition change is detected using the buffer property of the condition attributes. Multiple alarm messages can be generated based on the buffer contents.</p> <p>The alarm is time-stamped with the time the alarm was recorded at the field device, not the time at which the values were received and processed within the object.</p> <p>A set of alarm messages corresponding to the alarm state transitions are sent to its Notification Distributor.</p> <p>The TimeAlarmOff attribute's value is set to the time of the last off transition, and TimeAlarmOn to the last on transition.</p> <p>The InAlarm value, time and quality properties are set to the last alarm transition's state, time and quality.</p>
Notification	<p>When the Notification Distributor receives a set of alarm messages, it updates the state of the record for the corresponding alarm. The set of alarm messages are forwarded to all registered alarm clients.</p> <p>The Notification Distributor does not send a throttling signal as a result of receiving multiple alarm messages.</p> <p>When the Notification Distributor receives a set of event messages from ApplicationObjects as the result of processing an attribute's buffer, the set of event messages are forwarded to all registered event clients.</p>
Throttling	<p>Alarm throttling settings are ignored for alarm messages generated based on buffered data.</p> <p>Alarms will be generated for all alarm conditions in the buffers. If a potential throttling point is reached, the current version of Application Server provides a functionality for all alarms to be processed without system overload.</p>

<b>Alarm Function</b>	<b>Behavior</b>
Acknowledgement	If an alarm is associated with buffered data, acknowledging an alarm only acknowledges the last alarm, and not all alarms contained in the buffer.
Disabled Alarms	<p>If an alarm is disabled for a buffered attribute, the alarm's Buffer property is not processed. Alarm buffers for disabled alarms are cleared.</p> <p>If an alarm is silenced for a buffered attribute, the alarm's Buffer property is processed. Alarms will be generated but will not be displayed by the alarm client.</p>
Outages	Buffered alarms and events can be lost when there is an outage because the Notification Distributor will only keep and re-send the last on-alarm message and the last off-alarm message.

## Alarm and Event Types and Buffered Data

The following table describes buffered data-enabled behavior of alarm and event types.

<b>Alarm/Event Type</b>	<b>Behavior</b>
Data Change Event	<p>Application (data change) Events have the following fields:</p> <ul style="list-style-type: none"> <li>• Event type – data change (historized)</li> <li>• Timestamp – date/time of event (historized)</li> </ul> <p>When Log Event is checked for the buffered data-enabled attribute, a data change event will be reported for every VTQ in the attribute's buffer property.</p>

---

Alarm/Event Type	Behavior
Limit Alarm	<p>The limit alarm can provide a buffer of alarms on an attribute with buffered data enabled.</p> <p>If the base attribute has an associated Buffer property, then the limit alarm calculation is performed for each VTQ in the buffer. This results in a buffer of Boolean alarm conditions with associated time of the VTQ matching the times of the VTQ in input buffer, which is stored in the respective condition attribute's buffer property.</p> <p>The order of the condition buffer matches the order of the input buffer. The respective condition value and time reflects the last VTQ in the input buffer.</p> <p><b>Value Deadband:</b> If value deadband is configured for limit alarms, then the alarm condition is calculated using VTQs in the input buffer. All VTQs in the buffer are processed. Value deadband is used during execution.</p> <p><b>Time Deadband:</b> If time deadband is configured for limit alarms, when the alarm condition is met, the duration of the condition is calculated using the VTQs in the input buffer. If the last condition change remains undecided until the end of processing the current buffer, then a late decision will be made when the next input buffer is processed. If no more input buffer is received, the last alarm condition change is reported when the time deadband has elapsed as engine time.</p> <p>The time deadband is applied only to the On Rise event of the alarm state. It is not applied when the alarm state falls outside the limit condition.</p>
State Alarm	<p>The state alarm can provide a buffer of alarms on a Boolean attribute when buffered data is enabled.</p> <p>If the base attribute has an associated Buffer property, then the state alarm calculation is performed for each VTQ in the input buffer. This results in a buffer of Boolean alarm conditions with associated time of the VTQ matching the times of the VTQ in input buffer, which is stored in the respective Condition attribute's buffer property.</p> <p>The order of the condition buffer matches the order of the input buffer. The respective condition value and time reflects the last VTQ in the input buffer.</p>
Rate of Change Alarm	<p>If the input has an associated Buffer property, then the Rate of Change (ROC) calculation is performed for each VTQ in the input buffer. Interval is calculated using the difference of two successive VTQs. These calculations result in an array of Boolean alarm conditions with associated timestamps which are stored in the buffer property of the respective condition alarm attribute. The condition value reflects the last VTQ in the respective condition buffer.</p>

---

---

<b>Alarm/Event Type</b>	<b>Behavior</b>
Deviation Alarm	<p>If buffered data is enabled for a deviation alarm's attributes, buffered data is also enabled on the configured condition attributes Dev.Minor.Condition or Dev.Major.Condition or both.</p> <p>When buffered data is enabled on a deviation alarm's attributes and the AppEngine goes Off Scan, the buffers for the configured condition attributes, Dev.Minor.Condition or Dev.Major.Condition or both, are cleared of all VTQs.</p> <p>If the input has an associated Buffer property, then the deviation calculation is performed for each VTQ in the input buffer. These calculations result in an array of Boolean alarm conditions and associated timestamps which are stored in the buffer property of the respective condition alarm attribute. The condition value reflects the last VTQ in the respective Condition buffer.</p>

---

# CHAPTER 13

## Working with References

This section describes the concept of references and how to use reference strings in creating your Application Server application.

### Using Message Exchange and Attributes

References allow identification and communication between objects in the ArcestrA environment. Every object, every attribute, and every property can be uniquely referenced. Those references are communicated over the messaging system. The Message Exchange is the object-to-object communications protocol used by ArcestrA and the Application Server.

---

**Note:** ArcestrA is the framework for supervisory control and manufacturing information systems. It is an open and extensible technology based on a distributed, object-based design. For example, if you are using Application Server with InTouch, these products communicate with each other using the ArcestrA framework.

---

All object attributes have properties, such as Value and Quality. Any data read or written to or from these attributes over ArcestrA Message Exchange is tracked. If an operation cannot be performed, the requesting client is notified.

Message Exchange provides the following features and information:

- Guaranteed response
- Name signatures
- Status and data quality
- Message order preservation within a priority system
- AppEngine-to-AppEngine buffering
- Publish-subscribe heartbeats

### Reference Strings

Reference strings refer to an object or to data within an object's attributes. A reference string consists of an object's reference string plus an attribute's reference string.

```
object Reference + Attribute Reference
```

A reference string is the object name plus the attribute name: `ObjectName.AttributeName`.

In a multi-galaxy environment, a reference string to a remote galaxy is the galaxy name plus the object name plus the attribute name:

```
GalaxyName:ObjectName.AttributeName.
```

Use references and reference strings in a multi-galaxy environment as you would in a single-galaxy environment, with the exception of the galaxy name prefix, `<GalaxyName>`, followed by a colon as in the preceding example. For more information about using references and reference strings in a multi-galaxy environment, see *Accessing Multiple Galaxies* on page 407.

In `TIC101.PV`, `TIC101` is the object reference and `PV` is the attribute reference. The `AttributeName` can be omitted in a reference string, `PV` being assumed in such cases.

---

**Note:** Some objects have a PV attribute, while others do not.

---

Reference strings are concatenated substrings, each no more than 32 characters separated by periods. A substring cannot contain a period. Mathematical operator characters are not allowed. At least one character in each substring must be non-numeric.

Avoid assigning objects and attributes names such that the same reference string can refer to two different things. For example, you have two objects named A1 and B2, and inside A1 you create an attribute named B2 with a data type of Float. A1.B2 refers to the attribute Float named B2. If you then assign object B2 so that A1 is a container of object B2, the reference A1.B2 could refer either to the object B2 or the attribute B2 Float.

---

**Important:** The Galaxy resolves reference strings. If the GR is not available, resolution is done on a peer-to-peer level. After initial resolution, an object is provided an alias that handles references to its location across your network. If an object is relocated or renamed, the reference string resolution is repeated and a new alias provided.

---

## Relative References

References that go up the hierarchy to parent objects are called relative references. For more information about hierarchy, see *Application Object Containment* on page 60.

Relative references, such as `Me`, are valid reference strings. A valid reference string must always contain at least a relative reference or one substring.

The following are valid relative references that refer to the current object:

- `Me`
- `MyContainer`
- `MyArea`
- `MyPlatform`
- `MyEngine`
- `MyHost`

Relative references are especially useful in templates because absolute references typically do not apply or make sense.

When you use relative references, like `MyContainer`, you can refer to contained objects within that container. For example, a reference to `MyContainer.InletValve.PV` is equivalent to `Tank1.InletValve.PV` in the following hierarchy:

<code>Tank1</code>	Cannot reference at this level because this is not contained
<code>Inlet Valve (InletValve)</code>	Can reference at this level because this object is contained
<code>Outlet Valve (OutletValve)</code>	Can reference at this level because this object is contained

## Property References

Certain property names are reserved for ArcestrA. If a string has a reserved property name in the ArcestrA environment, you can still use it. The `PROPERTY` keyword must be part of the string, for example, `PROPERTY(propertyName)`. In all other cases, the case insensitive `PROPERTY` keyword is not required.

The Value property is assumed if no property reference is specified.

The following are property references:

- .Name
- .Value
- .Type
- .Quality
- .Time

syntax:

```
obj.int.PROPERTY (quality)
```

where:

- obj = object specifier
- int = attribute
- PROPERTY = keyword
- (quality) = property specifier

For example, you can address the time of an attribute in a scan group for a DIObject from within an InTouch application, as follows:

```
Galaxy: "<DIObject>.<scangroup>.<attribute>.Property (Time) "
```

This is the same as if you used .Time:

```
Galaxy: "<DIObject>.<scangroup>.Attribute (<attribute>) .Time"
```

You can directly address an item without having an attribute in the scan group. For this example, the item is MB1:

```
Galaxy: "<DIObject>.<scangroup>.MB1.Property (Time) "
```

and

```
Galaxy: "<DIObject>.<scangroup>.Attribute (MB1) .Time"
```

For objects with a default scan group, you must refer to the .Time, .Value, and .Quality properties using the .Property(time), .Property(value), and .Property(quality) notation.

The following is an example of the correct use of property:

```
LogMessage (ATTRIBUTE ("MyEngine.tagname.PROPERTY (securityclassification) ")
)
```

The following is an example of the incorrect use of property:

```
Logmessage (abtcpplc5_001.fast.changingpoint.property (quality) );
```

## Handling Time Zones with the Time Property

If you need to share time stamp values across different time zones (Platforms), use the Time data type in every time zone location. However, if you need to share it as string, remember that when converting the Time data type to a string (for example, in a script), it is automatically converted to local time, so you lose the ability to adjust it in a different time zone.

For example, to convert the Time property to a string GMT:

```
Dim localDateTime As System.DateTime;
```

```
localDateTime = System.DateTime.Parse( obj.attr.Time );
obj.udStringGMTfromLocalTime= localDateTime.ToUniversalTime().ToString();
```

To convert the string GMT to a string of local time:

```
Dim univDateTime As System.DateTime;
univDateTime = System.DateTime.Parse( obj.udStringGMTfromLocalTime );
Obj.udStringLocalTimeFromGMT = univDateTime.ToLocalTime().ToString();
```

## Preserving Time Stamps from the Publishing Source

In the following cases, if you want to pass only the time stamp, the subscriber gets the time stamp as converted to the local time zone of the publisher and not the time zone of the data source.

Examples of configurations that do not preserve the original time zone are as follows.

In this configuration, GalaxyB:Object1.TimeAttr shows the time adjusted to the local time zone of the GalaxyA FSGateway and not the time zone of the PLC:

```
PLC.Item <= GalaxyA Object1.IntAttr.Time <= FSGateway <= GalaxyB OPCClient <=
Object1.TimeAttr
```

In this configuration, GalaxyB:Object1.TimeAttr shows the time adjusted to the local time zone of the InTouch application and not the time zone of the PLC:

```
PLC.Item <= GalaxyA Object1.IntAttr.Time <= InTouch App I/O Message Tag <= GalaxyB
InTouchProxy <= Object1.TimeAttr
```

To avoid these problems, subscribe to the GalaxyA:Object1.IntAttr value property. This way, both the value and time stamp propagate to GalaxyB:Object1.IntAttr. You can then use the GalaxyB:Object1.IntAttr.Time. For example:

```
PLC.Item <= GalaxyA Object1.IntAttr <= FSGateway <= GalaxyB OPCClient <= Object1.IntAttr
PLC.Item <= GalaxyA Object1.IntAttr <= InTouch App I/O Integer Tag <= GalaxyB InTouchProxy <=
Object1.IntAttr
```

In this configuration, the time property propagates from InTouch to Object.IntAttr.Time:

```
PLC.Item <= InTouch I/O Integer Tag <= Galaxy InTouchProxy <= Object.IntAttr
```

## Arrays

A reference string can also refer to the Value property of an array attribute with an optional Array Element Reference that includes up to one dimension:

- [i] – individual element
- [] – entire array

The letter *i* represents an integer constant.

## Formatting Reference Strings

These symbols apply to the reference strings that follow:

This..	means..
::=	can be replaced by



This...	means...
	or
[]	contents optional
{}	contents can be left out, used one time or repeated

Quotation marks are not allowed in tag names, feature names, or attribute names.

## Using Literals

Items outside of angle brackets "<>" are literals. For example:

- `reference_string ::= <Automation_object_reference><attribute_reference> | <tag_name>`
- `Automation_object_reference ::= <absolute_reference>|<relative_reference>`
- `absolute_reference ::= <tag_name>{.<contained_name>}`
- `tag_name ::= <identifier>`
- `contained_name ::= <identifier>`
- `relative_reference ::= <relative_name> | <relative_contained_reference>`
- `relative_contained_reference ::= MyContainer.<contained_name> | MyArea.<contained_name>`
- `relative_name ::= Me | MyContainer | MyArea | MyHost | MyEngine | MyPlatform`
- `attribute_reference ::= <value_ref>|<property_ref>`
- `whole_attribute_ref ::= [.<feature>][.<attribute>] | [.<feature>][.ATTRIBUTE(attribute)]`
- `value_ref ::= <whole_attribute_ref>[<array_index>]`
- `array_index ::= <open_bracket> {<index>} <close_bracket> [<open_bracket><index><close_bracket>][<open_bracket><index><close_bracket>]`
- `property_ref ::= <whole_attribute_ref>.<property>`
- `property ::= Value|Type|Quality|BitField|Dimension1|SecurityClassification|Locked|Category | propertyref`
- `propertyref ::= PROPERTY(Value|Type|Quality|BitField|Dimension1|SecurityClassification|Locked|Category)`
- `BitField ::= .00, .01, .02, ..., .31 (valid ONLY for attributes of type MxInteger; otherwise Configuration error occurs at run time)`
- `attribute ::= <static_attribute>|<dynamic_attribute>`
- `static_attribute ::= [<static_attribute>.]<identifier>`
- `<dynamic_attribute> ::= <any_char_but>{<any_char_but>}`
- `feature ::= [<feature>.]<identifier>`

- `identifier ::= <valid_char>{<valid_char>}`
- `valid_char ::= <letter>|<digit>|<special_character>`
- `letter ::= any letter in alphabet of any language`
- `digit ::= any numerical character`
- `special_character ::= any graphics char, except the following:  
. + - * / \ = ( ) ` ~ ! % ^ & @ [ ] { } | : ; ' , < > ? " whitespace`
- `whitespace ::= CR, LF, Tab, Space, FF, as returned by iswspace()`
- `any_char_but ::= any character except whitespace`
- `open_bracket := [`
- `close_bracket := ]`
- `Galaxy_identifier ::= <letter> | <digit>`

## Notes

- `<tag_name>` is an object's unique name.
- `<contained_name>` is an object's optional contained name. It can be specified in a reference when an object is referred to as a contained child of another object.
- `<index>` is -1 or a positive integer from 1 to 32767.
- `<identifier>` is limited to a maximum of 32 characters.
- An `<attribute>` name or `<feature>` name can contain several `<identifier>` parts. The length of each `<identifier>` part can be up to 32 characters. Each `<identifier>` part is separated by a period. The maximum total length of the `<attribute>` name is 329. This name length applies to both static and dynamic attribute names. The maximum total length of the `<feature>` name is 329.
- `<relative_name>` and `<property>` replacements are case insensitive, including `PROPERTY()`.
- If no attribute reference is specified, `.PV` is assumed. If PV is an attribute of type array, the resulting reference is invalid. For arrays, the `.PV[]` part must be explicitly supplied.

The exception to this rule is a reference that is preceded with an @ sign. This reference refers to the object itself and not any particular attribute or property. Currently, this reference string format is used only in the **Execution Order** group on the **Object Information** page of the Object Editor.

- Do not use Property Names or InTouch Pseudo-Property Names for the names of features or attributes when enhancing an object's functionality on the **Scripts**, **Attributes** and **Field Attributes** pages.

Archestra Property Names include: Locked, Category, HasruntimeSetHandler, Name, Type, Quality, Dimension1, Value, SecurityClassification, 00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30 and 31.

InTouch Pseudo-Property Names include: #VString, #VString1, #VString2, #VString3, #VString4, #EnumOrdinal, #ReadSts, #WriteSts and #QString. For more information on InTouch Pseudo-Property Names, see the *InTouch® HMI Data Management Guide*.

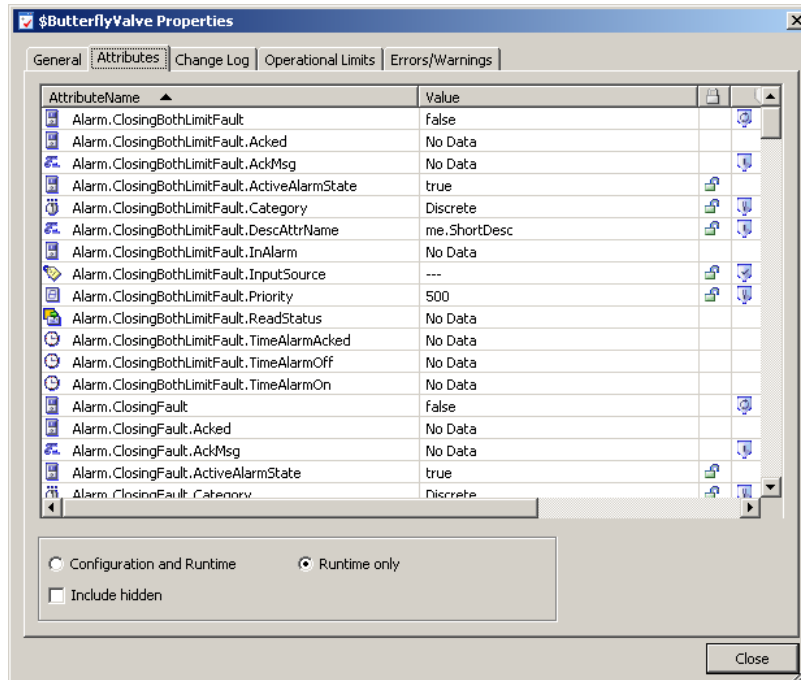
- If the object is a DIOject, use `obj.Attribute(attribute)[x]` rather than `obj.Attribute[x]` to avoid having to resolve the reference at the GR. Resolving an object reference at the GR can negatively impact performance.

## Viewing Attributes in Objects

Within the System Platform IDE, you can view the attributes in an object. This lets you see what attributes are available.

### To view the attributes in a selected object

1. Select an object.
2. On the **Galaxy** menu, click **Properties**.
3. To see the references for the selected object, click the **Attributes** tab.



This page shows you the selected attributes for the object, the current attribute values, and locked and security status. The information shown depends on whether **Configuration and Runtime** or **Runtime Only** is selected.:

#### AttributeName

The name of the attribute.

#### Value

Shows the value of the attribute.

#### Locked/Unlocked

Shows if the attribute is locked or unlocked

#### Security

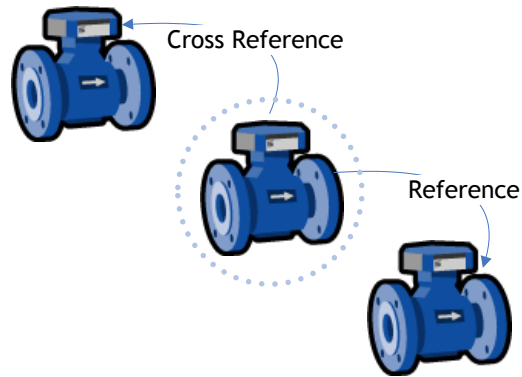
Shows the current security setting, if any.

4. To filter the view, select one or more of the following:
  - Configuration and Runtime:** Select to show both configuration and run-time attributes for the selected object.
  - Runtime only:** Select to show only run-time attributes for the selected object.
  - Include hidden:** Select to show hidden attributes for the selected object.
5. When you are done, click **Close**.

## Viewing References and Cross References

Objects have references and cross references.

- References are the objects the selected object is looking for.
- Cross-references are the objects looking for the selected object.



You can view references and cross-references for a selected object.

Some attributes are dynamic attributes. These are attributes that get created during run time and exist only in run-time. A device integration (DI) reference to a hardware register is a good example.

The attribute referencing a hardware register does not exist at configuration time by default. DI instances create the attribute dynamically during run time if the hardware register exists in the target device.

---

**Note:** References and cross-references shown in the **Properties** dialog box only refer to interobject communications. Area associations, containment, or host assignments are not shown.

---

### To view references and cross-references

1. Select an object.
2. On the **Galaxy** menu, click **Properties**.
3. To see the references for the selected object, click the **References** tab. You see:

#### Source Attribute

The attribute in the selected object referencing an attribute in another object in the application Galaxy.

#### Attribute Reference

The reference string within the **Source Attribute**. This is either an absolute reference or a relative reference.

#### Target Attribute

The absolute reference of the **Attribute Reference**. If this is a reference to a dynamic attribute, the **Target Attribute** only lists the Tagname name of the instance.

4. To see the cross references for the selected object, click the **Cross References** tab. You see:

#### Target Attribute

The absolute reference of the **Attribute Reference**. If this is a reference to a dynamic attribute, the **Target Attribute** only lists the Tagname of the instance.

#### Attribute Reference

The reference string within the **Source Attribute**. This is an absolute reference or a relative reference.

**Source Attribute**

The attribute in the selected object that is referencing an attribute in another object in the application Galaxy.

- When you are done, click **Close**.

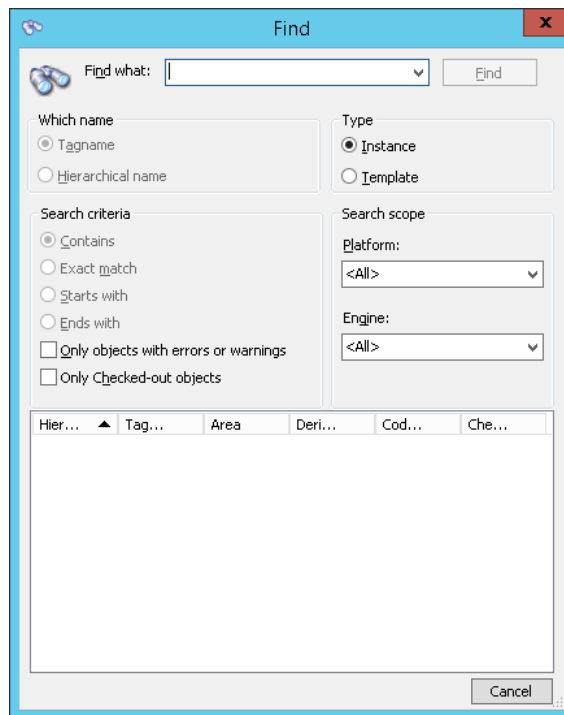
## Finding Objects

Your Galaxy can get very large and can include many objects. It can become difficult to find a specific object. You can search for templates or instances, and you can search by part or all of a tag name or hierarchical name. You can limit a search by platform and engine. Other search options are:

- Find objects with errors or warnings
- Find checked-out objects

### To search for objects

- On the **Edit** menu, click **Find**. The **Find** dialog box appears.



- Do some or all of the following:
  - In the **Find what** box, type some or all of the name of the object.
  - In the **Which name** area, select either **Tagname** or **Hierarchical name** as the type of name you entered in the **Find what** box.
  - In the **Type** area, specify if you are looking for an **Instance** or a **Template**. If you select **Template**, the **Which name** and **Search scope** groups are unavailable.
  - In the **Search criteria** area, specify how to search for the name in the **Find what** box. The options are: **Contains**, **Exact match**, **Starts with** or **Ends with**.
  - Search for objects in an error or warning state by selecting **Only objects with errors or warnings**.
  - Search for objects that are checked out by selecting **Only Checked-out objects**.
  - Limit the search scope by selecting from the **Search scope** lists.

3. When you are done specifying the search criteria, click **Find**. The search results appear in the bottom pane.
4. Double-click an object in the results pane. The object is located and selected for you in the **Application views** area.

If you double-click a Backup AppEngine, the IDE opens the **Deployment view** and the object is selected there. See *Working with AppEngine Redundancy* on page 547 for more information.

## Using Galaxy References in InTouch

You can use Galaxy references in InTouch. Use the InTouch Tag Browser in unlimited selection mode to browse and include references from an Application Server Galaxy in InTouch applications you are developing.

You also can use Galaxy references in InTouch in a multi-galaxy environment by using both the Galaxy: access name and the remote GalaxyName as part of the item name. For more information, see *Using InTouch with Multiple Galaxies* on page 409.

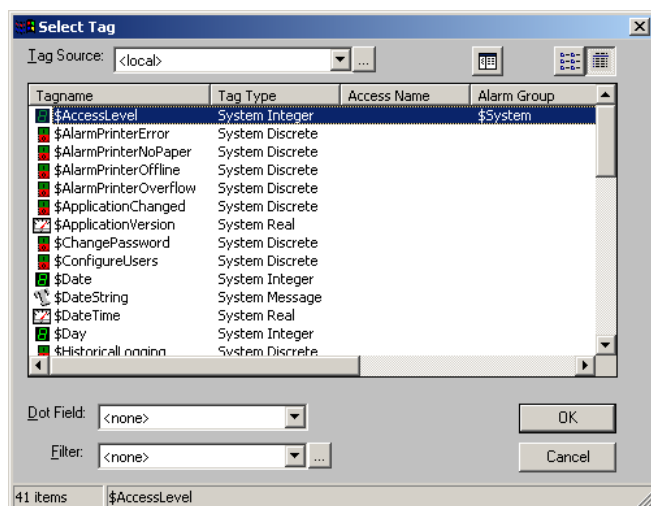
---

**Note:** You must install the Bootstrap and IDE on the InTouch node to create the TagSource to browse Galaxy objects.

---

The following lists the primary ways you can open the Tag Browser in InTouch to see unlimited selection mode:

- Double-click an animation link tagname or expression input box.
- Double-click an ActiveX or wizard tagname or expression input box.
- Double-click an empty area in any InTouch QuickScript window.
- In the InTouch QuickScript editor, select **Tagname** on the **Insert** menu.
- Press the **Alt+N** keys in the InTouch QuickScript editor.
- Double-click a blank **New Name** box in the **Substitute Tagnames** dialog box.
- Double-click the **Tagname** input box in the SQL Access **Bind List Configuration** dialog box.

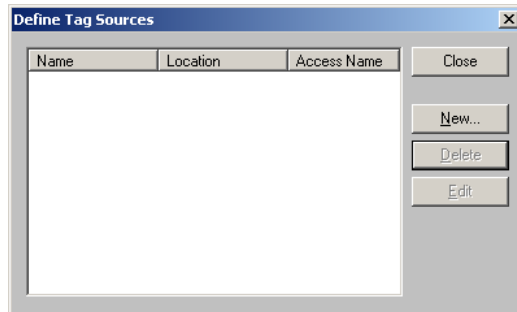


For complete information about using the InTouch Tag Browser, see the InTouch documentation.

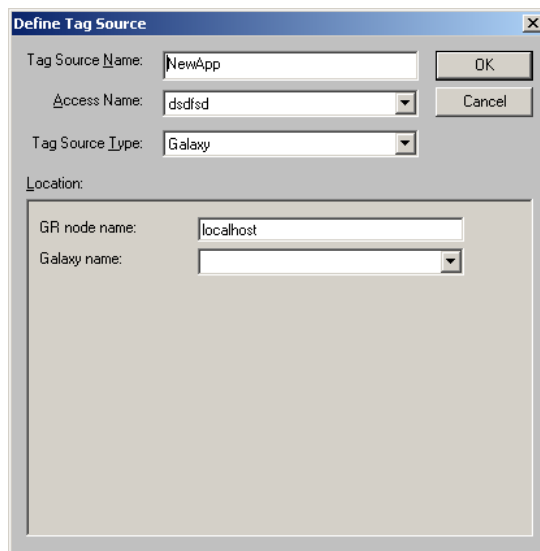
Before you can browse Galaxy references, you must define a new tag source.

## To define a new tag source

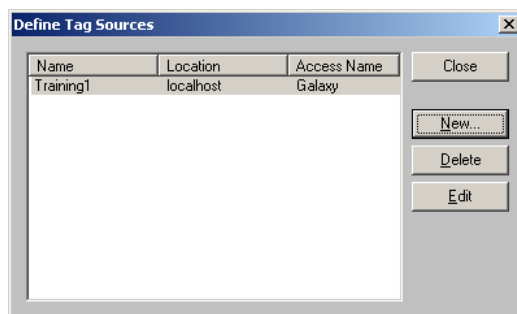
1. In the InTouch HMI **Tag Browser**, click the **Browse** button to the right of the **Tag Source** list. The **Define Tag Sources** dialog box appears.



2. Click **New** to open the **Define Tag Source** dialog box.



3. Type a **Tag Source Name**. This name appears in the **Tag Source** box of the Tag Browser. For example: Training1.
4. Select Galaxy from the list for **Access Name** and **Tag Source Type**.
5. In the **GR node name** box, type the Host Name of the computer on which the Galaxy is located. If the Galaxy is located on the same computer, use `localhost`.
6. In the **Galaxy Name** list, select the name of your Galaxy. For example: Training. Assuming the examples given in steps 3 and 5, the **Define Tag Sources** dialog box appears as follows.



7. Click **Close**. The Tag Browser appears. Now you can browse the TagNames.

### To browse attribute references in a Galaxy

1. Open the **InTouch Tag Browser** in unlimited selection mode.
2. In the **Tag Source** box, select the tag source you created in the previous steps (Training1 in the example). The **Attribute Browser** appears.
3. Select the object and attribute you want to reference in your InTouch application and click **OK**.

---

**Note:** The next time you open the Tag Browser, the **Attribute Browser** automatically opens. To change that, exit the **Attribute Browser** without selecting anything by clicking the blue arrow at top right. The Tag Browser appears and it defaults to the InTouch Tagname Dictionary.

---

For more information about using the Attribute Browser, see *Reference Objects Using the Galaxy Browser* on page 88.

## Telemetry Server Data References

The Telemetry Server interfaces with Application Server to link data from widely dispersed equipment, using DNP3, IEC-60870, or Modbus protocols. This section describes how to create dynamic and static data references to the Telemetry Server, and is organized as follows:

- *Telemetry Server Dynamic Tags* on page 465
- *DNP3 Data References* on page 467
- *IEC 60870-5 Dynamic Data References* on page 472
- *Modbus Data References* on page 475
- *Telemetry Server Static Data References* on page 478

---

**Note:** The IEC-60870 protocol is not supported in AVEVA Telemetry Server version 1.0.

---

Dynamic data references to the Telemetry Server use the following syntax:

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<Point Type>;<SyntaxSpecificItems>;[<PointName>]
```

- The <space> character is valid in Telemetry Server references.
- <ScopeName> is not case sensitive. However, the rest of the reference string is case-sensitive.
- The percent sign (%) that precedes the OutstationFullName indicates that the reference is dynamic. Static data references do not include the percent sign. See *Telemetry Server Static Data References* on page 478 for more information.
- If [<PointName>] is specified, it must be enclosed in brackets.

Note that **Telem\_Server\_** is always prepended to the ScopeName (computer name) at the beginning of the reference string. Dynamic tags automatically populate some of the required fields in the Telemetry Server dashboard. This helps to reduce the amount of duplicated configuration data needed for each point. For information about creating dynamic tags, see *Telemetry Server Dynamic Tags* on page 465.

Using attributes on UserDefined objects (UDOs) is a recommended best practice to create the I/O interface to the Telemetry Server. Once you create the attribute and enable I/O, create the I/O reference using the applicable protocol and syntax. See *Using the I/O Feature* on page 127 for more information about enabling I/O.

### Point Name Format

When creating a reference and you include the optional parameter, "Point Name," the Point Name MUST be enclosed in [brackets] as the following example shows:

```
Telem_Server_Node-12:%Group.DNP3_Outstation;AI;100;[Point_Name]
```



## Invalid Characters

The following are invalid characters that cannot be used within Telemetry Server OutStationFullNames or PointNames:

< > # % \ " ; : ? . @ & = + \$ { } | ^ [ ]

The % . [ ] and : characters are used in creating Telemetry Server references.

- The "%" character designates the reference as a dynamic tag.
- The "." character is used as a separator between the GroupName and OutstationName within the OutStationFullName, and can be used as part of a Point Name when creating a point in a different group than the Outstation. The syntax for this is <GroupName>.<PointName>.
- The ":" character is used to separate the ScopeName from the Telemetry Server tag information.
- The square bracket characters "[ ]" are used to enclose the Point Name, when the Point Name is specified and cannot be used anywhere else in the reference.

## Telemetry Server Dynamic Tags

Dynamic tags (data references) automatically populate some of the required fields in the Telemetry Server dashboard. This helps to reduce the amount of duplication when creating tags for the Telemetry Server. A dynamic tag takes the information entered as an I/O reference in Application Server, and creates or renames a point on the Telemetry Server.

Include a percent sign (%) in the reference string to designate a dynamic tags. The % character is inserted after the Scope Name and separator character ("Telem\_Server\_<ScopeName>:") in the reference string.

Note that **Telem\_Server\_** is always prepended to the Scope Name (computer name) at the beginning of the reference string.

## Dynamic Data References vs. Static Data References

Both dynamic data references (tags) and static data references can be used to access the Telemetry Server.

**Dynamic data references** are used to create the Telemetry Server points/tags remotely from Application Server. Use the reference syntax that corresponds with the protocol you are using. See *Telemetry Server Dynamic Tags* on page 465 for more information.

**Static data references** are used when the points/tags (discrete, analog, strings) already exist in the Telemetry Server. As such, the reference syntax is specific to the pre-existing tag. Use a static data reference Syntax to reference the properties of any object in the database (outstations, channels, points, etc) that already exist in Telemetry Server, or to invoke methods on these objects. See *Telemetry Server Static Data References* on page 478 for more information.

## Dynamic Tag Prerequisites

Before a dynamic tag entered in Application Server can function and automatically create a point in the Telemetry Server, the following items must be defined through the Telemetry Server dashboard:

- Group
- Channel
- Outstation Set
- Outstation

---

**Note:** *Group* is a Telemetry Server-specific term that refers to a folder in the configuration hierarchy. An outstation object is contained within a group. In the Telemetry Server, the configuration hierarchy is the tree at the left side of the dashboard.

---

## Naming Rules

When entering data references for an I/O attribute in Application Server, the following naming rules are used to create or rename points in the Telemetry Server:

- If a point with the specified name and address does not exist in the specified group and outstation, a new point is created with specified name. If a point name is not specified in the reference, the default naming convention is used to generate the point name. Typically, this is <PointType>\_<Address>.
- If the point name is specified in the data reference, the point name must be enclosed in square brackets [ ].
- If a point with the specified name and address already exists in the specified group and outstation, the existing point is used.
- If a point with the specified address but a different name is found in the specified group and outstation, the reference generates a configuration error. The point with specified address is not changed and a new point is not created. See *Configuration Errors* on page 466 for more information.
- If a point with the specified name exists in the specified group, but is attached to a different outstation, the reference generates a configuration error. The named point is not changed and a new point is not created. See *Configuration Errors* on page 466 for more information.
- If a point with the specified name exists in the specified group and outstation but has a different point address, the reference generates a configuration error. The named point is not changed and a new point is not created. See *Configuration Errors* on page 466 for more information.
- If any other database object with the specified name exists in the specified group, the reference generates a configuration error. A new point is not created. See *Configuration Errors* on page 466 for more information.
- If the point name is invalid, for example, if it contains invalid characters, is too long, or has some other error, the reference generates a configuration error and a new point is not created. See *Configuration Errors* on page 466 for more information.

## Configuration Errors

To resolve configuration errors, refer to the *Telemetry Server User Guide* for detailed information. The basic way to resolve an error is to make one or more of the following changes, as applicable:

- Check that the <PointName> in the dynamic tag reference is valid to allow the point to be created in the Telemetry Server database.
- Check that the <PointType> and *address* in the dynamic tag reference are valid for the protocol and correct for the particular outstation. For some point types, such as Modbus digital points, you must add additional items, such as bit offset and bit count.
  - In DNP3, address refers to PointNumber.
  - In IEC 60870, address refers to IOA.
  - In Modbus, address refers to PointAddress

- Check that the <OutstationFullName> in the dynamic tag reference exactly matches the name of the outstation in the Telemetry Server database and that this is the correct outstation. The name is case sensitive.
- Rename, or delete, the object that has the same name in the group, or move the object to a different group.

## DNP3 Data References

A DNP3 point is defined by its outstation name, point type, point number, and optionally, a point name. When adding points to an outstation, use the following general guidelines:

- All references begin with "Telem\_Server\_" prepended to the Scope Name.
- Point numbers are unique within each point type. For example "Analog Input 1" (AI\_1) and "Analog Output 1" (AO\_1) are different points. In the case of binary output points, which contains three subtypes, the subtype does not provide differentiation. Thus, "Digital Output 1", "Pulse (NULL) 1" and "Pulse Trip-Close 1" all refer to the same binary output point. For example "Digital Output 1", "Pulse (NULL) 1" and "Pulse Trip-Close 1" all refer to the same binary output point.
- Point numbers must be unique within each data type. The range of valid point numbers may vary between outstations. Point numbers do not need to be contiguous.
- The point numbers (in an outstation) are configurable. Refer to the outstation configuration to determine which point numbers are valid for a particular outstation.
- The valid range of DNP3 point numbers is 0 to 65535. The Telemetry Server does not limit this range, however, values are device-specific and many devices will impose a much smaller number than 65535 for their upper limit.

### Point Types

A point in DNP3 can be defined as any of the following data types:

Point Type	I/O Type	Details
AI	Analog Input	<i>DNP3 Analog Input and Output Points</i> on page 468
AO	Analog Output	<i>DNP3 Analog Input and Output Points</i> on page 468
DI	Binary Input	<i>DNP3 Binary Input and Output Points</i> on page 469
DO	Binary Output	<i>DNP3 Binary Input and Output Points</i> on page 469
DBI	Double Binary Input	<i>DNP3 Double-Bit Binary Input Points</i> on page 470
RC	Counter	<i>DNP3 Counter Input Points</i> on page 470
OS	String	<i>DNP3 String Input Points</i> on page 471
PN	Pulse (NULL)	<i>DNP3 Pulse Output Points</i> on page 471
PTC	Pulse (Trip/Close)	<i>DNP3 Pulse Output Points</i> on page 471

## Reference Syntax

When creating a Telemetry Server reference, adding a percent sign (%) before the full outstation name makes the reference a dynamic reference. The full outstation name includes the group name.

The general syntax for dynamic references is as follows :

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<PointType>;<PointNumber>;[<PointName>]
```

To create the point in a different group, add the group name to the PointName with a "." separator, as follows:

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<PointType>;<PointNumber>;[<GroupName>.<PointName>]
```

Note that all point types contain all the elements shown in the general syntax, but some point types contain additional parameters. For example, the syntax for Pulse data types is as follows:

```
Telem_Server_<ScopeName>:%<OutstationFullName>;PTC;<PointNumber1>;<PointNumber2>;[<PointName>]
```

When the [<PointName>] parameter is not specified and the point does not already exist, a default name is automatically assigned to the point. The name is in the format "<PointType>\_<PointNumber>."

If you specify the PointName, enclose the name in brackets.

## DNP3 Analog Input and Output Points

### DNP3 Analog Point Syntax

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<PointType>;<PointNumber>;{ZSValue};{FSValue};{DBValue}{%};[PointName]
```

where items enclosed in { curly brackets } are optional.

- **PointType:** AI or AO
- **PointNumber:** a decimal number from 0 to 65535 that maps to the "Point Number" field in the Telemetry Server dashboard. The maximum point number is device-dependent.
- **Optional fields:** ZSValue, FSValue, DBValue, and PointName. You can enter these fields in any order. If no value is entered, the default values for these fields are used.
  - **ZSValue:** This sets the zero scale limit of the tag in the Telemetry Server, and uses the format "ZS<value>", where <value> is a double data-type value and includes both positive and negative numbers. If <value> is out of range, point creation fails.
  - **FSValue:** This sets the full scale limit of the tag in the Telemetry Server. It uses the format "FS<value>", where <value> is a double data-type value and includes both positive and negative numbers. If the value given is out of the range, point creation fails.
  - **DBValue:** This sets the Significant Change option (deadband) of the tag in the Telemetry Server. You can enter the deadband as an absolute or percentage value, as follows:
    - DB<value> sets the **Significant Change** field to **Absolute**, and sets the **Deadband** field to <value>.
    - DB<value>% sets the **Significant Change** field to **Percentage of Change**, and sets the **Deadband** field to <value>.
    - The <value> specified for both **Absolute** and **Percentage of Change** must be a positive double data-type value.
    - When **Percentage of Change** is used, the maximum value is 100.

- **PointName:** a descriptive name for the point, for example, [LiftStationPump\_01]. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, AI\_100.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### Analog Reference Examples

Without optional parameters:

```
Telem_Server_Node-12:%Group.DNP3_Outstation;AI;100
```

References analog input point 100, which is given the name **AI\_100**

With the optional zero scale, full scale, and deadband parameters:

```
Telem_Server_Node-12:%Group.DNP3_Outstation;AI;100;ZS-10;FS250;DB25%
```

References the same analog input point as above, but defines the zero scale limit as -10, the full scale limit as 250, and the deadband as 25%

With the optional point name parameter that includes group name (include the group name when creating the point in a different group than the Outstation):

```
Telem_Server_Node-12:%Group.DNP3_Outstation;AI;100;[Group2.Point_Name]
```

References analog input point 100, which is named **Point\_Name** and is created in Group2.

## DNP3 Binary Input and Output Points

### DNP3 Binary Point Syntax

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<Point Type>;<PointNumber>{;<PointName>}
```

where items enclosed in { curly brackets } are optional.

- **PointType:** DI or DO

---

**Note:** When referencing a binary output point using the DO point type, the Telemetry Server uses the "latch on/off" operation type.

---

- **PointNumber:** a decimal number from 0 to 65535 that maps to the "Point Number" field in the Telemetry Server dashboard. The maximum point number is on the device-dependent.
- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, DI\_5.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### Binary Reference Examples

Without optional PointName:

```
Telem_Server_Node-12:%Group.DNP3_Outstation;DI;5
```

References binary input point 5, which is given the name **DI\_5**

With optional PointName:

```
Telem_Server_Node-12:%Group.DNP3_Outstation;DI;5;[Tag_Name]
```

References binary input point 5, which is named **Tag\_Name**

## DNP3 Double-Bit Binary Input Points

### DNP3 Double-Bit Binary Point Syntax

Telem\_Server\_<ScopeName>:%<OutstationFullName>;<Point Type>;<PointNumber>{;<PointName>}}

where items enclosed in { curly brackets } are optional.

- **Point type:** DBI
- **Point number:** a decimal number from 0 to 65535 that maps to the "Point Number" field in the Telemetry Server dashboard. The maximum point number is device-dependent.
- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, DBI\_5.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### Double-Bit Binary Reference Examples

Without optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;Group.DBI;5

References double-bit binary point 5, which is given the name **DBI\_5**

With optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;DBI;5;[Tag\_Name]

References double-bit binary point 5, which is named **Tag\_Name**

## DNP3 Counter Input Points

### DNP3 Running Counter Point Syntax

Telem\_Server\_<ScopeName>:%<GroupName.OutstationName>;<Point Type>;<PointNumber>{;<PointName>}}

- **PointType:** RC
- **PointNumber:** a decimal number from 0 to 65535 that maps to the "Point Number" field in the Telemetry Server dashboard. The maximum point number is device-dependent.
- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, RC\_3.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### Counter Reference Examples

Without optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;RC;3

References running counter point 3, which is given the name **RC\_3**

With optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;RC;3;[Tag\_Name]

References running counter point 3, which is named **Tag\_Name**

## DNP3 String Input Points

### DNP3 Octet String Point Syntax

Telem\_Server\_<ScopeName>:%<OutstationFullName>;<Point Type>;<PointNumber>{;[<PointName>]}

- **PointType:** OS
- **PointNumber:** a decimal number from 0 to 65535 that maps to the "Point Number" field in the Telemetry Server dashboard. The maximum point number is device-dependent.
- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, OS\_6.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

#### String Reference Examples

Without optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;OS;6

References octet string point 6, which is given the name **OS\_6**

With optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;OS;6;[Tag\_Name]

References octet string point 6, which is named **Tag\_Name**

## DNP3 Pulse Output Points

### DNP3 Pulse-Null Point Syntax

Telem\_Server\_<ScopeName>:%<OutstationFullName>;<Point Type>;<PointNumber>{;[<PointName>]}

- **PointType:** PN
- **PointNumber:** a decimal number from 0 to 65535 that maps to the "Point Number" field in the Telemetry Server dashboard. The maximum point number is device-dependent.
- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, PN\_12.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

#### Pulse Reference Example

Without optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;PN;12

References binary output point 12 using "pulse-on" operator type and "NULL" trip-close code, which is given the name **PN\_12**

With optional PointName:

Telem\_Server\_Node-12:%Group.DNP3\_Outstation;PN;12;[Tag\_Name]

References pulse (null) point 12 using "pulse-on" operator type and "NULL" trip-close code, which is named **Tag\_Name**

## DNP3 Pulse (Trip/Close) Output Points

### DNP3 Trip-Close Pulse Point Syntax

Telem\_Server\_<ScopeName>:%<OutstationFullName>;<Point Type>;<PointNumber1>;<Point Number 2>;[<PointName>]}

- **PointType:** PTC
- **PointNumber:** a decimal number from 0 to 65535 that maps to the "Point Number" field in the Telemetry Server dashboard. The actual range of valid point numbers is dependent on the specific outstation.

For the Trip-Close Pulse function, two point numbers are defined, one for each action (that is, one point number for **trip** and the other for **close**). Generally, these two numbers are the same, however, some non-standard outstations require different point numbers.

- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and point numbers in the format **PointType\_PointNumber1\_PointNumber2**, for example, P\_5\_5. Note that both point numbers are used for automatic naming.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### Pulse (Trip-Close) Reference Example

Without optional PointName:

```
Telem_Server_Node-12:%Group.DNP3_Outstation;PTC;5;5
```

Reference binary output point 5 using "pulse-on" operator type and "trip-close" codes, which is given the name **PTC\_5\_5**

With optional PointName:

```
Telem_Server_Node-12:%Group.DNP3_Outstation;PTC;5;5;[Tag_Name]
```

References binary output point 5 using "pulse-on" operator type and "trip-close" codes, which is named **Tag\_Name**

## IEC 60870-5 Dynamic Data References

---

**Note:** The IEC-60870 protocol is not supported in AVEVA Telemetry Server version 1.0.

---

An IEC60870-5 point is defined by its outstation name, point type, Information Object Address (ioa), and optionally, a point name. When adding points to an outstation, use the following general guidelines:

- All references begin with "Telem\_Server\_" prepended to the Scope Name.
- Each point type (AI, AO, etc.) has its own unique set of point numbers. Note that AI\_1 is different than AO\_1.
- Information Object Addresses (IOAs) must be unique across the outstation.
- The valid range of IEC 60870 IOAs is 1 to 16,777,216. The Telemetry Server does not limit this range, however, values are device-specific and many devices will impose a much smaller number than 16,777,216 for their upper limit.

### Point Types

A point in IEC 60870-5 can be defined as any of the following data types:



Point Type	I/O Type	Details
AI	Analog Input	<i>IEC60870 Analog Input and Output Points on page 473</i>
AO	Analog Output	<i>IEC60870 Analog Input and Output Points on page 473</i>
DI	Digital Input	<i>IEC60870 Digital Input and Output Points on page 474</i>
DO	Digital Output	<i>IEC60870 Digital Input and Output Points on page 474</i>

## Reference Syntax

The general syntax for references is as follows, but there are variations between analog and digital point types:

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<PointType>;<IOA>;[<PointName>]
```

If you specify the PointName, enclose the name in brackets.

When the optional <PointName> parameter is not specified, a default name is automatically assigned to the point. The name is in the format "<PointType>\_<IOA>."

To create the point in a different group, add the group name to the PointName with a "." separator, as follows:

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<PointType>;<IOA>;[<GroupName>.<PointName>]
```

## IEC60870 Analog Input and Output Points

### IEC 60870 Analog Point Syntax

---

**Note:** The IEC-60870 protocol is not supported in AVEVA Telemetry Server version 1.0.

---

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<PointType>;<IOA>;{;ZSValue};{;FSValue};{;DBValue}{%};{;[PointName]}
```

- **PointType:** AI or AO
- **IOA:** Information Object Address. This is a decimal number from 1 to 16,777,216, and maps to the "IOA" field in the Telemetry Server dashboard. The actual range of valid IOA numbers is dependent on the specific outstation.
- **Optional fields:** ZSValue, FSValue, DBValue, and PointName. You can enter these fields in any order. If no value is entered, the default values for these fields are used.
  - **ZSValue:** This sets the zero scale limit of the tag in the Telemetry Server, and uses the format "ZS<value>", where <value> is a double data-type value and includes both positive and negative numbers. If <value> is out of range, point creation fails.
  - **FSValue:** This sets the full scale limit of the tag in the Telemetry Server. It uses the format "FS<value>", where <value> is a double data-type value and includes both positive and negative numbers. If the value given is out of the range, point creation fails.
  - **DBValue:** This sets the Significant Change option (deadband) of the tag in the Telemetry Server. You can enter the deadband as an absolute or percentage value, as follows:
    - DB<value> sets the **Significant Change** field to **Absolute**, and sets the **Deadband** field to <value>.
    - DB<value>% sets the **Significant Change** field to **Percentage of Change**, and sets the **Deadband** field to <value>.

- The <value> specified for both **Absolute** and **Percentage of Change** must be a positive double data-type value.
- When **Percentage of Change** is used, the maximum value is 100.
- **PointName**: a descriptive name for the point, for example, [LiftStationPump\_01]. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, AI\_100.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

## Analog Reference Examples

Without optional parameters:

```
Telem_Server_Node12:%Group.IEC60870_Outstation;AI;100
```

References analog input point 100, which is given the name **AI\_100**

With the optional zero scale, full scale, and deadband parameters:

```
Telem_Server_Node12:%Group.IEC60870_Outstation;AI;100;ZS-10;FS250;DB25%
```

References the same analog input point as above, but defines the zero scale limit as -10, the full scale limit as 250, and the deadband as 25%

With the optional point name parameter that includes group name (include the group name when creating the point in a different group than the Outstation):

```
Telem_Server_Node12:%Group.IEC60870_Outstation;AI;100;[Group2.Point_Name]
```

References analog input point 100, which is given the name **Point\_Name** and is created in Group2.

## IEC60870 Digital Input and Output Points

### IEC 60870 Digital Point Syntax

---

**Note:** The IEC-60870 protocol is not supported in AVEVA Telemetry Server version 1.0.

---

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<PointType>;<IOA>;<BitCount>;{;<PointName>}}
```

- **PointType**: DI or DO
- **IOA**: Information Object Address. This is a decimal number from 1 to 16,777,216, and maps to **Information Object Address** in the Telemetry Server dashboard. The maximum IOA number is dependent on the outstation.
- **BitCount**: a decimal number between 1 and 3. This number maps to **Number of Bits** in the Telemetry Server dashboard.
- **PointName (optional)**: a descriptive name for the point, for example, LiftStationPump\_01.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, DI\_5.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

## Digital Reference Examples

Without optional PointName:

```
Telem_Server_Node12:%Group.IEC60870_Outstation;DI;5;1
```

References 1-bit digital input point 5, which is given the name **DI\_5**

With optional PointName:

```
Telem_Server_Node12:%Group.IE C60870_Outstation;DI;5;1;[Tag_Name]
```

References 1-bit digital input point 5, which is named **Tag\_Name**

## Modbus Data References

A Modbus point is defined by its outstation name, point type, point address, and optionally, point name. When adding points to an outstation, use the following general guidelines:

- All references begin with "Telem\_Server\_" prepended to the Scope Name.
- Modbus point addresses have 5 or 6 digits, depending on the Point Address Range setting of the outstation. The default setting is 5 digits. The range of valid addresses vary by point type.

### Point Types

A point in Modbus can be defined as any of the following types:

Point Type	I/O Type	Details
AI	Analog Input	<i>Modbus Analog Input and Output Points on page 475</i>
AO	Analog Output	<i>Modbus Analog Input and Output Points on page 475</i>
DI	Digital Input	<i>Modbus Digital Input and Output Points on page 477</i>
DO	Digital Output	<i>Modbus Digital Input and Output Points on page 477</i>
STR	String	<i>Modbus String Points on page 477</i>

### Reference Syntax

The general syntax for references is as follows, but there are variations between analog, digital and string point types:

```
Telem_Server_<ScopeName>%<OutstationFullName>;<PointType>;<PointAddress>;<DataType>;
[<PointName>]
```

When the optional *<PointName>* parameter is not specified, a default name is automatically assigned to the point. The name is in the format "*<PointType>\_<PointAddress>*."

If you specify the PointName, enclose the name in brackets.

To create the point in a different group, add the group name to the PointName with a "." separator, as follows:

```
Telem_Server_<ScopeName>%<OutstationFullName>;<PointType>;<PointAddress>;<DataType>;
[<GroupName>.<PointName>]
```

## Modbus Analog Input and Output Points

### Modbus Analog Point Syntax

```
Telem_Server_<ScopeName>:%<GroupName>.OutstationName>;<Point Type>;<PointAddress>;<Data
Type>;{;ZSValue};{;FSValue};{;DBValue};{;};[PointName]}
```

- **PointType:** AI or AO

- **PointAddress:** a 5 or 6 digit decimal number, depending on the Point Address Range setting of the outstation. The default setting is 5 digits. The range of valid addresses vary by point type and maps to the "Address" field in the Telemetry Server dashboard.
- **DataType:** one of the values listed in the following table, for example UINT16.

Data Type	Definition
UINT16	Unsigned 16-bit integer
UINT32	Unsigned 32-bit integer
INT16	Signed 16-bit integer
INT32	Signed 32-bit integer
SINGLE	Single precision float
DOUBLE	Double precision float

- **Optional fields:** ZSValue, FSValue, DBValue, and PointName. You can enter these fields in any order. If no value is entered, the default values for these fields are used.
  - **ZSValue:** This sets the zero scale limit of the tag in the Telemetry Server, and uses the format "ZS<value>", where <value> is a double data-type value and includes both positive and negative numbers. If <value> is out of range, point creation fails.
  - **FSValue:** This sets the full scale limit of the tag in the Telemetry Server. It uses the format "FS<value>", where <value> is a double data-type value and includes both positive and negative numbers. If the value given is out of the range, point creation fails.
  - **DBValue:** This sets the Significant Change option (deadband) of the tag in the Telemetry Server. You can enter the deadband as an absolute or percentage value, as follows:
    - DB<value> sets the **Significant Change** field to **Absolute**, and sets the **Deadband** field to <value>.
    - DB<value>% sets the **Significant Change** field to **Percentage of Change**, and sets the **Deadband** field to <value>.
    - The <value> specified for both **Absolute** and **Percentage of Change** must be a positive double data-type value.
    - When **Percentage of Change** is used, the maximum value is 100.
  - **PointName:** a descriptive name for the point, for example, [LiftStationPump\_01]. If specified, the PointName must be enclosed in brackets.
    - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointNumber**, for example, AI\_100.
    - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### Analog Reference Examples

Without optional parameters:

Telem\_Server\_Node-12:%Group.Modbus\_RTU;AI;40001;UINT16

References the analog input point mapped to holding register 40001 with UINT16 data type, which is given the name **AI\_40001**

With the optional zero scale, full scale, and deadband parameters:

Telem\_Server\_Node-12:%Group.Modbus\_RTU;AI;40001;UINT16;ZS-10;FS250;DB25%

References the same analog input point as above, but defines the zero scale limit as -10, the full scale limit as 250, and the deadband as 25%

With the optional point name parameter (include the group name when creating the point in a different group than the Outstation):

```
Telem_Server_Node-12:%Group.Modbus_RTU;AI;40001;UINT16;[Group2.Point_Name]
```

References the analog input point mapped to holding register 0001 with UINT16 data type, which is named **Point\_Name** and is created in Group2.

## Modbus Digital Input and Output Points

### Modbus Digital Point Syntax

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<Point Type>;<PointAddress>;<BitOffset>;<BitCount>;{;<PointName>}}
```

- **PointType:** DI or DO
- **PointAddress:** a 5 or 6 digit decimal number, depending on the Point Address Range setting of the outstation, which is set through the Telemetry Server dashboard. The default setting is 5 digits. The range of valid addresses vary by point type and maps to the "Address" field in the Telemetry Server dashboard.
- **BitOffset:** a decimal number between 0 and 15. This number maps to the "BitOffset" field in the Telemetry Server dashboard. The maximum offset varies and depends on the bit count.
- **BitCount:** a decimal number between 1 and 3. This number maps to the "BitCount" field in the Telemetry Server dashboard.
- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets. If point name is not specified, the point is assigned a default name that uses point type and point address, in the format **PointType\_PointAddress**, for example, DI\_30001.
  - If PointName is not specified, the point is assigned a default name that uses point type and number, in the format **PointType\_PointAddress**, for example, DI\_30001.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### Digital Reference Examples

Without optional PointName:

```
Telem_Server_<ScopeName>:%Group.Modbus_RTU;DI;30001;0;1
```

References the 1-bit digital input point mapped to the least significant bit of input register 30001, which is given the name **DI\_30001**

With optional PointName:

```
Telem_Server_<ScopeName>:%Group.Modbus_RTU;DI;30001;0;1;[Tag_Name]
```

References the 1-bit digital input point mapped to the least significant bit of input register 30001, which is named **Tag\_Name**

## Modbus String Points

Modbus string points can function as either input (read) only or input/output (read/write) points. This is set by checking the "Write" checkbox in the Telemetry Server dashboard.

### Modbus String Point Syntax

```
Telem_Server_<ScopeName>:%<OutstationFullName>;<Point Type>;<PointAddress>;<Length>;<StringType>;{;<PointName>}}
```

- **PointType:** STR
- **PointAddress:** a 5 or 6 digit decimal number, depending on the Point Address Range setting of the outstation. The default setting is 5 digits. The range of valid addresses vary by point type and maps to the "Address" field in the Telemetry Server dashboard.
- **Length:** a decimal number from 1 to 247 that sets the maximum string length and maps to **Requested String's Length** in the Telemetry Server dashboard.
- **String type:** one of the values listed in the following table, for example, ASCIILOW. This value maps to the "StringType" field in the Telemetry Server database.

String Type	Definition
ASCIILOW	Single byte ASCII, first character in low byte
ASCIHIGH	Single byte ASCII, first character in high byte
UNICODE	Two byte Unicode

- **PointName (optional):** a descriptive name for the point, for example, LiftStationPump\_01. If specified, the PointName must be enclosed in brackets.
  - If PointName is not specified, the point is assigned a default name that uses point type and point address, in the format **PointType\_PointAddress**, for example, STR\_30002.
  - If you want to create the new point in a different group than the Outstation group, include the group name: **GroupName.PointName**.

### String Reference Example

Without optional PointName:

```
Telem_Server_Node-12:%Group.Modbus_RTU;STR;30002;20;ASCIILOW
```

References the 20-character ASCII string point mapped to a block of input registers starting at 30002, which is given the name **STR\_30002**

With optional PointName:

```
Telem_Server_Node-12:%Group.Modbus_RTU;STR;30002;20;ASCIILOW;[Tag_Name]
```

References the 20-character ASCII string point mapped to a block of input registers starting at 30002, which is named **Tag\_Name**

## Telemetry Server Static Data References

Static data references are used when the objects have already been defined in the Telemetry Server. Static data reference syntax is common to all protocols in Telemetry Server, whereas dynamic data reference syntax varies by protocol. Static data references can use any of the following syntaxes:

```
Telem_Server_<ScopeName>:<Item>
```

```
Telem_Server_<ScopeName>:<Item>.<Property>
```

```
Telem_Server_<ScopeName>:<Item>.<DatabaseAggregate>
```

```
Telem_Server_<ScopeName>:<Item>.<DatabaseAggregate>.<Property>
```

```
Telem_Server_<ScopeName>:<Item>.<Method>
```

```
Telem_Server_<ScopeName>:<Item>.<DatabaseAggregate>.<Method>
```

### *Where*

- <ScopeName> is by default the name of the computer where the Telemetry Server instance is installed. You can customize the "ScopeName" field in the Telemetry Server to accommodate redundancy.
- <Item> is the location of the database item.
- <Property> is the name of the property tag that you want to reference. If you omit <Property>, the tag uses the default property for the defined database item or aggregate. The default property for an item or aggregate varies according to the type of item or aggregate. For example, the default property for a point is CurrentValue.
- <Database Aggregate> is the name of the database aggregate that has the required property. Database Aggregates are extensions to a table in the database and they have their own name, table, properties and methods.
- <Method> is the name of the method that you want to reference. Method static tags are write-only. Attempting to read a method tag generates an error.
  - Method static tags are write only. Writing to a method tag invokes the method, and thus triggers the associated action. Attempting to read a method tag generates an error.
  - Only methods with zero or one arguments are supported in static tags. When you write to a method tag with zero arguments, the value being written is discarded. When you write to a method tag with one argument, the value being written is passed as the argument to the method. Therefore, the argument must match the value.

### **Static Data Reference Examples**

Example of the static data reference syntax:

```
Telem_Server_Node12:Group01.Group02.Tank_Level
```

You can trigger the Telemetry Server to call a dial-up outstation by invoking the one shot method. Writing to this tag invokes the "OneShot" method on the "PSTN" database aggregate of an outstation called "Group.Outstation."

```
Telem_Server_XXXXX:Group.Outstation.PSTN.OneShot
```

You can apply an override to an analog point by invoking the override method. Writing a value to this tag invokes the "Override" method on the point "Group.AI\_1" to change the point to the specified value. Overriding must be enabled on the point:

```
Telem_Server_XXXXX:Group.AI_1.Override
```





# CHAPTER 14

## Configuring Security

This section describes the architecture of ArcestrA security, and how to use it to manage access to configuration and run-time aspects of your Application Server application.

For the latest information about mitigating security threats that can affect Application Server and Microsoft products, sign into the AVEVA Global Customer Support website:

<https://softwaresupport.aveva.com/> <http://softwaresupport.aveva.com>

Then, navigate to **Security Central**.

### About Security

Galaxies are created without security. After creating a Galaxy, you can add security to manage access to it. User permissions are set within the IDE. Adding security to a Galaxy authenticates users across all editors and interfaces and controls access to the:

- IDE
- ArcestrA System Management Console (SMC)
- Layout Editor
- Screen Profile Editor
- ViewApp Editor

Security lets you set specific permissions for operations within the IDE and SMC, such as:

- Starting the IDE
- Importing, exporting, creating, and modifying instances, derived templates, graphics and ViewApps
- Deploying and undeploying instances
- Uploading run-time changes
- Managing alarms
- Performing maintenance and system administration functions within the SMC.

You can also configure credentials that AVEVA OMI apps can use for connecting to resources that do not recognize Windows credentials.

While there are not specific permissions that can be set for the Layout, Screen Profile, and ViewApp Editors, restricting permissions for graphics and objects in the IDE will also restrict a user's ability to modify objects and graphics in these editors. See *Assigning Users to Roles* on page 500 for additional information.

### Security Model for Galaxies

Security settings in the IDE controls access to:

- User and editor interfaces in the ArcestrA environment. The signed-in user remains authenticated across all editors and interfaces. The user does not have to sign in again when switching to a different editor, for example, when switching from the IDE to the Screen Profile Editor.

---

**Note:** Regardless of the Galaxy security setting, a user must be a member of the **aaConfigTools** group to connect to a Galaxy from the IDE, or the user must launch the IDE using the "Run as administrator" option. Use the Windows Control Panel to add a user to the aaConfigTools group. Since aaConfigTools is an OS user group, you cannot use Galaxy Security Configuration to add a user to the group.

---

- Object attributes and the data they represent.
  - Connection to the SQL Server database used for the Galaxy Repository.
- 

**Note:** For additional information about SQL Server security, see the section on SQL Server Rights Requirements in the *System Platform Installation Guide*.

---

- Credentials that allow a logged-in user to an AVEVA OMI ViewApp to access third-party resources used by the ViewApp.

Each Galaxy in the Galaxy Repository manages its own security model. The security schema managed in a Galaxy is a three-level configuration model to create and maintain the following:

- Users associated with specific roles
- User roles associated with specific system administration, configuration and run-time (operational) permissions, which map to security groups
- Security groups associated with specific objects in the Galaxy

The default Galaxy Security model includes:

- Two users: DefaultUser and Administrator, both with full access to everything.
- 

**Important:** Be sure to change the passwords for the Administrator and DefaultUser accounts after enabling galaxy security. We recommend that you use Microsoft Active Directory to implement your galaxy password policy because it is the easiest and most effective way to set the policy.

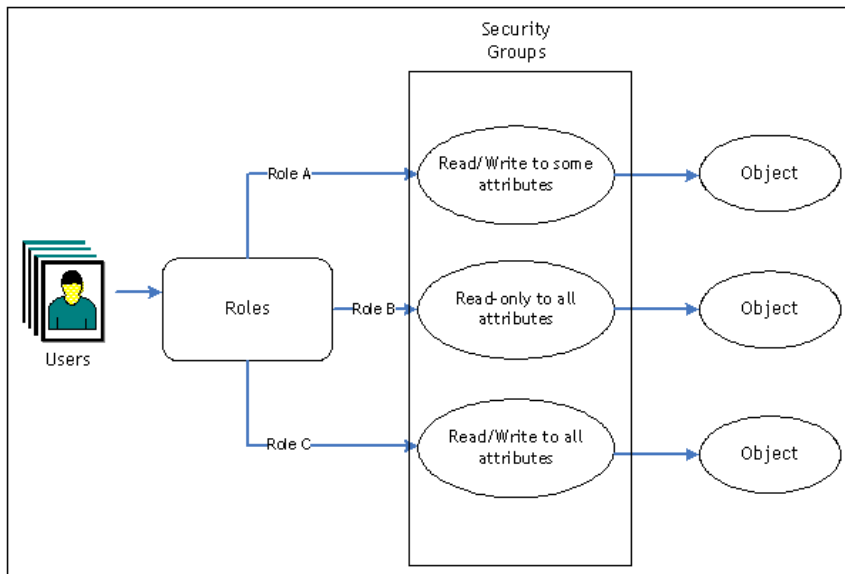
---

- One security group named Default.
  - Two security roles: Default and Administrator, both with full privileges.
- 

**Important:** We recommend that you set the privileges for the Default role to the minimum level needed by all users.

---

The security matrix defines a cascading model of users associated with specific roles that are associated with specific security groups that are associated with specific objects. User run-time permissions can vary from object to object, action to action, and process to process. The security icons associated with object attributes map directly to control points in the Archedra security model.



## About Authentication Modes

You can select from three authentication modes to assign security:

- **Galaxy:** Uses local Galaxy configuration to authenticate users. All security for the Galaxy is specified and contained at the specific Galaxy level. When the user logs on, security credentials are checked and access to areas and activities is granted at the Galaxy level.

---

**Note:** If you are implementing a multi-Galaxy environment or creating credentials for ViewApps, Galaxy Authentication Mode is not supported. Use one of the OS-based authentication modes instead. See *Working with Multiple GR Nodes and Galaxies* on page 403 and *About Credentials* on page 489 for more information.

---

- **OS User Based:** Uses the operating system's user authentication system on an individual user level. All security for the Galaxy is specified and contained in the operating system (OS) on a user level basis. When the user logs on, security credentials are checked and access to areas and activities are decided at the OS user level.
- **OS Group Based:** Uses the operating system's user authentication system on a group basis. All security for the Galaxy is specified and contained in the user-to-roles mapping you created in the OS to assign security. When a user logs on, security credentials are checked and verified at the OS group level. OS groups are mapped to security roles in the Galaxy to allow access to areas and activities in the Galaxy. For more information, see *About OS Group-based Security* on page 503.

---

**Note:** If you are using OS user-based security or OS group-based security and you have permissions to use the IDE, the Log In dialog box does not appear.

---



---

**Important:** Do not use the "Administrator" user account to log in to Application Server, InTouch ViewApps, or other System Platform components. "Administrator" is a reserved System Platform name. Some modules of Application Server and System Platform view "Administrator" as a system admin, while other modules view it as a Galaxy admin.

---

## Multiple Accounts Per User

Regardless of the security system, a single user can have multiple accounts. For example, a user can have an account that provides permissions for working with instances but not templates. The same person can have another supervisory account for working with templates and managing users in the ArchestrA environment.

Each account requires a different user name and password. For example:

User Name	Password	Access
bsmith	password	Instances, not templates
bobsmith	super	Instances, templates, not managing users
Robertsmith	admin	Instances, templates, managing users

## Changing Security Settings

After you change security for a Galaxy, you see the following behaviors and conditions:

- When you change the authentication mode security, the IDE restarts.

---

**Important!** Make sure that you have a valid login for the IDE before changing the authentication mode, especially if changing from "None." If not, create a login before changing authentication modes.

---

- To switch users, the person must log on as the new user by clicking **Change User** on the **Galaxy** menu.
- If you previously configured security under one authentication mode and then switch authentication modes, only those users created while configuring the new mode are available. Other users are not deleted, just unavailable in the new mode.
- Objects that are reassigned to different security groups are marked as "pending update" and require redeployment for the change in security group to take effect.
- If security was previously configured for an OS-based authentication mode, reconfiguring security synchronizes the user's full name and OS groups if some data in the OS has changed.

## About Security Groups

Every object in the Galaxy belongs to only one security group. You can create and manage security groups that make sense for your organization. These security groups are mapped to roles on the **Roles** page.

Permissions determine what kind of access users have for each attribute. There are five basic operational permissions:

- Acknowledge alarms
- Change the value of attributes with security mode Configure
- Change the value of attributes with security mode Operate; this includes also security modes Secured Write and Verified Write

- Change the value of attributes with security mode Tune
- Confirm writes to attributes that require Verified Writes

By default, all currently used objects are assigned to a security group called Default.

A user who is a member of a role assigned to Security Role "Default" has permission to:

- Acknowledge alarms
- Change attribute values with "configure" security mode
- Change attribute values with "operate" security mode, including "secured write" and "verified write"
- Change attribute values with "tune" security mode
- Verify writes to Attributes with "verified write" security mode

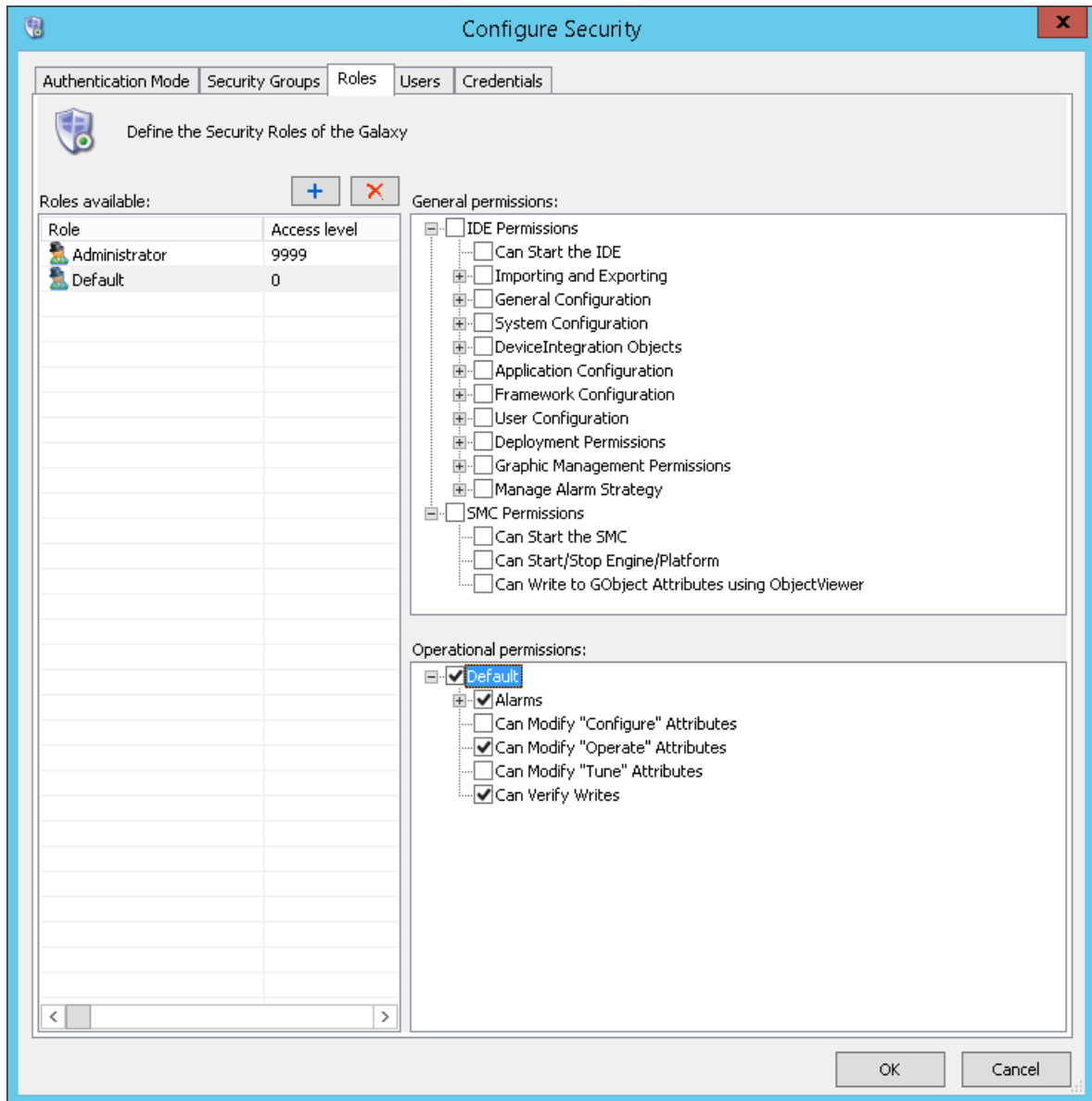
For example, you want users in certain roles to only have permission to acknowledge alarms that are generated from objects contained in Area1. You have a role named Area1Acknowledgers. You need to:

1. Create a new Security Group, for example **SecGrpArea001**.
2. Assign all objects that are contained in area Area1 to Security Role **SecRoleArea001**.
3. On the **Roles** page, select the **Area1Acknowledgers** role. In the **Operational Permissions the Security Group** for **SecGrpArea001**, select **Can Acknowledge Alarms**.
4. Any user that belongs to the **Area1Acknowledgers** role can at least acknowledge alarms of objects contained in the security group **SecGrpArea001**. They do not have any other operational permissions for those objects.

## About Roles

You can create and manage user roles that apply to your organization's processes and work-based authorities. A role defines a set of permissions. After defining roles, you assign roles to authorized users as needed to provide the users with the permissions that they need. A user can be given multiple roles.

Application Server automatically creates two roles called **Administrator** and **Default** and gives both roles full permissions to everything. When defining roles, first remove any permissions that are not needed by all users from the **Default** role because all configured users will be assigned to the Default role and you cannot remove the Default role from a user. For example, if the minimum set of permissions required by all users is the ability to acknowledge alarms and modify "Operate" attributes, remove all other permissions. Add new roles by selecting the **+** button above **Roles available**, then enter a name for the role and an access level.



You can specify General and Operational Permissions for each role.

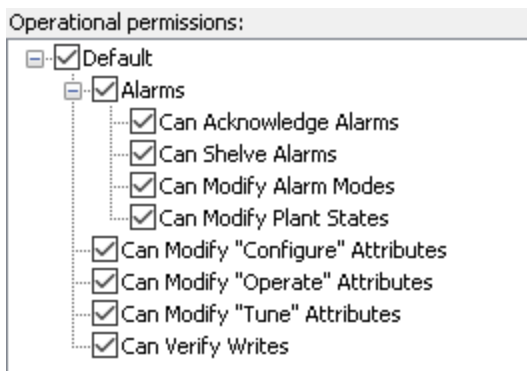
- **General permissions** relate to application configuration and administration tasks within the IDE and the System Management Console (SMC). By default, the Administrator and Default roles have all general permissions. Remove any unnecessary general permissions from the Default role. Leave only the permissions that are needed by all users.

**Note:** You cannot remove any General permissions from the Administrator role. You can, however, remove Operational permissions.

General permissions:

- IDE Permissions
  - Can Start the IDE
  - Importing and Exporting
    - Can Import
    - Can Utilize GalaxyLoad/GalaxyDump
    - Can Export
  - General Configuration
    - Can Modify Deployed Instances
    - Can Disable Change Comments
    - Can Override Checkout
    - Can Upload from Runtime
    - Can Modify Communications Management
  - System Configuration
    - Can Create/Modify/Delete System Object Templates (Platforms and Engines)
    - Can Create/Modify/Delete System Object Instances (Platforms and Engines)
    - Can Create/Modify/Delete Area Objects
  - DeviceIntegration Objects
    - Can Create/Modify/Delete DeviceIntegration Object Templates
    - Can Create/Modify/Delete DeviceIntegration Object Instances
  - Application Configuration
    - Can Create/Modify/Delete Application Object Templates
    - Can Create/Modify/Delete Application Object Instances
  - Framework Configuration
    - Can Modify the Security Model
    - Can Modify the Time Synchronization Settings
  - User Configuration
    - Can Create/Modify/Delete Users
    - Can Modify Own User Information
  - Deployment Permissions
    - Can Deploy/Undeploy System Objects and Manage ArchestrA Services
    - Can Deploy/Undeploy Area Objects
    - Can Deploy/Undeploy Application Objects
    - Can Deploy/Undeploy DeviceIntegration Objects
    - Can Mark an Object as Undeployed
  - Graphic Management Permissions
    - Can Import Graphics
    - Can Export Graphics
    - Can Create/Modify/Delete Graphics within Toolsets
    - Can Create/Modify/Delete Attached Object Graphics in Template
    - Can Create/Modify/Delete Attached Object Graphics in Instance
    - Can Create/Modify/Delete ViewApplications
    - Can Deploy/Undeploy ViewApplications
    - Can Edit Galaxy Style Library
  - Manage Alarm Strategy
    - Can Modify Alarms and Events Configuration
- SMC Permissions
  - Can Start the SMC
  - Can Start/Stop Engine/Platform
  - Can Write to GObject Attributes using ObjectViewer

- Operational permissions** relate to the security groups listed on the **Security Groups** page. For each role, modify security group permissions as needed. The Administrator and Default roles initially are given all operational permissions for the Default Security Group. If you created any new security groups on the Security Groups page, the Administrator and Default roles are not automatically granted any permissions to these security groups. If you do not intend to have users with the Administrator role working with alarms or attributes at run time, you can limit operational permissions from the Administrator role.



Operational permissions are defined as follows:

- Can Acknowledge Alarms: Allows users to manually acknowledge an alarm in the run-time environment.
- Can Shelve Alarms: Allows users to manually shelve and unshelve alarms.
- Can Modify Alarm Modes: Allows users to modify the mode of an alarm.
- Can Modify Plant States: Allows users to modify plant states for state-based alarming.
- Can Verify Writes: Allows users to provide an authentication signature for attributes configured with Verified Writes security classification. Only users with this permission can verify a task performed by users with the Can Modify "Operate" Attributes permission.
- Can Modify "Operate" Attributes: Allows users with operational permissions to do certain normal day-to-day tasks like changing setpoint, output and control mode for a PID object, or commanding a Discrete Device object.
- Can Modify "Tune" Attributes: Allows users to tune the attribute in the run-time environment. Examples of tuning are attributes that adjust alarm setpoints and PID sensitivity.
- Can Modify "Configure" Attributes: Allows users to configure the attribute's value. Requires that the user first put the object Off scan. Writing to these attributes is considered a significant configuration change, for example, a PLC register that defines a Discrete Device input.

Once you have defined a role by setting operational permissions and general permissions, you can assign users to that role.

## About Users

If you select either OS based authentication mode, users with local accounts are added to the **Authorized Users Available** list in the following format: `.\<username>`. Domain users are added in the format `<domain>\<username>`.

If you select **OS Group Based** authentication mode and use local accounts, the local account must exist on each node in the Galaxy for successful authentication of that user on any computer in the Galaxy. OS groups are added in the format `<domain>\<groupname>`.



Two users are defined by default when a new Galaxy is created: Administrator and DefaultUser. These cannot be deleted in an open security setting and both users are associated with the default roles, Administrator and Default. See *About Roles* on page 485 for more information.

---

**Important:** Do not use the "Administrator" user account to log in to Application Server, InTouch ViewApps, or other System Platform components. "Administrator" is a reserved System Platform name. Some modules of Application Server and System Platform view "Administrator" as a system admin, while other modules view it as a Galaxy admin. When using OS-based security, do not use the local ArchestrA user account for Application Server user authentication purposes.

---

## About Credentials

The **Credentials** tab lets you add login credentials that some AVEVA OMI ViewApps may need for access to third-party data and applications that do not support standard authentication methods, such as Windows OS credentials, Active Directory, OpenID Connect, or other standard authentication method.

Initially, a galaxy does not include any credentials. Credentials are created on a galaxy-wide basis, not for individual ViewApps. Although you enter credentials as plain text, the credentials are encrypted when saved, and the credentials are sent to run-time nodes as encrypted data.

Each credential you create is associated with one and only one OS user group. In order to create credentials, OS group-based or OS user-based security must be configured. The credentials you create here are associated with an OS group name. When a user logs into a ViewApp that has an app with a configured credential, the logged-in user obtains access to the credential, provided that the user is associated with the same OS group that matches the OS group for the credential. For example, if you create a credential that is associated with the OS group "Operators," a user that is also a member of "Operators" has access to that credential. When you configure an app that uses credentials

App developers can use the Touch OMI SDK to include a dropdown menu that lists configured credentials. When you add an app that uses credentials to a layout, you can select one of the credentials as you configure the app. This is the credential that the app will use at run time, when deployed in a ViewApp. Only one credential can be configured per layout, and users cannot select a different credential from the running ViewApp.

---

**Note:** Credentials are supported only when **OS User based** or **OS Group based** authentication mode is enabled. If authentication mode is set to **None** or **Galaxy**, you cannot add or delete credentials, even if the credentials were created while the authentication mode was set to OS User or OS Group.


---

## Create Named Credentials

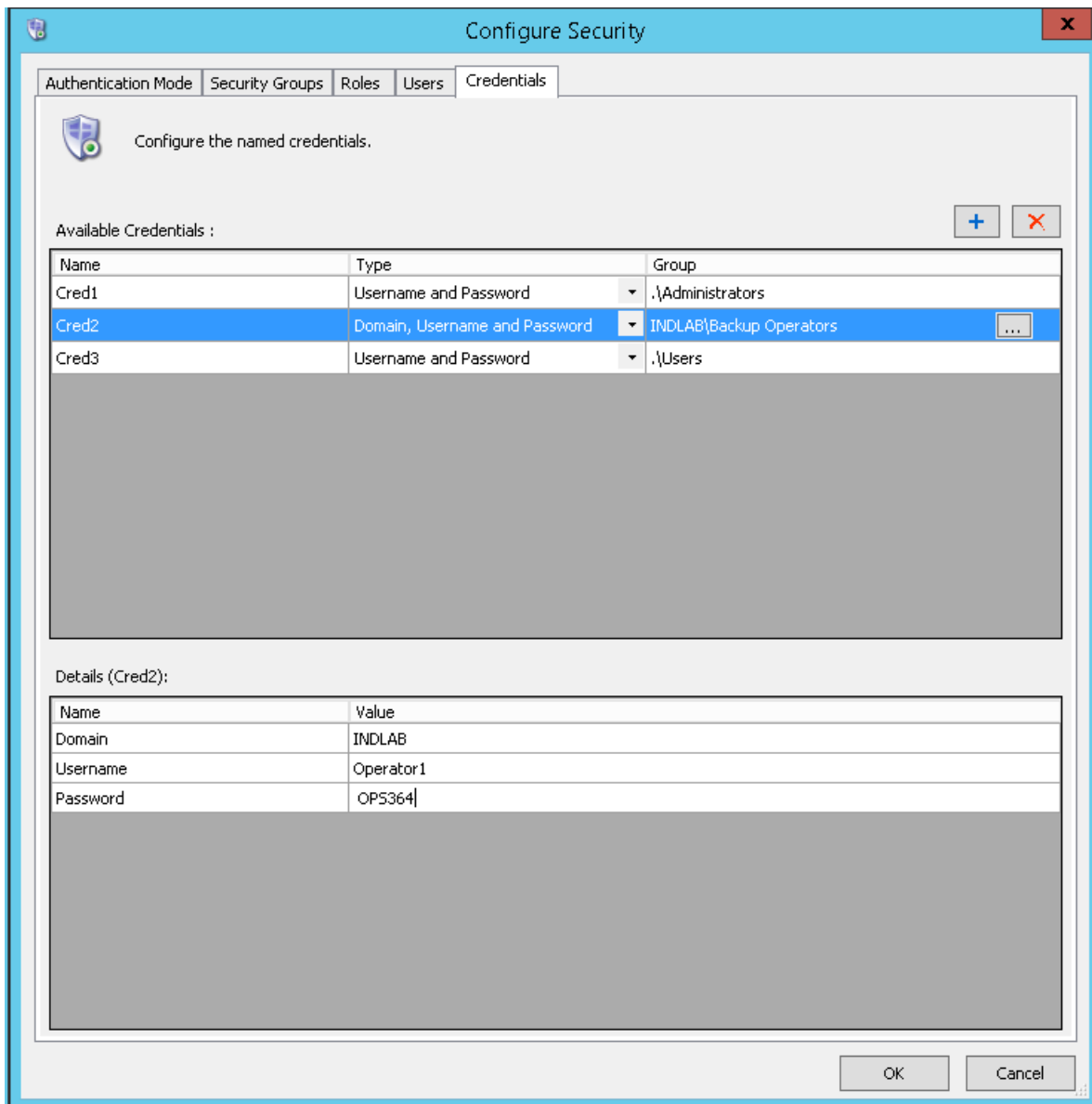
From the **Configure Security** dialog, select the **Credentials** tab. The **Credentials** tab is divided in three columns:

- Name
- Type
- Group

### To create a new credential

1. Click the Add Credential  button. A new credential is added to the **Available Credentials** window.
2. Rename the credential by selecting the default name from the **Name** column and then type the new name.
3. Set the credential **Type** by selecting the down arrow at the right edge of the **Type** column. Type can be one of the following:

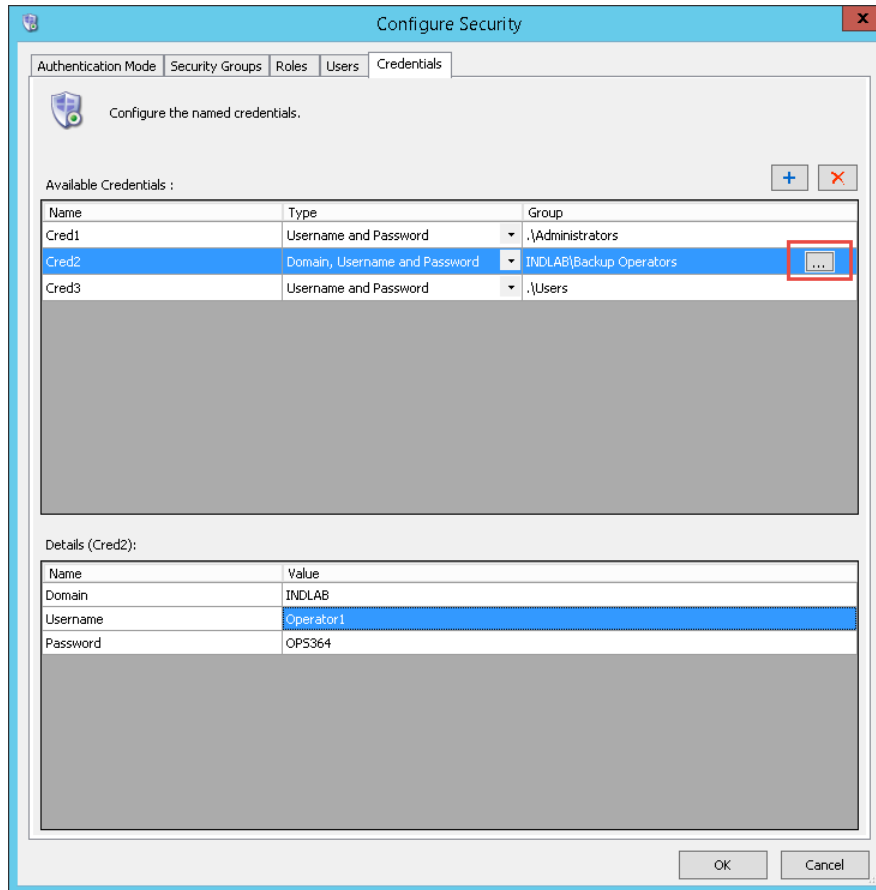
- Username and Password
  - Domain, Username and Password
4. To select **Group** name to associate with the credential:
    - a. Clicking on the ellipses button (...) at the right of **Group** column. The **Select Groups** dialog opens. See *Associate an OS Group with a Credential* on page 491 for more information.
    - b. Choose the applicable OS group name from the list of available OS groups, or type the name of an existing OS group.
  5. Configure the credential after you have created a credential by entering values for Domain, Username, and Password. See *Configure Credentials* on page 493 for additional information.



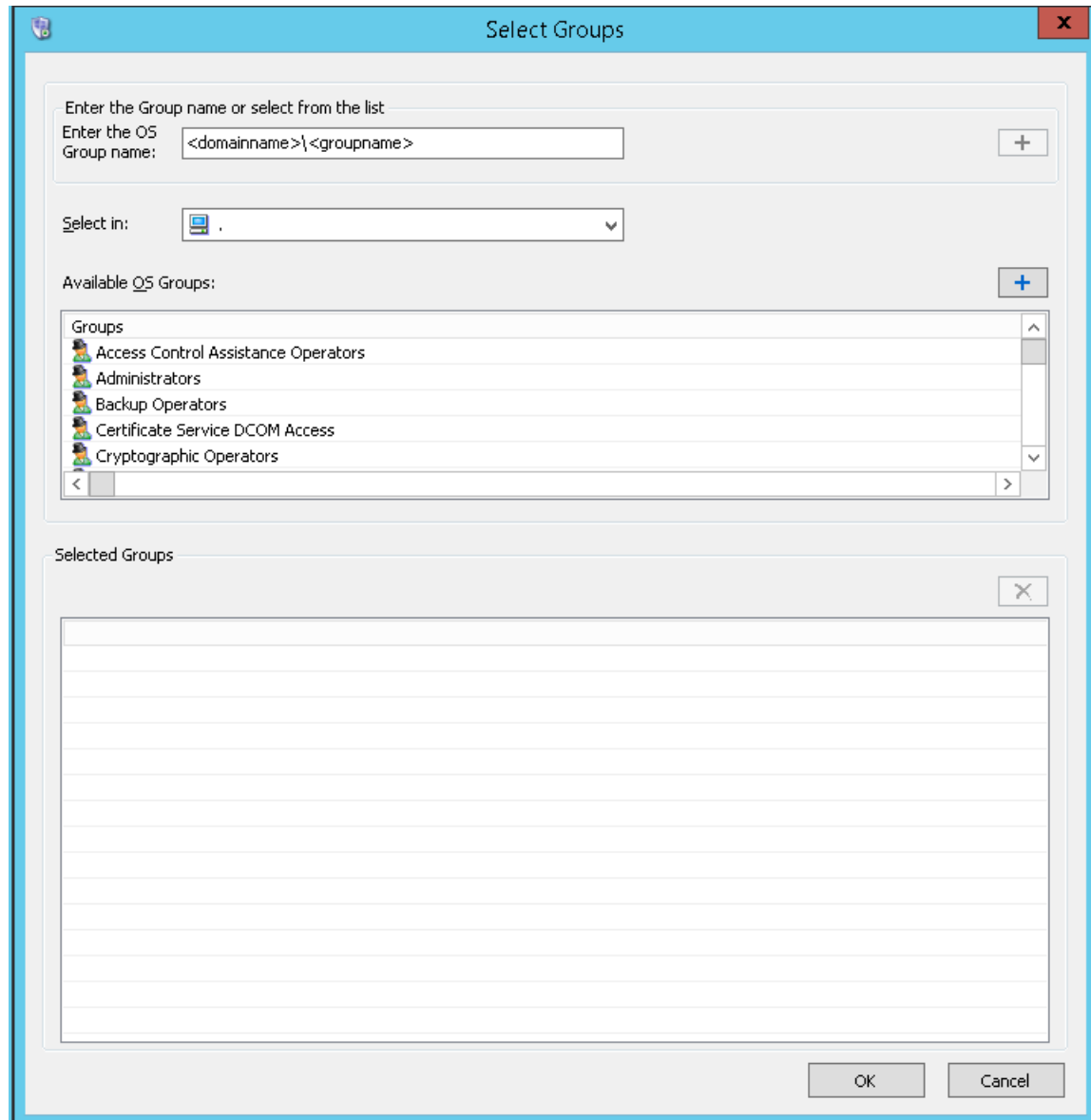
## Associate an OS Group with a Credential

### To associate an OS group with a credential

1. Select the the ellipses button (...) next to the group name in the **Credentials** tab.



The **Select Groups** dialog opens with the local machine selected, and the list of OS groups that exist on the local machine will populate the list of **Available OS Groups**.



2. Choose an existing OS group from the **Available OS Groups** window. Or, you can type the domain and OS group name in the **Enter OS Group name** text box.
3. Add the OS group by clicking the **+** button above the list, on the right side. The OS group is added in the **Selected Groups** window.
  - If you are creating a credential that includes domain, select the applicable domain from the **Select in** dropdown list, then select the OS group name from the **Available OS Groups** window. If the credential type you created does not include domain, you will not be able to select another domain.
  - If you manually enter the domain and/or the OS group name, use a backslash to separate the domain from the group name.
  - If the OS group is on the local machine, you must include ".\" before the group name.

---

**Note:** Only one OS group can be added per credential.

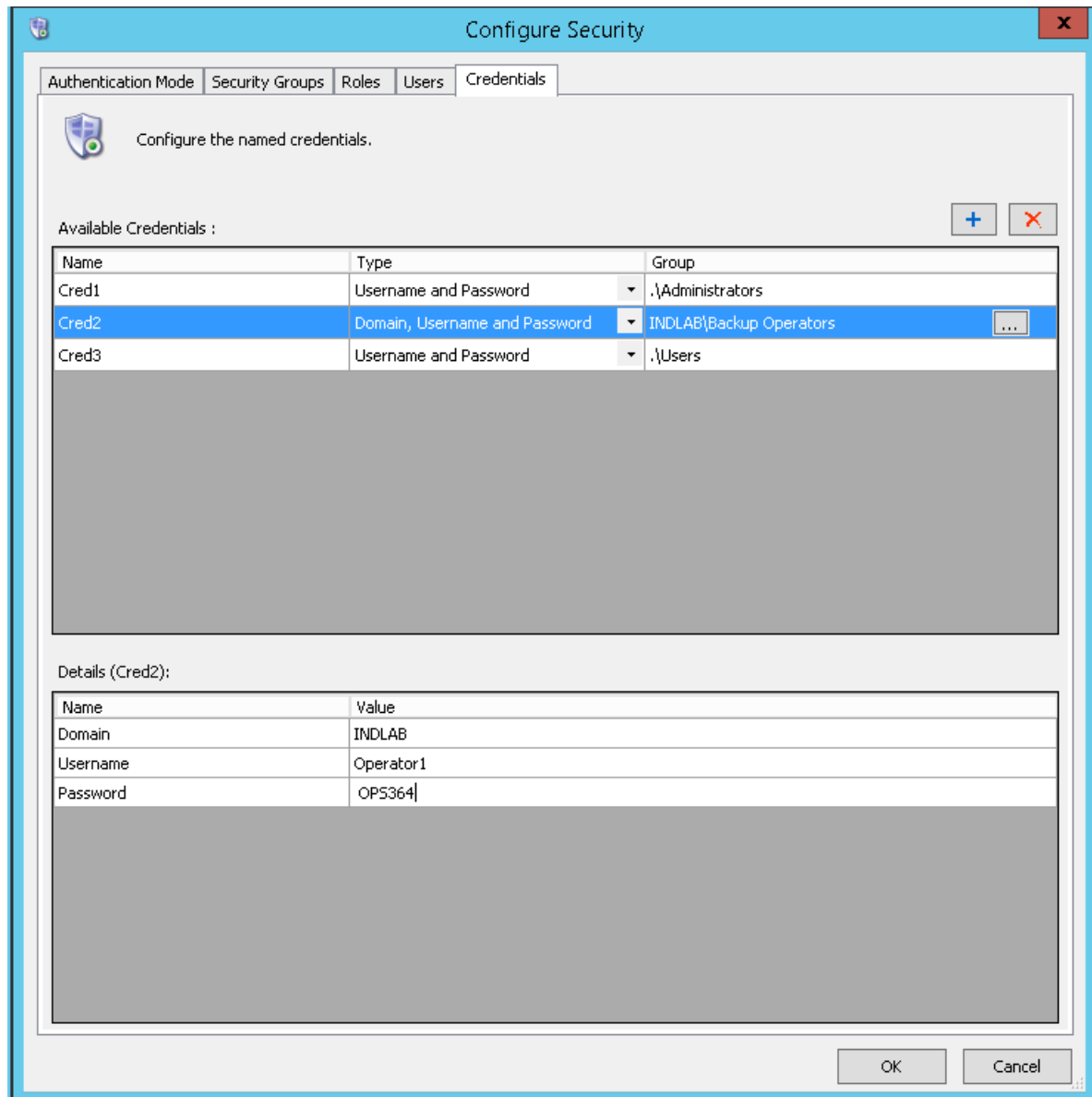
---

4. Click **OK** to close the **Select Groups** dialog and return the **Credentials** tab.

## Configure Credentials

### To configure a credential

1. Select the credential to be configured in the **Available Credentials** window.



2. In the **Details** window, enter a value for each key name.
  - There are two key names for Username and Password credentials.
  - There are three key names for Domain, Username and Password credentials.
  - Null values can be used for Username and Password. If a domain value is required, enter the domain name that was used to create the credential.

## Credential Definitions

**Name:** This is the name of credential. The default credential name is "Credential," and subsequent names are appended by a number that automatically is incremented by one. To rename a credential, just highlight the name and enter new one. Credential names must contain at least one character and must be unique.

Configured credentials are typically shown in a dropdown menu in a ViewApp that has been configured to use credentials. The user can then select the applicable credential. In some cases, an app developer may choose to hard-code a credential name. In this case, no user action is required to access the credential at runtime.

**Type:** Two types of credentials are supported:

- **Username and Password.**
- **Domain, Username and Password.**

---

**Note:** Null values are valid for all fields, except the credential name.

---

**Group:** When you add credentials, you must associate each credential with an OS Group name (local or domain) that has been defined previously.

To associate a credential with an OS group, simply highlight the group name when you add the credential. You can change the group at any time in the **Credentials** tab, but doing so may impact AVEVA OMI apps that have been configured to use the credential. Associations are saved when you click **OK**. See *Associate an OS Group with a Credential* on page 491 for more information.

## About SQL Server Security

SQL Server has its own security settings. When you install Application Server, the following operating system groups and users and user accounts are created:

- aaAdministrators group
- ArcestrA user account
- aaGalaxyOwner user account

---

**Caution:** aaGalaxyOwner and ASBService are reserved OS user names. aaAdministrators and ASBSolution are reserved OS group names. Do not create users or groups with these names.

---

The aaAdministrators group, ArcestrA user account, and aaGalaxyOwner user account are defined for use with SQL Server, and are used for Galaxy operations. Do not delete this group or these accounts. See the section on SQL Server Rights Requirements in the *System Platform Installation Guide* for additional information. This section also describes how to set the SQL Server Security Mode with the **aaConfig SQL** utility.

## Configuring Security

Before you open the security editor for a Galaxy, make sure:

- No other user is connected to the Galaxy.
- All objects in the Galaxy are checked in.
- Your user profile has configuration permissions to change Framework Configuration/Modify Security Model, if security was previously configured.

If you try to open the security editor before these conditions are met, a warning message appears and you are denied access.

---

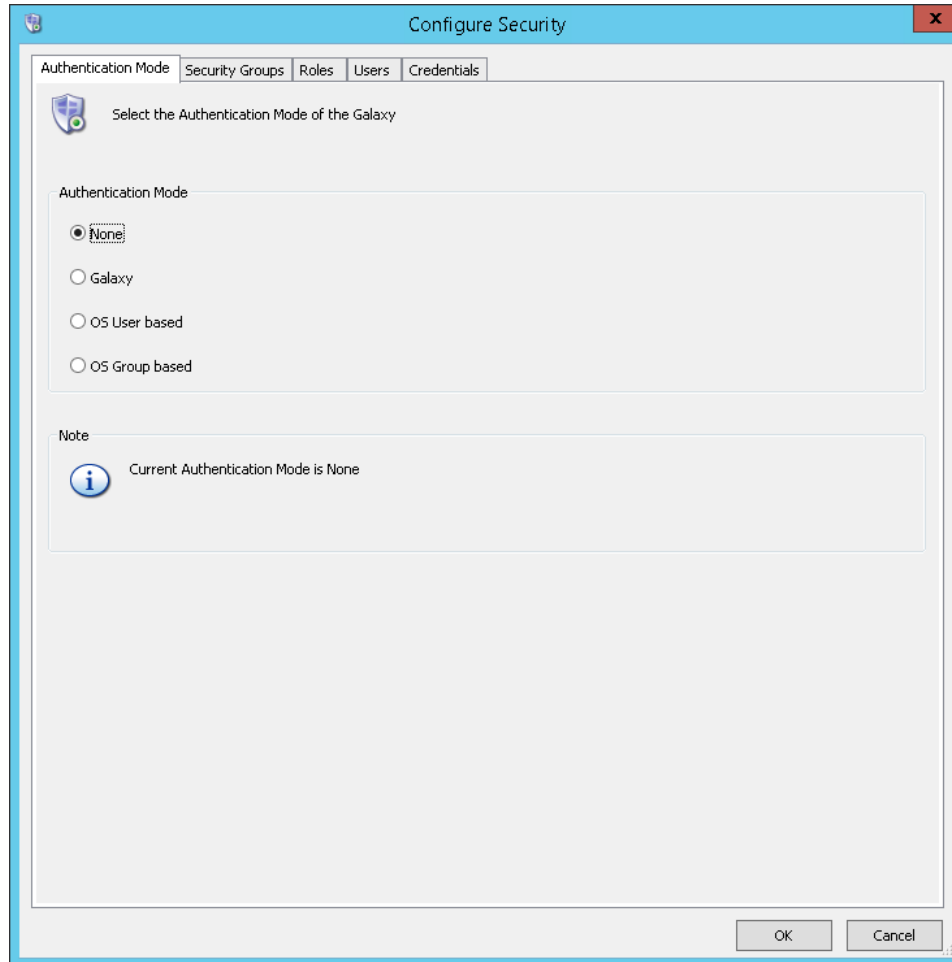
**Caution:** Do not configure security settings of the IDE while an IDE-managed InTouch application is opened for editing in WindowMaker.

---

Other users who try to open the Galaxy while you are configuring security are denied access to the Galaxy.

### To configure Galaxy security

1. On the **Galaxy** menu, click **Configure** and then click **Security**. The **Configure Security** dialog box appears.

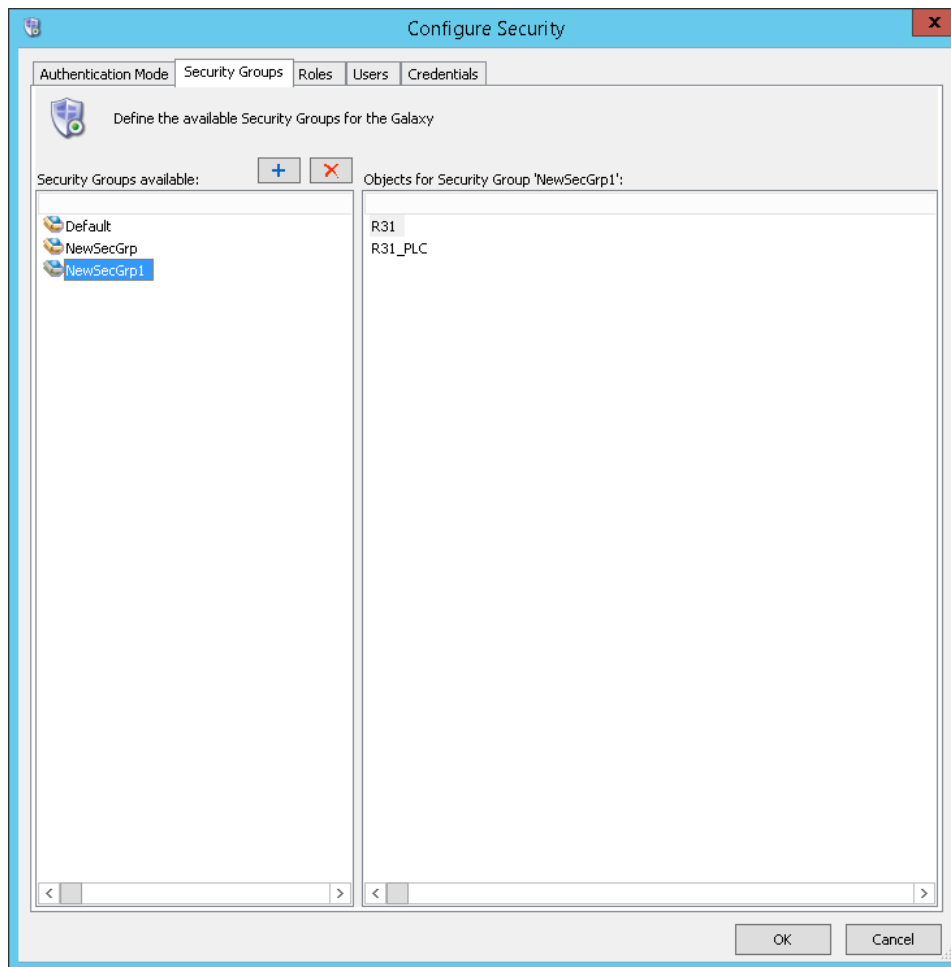


2. On the **Authentication Mode** tab, select the security type you want: Galaxy, OS User Based, or OS Group Based.

If you select the **OS Group-based** authentication mode, you can specify the time-out period for login attempts and role verification. Allowing a longer interval can be helpful if your network connection is slow or intermittent.

- **Login Time:** The time-out period (measured in milli-seconds) during which the system validates the user's membership against the OS groups selected as ArcestrA Roles. Minimum value is 0 (zero), maximum is 9,999,999. The default value is 1,000. If the login time is set to 0 (zero), which turns this feature off, the operation does not time out. Specify a value, based on the speed of your network and the number of groups configured in ArcestrA. The slower the network or the larger the number of groups, the greater the value.

- **Role Update:** The time between each validation attempt per OS group for the user's membership when a log on is attempted. The user membership update is done one role per **Role Update** interval to minimize network usage. The minimum allowed value is 0 (zero) and the maximum is 9,999,999. The default value is 0 (zero), which turns off this feature so the operation does not pause between validating user membership and groups. This option operates independently of the **Login Time** option. Even if **Login Time** times out, the role update operation continues in the background and eventually updates user-to-role relationships for this user in the local cache.
3. Select the **Security Groups** tab to add and configure new security groups. Security groups define which objects (templates and instances) the logged-in user can access. To assign objects to a new Security Group, select the objects listed under the Default Group and drag them to the new group. Security Groups are displayed in Roles tab, under "Operational Permissions." See *Assigning Users to Roles* on page 500 for more information.



- Create a new security group by clicking the **Add** button. Type a unique name for the new group in the **Security Groups Available** pane. Security group names can be up to 32 alphanumeric characters, including a period. The name must include at least one letter and cannot start with \$.

---

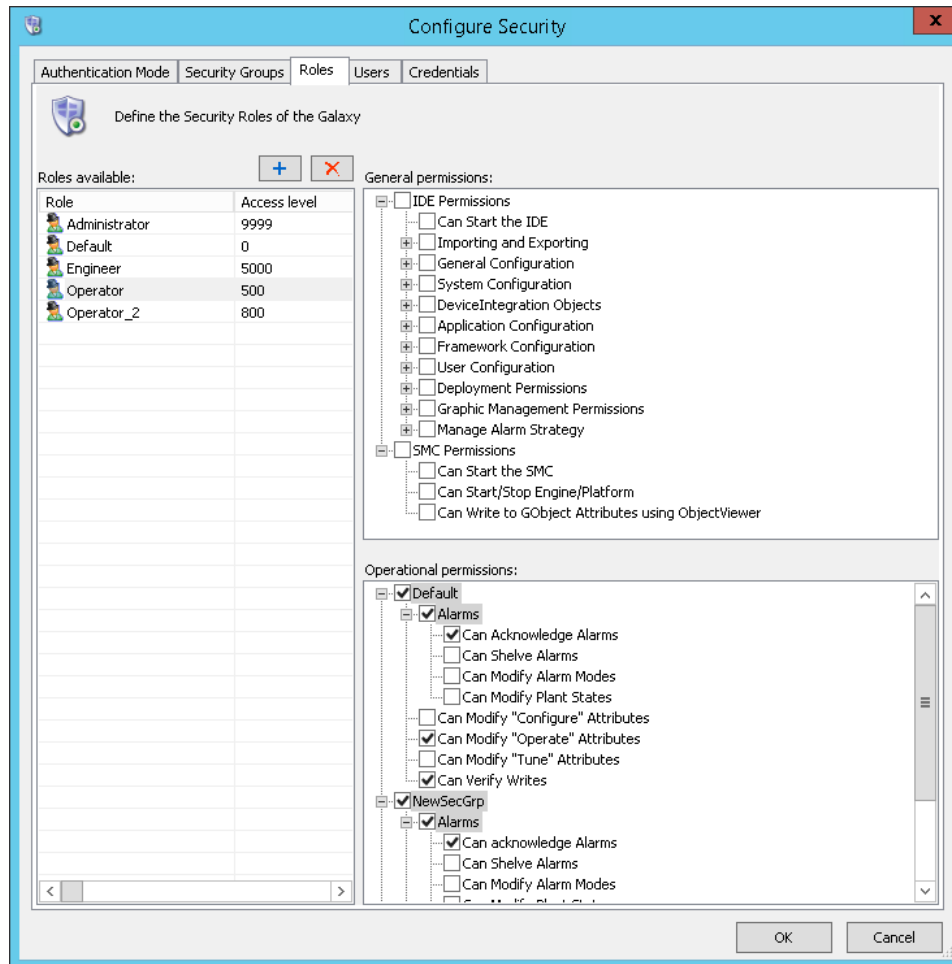
**Note:** Security group names are not case sensitive. Admin is the same as admin.

---

- Objects can only be associated with one security group. To move objects from the Default security group to the new security group, click the **Default** group, then drag objects to the new security group.

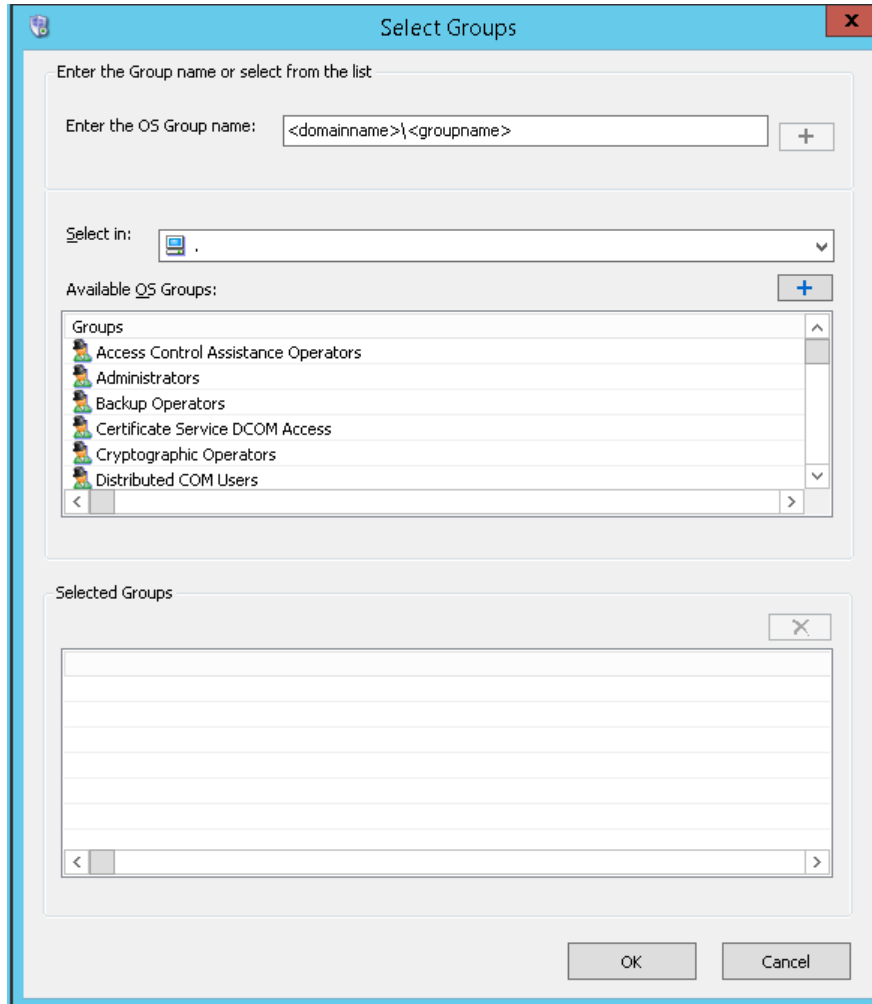


- Select the **Roles** tab to set the permissions (general and operational) for each Security Role, and to create new roles or delete existing roles.



- Remove any unnecessary general and operational permissions from the **Default** role (you can also remove operational permissions from the Administrator role). Leave only the permissions that are needed by all users. See *About Roles* on page 485 for more information.
- If you selected **Galaxy** or **OS User based** authentication mode, you can create new roles by clicking the **Add** button. Type a name for the new role in the **Roles Available** pane. Role names can be up to 512 alphanumeric characters, including a period.

- If you selected the **OS Group based** authentication mode, the **Select Groups** dialog is displayed after you click the **Add** button. This lets you add roles, based on existing OS groups.



- In the "Enter the OS Group name" text box, type the group name (preceded by the domain name, if not part of the local domain) and click the **Add** button, or,
- In the "Select in" dropdown, select the domain that has the OS group you want to add as a role. Then, select the group from the "Available OS Groups" dropdown and click the **Add** button.
- Select the **General** and **Operational Permissions** for the new role. General permissions define Galaxy configuration and management actions that a user is allowed in the IDE and the SMC. Operation permissions define the actions a user can take at run time, for example, permission to acknowledge alarms or modify attribute values.

---

**Important:** In the **General permissions** area, clearing the **Can Start SMC** check box will still allow a user assigned to this role to start the SMC, but not to connect to Platform Manager.

---



---

**Important:** In the **General permissions** area, clearing the **Can Start/Stop the Engine/Platform** check box will still allow the user assigned to this role to set the engine or platform On Scan or Off Scan.

---

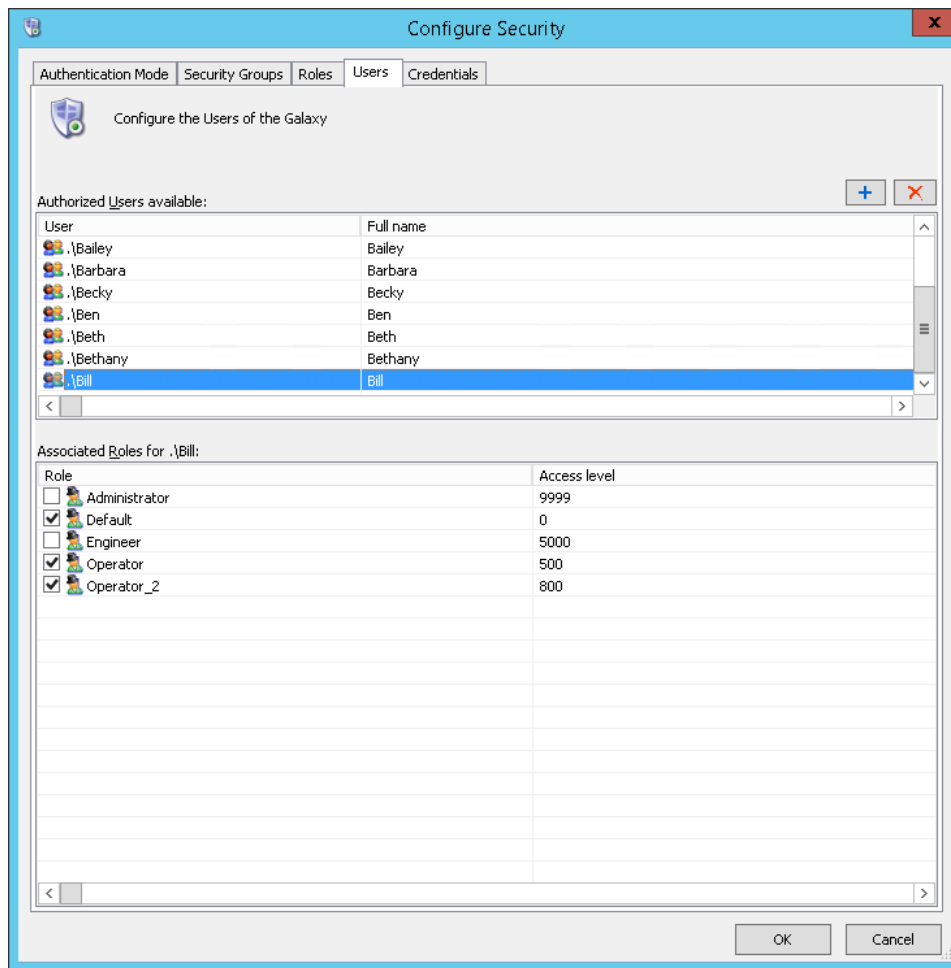
**Important:** If a role is given "Can Modify Deployed Instances" permission, make sure "Can Create/Modify/Delete..." permissions in the System Configuration, Device Integration Objects, and Application Configuration groups are also selected. This provides the role with check in and undo checkout abilities.

5. Select the **Users** tab to associate authorized users with their roles, add new users, or delete existing users.

**Note:** Bold red text indicates that the User or Role is invalid within the selected Authentication Mode.

If you selected **Galaxy** or **OS User based** authentication, create a new user by clicking the **Add** button. User names can be up to 255 alphanumeric characters with no spaces. To configure user roles, see *Assigning Users to Roles* on page 500.

**Note:** You cannot add users if OS Groups based authentication mode is selected.



While viewing Application Server events and alarms in InTouch, the "." appears as the user's domain if it is a local name. Otherwise, it appears as <domain name>\<username>.

6. When you are done, click **OK**. You are prompted to log on to the currently open Galaxy.

**Important:** To improve security, do not use the default user accounts (Administrator or DefaultUser) when logging in to the IDE or other System Platform components. These are reserved system names. In general, you should avoid using generic or easily-guessed user names such as "Admin," "Administrator," "DefaultUser," or "User." Be sure to change the passwords for these accounts to prevent malicious or accidental access to the Galaxy.

## Assigning Users to Roles

After you create users and roles, you can assign users to roles. On the **Users** page, all users in the Galaxy and the roles they are assigned are listed.

New users are automatically associated with the Default role (every user must belong to the Default role). To assign additional roles to a user, select the checkbox for the role while the user name is highlighted.

---

**Note:** All users are automatically assigned to the Default role in addition to any new roles you create and assign to them. The best way to manage permissions is to limit the permissions of the Default role to those permissions you want everyone to have. Then, add users to other roles with permissions for other parts of Application Server.

---

The Administrator user can log on any authentication mode except when security is disabled. When you are logged in as Administrator on the Galaxy Repository node, you can change the password of any Galaxy user without providing the old password.

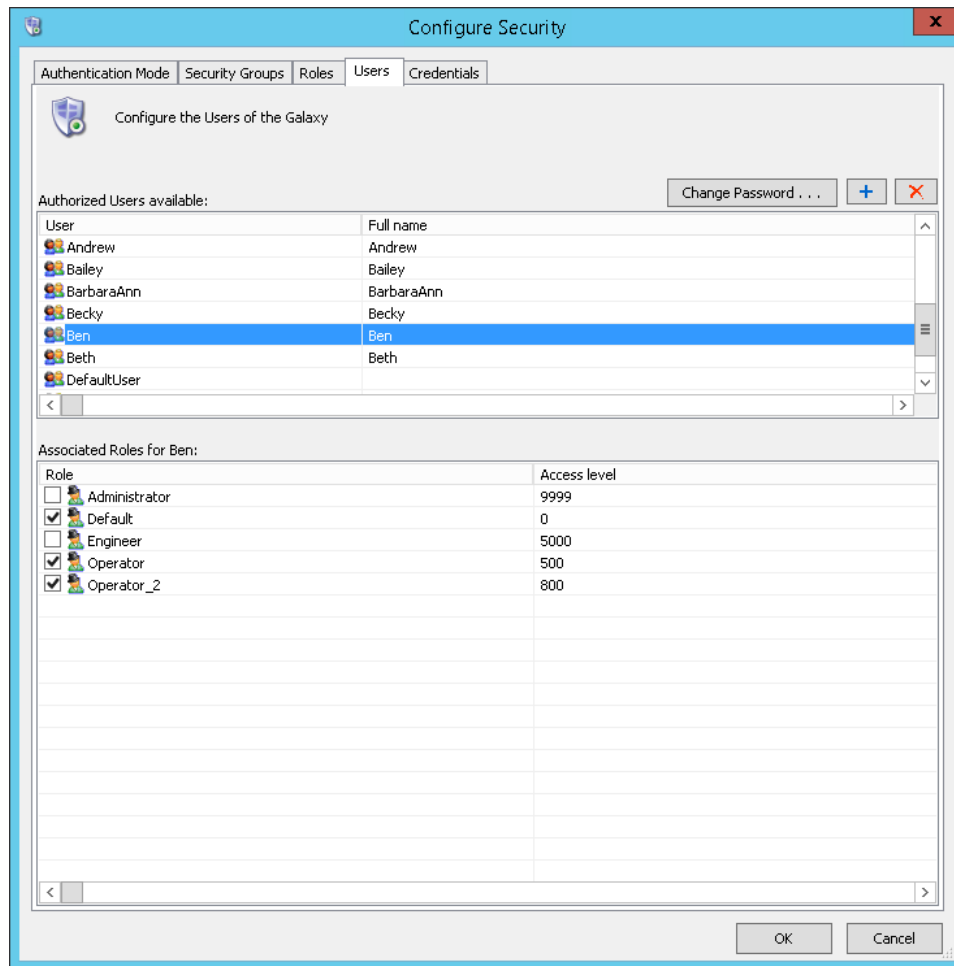
- In Galaxy authentication mode, you can edit the **User Name** in the **Change Password** dialog box.
- In OS-based authentication modes, the **User Name** of the OS user is shown. User information cannot be edited.

You can assign users to more than one role.

### To assign a role to a user

1. On the **Galaxy** menu, click **Configure** and then click **Security**. The **Configure Security** dialog box appears.
2. Click the **Users** tab.

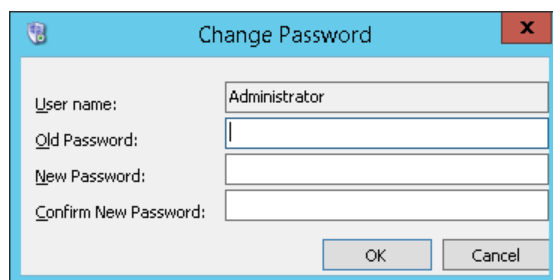
**Note:** The **Change Password** button is displayed only if **Galaxy authentication mode** is selected.



3. Select the user in the **Authorized Users available** area. Select a role in the **Associated Roles for <user name>** area.

Every user is assigned to the Default role. Add other roles that they need to perform their job functions.

4. Provide each user with a password. Make sure that you set new passwords for the **Administrator** and **DefaultUser** accounts.
  - If you are using Galaxy Authentication Mode, click **Change Password**. The **Change Password** dialog box appears.



You can leave the "Old Password" field blank when you initially set the password for the Administrator and DefaultUser accounts.

- If you are using an OS-based security mode, use the Windows Control Panel to manage passwords.

---

**Important:** If an OS-based security mode is selected on the **Authentication Mode** page, changing a user's password changes the OS password for the user. Any ArcestrA password may be set to a maximum of 127 characters.

---

5. When you have finished configuring users, click **OK**. The security information is used in the configuration, administration and run-time environment to authenticate users.

## Deleting Security Groups

You can delete a security group you no longer need. Before you can delete a security group, make sure no objects are associated with it.

You cannot delete the Default security group.

---

**Note:** The aaAdministrators group is required. If you accidentally delete it from Windows, you can run either the **Change Network Account** utility or the **aaConfig SQL** utility to restore it. If you delete the group from SQL Server, you must run **aaConfig SQL**. See the *System Platform Installation Guide* for more information.

---

### To delete a security group

1. On the **Galaxy** menu, click **Configure** and then click **Security**. The **Configure Security** dialog box appears. Click the **Security Groups** page.
2. On the **Security** page, select the group you want to delete.
3. Click the **Delete** button.

## Deleting Roles

You can delete roles you no longer need. You cannot delete a role if any users are associated with it.

You cannot delete the Default and Administrator roles.

### To delete a role

1. On the **Galaxy** menu, click **Configure** and then click **Security**. The **Configure Security** dialog box appears. Click the **Roles** tab.
2. On the **Role** page, select the role you want to delete.
3. Click the **Delete** button.

## Deleting Users

You can delete users you no longer need.

---

**Note:** You cannot delete a user who is currently logged on.

---

### To delete a user

1. On the **Galaxy** menu, click **Configure** and then click **Security**. The **Configure Security** dialog box appears. Click the **Users** tab.
2. On the **User** page, select the user you want to delete.
3. Click the **Delete** button.

---

**Note:** The ArcestrA user account and aaGalaxyOwner account are required. If you accidentally delete the ArcestrA user account from Windows, you can run either the **Change Network Account** utility or the **aaConfig SQL** utility to restore it. If you delete either account from SQL Server, you must run **aaConfigSQL** to restore it. *See the System Platform Installation Guide* for more information.

---

## About OS Group-based Security

If you use OS Group-based Authentication Mode, make sure you understand the Windows operating system, particularly its user permissions, groups and security features. ArcestrA OS Group-based security uses these Windows features. For more help, see the Microsoft online help or a 3rd party book about Windows security.

When you use local OS Groups as Roles, each node within the Galaxy must have the same OS Users, Groups, and user-group mappings to get the same level of access to the user at each node.

---

**Note:** Application Server uses Pre-Windows 2000 Group names for OS Group-based Authentication Mode, not the Active Directory Common Name.

---

## Connecting to a Remote Node for the First Time

A newly-added user working on a computer with no access to the Galaxy Repository node cannot write to an attribute on a remote node if that user has never logged on to the remote node. This is true even if the user is given sufficient run-time operational permissions to do writes. To enable remote writing capabilities, log on to the remote node at least one time.

When attempting to access a remote Galaxy Repository node that contains an IDE, the user who is logging in must have admin privileges.

## Cached Data at Log In

If you log on to ArcestrA on a workstation that belongs to Domain A and Domain Controller A fails, locally cached login data is used on subsequent log on attempts. When the domain controller returns to operation, your log on fails during the time period that trusts are being reestablished by the controller.

If, during the controller outage, your username/password data changed, you can use the old log on data if you intend to work locally.

If you want to perform remote operations, you should log on with the new log on data. If that fails, the trusts are being reestablished by the controller, and you should retry at a later time.

The user's full name is not available to any client (for example, an InTouch window) if the domain controller is disconnected from the network when the user logs on to ArcestrA for the first time.

If the user previously logged on to ArcestrA when the domain controller was connected, the user's full name is still available to the client from data stored in cache even if the domain controller is disconnected when the user subsequently logs on to ArcestrA.

## Mixed or Native Domains

Your unique listing maps to the list of domains and user groups you see when you use the Windows tool for managing your domain. The list of domains and user groups appears differently in the group browser depending on whether your domain is configured as a Mixed or Native domain.

The two modes cannot be used interchangeably. We recommend using Native-mode domains for the mode's advantages in security and use of domain local groups.

## Using Domain Local Groups

Domain local groups (DLGs) cannot be used in a mixed-mode domain. If you use DLGs to define role permissions, when group members attempt to access secured content, those group members might see "access denied" error messages.

This problem can occur if your domain is operating in mixed mode. In mixed mode, the scope of a DLG is limited to the domain controllers only. The DLG is not valid for member servers. The DLG can still appear valid as certain applications, such as SharePoint Portal Server resources, do not filter out invalid DLG entries.

If you grant the necessary role permissions to the domain user accounts individually, those users can gain access to the secured content, but this approach nullifies one of the advantages of using DLGs.

## Using Security and Distribution Domain Configurations

You must select one of two group types when you create a new group.

- Security Group Type

Use security groups to manage user and computer access to shared resources and to control which users receive group policy settings.

- Distribution Group Type

Use distribution groups as email distribution lists with email applications such as Microsoft Exchange or Microsoft Office Outlook. You cannot use distribution groups to assign permissions or to filter group policy settings.

For this reason, a domain group configured as a distribution type rather than a security type cannot be used for security purposes.

## Using InTouch Access Levels Security

The **Roles** page includes the **Access Level** column. The **Access Level** is an InTouch function.

In InTouch, access levels are a schema for prioritizing run-time functions. In the ArchestrA security model, it only maps to InTouch values and has no prioritizing characteristics.

The maximum value is 9999 and the minimum is 0 (zero). If a user is assigned more than one role with different access levels, the higher access value is passed to InTouch.

## Using Secured and Verified Writes

You can assign Secured Write or Verified Write security classification to a configurable attribute. These security classifications require authentication to perform run-time writes to the configured attribute. With authentication, users can write to such attributes by:

- Any assignment in a script that sets the value of the attribute, such as "A=B";  
where A references an attribute that is configured for Secured Write or Verified Write security classification.
- Any action on an animation graphic that alters the value of an attribute that has Signed Write or Verified Write security classification, such as a user input, a slider, an up/down button on a counter, or any other such actions.
- A script that uses the SignedWrite() function.

For information about the SignedWrite() function, see Chapter 2, "QuickScript .NET Functions," in the *Application Server Scripting Guide*.



The operator and verifier provide credentials for authentication by entering a valid security account (domain name, user name, and password), or by using a Smart Card if a Smart Card reader is attached to the system.

---

**Note:** Smart Card authentication is not supported in multi-galaxy environments for read/write operations to remote galaxies.

---

Operators can write to attributes configured with Secured Write or Verified Write security classification even if another user is logged on. This does not affect the session of the logged-on user.

- The operator must have the Can Modify Operate Attributes operational permission to perform the Verified Write.
- The verifier must have the Can Verify Writes operational permission to confirm the Verified Write.

For more information, see *Set Object Security* on page 73 and *About Roles* on page 485.

---

**Note:** The Secured and Verified Write features work only when security is enabled on both the Galaxy and InTouch applications and do not work if either Galaxy or InTouch security is set to None.

---

## Using Configurable Descriptions and Comments for Secured and Verified Writes

At run time, the Secured Write or Verified Write dialog boxes appear when the operator attempts to write to an attribute configured for Secured Write or Verified Write. The dialogs enable configurable user input that can be used to provide information about the Secured or Verified Write.

Use the SignedWrite() script function to configure these inputs. The script function can be used only for Archedra attributes configured with Secured Write or Verified Write. It can be used only for symbol scripts and not for application object scripts.

### Reason Description

The reason description is specific to an Archedra attribute. It explains the purpose of the attribute and the impact of changing it. If you do not configure a reason description, the reason description area is blank.

It is possible to use a script to directly assign a value to an attribute or field attribute that requires Secured or Verified Write. When the script is executed, the Secured or Verified Write dialog box appears and the reason description area displays an Attribute description, if there is one. If the Attribute does not have a description, then the reason description area displays the description of the Application Object to which the attribute belongs.

### Predefined Comment List

The predefined **Comment** list enables the operator to comment on changing the attribute by selecting from a predefined list of comments.

### Editable Comment Box

You can enable the operator to enter a comment in the **Comment** box.

For more information on configuring the reason description and the predefined **Comment** list and box, see "Working with the SignedWrite() Function for Secured and Verified Writes" in Chapter 3, "Managing Symbols," in the *Creating and Managing Industrial Graphics User Guide*.

---

**Note** The reason message and **Comment** box and list are displayed in the Secured Write or Verified Write dialog box only in InTouch WindowViewer and not in ObjectViewer.

---

## About Secured and Verified Writes Logged Information

Following a Secured or Verified Write a security Event is written to the event log, including the following information:

- The signee name
- Verifier name, if any
- Type of write: "Secured Write" or "Verified Write"
- Comment, if any entered by user
- Reason Description, if any provided
- Attribute or field attribute description, if any, or the Short Description of the Application Object, if no Attribute description exists

# CHAPTER 15

## Working with Languages

This section describes the features and procedures for configuring languages and enabling run-time language switching.

### Defining and Configuring Galaxy Languages

Application Server language features enable you to develop applications that can be switched to another language at run time. To enable run-time language switching, you must:

- Configure multiple languages for the application.
- Export your application text for offline translation.
- Translate one or more exported dictionary files.
- Import one or more translated dictionary files.

You can set the Galaxy default language and you can define additional languages for purposes of switching symbol and alarm comment languages at run time.

The default language is set and languages are added only at the Galaxy level.

### Graphics Language Switching

Run-time language switching applies to all portions of the graphics or animations that you create or configure using the Industrial Graphics Editor or InTouch WindowMaker. The language to display is set when the application is being shown at run time.

You can only view translations at design time for Industrial Graphics in the Graphic Editor. You cannot switch languages in WindowMaker at design time to see the translations.

### Alarm Comment Language Switching

The alarm comment language switching feature allows you to format and export the configured alarm comments from attributes in a Galaxy to external files. The purpose of exporting the alarm comments to external files is to produce InTouch-compatible localized files to support language switching of alarm comments in the InTouch runtime environment.

You can import the alarm comment files back into the Galaxy after translation for convenient updating and re-export.

If you change an alarm comment after exporting the alarm comments, you must re-export the alarm comments. The translated alarm comments imported into InTouch and displayed at run time do not change dynamically when the alarm comment is edited.

### Workflow

A typical workflow for a large Galaxy might consist of the following:

1. Export the Galaxy elements to be translated as a single file that is more manageable by the translator.
2. Import the translated file back into the Galaxy.
3. Export the translated file by Areas for use by InTouch applications.

4. Import the translated language file into InTouch.
  - In stand-alone InTouch, you can perform a single import operation.
  - In managed InTouch, you must import the language file into each managed InTouch application.
5. For InTouch applications that only target subareas of the Galaxy, exporting by Area is a more optimal workflow.

---

**Important:** Exporting, translating and importing the translated files back into the Galaxy are important steps for translating and managing translated files. Note that run-time language switching will not function if you do not import the translated language file as described in number 4.

---

## Configuring Languages for a Galaxy

The language settings of the Galaxy control which languages are available to symbols and alarm comments. You cannot add a language at the symbol or attribute level. Languages are only added at the Galaxy level using the IDE.

When you open a symbol for editing using the Graphic Editor or open a managed InTouch application in WindowMaker, the language settings are retrieved from the Galaxy.

For example, you configure English and French languages for the Galaxy. You open symbol S1 in the Graphic Editor. Symbol S1 is now configured with English and French languages. Using the IDE, you add German to the list of configured languages. You must close the Graphic Editor and open symbol S1 again to see the German language available for the symbol.

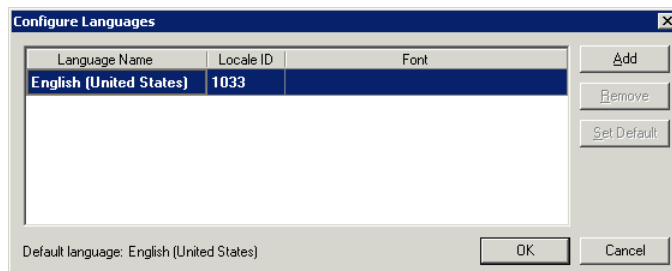
For more information about language switching in Industrial Graphics, see the *Creating and Managing Industrial Graphics User Guide*.

## Adding a Language to a Galaxy

Every Galaxy is associated with a base language. You must configure any additional languages that you want to support.

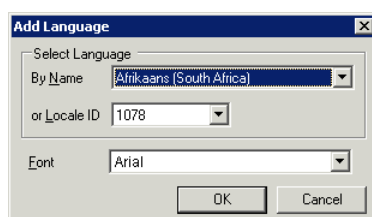
### To add a language for a Galaxy

1. Using the System Platform IDE, open the Galaxy for which you want to add a language.
2. On the **Galaxy** menu, point to **Configure**, and then click **Languages**. The **Configure Languages** dialog box appears.

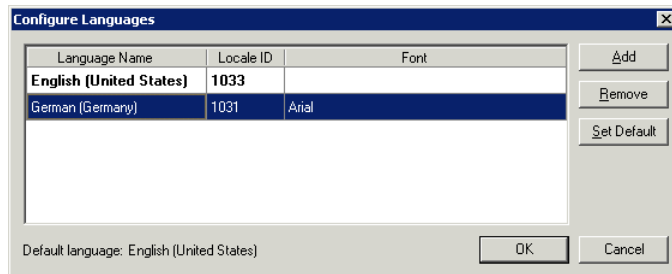


The **Configure Languages** dialog box shows the base (default) language of the Galaxy.

3. Click **Add**. The **Add Language** dialog box appears.



4. Specify the language and font for the translated text.
  - In the **By Name** or the **Locale ID** list, click the language to add. If you select the language by name, the corresponding locale ID appears in the **Locale ID** list, and vice versa.
  - In the **Font** list, click the name of the font.
5. Click **OK** to close the **Add Language** dialog box. The language you configured is listed in the **Configure Languages** dialog box.



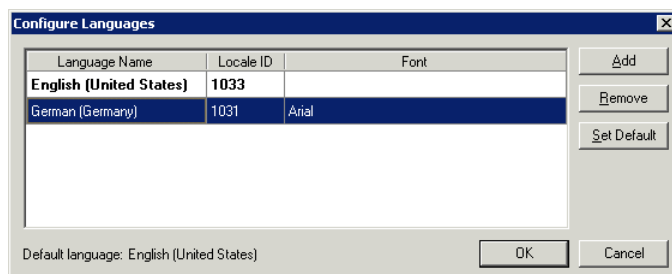
6. To add more languages, repeat steps 3 through 5.
7. To remove a language, select the row in the list and then click **Remove**.
8. To specify a particular language as the default, select the row in the list and then click **Set Default**.
9. When you are done, click **OK**.

## Removing a Language from a Galaxy

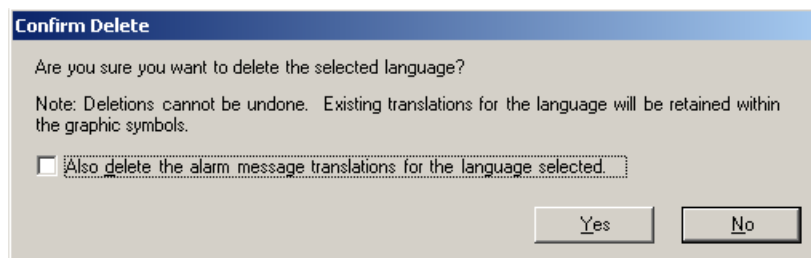
You cannot remove the default language. At least one language must be configured for a Galaxy.

### To remove a language for a Galaxy

1. Using the System Platform IDE, open the Galaxy for which you want to remove a language.
2. On the **Galaxy** menu, point to **Configure**, and then click **Languages**. The **Configure Languages** dialog box appears.



3. Select the row of the language to remove and then click **Remove**. A **Confirm Delete** dialog box appears.



4. The dialog box provides an additional language deletion option. Select the check box to delete alarm comment translations for the language selected. Leave the check box empty (unselected) if you want to keep the alarm comment translations for the selected language.

- Click **Yes** to confirm deletion of the selected language as well as the alarm comment translations for that language if also selected.

## Modifying the Font for a Language

The default font for all languages is Arial. You can change the font setting for a language that you have already added. You cannot edit the font for the default language.

The configured font for a language is used in design time when a specific translation is supplied for a language. It is also propagated to all translations for secondary languages when the fonts are not specifically overridden by you.

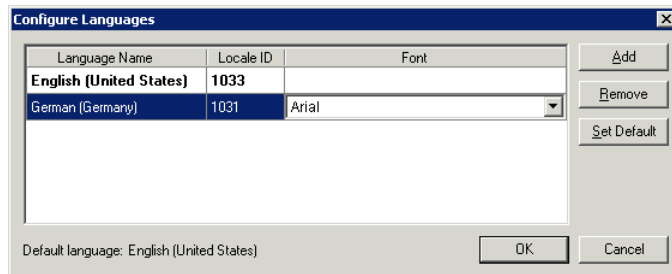
For example, if you create a text element and provide translation for the secondary language without modifying the font, the text is shown using the font from the Galaxy. However, if you manually modify the font, then the text will always use the chosen font. This same behavior also applies to run time.

For an Industrial graphic to show the updated font from the Galaxy, WindowViewer must be restarted. No notification is provided to WindowViewer when the Galaxy font changes.

All managed InTouch applications are updated to use the new font for the selected language.

### To change the font settings for a configured language

- Using the System Platform IDE, open the Galaxy for which you want to change the font for a configured language.
- On the **Galaxy** menu, point to **Configure**, and then click **Languages**. The **Configure Languages** dialog box appears.



- In the list of languages, select the target language.
- In the **Font** column, double-click on the name of the font so that a drop-down arrow appears.
- Click the name of the new font.
- Click **OK**.

## Changing the Default Language for a Galaxy

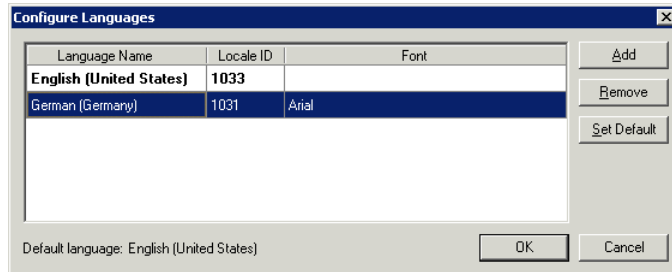
When you change the default language for the Galaxy:

- The new default language is shown when the Graphic Editor opens.
- The base language for translation export is changed to the new default language.
- You cannot import translations for that language.
- Symbols opened in design time use the default language when showing text that does not have a specific translation in a secondary language.
- Symbols shown at run time use the default language if specific translated values are not provided for the currently viewed language.
- Thumbnails for symbols are shown using the default language.

The default language for managed InTouch applications is always the same as the InTouch installed language.

### To change the default language for a Galaxy

1. Using the System Platform IDE, open the Galaxy for which you want to change the default language.
2. On the **Galaxy** menu, point to **Configure**, and then click **Languages**. The **Configure Languages** dialog box appears.



The **Configure Languages** dialog box shows the base language of the Galaxy.

3. In the list of languages, select the target language.
4. Click **Set Default**.
5. Click **OK**.

---

**Note:** Changing the default language does not change the alarm comment default language. Alarm comments are always stored as the Galaxy Repository default language, which is the locale of the operating system on which the Galaxy is created.

---

## Exporting Symbol Text for Offline Translation

If your application has many strings, you typically send the text strings for your graphic symbols out for bulk translation. You can export symbol strings for translation and modify them using a text editor, an XML editor, or a spreadsheet program like Microsoft Excel.

We recommend that you do not make changes to symbols while symbol text is being translated.

If you make changes to your application after you export your dictionary files, you must export the dictionary file again. For more information, see *Exporting Symbol Text to an Existing Dictionary File* on page 515.

You can only export the text strings for one language at a time. You cannot export text strings for the default language.

Each symbol in a Galaxy is only exported one time, no matter how many times it is referenced.

When you export the text, you specify a folder for the dictionary files. Creating a new folder to export phrases for each language makes it easy to manage dictionary files. For example, ...\*Galaxy1*\My German Files\.

Also, all exported files have the following convention: *<GalaxyName>\_<LanguageID>.xml*. If you will be exporting language data for different objects at different times, use separate target directories to prevent subsequent exports from overwriting the first export.

When you export the dictionary for an application, the files are .xml files that you can edit using Microsoft Excel.

## Types of Language Dictionary Files

A language export creates the following types of dictionary files:

- Dictionary file for all Graphics Toolbox symbols, template symbols, and AutomationObject symbols. The file naming convention is: <GalaxyName>\_<LanguageID>.xml. For example, if the Galaxy name is TestSample and the language being exported is French (Language ID = 1036) then the file name is TestSample\_1036.xml.
- Dictionary file for each managed InTouch application.
- Dictionary file for each SmartSymbol in managed InTouch applications.

---

**Note:** The alarm comment export files are named and formatted differently from symbol dictionary files. For further information see *Exporting Alarm Comments for Offline Translation* on page 521.

---

## Exporting Language Data for All Symbols in a Galaxy

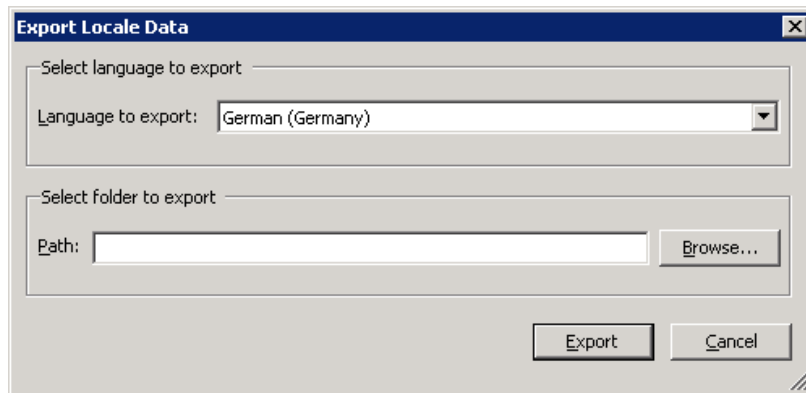
You can export language data for all symbols in a Galaxy at one time. The export contains language data for:

- Graphic Toolbox symbols.
- All symbols contained in AutomationObject templates, except for an \$InTouchViewApp template.
- All symbols contained in AutomationObject instances.

The export operation only applies to symbols that are checked in.

### To export language data for all symbols

1. Using the System Platform IDE, open the Galaxy for which you want to export symbol text.
2. On the **Galaxy** menu, point to **Export**, and then click **All Symbols**. The **Export Locale Data** dialog box appears.



3. Configure the export settings.
  - In the **Languages to export** list, select the language dictionary to export. The default language is not listed.
  - In the **Path** box, type the folder to which you want to export the dictionary. Click **Browse** to select an existing folder or create a new folder.
4. Click **Export**. The export progress is shown.
5. Click **Close**.

## Exporting Language Data for Specific Objects

You can export language data for these types of Galaxy objects:

- Graphic ToolBox symbols

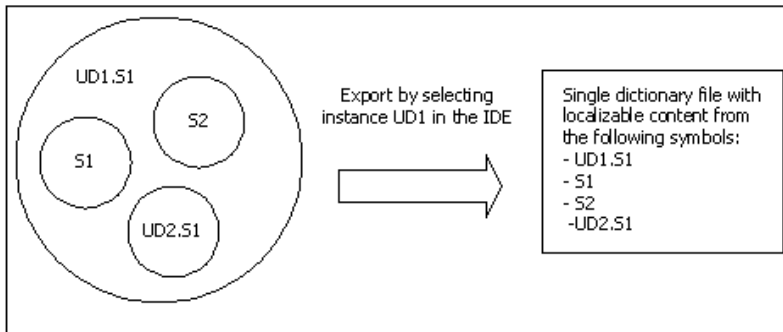


- AutomationObject templates and instances
- \$InTouchViewApp templates and instances.

All symbols in an instance or template are exported.

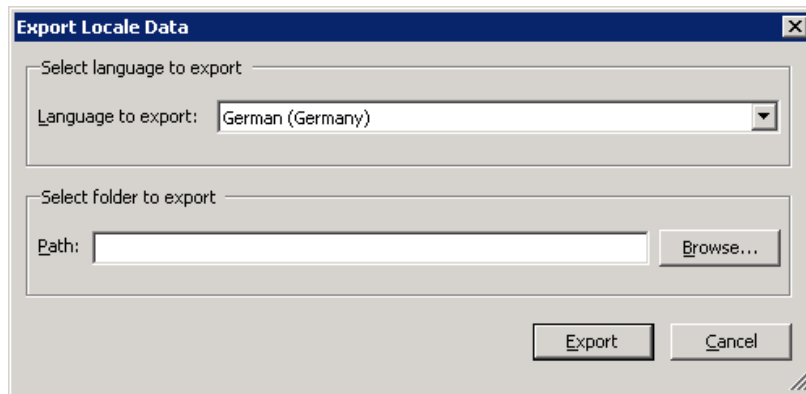
When you export the language data for a symbol, the language data for all symbols referenced by the symbol is also exported. The exported symbols can be referenced through direct embedding or through a show symbol animation.

For example, a Galaxy has Symbol1 and Symbol2 defined in the Graphic Toolbox. There are two instances called UserDefined1 and UserDefined2. UserDefined1 includes Symbol1. UserDefined2 includes Symbol1 and Symbol2. The instance symbol UserDefined1.Symbol1 embeds Symbol1 and Symbol2 from the Graphics Toolbox and one instance symbol UserDefined2.Symbol1. If you select the instance UserDefined1 and export the language data, then the language data would also be exported for the symbols Symbol1, Symbol2, and UserDefined2.Symbol1.



### To export language data for specific objects

1. Using the System Platform IDE, open the Galaxy for which you want to export symbol text.
2. Select one or more objects to export.
3. On the **Galaxy** menu, point to **Export**, and then click **Localization(s)**. The **Export Locale Data** dialog box appears.



4. Configure the export settings.
  - In the **Language to export** list, select the language dictionary to export. The default language is not listed.
  - In the **Path** box, type the folder to which you want to export the dictionary. Click **Browse** to select an existing folder or create a new folder.
5. Click **Export**. The export progress is shown.
6. Click **Close**.

## Exporting Symbol Language Data for a Managed InTouch Application

You export language data for a managed InTouch application using the IDE. You cannot export translations from within the InTouch HMI.

The export includes strings for:

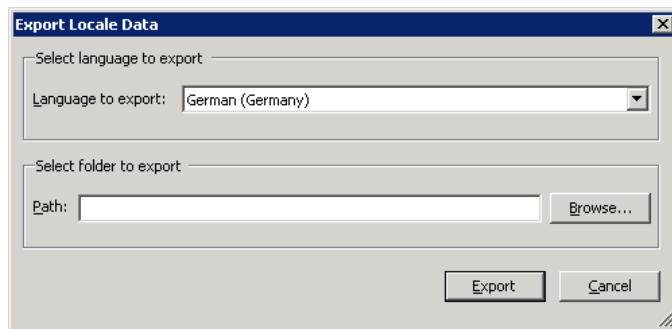
- All InTouch windows
- All SmartSymbols
- Industrial Graphics referenced by the \$InTouchViewApp template

The export causes a cascade export of all referenced Industrial Graphics.

When you export language data for a managed InTouch application, the default language for the Galaxy is ignored. The InTouch default locale is always the installed InTouch locale. If the InTouch installed locale and the Galaxy default language are not the same, the export is skipped for the selected InTouchViewApp and a message is logged.

### To export language data for a managed InTouch application

1. Using the System Platform IDE, open the Galaxy for which you want to export symbol text.
2. Select one or more managed InTouch applications.
3. On the **Galaxy** menu, point to **Export**, and then click **Localization(s)**. The **Export Locale Data** dialog box appears.



4. Configure the export settings.
  - In the **Language to export** list, select the language dictionary to export. The default language is not listed.
  - In the **Path** box, type the folder to which you want to export the dictionary. Click **Browse** to select an existing folder or create a new folder.
5. Click **Export**. The export progress is shown.
6. Click **Close**.

## Exporting Symbol Language Data for a Published InTouch Application

If you export languages for a published InTouch application, the following are not included:

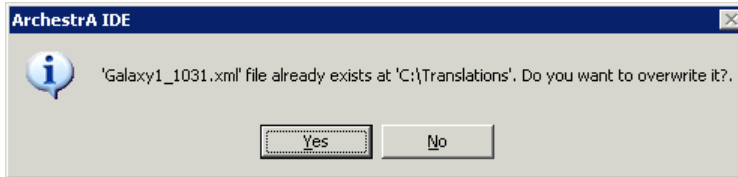
- ArchestrA embedded symbols
- Custom property overrides
- String overrides.

The export only includes native InTouch translations.

## Exporting Symbol Text to an Existing Dictionary File

If you change your application after you translate the text strings, you need to export the text again. For more information, see *Exporting Symbol Text for Offline Translation* on page 511.

If you export more than one time to the same directory, a message box appears.



If you click **Yes**, the existing .xml files are deleted and the current text for symbols in the Galaxy are exported as a new .xml file.

Any existing translations for a symbol are reflected in the new .xml file.

## Translating Exported Symbol Language Files

The procedures and tools for translating the exported language files are similar for both symbol text and for alarm comment text. However, there are important differences. Procedures for each are described in this section.

### Translating Exported Symbol Text Dictionary Files

After you export the dictionary file containing your application text, use Microsoft Excel to edit the text.

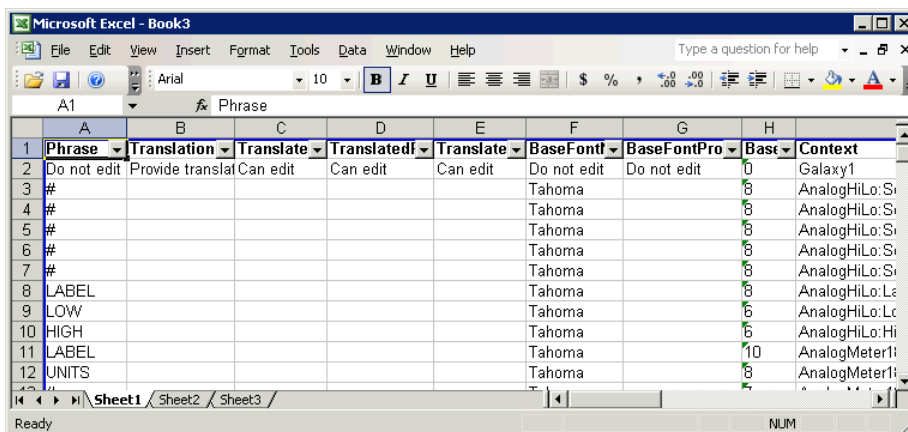
#### To translate an exported dictionary file

1. Open the XML file in Excel. The **Open XML** dialog box appears.
2. Click **As an XML** list, then click **OK**. A message may appear informing you that an XML schema will be created.
3. Click **OK**.

The XML file opens in Excel with columns for the:

- Phrases in your application.
- Translated phrases from the translator.
- Translated font name.
- Translated font properties.
- Translated font size.
- Base font properties.
- Base font size.

- Context, phrase ID, language ID and foreign language ID.



**Important:** Only modify data in the **Translation**, **TranslatedFontSize**, **TranslatedFontName**, and **TranslatedFontProperty** columns. Do not change any column header. Do not insert or delete rows.

4. Type the language-specific text in the **Translation** column in the row that corresponds with the base language string in the **Phrase** column.
  5. If necessary, change the font parameters for the translated strings. If you only provide a translation, the Galaxy-configured font for the language is used to render the text after the translation is imported. If you specify a font, it overrides the Galaxy-configured font.
    - In **TranslatedFontName** column, type the font name.
    - In the **TranslatedFontProperty** column, type the notation for the font properties:
      - B = **bold**
      - I = *italic*
      - U = underline
- For example, if you want the text to be bold, type **B** in the **TranslatedFontProperty** column. If you want the text to be bold and underlined, type **BU** in the **TranslatedFontProperty** column.
6. Save the file using XML Data as the file type.

**Important:** If you save as another file type, such as XML Spreadsheet, Excel changes the schema and the Galaxy cannot load the file. If you change the name of the XML file, the file will not import properly into the Galaxy.

## Importing Translated Symbol Language Files

You can import alarm comment language files for re-export by area, to facilitate the export of new, untranslated alarm comments, and to facilitate re-export of previously translated alarm comments that have been changed.

For symbol text, you must import the translated dictionary files for each language to enable run-time language switching for those languages. All dictionary files for a given language should be placed in the same folder.

You can import files for only one language at a time. When you import, you select the desired language and specify the files to import.

## Importing Translated Symbol Dictionary Files

All affected symbols and relevant objects are checked out before the import begins and are checked in when the import is done. If an affected symbol or object is already checked out, a message appears in the progress dialog box, and the import is skipped for the checked out object.

For a published InTouch application, only the native InTouch translations are imported.

You can configure how you want Galaxy and symbol/object mismatches handled during the import.

The import is skipped if:

- If default locale specified in the .xml translation file does not match the current default locale of the Galaxy.
- An InTouchViewApp's installed locale does not match the current default locale of the Galaxy.

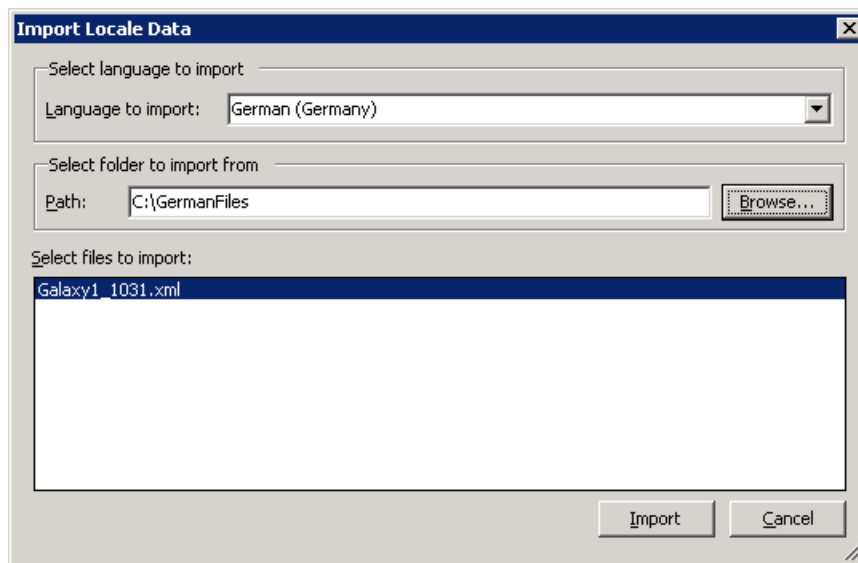
---

**Important:** You cannot cancel the import after it starts.

---

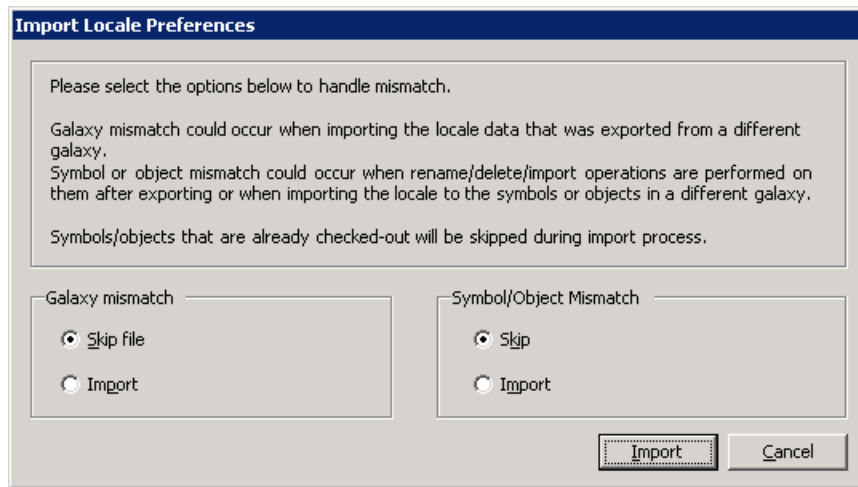
### To import a translated dictionary file

1. Using the System Platform IDE, open the Galaxy for which you want to import symbol text.
2. Close all editors and check in all Galaxy objects.
3. On the **Galaxy** menu, point to **Import**, and then click **Localization**. When a message appears, click **OK**. The **Import Locale Data** dialog box appears.



4. Configure the import settings.
  - In the **Language to import** list, select the language dictionary to import.
  - In the **Path** box, specify the folder that includes the dictionary file to import.
  - In the **Select files to Import** box, select the .xml files to import. Only files that include the current Galaxy name and the locale ID for the selected language are shown.

5. Click **Import**. The **Import Locale Preferences** dialog box appears.



6. In the **Galaxy mismatch** area, configure how you want Galaxy mismatches handled. A Galaxy mismatch occurs when you try to import a translation file that was exported from a different Galaxy. The Galaxy name in the translation .xml file is used to match the current name of the Galaxy.
- Click **Skip file** to skip all the files that do not contain the current Galaxy name in the file name.
  - Click **Import** to import all the selected files, regardless of what the Galaxy name is in the .xml filename.
7. In the **Symbol/Object Mismatch** area, configure how you want symbol and object mismatches handled. A symbol or object mismatch occurs when the name of the symbol and the internal ID (GObjectID) of the symbol do not match what is within the .xml file. Objects include AutomationObjects and InTouchViewApp objects.
- Click **Skip** to skip the symbols and objects that have mismatch names or mismatch IDs in the .xml file
  - Click **Import** to import a symbol or object only if it has a matching name or matching ID. If the name resolves to one object, and the ID resolves to another object, then the import is skipped.
- For examples, see *Examples of Symbol or Object Mismatch Handling during Language Imports* on page 518.
8. Click **Import**. The import progress is shown.
9. Click **Close**. The **Import Language Dictionary Files** dialog box appears.
10. Click **Check In**. The check in progress is shown.
11. Click **Close**. A summary of the import is shown.
12. Click **OK**.

## Examples of Symbol or Object Mismatch Handling during Language Imports

The following table shows an example of handling mismatch conditions for a toolbox symbol. The bold name/ID is the matching name/ID in the .xml file and current Galaxy during import.

	<b>Symbol Name and ID in the Galaxy while Exporting</b>	<b>Symbol Name and ID in the Galaxy while Importing</b>	<b>Change made after export and before import</b>	<b>Skip Option Selected</b>	<b>Import Option Selected</b>
1	S1, 100	<b>S1, 100</b>	No change	Import to S1	Import to S1
2	S1, 100	<b>S2, 100</b>	Rename S1 to S2	Skip	Import to S2
3	S1, 100	<b>S1, 200</b>	Delete S1, Create/Import S1	Skip	Import to S1
4	S1, 100	<b>S1, 200</b> <b>S2, 100</b>	Rename S1 to S2 Create/Import S1	Skip	Ambiguous, skip import
5	S1, 100	No S1 and No 100 in the Galaxy	Deleted S1	Skip	Skip import

The following table shows an example of handling mismatch conditions for an AutomationObject symbol. The bold name/ID is the matching name/ID in the .xml file and current Galaxy during import.

	<b>Symbol Name and ID in the Galaxy while Exporting</b>	<b>Symbol Name and ID in the Galaxy while Importing</b>	<b>Object Name and ID in the Galaxy while Importing</b>	<b>Change made after export and before import</b>	<b>Skip Option Selected</b>	<b>Import Option Selected</b>
1	Pump1, 10	S1, 100	<b>Pump1, 10</b>	No change	Import to S1 only if Pump1 has S1 with an ID of 100	Import to S1 only if Pump1 has S1 with an id of 100
2	Pump1, 10	S1, 100	<b>Pump2, 10</b>	Rename Pump1 to Pump2	Skip	Skip import if S1 and 100 pointing to two different symbols in Pump2 (ambiguous)  Import to S1 only if Pump2 has S1 or a symbol with id of 100.

	<b>Symbol Name and ID in the Galaxy while Exporting</b>	<b>Symbol Name and ID in the Galaxy while Importing</b>	<b>Object Name and ID in the Galaxy while Importing</b>	<b>Change made after export and before import</b>	<b>Skip Option Selected</b>	<b>Import Option Selected</b>
3	Pump1, 10	S1, 100	<b>Pump1</b> , 20	Delete Pump1 Create and Import Pump1	Skip	Skip import if S1 and 100 pointing to two different symbols in Pump1 (ambiguous)  Import to S1 only if Pump1 has S1 or a symbol with id of 100.
4	Pump1, 10	S1, 100	<b>Pump1</b> , 20 <b>Pump2</b> , 10	Rename Pump1 to Pump2  Create and Import Pump1	Skip	Ambiguous, skip import
5	Pump1, 10	S1, 100	No Pump1 and No 10 in the galaxy	Deleted Pump1	Skip	Skip import S1

## Language Data Handling for Galaxy Operations

If you import or export objects, all language data within the associated symbols is imported or exported.

You can export all objects in the Galaxy, selected instances/templates, or selected symbols from the Graphics Toolbox. All language data is exported as part of the graphics definition, regardless of the languages configured in the Galaxy.

You can import Galaxy objects that contain language data from the same Galaxy or a different Galaxy. All language data in the Industrial Graphics is imported, regardless of what languages are configured for the target Galaxy.

No Industrial Graphic or InTouchViewApp language data is read or modified during the import operation.

If you import an unmanaged InTouch application, the configured locales in the Galaxy are applied to the unmanaged InTouch application. No element translations in the InTouch application are removed during import, even though languages may no longer be visible in the InTouch HMI or available at run time in WindowViewer.



## Exporting Alarm Comments for Offline Translation

As with symbol text export for translation, you typically send the alarm comment text out for bulk translation. You can export alarm comments and modify them using a text editor or a spreadsheet program such as Microsoft Excel. Once translated the files can be reimported to Application Server and can be imported by InTouch for run-time alarm comment language switching.

### Guidelines and Recommendations

The following guidelines and recommendations will help you make best use of the alarm comment language switching feature:

---

**Important:** The exported file name is generated automatically and must not be changed.

---

### Organizing Your Export

- You can only export the alarm comment text for one language at a time.
- When you export text, you specify a folder for the language files. We recommend that you create a separate folder for each language. For example, ...\\Galaxy01\\ChineseFiles\\.
- For large Galaxies, exporting files for each area, after being translated and re-imported in the Galaxy, will perform faster when the alarm clients access those files.
- For small Galaxies or for InTouch applications, we recommend that you export the entire Galaxy for translation.

### Translation File Formatting and Editing

Microsoft Excel is recommended for translation file formatting and editing, since it supports large worksheets and provides enhanced formatting capabilities to convert the text file into tabular format. See *Exporting Alarm Comments from Very Large Galaxies* on page 522.

### Reimporting

- We highly recommend reimporting the translated files to the Galaxy after translation. Although reimport is not mandatory, you can use the reimport to optimize the export files to be used in the InTouch application.
- If you make changes to your alarm comments after you export comment text, you must export the alarm comments again. For this reason, we further recommend using the import functionality in Application Server to reimport the language file after each translation. This practice will avoid the need to retranslate alarm comments you have already translated.

For further information and procedures, see the following sections in this chapter:

*Exporting Alarm Comments from Very Large Galaxies* on page 522

*Exporting Alarm Comments by Area* on page 523

*Importing Translated Alarm Comment Language Files* on page 526

### About the Alarm Comments Language File

The alarm comment language switching feature exports alarm comments to a .txt file. That file can be edited directly, but working in a text editor typically is more difficult and prone to errors.

- All files export as Unicode. We recommend that you save the exported files as Unicode.
- We recommend that you edit the .txt file in Microsoft Excel. Be sure to save the edited file as a .txt file.

The language file name is not user-configurable. The file naming convention for alarm comment export is:

*Galaxy\_<GalaxyName>\_<localeID>\_Alarm\_Comments.txt*, or

*Galaxy\_<GalaxyName>\_<AreaName>\_<localeID>\_Alarm\_Comments.txt*, where the locale ID is the selected language decimal identification number.

For example, if the Galaxy name is "TestSample" and the exported language is Chinese, the language file name would be *Galaxy\_TestSample\_2052\_Alarm\_Comments.txt*.

For example, if the Galaxy name is "TestSample" the Area name is "Area001", and the exported language is German, the language file name would be *Galaxy\_TestSample\_Area001\_1031\_Alarm\_Comments.txt*

See *Translating Exported Alarm Comment Language Files* on page 524 for further information about alarm comment language files.

## Exporting Alarm Comments from Very Large Galaxies

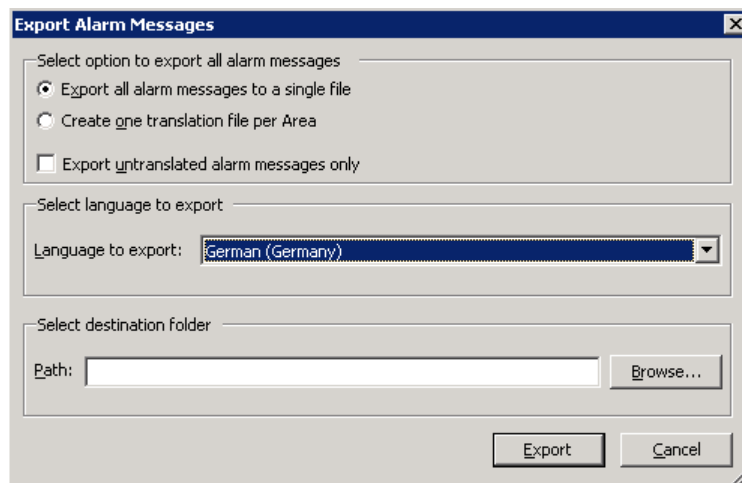
You can export language data in one operation for all alarm comments in a Galaxy. However, very large Galaxies with numerous alarm comments require longer export processing time and can result in a very large .txt language file. For handling large Galaxies and a large number of alarm comments, we recommend that you export alarm comments by Area rather than exporting alarm comments for the entire Galaxy to a single language file. See *Exporting Alarm Comments by Area* on page 523.

## Exporting All Galaxy Alarm Comments

You can export language data for all alarm comments in a Galaxy at one time. The export operation only applies to objects that are checked in.

### To export all Galaxy alarm comments to a single file

1. Using the System Platform IDE, open the Galaxy for which you want to export alarm comments.
2. On the **Galaxy** menu, select **Export**, then select the **Localization** menu option, and then select the **All Alarm Messages** menu option. The **Export Alarm Messages** dialog box appears.



3. Configure the export settings:
  - a. In the **Select option to export all alarm messages** area, select the **Export all alarm messages to a single file** radio button.
  - b. Check the **Export untranslated alarm messages only** check box if you have already exported, translated and re-imported your language file and you want to export only newer, untranslated messages. Otherwise, leave the check box unchecked.

- c. In the **Language to export** pull-down list, select the language to export. You can export only one language at a time.
  - d. In the **Path** box, type the folder to which you want to export the language file. Click **Browse** to select an existing folder or create a new folder.
4. Click **Export**. The export progress box appears.
  5. Click **Close** when the export completes.

## Exporting Alarm Comments by Area

You can export all alarm comments by Area or you can export Alarm comments for a specific Area.

### Using File Names

The *Galaxy\_<GalaxyName>\_<AreaName>\_<localeID>\_Alarm\_ Comments.txt* file name convention applies to specific Area alarm comment exports. Area names will be included in the file name. For example, given a Galaxy name "TestSystem", an Area name "Area001", and an export to Chinese (language designation 2052), the language file name would be *Galaxy\_TestSystem\_Area001\_2052\_Alarm\_ Comments.txt*.

### Exporting Objects Not Assigned to an Area

System Objects typically are not assigned to an Area. These are:

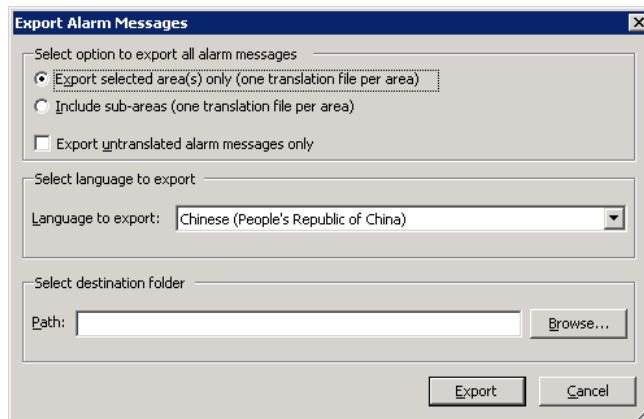
- Platform Objects
- Device Integration Objects
- Engine Objects

For purposes of language export, these System Objects form their own "Area" or become notification distributors for the purpose of sending alarms. Exporting System Objects follows these important conventions:

- System Object can only be exported from the Galaxy menu. They cannot be selected and exported using a context menu.
- System Objects can only be exported by exporting alarm comments for all Areas.

#### To export alarm comments for all Areas.

1. Using the System Platform IDE, open the Galaxy for which you want to export alarm comments.
2. On the **Galaxy** menu, select **Export**, then select the **Localization** menu option, and then select the **All Alarm Messages** menu option. The **Export (Area) Alarm Messages** dialog box appears.



3. Configure the export settings:

- a. In the **Select option to export all alarm messages** area, select the **Export selected area(s) only** radio button, or select the **Include sub-areas** radio button to include all sub areas for each selected Area.
  - b. Check the **Export untranslated alarm messages only** check box if you have already exported, translated and re-imported your language files by Area and you want to export only newer, untranslated messages. Otherwise, leave the check box unchecked.
  - c. In the **Language to export** pull-down list, select the language to export. You can export only one language at a time.
  - d. In the **Path** box, type the folder to which you want to export the language file. Click **Browse** to select an existing folder or create a new folder.
4. Click **Export**. The export progress box appears.
  5. Click **Close** when the export completes.

### To export alarm comments for a selected Area or Areas

1. Using the System Platform IDE, open the Galaxy for which you want to export alarm comments.
2. Expand the **Application View** tree and select the Area(s) you want to export.
3. Right click to view the context menu. Select **Export** on the context menu, then select **Localization**, and then select **Alarm message(s) of selected Area(s)**.

As an alternate method to using the context menu, you can select the Area you wish to export, then use the Galaxy menu as in the preceding procedures.

4. Configure the export settings:
  - a. In the **Select option to export all alarm messages** area, select the **Export selected area(s) only** radio button, or select the **Include sub-areas** radio button to include all sub areas for each selected Area.
  - b. Check the **Export untranslated alarm messages only** check box if you have already exported, translated and re-imported your language files by Area and you want to export only newer, untranslated messages. Otherwise, leave the check box unchecked.
  - c. In the **Language to export** pull-down list, select the language to export. You can export only one language at a time.
  - d. In the **Path** box, type the folder to which you want to export the language file. Click **Browse** to select an existing folder or create a new folder.
5. Click **Export**. The export progress box appears.
6. Click **Close** when the export completes.

## Translating Exported Alarm Comment Language Files

Alarm comments are exported to a .txt file, located in the selected directory. You can edit the files with a text editor, but we recommend Microsoft Excel as the editor. Using the format-as-table feature in Excel greatly simplifies editing the specific text to be translated.

## To translate an exported alarm comment file

1. Open the .txt file in Excel. The Excel **Text Import Wizard** appears. Follow the wizard instructions to import the text file in tab delimited, general column data format. The file appears unformatted in Excel.

	A	B	C	D	E	F	G	H	I	J	K
1	Column1	Column2	Column3								
2	Unique Phrases:										
3											
4	Phraselid (Default Message Translated Message (EDIT THIS COLUMN))										
5	1	<<< NO CONFIGURABLE MESSAGE >>>									
6	2	The UserDefined object provides a starting point for creating custom built objects that include features, scripts, and attributes.									
7	3	tttt	german translation for weird tttt alarm message								
8	5	grouping of objects for modeling and alarm									
9	6	me.ShortDesc									
10											
11											
12	Alarms:										
13											
14	Phraselid (Alarm (DO NOT EDIT))										
15	6	Area_002.udbool1									
16	5	Area_002.udbool2									
17	3	UserDefined_001.abc									
18	2	UserDefined_001_001.RickAlarm									
19	1	WinPlatform_001.CheckpointFileCorruptionAlarm									
20											
21											

2. Format the text as a table in Excel:
  - a. Select all of the rows containing data in Column A through Column C.
  - b. Select the **Home** tab, then select **Format as Table**.
  - c. Select the color pattern you want and click **OK** in the **Format as Table** dialog box. Excel will format the region to be edited.

	A	B	C	D
1	Column1	Column2	Column3	
2	Column1	Column2	Column3	
3	Unique Phrases:			
4				
5	Phraselid (DO NOT EDIT)	Default Message (DO NOT EDIT)	Translated Message (EDIT THIS COLUMN)	
6	1	<<< NO CONFIGURABLE MESSAGE >>>	translated message in Germany for << NO CONF MESSAGE >>	
7		The UserDefined object provides a starting point for creating custom built objects that include features, scripts, and attributes.	german translation for big alarm message	
8	3	tttt	german translation for weird tttt alarm message	
9	5	grouping of objects for modeling and alarm		
10	6	me.ShortDesc		
11				
12				
13	Alarms:			
14				
15	Phraselid (DO NOT EDIT)	Alarm (DO NOT EDIT)		
16	6	Area_002.udbool1		
17	5	Area_002.udbool2		
18	3	UserDefined_001.abc		
19	2	UserDefined_001_001.RickAlarm		
20	1	WinPlatform_001.CheckpointFileCorruptionAlarm		
21				
22				
23				

The Excel display is divided into four regions:

- Column 1 contains Phrase ID numbers used internally in the language file. Do not edit this column.
- Column 2 contains the exported alarm messages. Do not edit this column.
- Column 3 contains the alarm message translations. Edit this column with your translation text.

- Below the alarm comment text columns are the alarms and phrase IDs, which map to column 1 and are used internally in the language file. Do not edit this area.
3. Translate the alarm comment text. Type the alarm comment translations into the appropriate rows in column 3.

---

**Important:** Only edit the text in column 3, translations. No other text in the file may be edited. You can edit the original alarm comment text only within the System Platform IDE.

---

4. Save the file as a .txt file in the directory you originally selected for export.

---

**Important:** Do not change the file name or the file type, otherwise it will not import correctly into the Galaxy.

---

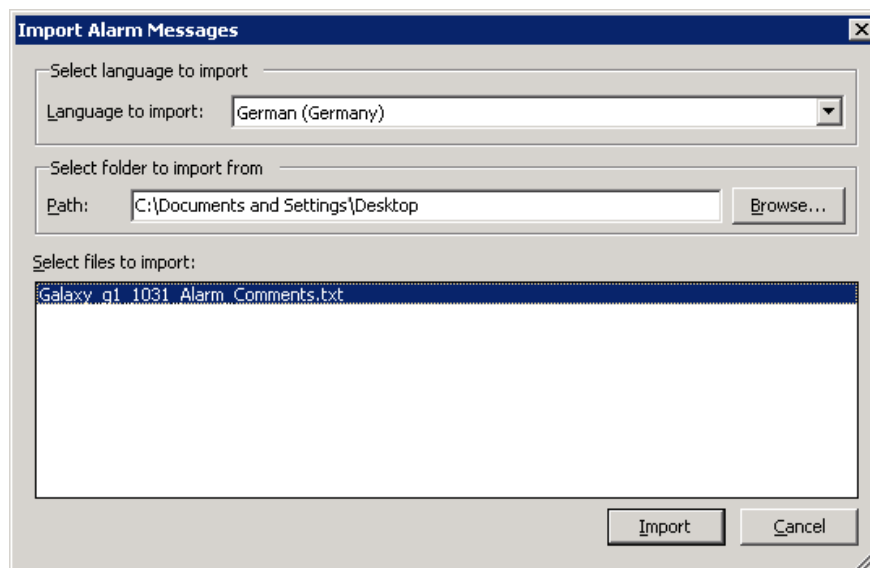
## Importing Translated Alarm Comment Language Files

You can import translated alarm comment language files only from the Galaxy menu, not from the context menu for specific Areas.

To import alarm comment language files, you must first have exported alarm comments for translation of some or all of the alarm comments.

### To import a translated alarm comment language file

1. Using the System Platform IDE, open the Galaxy for which you want to import translated alarm comments.
2. On the **Galaxy** menu, select **Import**, then select the **Localization** menu option, and then select the **Alarm Message(s)** menu option. The **Import Alarm Messages** dialog box appears.



3. Configure the import settings:
  - a. In the **Language to import** pull-down list, select the language to import. You can import only one language at a time.
  - b. In the **Path** box, type or browse to the folder where you previously exported your language file. The available language files appear in the **Select files to import** list.
  - c. Select a file or files to import.
4. Click **Import**. The import progress box appears.
5. Click **Close** when the import completes.

---

**Note:** If an alarm name or message do not match, the alarm comment import will display a message for each failed alarm message import.

---

### To import a translated alarm comment language file into InTouch

1. Open the application in WindowViewer.
2. On the **Special** menu, select **Language**, then select the name of the language you want to switch to. Information from the corresponding translated dictionary file (if one exists and has been imported into InTouch) loads and displays.
3. Click **Close** when the import completes.

## Re-exporting Alarm Comments

After exporting and translating alarm comments, you can re-import them to ArchestrA to help maintain up-to-date alarm comment language files.

After adding new alarm comments that require translation or after modifying existing, already translated alarm comments, you must re-export the language file to update the translations.

## Exporting New Untranslated Alarm Comments

In the **Export Alarm Messages** dialog box, select the **Export untranslated alarm messages only** check box. This creates a file named `_untranslated.txt`. For example, exporting untranslated alarm comments in Chinese for a Galaxy name "TestSystem" would create a file named `Galaxy_TestSystem_2052_Alarm_Comment_Untranslated.txt`.

## Exporting Modified Existing Alarm Comments

You can make changes to alarm comments that you previously exported, translated and imported back into ArchestrA. You must re-export the alarm comment language file(s) to update the translations.

In the **Export Alarm Messages** dialog box, select the same language option and folder you selected for the previous export. When asked, confirm that you want to overwrite the existing file.

## Testing the Language Switching Functionality at Run Time

We recommend that you test the run-time language switching functionality. Follow the procedures outlined in this chapter to enable run-time language switching in your application. Then import the appropriate language file into InTouch. The import step applies to both stand-alone and managed InTouch applications.

### To test the language switching functionality

1. Open the application in WindowViewer.
2. On the **Special** menu, point to **Language**, and then click the name of the language to switch to.  
The information from the corresponding translated dictionary file (if one exists and has been imported into InTouch) loads and appears.
3. When you are done, click **Close**.





# CHAPTER 16

## Managing Galaxies

You can back up and restore Galaxies, change the Galaxy you are working with, delete a Galaxy, and export and import all or part of a Galaxy.

If you want to create a Galaxy, see *Create a Galaxy* on page 20.

## Backing Up and Restoring Galaxies

Periodically, you should back up your Galaxy. Backing up your Galaxy helps if you have a computer failure or other problem. If there is a failure, you can restore your Galaxy from the backup.

Use the Galaxy Database Manager to back up and restore your Galaxy. The Galaxy Database Manager is part of the suite of ArcestrA System Management Console utilities.

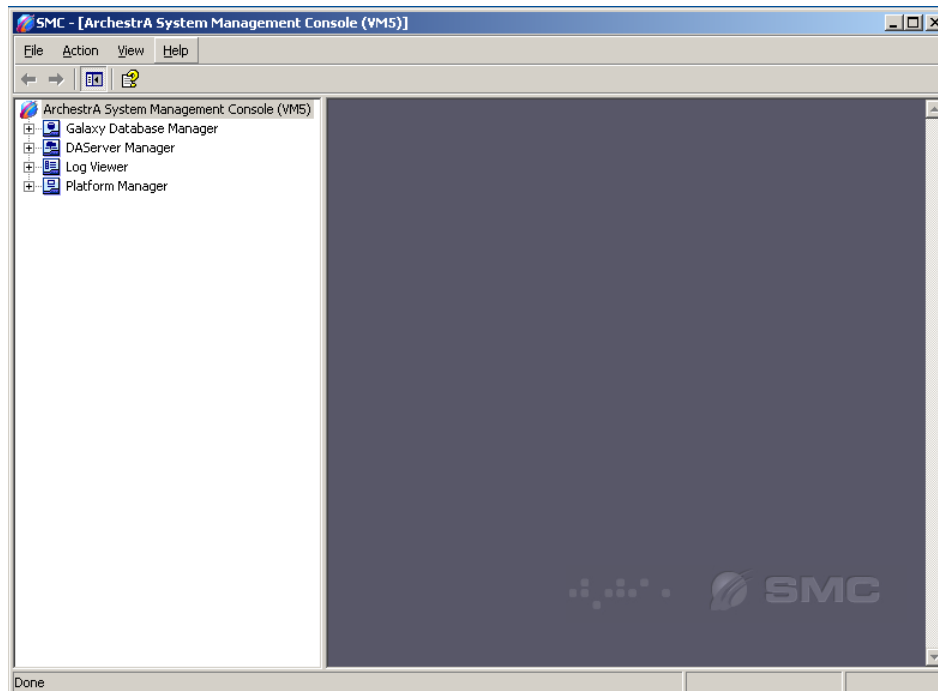
The aaConfigSQL utility is another utility that you may need to use when restoring Galaxies. aaConfigSQL is used to set ArcestrA user permissions for SQL Server. In most cases, the default level of permissions is adequate for restoring Galaxies.

However, if you are restoring a Galaxy created with an older version of Application Server, you may need to change the permissions level.

See the section about SQL Server Rights Requirements in the *System Platform Installation Guide* for additional information about the aaConfigSQL utility.

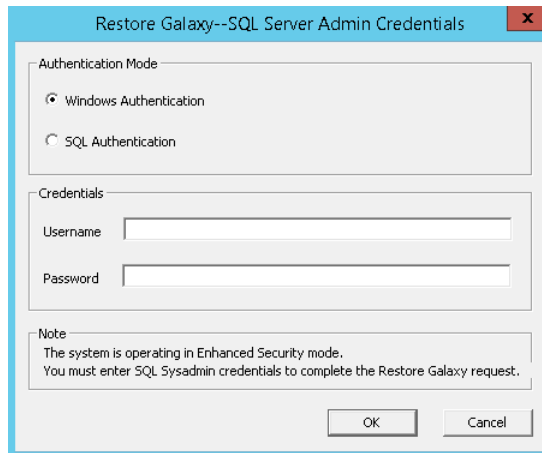
### To open the Galaxy Database Manager

- From the **Start** menu, navigate to the **AVEVA** folder, and then select **System Platform Management Console**.



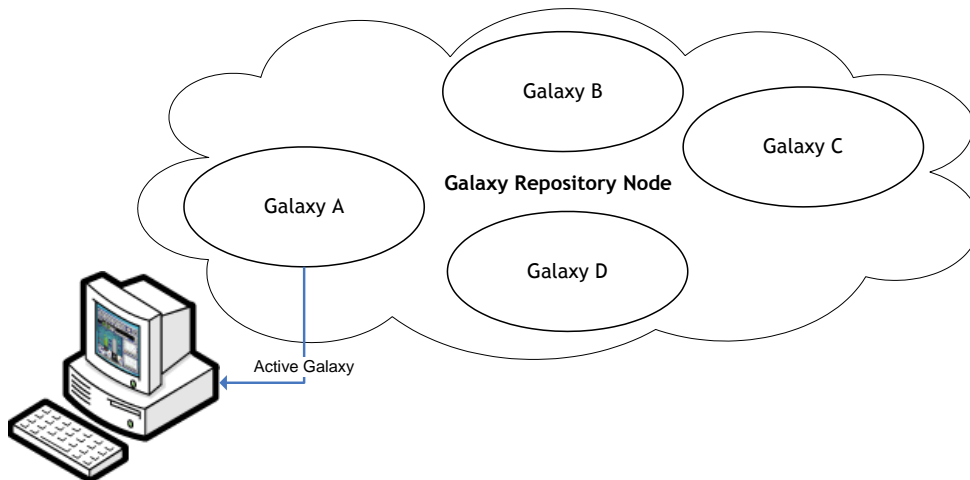
See the *Galaxy Database Manager* documentation for more information about backing up or restoring your Galaxy.

If Enhanced Security mode is in effect, you will be prompted to enter SQL SysAdmin credentials before you can restore a Galaxy created in an older version of Application Server.



## Changing Galaxies

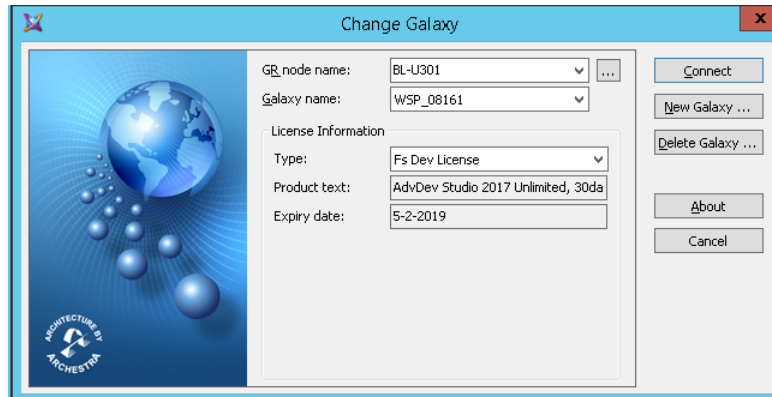
You can have many Galaxies in the Galaxy Repository. If you are a systems integrator, you have at least one Galaxy for every client.



For a Galaxy to deploy objects to other computers in the Galaxy, the Galaxy Repository (GR) must host a platform defined in that Galaxy. Because of this, you can only deploy one Galaxy from the Galaxy Repository at a time to the computers on your network. For more information about deploying, see *Deploying Objects* on page 333.

## To change from one Galaxy to another

1. On the **Galaxy** menu, click **Change Galaxy**. The **Change Galaxy** dialog box appears.



2. Do one of the following:
  - Select another Galaxy from the **Galaxy Name** list.
  - Select another Galaxy Repository node from the **GR Node Name** list.
3. Click **Connect**.

## Deleting a Galaxy

You can delete a Galaxy. Deleting a Galaxy removes all of the Galaxy information from your computer.

Before you delete a Galaxy, you must undeploy all objects within it. For more information about undeploying objects, see *Undeploying Objects* on page 338.

Make sure you select the right Galaxy to delete. After you delete a Galaxy, you cannot undelete it. You can only recreate it by restoring from a backup. For more information, see *Backing Up and Restoring Galaxies* on page 529.

### To delete a Galaxy

1. Undeploy all objects from the Galaxy you want to delete.
2. Close any Galaxy and connected IDE you have open.
3. On the **File** menu, click **Change Galaxy**. The **Connect to a Galaxy** dialog box appears.
4. Select the Galaxy you want to delete, except the connected Galaxy.
5. Click **Delete Galaxy**.
6. At the prompt, click **Yes**.
7. When the Galaxy is deleted, click **Close**.

## Galaxy Object Components Synchronization

The Galaxy Object Components Synchronization feature provides a detection and recovery mechanism when the client is out-of-sync with the server. Synchronization means synchronizing the installed object components on the client node with respect to the connected GR.

Synchronization of object components as defined in this section is fundamental to the subsequent functionality. For that reason, it is unnecessary to synchronize script libraries, graphics or InTouchViewApp-related files, client controls, IDE extensions, run-time components, security, or history data.

In the case of a run time out-of-synchronization condition, recovery is accomplished by redeploying the run-time platform.

## Definitions

The following terms are important in understanding the Galaxy Object Components Synchronization feature:

- **Object Component:** Base template-dependent files
- **Server:** The GR node, which maintains the object component information
- **Client:** The IDE

## Typical Out-of-Sync Scenarios

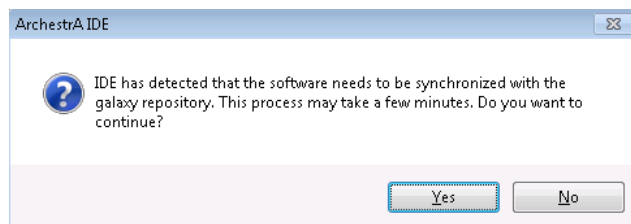
The GR and the client nodes can become out-of-sync in a number of ways. The following are the typical causes:

- Re-imaging the client or server node or both
- Restoring a Galaxy from a backup
- Deleting and creating a Galaxy on the server node

## Using the Synchronization Feature

The synchronization detection and resolution process is the same for all scenarios. Out-of-sync conditions between a client and a Galaxy are detected and resolved at connection time.

If the system recognizes that the minimum connectivity requirements are met, and there are object versioning conflicts, then you are prompted about the out-of-sync state with the option either to synchronize or to reject the synchronization operation.



Select the option you prefer. If you select to synchronize, the synchronization operation will run without further prompts.

A warning message is placed in the Logger for any out-of-sync detection and synchronization issues.

## Exporting a Galaxy Dump File

You can export instances and their configuration data in a Galaxy to a comma-separated values file with a .csv file name extension. After you export the Galaxy to a .csv file, you can edit it with Microsoft Excel. This makes it very easy to do large editing changes, such as search and replace.

Exporting only exports instances. Templates cannot be exported in .csv file format.

The .csv file contains the configuration for the checked-in versions of the selected instances and the checked-out instances of the user who does the Galaxy dump. If an instance is checked out by another user when you export the Galaxy, the checked in version is exported.

The file contains only those attributes that are unlocked and configuration time-writeable, one column per attribute. The following are not exported:

- Scripts libraries are not exported. Scripts within an object are exported.

- Attributes that are not text-based are not exported. For example, type QualifiedStruct is not exported.
- Custom object online help files are not exported.

---

**Note:** I/O Auto Assignment information is not included in the Galaxy dump file. For information about manually adding I/O Auto Assignment information to a Galaxy dump file for inclusion in the Galaxy load process, see *Enabling I/O Auto Assignment in the Galaxy Dump File* on page 535.

---

See *About the Galaxy Dump File Structure* on page 533 for specific information about the structure of the .csv file. Before you start, make sure you know what instances you want to export to a file.

### To export objects to a Galaxy dump file

1. In the **Application views** area, select at least one instance. Shift+click to select multiple instances. You can export all instances derived from a template by selecting the template.
2. On the **Galaxy** menu, click **Export** and then click **Galaxy Dump**. The **Galaxy Dump** dialog box appears.
3. Browse to the location of the .csv file to which you want to dump the selected instances. Type the name of the file. Click **Save**.
4. When the Galaxy export process is done, click **Close**. A .csv file is created containing the selected objects and configuration data.

## Editing the Galaxy Dump File

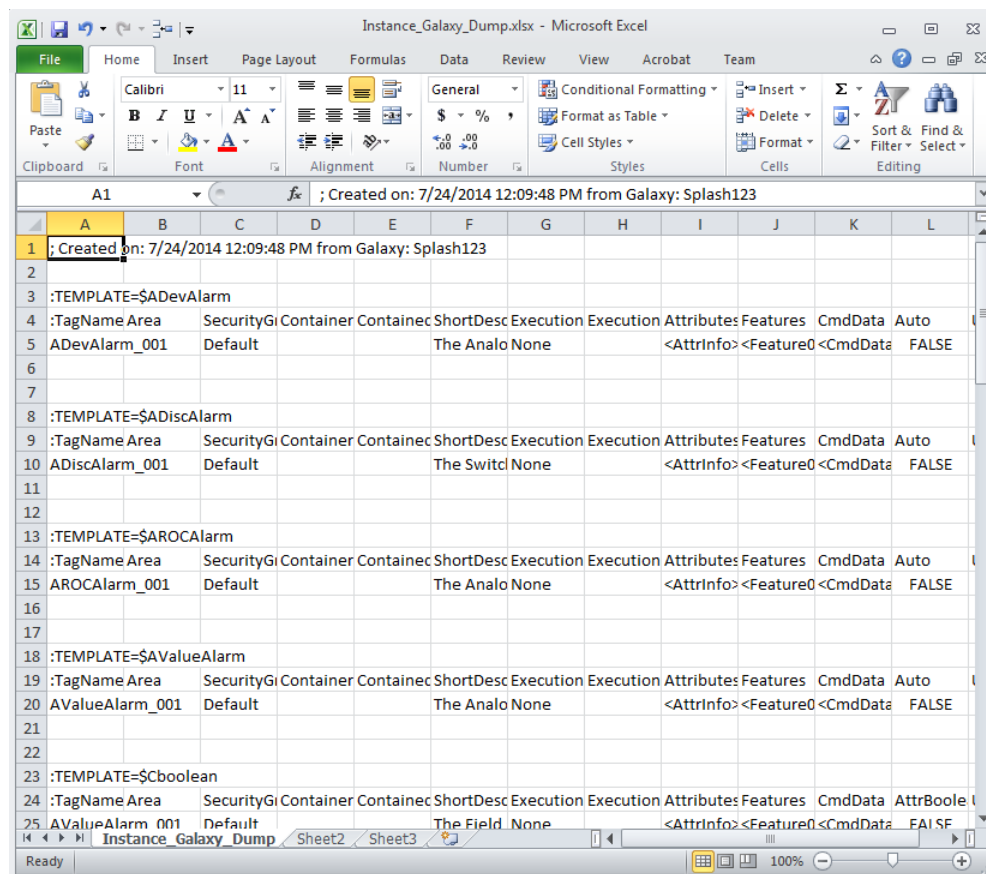
You can open the .csv file created by the Galaxy dump process in a text editor such as Notepad or in Microsoft Excel. When you have finished editing the file, save it as plain text .csv (comma delimited) file to import back into the Galaxy.

## About the Galaxy Dump File Structure

The Galaxy .csv file has a specific structure. This section describes that structure.

Objects are organized in the .csv file based on the template each is derived from. A header row per template indicates the instance's columns' reference.

Other information is organized in columns. This makes it easy for you to read the information and carefully make any changes you need. You can easily import the .CSV file into a text editor or into Microsoft Excel.



Add comments by adding a line with a semi-colon as the first character in the comment.

## Host Attributes

Galaxy dump files contain a column for the Host attribute of the objects being dumped. In the case of Platform objects, Host is always the name of the Galaxy from which the object is being dumped.

This data is ignored in subsequent Galaxy Load operations because the Host for Platform objects is automatically the name of the Galaxy into which it is being loaded, regardless of the name of the Galaxy from which it was dumped.

Using a text editor, you can delete the Host attribute column, like any other data in the Galaxy dump file. This has no effect on Platform objects in subsequent Galaxy Load operations because they take the Galaxy name as their Host.

## About Quotation Marks and Carriage Returns

Carriage returns in scripts associated with dumped objects are replaced with \n in the .CSV file. If you edit the dump file, **do not** delete the \n characters. If you edit scripts in the dump file, use \n for a carriage return. This character set is interpreted as a carriage return when the dump file is used in a Galaxy Load operation.

When editing a script in a dump file, use \\n if you want to include the backslash character (\) followed by the letter n in a script. This character set is not converted to a carriage return in a Galaxy Load function.

Be careful when adding or editing quotation marks in the `.csv` file. Type all single quotation marks as two single quotation marks and surround the entire string with opening and closing quotation marks.

Make sure the string contains an even number of quotation marks. When the object is loaded in a Galaxy Load operation, the extra quotation marks are stripped from the string.

For example, if you want to enter `3"Pipe` as a Short Description, add a second quotation mark (`3""Pipe`) and then surrounding quotation marks (`"3""Pipe"`).

## Time Formats in Excel

If you edit a Galaxy dump file in Microsoft Excel, be careful typing time entries. Excel can change the time format and the resulting entries do not work when you reload the changed Galaxy.

Galaxy Load accepts two formats:

- `DAYS HH:MM:SS:SSSSSS` - the number of `DAYS` is followed by a space.
- `HH:MM:SS:SSSSSS` - Excel automatically changes the entry to an incompatible format.

When you type time entries in Excel, use the following format:

`DAYS HH:MM:SS:SSSSSS`.

For example:

```
0000 01:02:12.123: 1
04:03:06.12:120
22:66:88:123456
```

## Enabling I/O Auto Assignment in the Galaxy Dump File

You can use Galaxy dump to convert a Galaxy that contains traditional I/O hard coding to use I/O auto assignment instead. If you are planning to expand an existing Galaxy by adding many objects, or if you are upgrading PLCs, converting hard coded I/O references to auto assignment can save you time. To convert the references, edit the Galaxy dump file as described below. This bulk edit is faster than changing each object individually in the IDE object editor.

The Galaxy dump file displays a flag, `---Auto---`, for input and output attributes that use I/O auto assignment. This flag indicates that I/O auto assignment is enabled for the attribute. The actual I/O auto assignment to a DI Object and scan group is not preserved in the dump file. If you want to preserve existing I/O auto assignment settings, use the export objects function instead of Galaxy dump. See *Export Objects* on page 102 for additional information.

To convert hard coded I/O references, edit the Galaxy dump file to replace the hard coded reference with `---Auto---` for each `<attribute>.InputSource` and/or `<attribute>.Output.Dest`. Import the edited objects back into the Galaxy, and then assign the objects in the **IO Devices** view of the IDE to a DI Object and scan group.

For more information about I/O auto assignment, see *Using I/O Auto Assignment* on page 148.

### To convert hard coded I/O references to I/O auto assignment

1. Use Galaxy dump to export the objects with hard coded I/O references that you want to convert.
2. Open the exported `.CSV` file in a text editor or Excel.
3. In the data row of the `Attribute.InputSource` column, delete the hard coded I/O assignment. Enter `---Auto---` in the data row.
4. Repeat for each object that you want to enable for I/O auto assignment.
5. In the data row of the `Attribute.OutputDest` column, delete the hard coded I/O assignment. Enter `---Auto---` in the data row.
6. Repeat for each object that you want to enable for I/O auto assignment.

7. Save your changes.

### To complete the process of activating I/O auto assignment

1. Use Galaxy load to import the objects in the edited Galaxy dump file. See *Importing a Galaxy Load File* on page 536 for additional information.

Once imported, the objects will appear in the **IO Devices** view, under the Unassigned IO Device folder.

2. Drag and drop objects in the **IO Devices** view to link the objects to the applicable DI Object and scan group. See *Using I/O Auto Assignment* on page 148 for additional information.

## Importing a Galaxy Load File

After you are done editing a .csv file, you can import it back into your Galaxy.

A load file contains only instances. Templates cannot be dumped or loaded. See for more information about the contents of the .csv file.

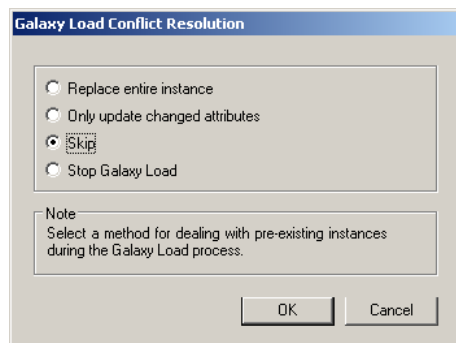
---

**Note:** A comment line in a .csv file created in Microsoft Excel can create an unintended default-value object. To avoid this, open the .csv file in Notepad to make sure the comment line does not contain quotation marks.

---

### To import a .csv file

1. On the **Galaxy** menu, click **Galaxy Load**. The **Galaxy Load** dialog box appears.
2. Browse to find the .csv file that contains the objects and configuration data you want to import. Select the file and click **Open**.
3. The **Galaxy Load Conflict Resolution** dialog box appears. Use it to resolve conflicts that can occur if objects you want to load already exist in the Galaxy.



4. Select one of the following:
  - Replace entire instance** if an instance of an object with the same name already exists and you want to replace it entirely with the object in the import file.
  - Only update changed attributes** if an instance with the same name already exists and you want to replace only the attributes of the object where the values are different.
  - Skip** if an instance with the same name already exists and you want to keep the version already in the Galaxy.
  - Stop Galaxy Load** if an instance with the same name already exists and you want to cancel the Galaxy Load.

5. Click **OK**. A progress box appears showing the Galaxy load process.

When the load is done, all objects changed or created during the Galaxy Load process are checked in.



## Hosting Multiple Galaxies in One Galaxy Repository

You can create and configure multiple Galaxies in a single Galaxy Repository on the same computer. You can configure one Galaxy from an IDE and then change Galaxies and configure the second one using the same IDE.

---

**Note:** If you try to deploy objects from the second Galaxy to a computer that hosts deployed objects from the first Galaxy, the deploy fails.

---

## Disk Space Requirements

After Application Server is installed, certain operations require at least 100 MB of available disk space. These operations include

- Creating a Galaxy
- Deploying objects
- Importing and exporting objects
- Loading and dumping a Galaxy
- Restoring and backing up a Galaxy.

This minimum requirement applies to the Galaxy node as well as any remote IDE nodes.

If your computer has less than 100 MB of available hard disk space, running any of these operations can result in erratic behavior.

## Managing Communication between Galaxy Nodes

You can use Application Server in a distributed environment. All computers with ArcestrA-enabled software installed must be able to communicate with each other.

Two items must be considered to ensure communication between nodes:

- ArcestrA user accounts
- Multiple network interface cards in computers

All Platforms communicate with several attributes on the GR Platform to detect configuration changes. This communication sends NMX heartbeats from each Platform to the GR Platform. For example, the attributes include "time of last deploy," and "time of last configuration change."

In addition to NMX heartbeats, all Bootstraps on each Platform in the Galaxy send each other heartbeats. These heartbeats are for Platform status information, occur every two seconds, and are configurable.

## Mapping Network Drives with User Account Control Enabled

Mapping network drives with Windows User Account Control (UAC) enabled can create issues with subsequent access to those drives. Windows considers mapping a network drive and later access to that drive, to import objects from a network location, for example, as separate logon contexts, and issues separate credentials. Without correct credentials, the mapped network drives can become inaccessible.

You can ensure that no issues occur with mapped drives by using one of two methods to map the drives:

1. Map drives from the command prompt as an administrator.

From Windows **Start/All Programs**, click **Accessories**, then right-click **Command Prompt** and select **Run as administrator** on the context menu.

At the command prompt, type the "net use" command, using the following format:

```
net use \\<computername>\<sharename> /user:<username>
```

Example:

```
net use z: \\MainComputer\ObjectFiles /user:DRoberts
```

where the drive being mapped is z:, the computer name is MainComputer, the share name is ObjectFiles, and the user name is DRoberts.

2. Map drives from the IDE using elevated privileges.

For example, you can map drives from the **Import Object(s)** dialog.

## About ArcestrA User Accounts

Communication occurs with an ArcestrA-specific user account set up during the initial installation of each ArcestrA component on each computer, including the IDE.

---

**WARNING! The user account that is used for ArcestrA communication is a standard Windows operating system account located on the local computer or on a domain. Do not delete this account with operating system account management tools. If you do, the IDE stops functioning.**

---

If you delete the ArcestrA user account on an IDE node, you must recreate it with the Change Network Account utility. You must have Administrator privileges on the computer to make changes with the Change Network Account utility.

Before you start, find out the user name and password that is created on all computers with ArcestrA-enabled software installed.

---

**Important! Completing the account change requires a system restart. Once you press OK, you cannot cancel the account name change or the system restart.**

---

### To recreate an ArcestrA user account

From the **Start** menu, navigate to the **AVEVA** folder, and then select **Change Network Account**.

1. The domain name or local machine name for the old ArcestrA user account is shown. If you need to change to a different domain, use the short domain name. Do not use a fully qualified domain name (FQDN). For example, use "DomainName" and not "DomainName.com" or "DomainName.local."
2. Do one of the following:
  - In a single-node ArcestrA system, enter a new or existing account name.
  - In a multi-node ArcestrA system, create the same user account with the same user name and password on all other ArcestrA computers.

If you are using an existing account name or a domain account, leave the "Create Local Account" checkbox unchecked.

3. Click **OK** or **Cancel**. If you click OK, the system will warn you to shut down all open applications and that the computer will be restarted. You cannot cancel the account change or system restart at this point. After you recreate the user account, the Microsoft Windows security component on your computer can take several minutes to update this information on the ArchedstrA Galaxy node. Until that happens, your IDE might not function properly. Restarting the Galaxy node applies this update immediately.

## Using Multiple Network Interface Cards

You can use nodes with more than one network interface card (NIC). These nodes must be configured properly so they can communicate with other ArchedstrA nodes.

If a node contains multiple NICs for redundancy reasons, see *Working with Redundancy* on page 545 for more information.

If a multiple NIC computer in your Galaxy uses only one NIC, you need to disable all cards except the supervisory network.

## Defining the Order of the NIC

For any multiple NIC ArchedstrA node to communicate with all other Galaxy nodes, you must define the correct order of network connections in the network services of the computer.

Before you start configuring multiple network cards, you need the IP address for the second and subsequent nodes.

### To define the correct order (Windows 8.1 and Windows Server 2012 and 2012 R2)

1. Open **Network Connections** through the **Control Panel** (access **Network Connections** through **Control Panel > Network and Sharing Center**, then select **Change adapter settings**).
2. Rename each network card with a clearly identifiable function, for example, `Supervisory Net` and `PLC net`.
3. Select **Advanced** from the menu bar (press the Alt key if the menu bar is not visible), and click **Advanced Settings**.
4. In the **Advanced Settings** dialog box, click the up and down arrows to define the correct order of Connections.
  - The first connection in the list must be the supervisory network card.
  - If a computer contains more than two network cards, for example, a supervisory connection, a PLC connection, and a Redundancy Message Channel (RMC) connection for ArchedstrA redundancy, the supervisory network must be listed first. The others can be listed in any other position.
5. Click **OK** to accept the changes.

### To define the correct order (Windows 10 and Windows Server 2016)

See *Configuring Multiple NICs* on page 540 for instructions.

## Configuring the IP Address and DNS Settings

After you specify the order of the network cards, you are ready to configure several other parameters to ensure successful node-to-node communications in the ArchedstrA environment.

You must configure the IP address and DNS settings as follows for each network card to function properly.

## To configure the IP address and DNS settings

1. In the **Network Connections** dialog box, right-click the network connection and click **Properties** in the shortcut menu. The **Properties** dialog box for this connection appears.
2. In the list of components used by this connection, select **Internet Protocol (TCP/IP)** and click **Properties**. The **Internet Protocol (TCP/IP) Properties** dialog box appears.
3. For the supervisory network, select **Obtain an IP address automatically**.  
For the other network connections, select **Use the following IP address**. Type the correct IP address for the secondary and further cards. See your network administrator for the proper settings for the remainder of the parameters in this group.
4. Click **Advanced**. The **Advanced TCP/IP Settings** dialog box appears. Click the **DNS** tab.
5. For the supervisory network, select **Register this connection's addresses in DNS**.  
For the other network connections, clear this check box.
6. Click **OK**.

## Configuring Multiple NICs

Network profiles are identified and assigned during computer startup and each time a connection changes. The three profiles currently in use in the supported Windows operating systems are:

- **Domain profile:** Active only when the computer can authenticate with a domain controller on all active interfaces (for example, LAN, wireless, and VPN). The domain profile may be more or less restrictive than the other two profiles depending on network security policies.
- **Private profile:** Active whenever the network type for all active network connections on the computer are identified as private networks. The private profile typically is used in a more trusted environment and is less restrictive than the public profile to allow for network discovery.
- **Public profile:** Active in all other circumstances. The public profile typically is more restrictive than the private profile because the computer often is connected to the Internet in an insecure location. Network discovery and remote access are disabled rather than explicitly blocking specific traffic.

All programs allowed to communicate through the firewall are blocked if one or more network interface cards (NICs) are configured as "Public Network." When the IP address of a network card changes, the NIC is automatically updated to "Public Network." If even one NIC in a computer with multiple NICs is configured as "Public Network", the firewall exceptions are disabled.

This is particularly important for computers configured to run as a redundant pair. If the TCP/IP properties for a network card are set to obtain an IP address automatically, there is a possibility that the network address will change when the computer restarts. The computer may suddenly change from having a "Private Network" to having a "Public Network," and potentially block the programs that were previously allowed to communicate through the firewall.

---

**Note:** An exception to this would occur if the NICs are configured and set to the Public Profile before Application Server is installed. The Application Server installation would allow the necessary programs to communicate through the firewall in the current (Public) profile.

---

The following describes the key NIC settings for a redundant pair. You configure these settings for both the primary and backup computers. These settings apply if more than two NICs are installed for redundancy (RMC) or other networking purposes, such as networking PLCs.

---

**Important!** When configuring NICs for a redundant pair, speed and duplex settings for the standby NIC must match the settings for the primary NIC.

---

For detailed instructions on how to configure multiple network interface card binding order settings, see *Defining the Order of the NIC* on page 539 and *Configuring the IP Address and DNS Settings* on page 539.

## To configure multiple NICs

1. Under **Internet Protocol Version 4 (TCP/IPv4) Properties**, select the **Use the following IP address option** and provide an IP address, subnet mask, and default gateway.
  - a. For the primary computer, the default gateway should be the IP address that you configure for the backup computer. For example:  
IP address: 100.100.100.94  
Subnet mask: 255.255.255.0  
Default gateway: 100.100.100.95
  - b. For the backup computer the default gateway should be the IP address that you configure for the primary computer. For example:  
IP address: 100.100.100.95  
Subnet mask: 255.255.255.0  
Default gateway: 100.100.100.94
  - c. A NIC acting as Redundancy Message Control (RMC) does not require a default gateway. The IP address and Subnet mask can be configured as described for the primary and backup computers. Windows will identify this NIC and assign it a private profile. For example:  
IP address 100.100.100.96  
Subnet mask: 255.255.255.0
2. Under **Advanced Settings**, check that the **Register this connection's address in DNS option** is not selected.

## To set the binding order in Windows 8.1 and Windows Server 2012 / 2012 R2

1. Make sure that the primary network adapter is first in the binding order. This places it above the RMC network in the binding order.
2. Make sure that ALL network cards are configured as "Private Network."
3. Configure the RMC NIC to have a persistent (across reboots) Private profile.
  - a. Open a command window and enter the command `secpol.msc`. The **Security Policy** window appears.
  - b. Select **Network List Manager Policies**.
  - c. Select **Unidentified Network**.
  - d. Right click, then select **Properties**, and then change the **Location** type from **Not configured to Private**.
  - e. Close the command window.

It may be necessary to restart the computer for the changes to take effect.

## To set the binding order in Windows 10 and Windows Server 2016

Microsoft has changed how network protocol bindings are handled in its newest operating systems. Windows now uses an Interface Metric number to determine priority.

To check the existing priorities of your network cards:

1. Launch the Windows PowerShell as administrator.
2. To view the current priority, enter the following cmdlet:

**Get-NetIPAddress**

```

Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> Get-NetIPAddress

ifIndex InterfaceAlias           AddressFamily NlMtu(Bytes) InterfaceMetric Dhcp    ConnectionState PolicyStore
-----
1       Loopback Pseudo-Interface 1 IPv6          4294967295   75     Disabled Connected ActiveStore
4       isatap.dev.wonderware.com IPv6          1280        75     Disabled Disconnected ActiveStore
2       RMC Primary                 IPv4          1500        100    Enabled Disconnected ActiveStore
5       Primary                     IPv4          1500        5      Enabled Connected ActiveStore
1       Loopback Pseudo-Interface 1 IPv4          4294967295   75     Disabled Connected ActiveStore

PS C:\WINDOWS\system32>

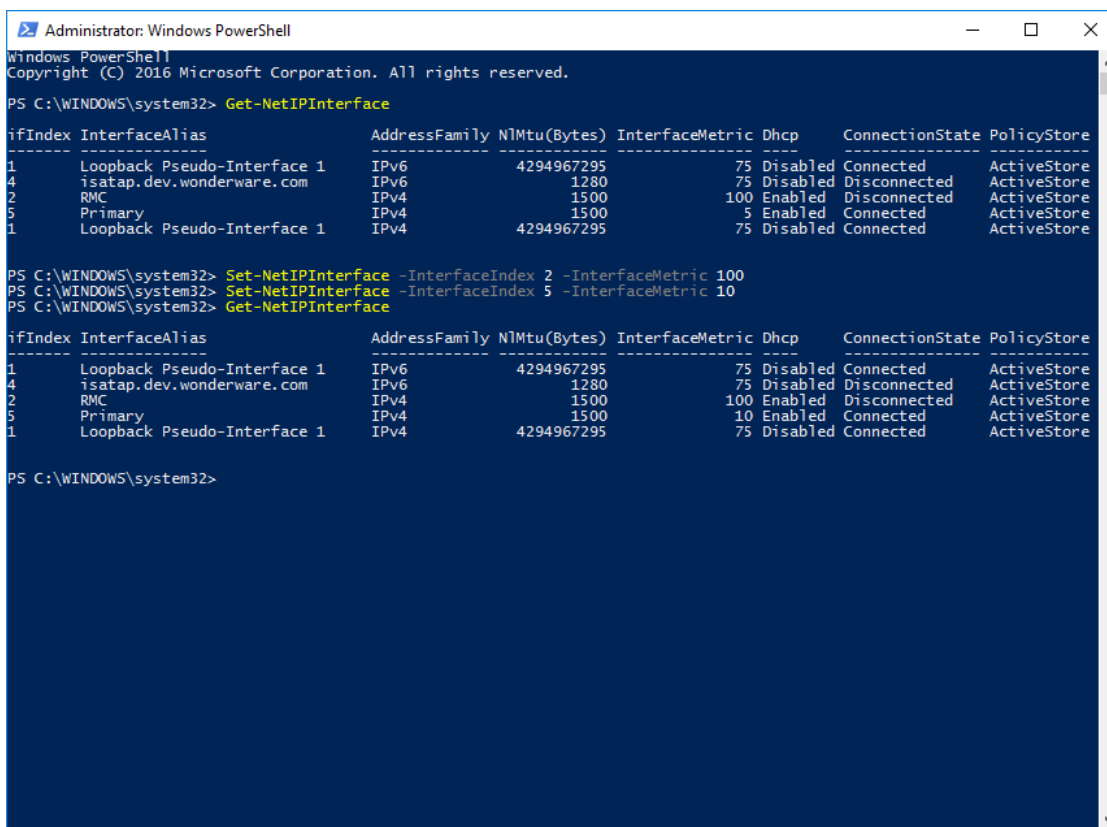
```

This displays the Interface List. An interface metric number is listed for each network interface card/adaptor. Note that the lowest number has the highest priority.

3. Identify the NIC cards and define the binding order (priority). Check the InterfaceIndex numbers for each NIC. It could be that your cards already have the correct priorities (higher InterfaceIndex numbers have lower priority).
4. If you need to change the priority of your interface cards, use the following cmdlet to change the InterfaceIndex for one or more network cards. For example:

**Set-NetIPAddress -InterfaceIndex 2 -InterfaceMetric 100** (lower priority)

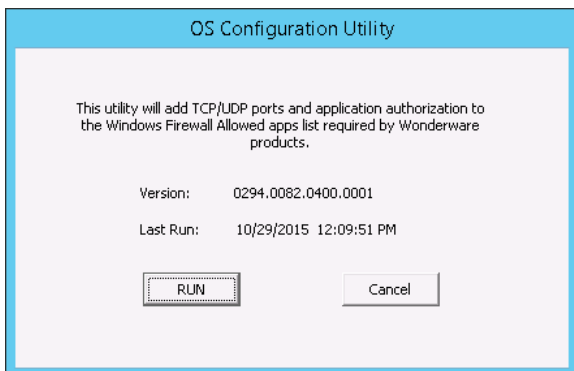
**Set-NetIPInterface -InterfaceIndex 5 -InterfaceMetric 10 (higher priority)**



5. Close the PowerShell.

If the two computers are set up properly, you can install Application Server and other System Platform software, and the OSConfigurationUtility sets up the firewall exceptions appropriately.

If you install Application Software and other System Platform software before configuring the NICs properly, configure the NICs as described in this section and then run the OSConfigurationUtility again. To do this, run the OSConfigurationUtility.exe, located in the **C:\Program Files (x86)\Common Files\ArcheStrA** folder.



To verify that the firewall exceptions are set, open **Control Panel** and then open the **Windows Firewall** application. Verify that the executables for the System Platform products you have installed are allowed to communicate through the firewall.





# CHAPTER 17

## Working with Redundancy

You can set up and run Application Server in a redundant environment. For more information about redundancy, see the AVEVA Software Global Customer Support website.

### About Redundancy

A failover is the condition during which run-time operations are moved from one critical component to another. Failover can occur due to failure conditions or it can be forced manually, called a forced failover.

Application Server provides redundancy in two critical functions:

- **AppEngine:** You must configure both an AppEngine and two WinPlatforms.
- **Data acquisition:** You must configure two DIOjects (data sources) and a RedundantDIOject.

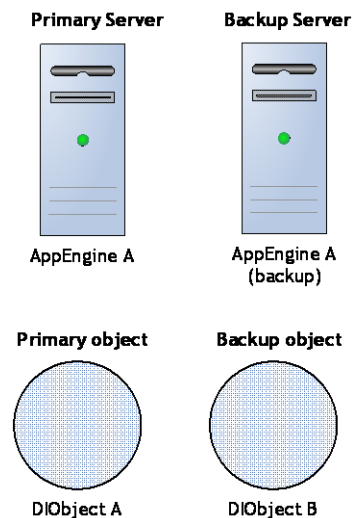
### Configuring AppEngine Redundancy

The Primary/Backup AppEngines form a redundant pair. The ArchestrA infrastructure will generate the backup AppEngine automatically when redundancy is enabled for an AppEngine. Hierarchy of ApplicationObjects can only be assigned to the primary AppEngine. The primary and backup engines need to be assigned to redundancy enabled platforms, and they can be deployed separately.

For data acquisition, the Primary/Backup DIOjects (the data sources) must be separately created, configured, and deployed. Also, you must create, configure, and deploy a RedundantDIOject to control failovers between the two data source objects.

In a redundant system, install and configure the primary OPC server on the backup engine node.

When you configure redundancy, you configure the Primary object and the Backup object.

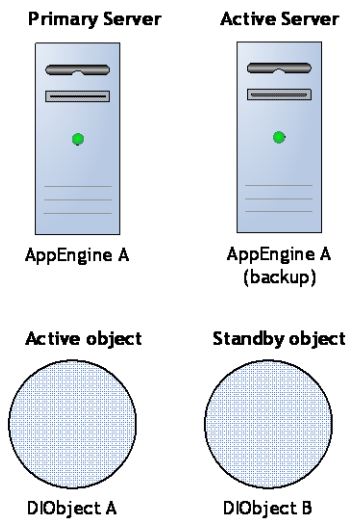


- **Primary object:** The main or central object that provides the functionality during run time. For AppEngines, this is the object you enable for redundancy. For data acquisition, this is the DIOject you intend to use first as your data source in run time.

- **Backup object:** The object that provides the functionality of the Primary object when the Primary object fails. For AppEngines, this is the object created by the ArchestrA infrastructure when the Primary object is enabled for redundancy. For data acquisition, this is the DIO object you do not intend to use first as your data source in the run-time.

## Redundancy during Run Time

When you deploy these objects to a run-time environment, the configuration objects become the Active object and the Standby object.



- **Active object:** The object that is currently executing functions. For AppEngines, it is the object that is hosting and executing ApplicationObjects. For data acquisition, it is the object that is providing field device data through the RedundantDIO object.
- **Standby object:** The object that is waiting for a failure in the Active object or for a force-failover. For AppEngines, it is the object that monitors the status of the Active AppEngine. For data acquisition, it is the object that is not providing field device data through the RedundantDIO object.

In the AppEngine redundancy environment, the Active and Standby objects monitor each other's statuses and switch during a failover situation.

During a failover, when an Active engine fails, it restarts and the Standby engine becomes active. The Active engine then turns to the standby mode. In the data acquisition environment, the RedundantDIO object monitors the status of the two DIO object data sources, and handles the switching from Active to Standby objects.

The relationship between the configuration time (Primary/Backup) and run-time (Active/Standby) object pairs is not static. In the run time, either the Primary or Backup object can be the Active object at any particular time. Whenever one becomes the Active object, the other automatically becomes the Standby.

## Engine Restart After Failure

An AppEngine will restart automatically after an engine failure such as a crash or hang. If the engine is redundant, it will synchronize with its redundant partner, and running applications will resume.

Engine restart on failure is automatic and cannot be disabled.

## CPU Load Balancing

CPU load balancing for redundant AppEngines can be used to maintain system stability by limiting high CPU usage during AppEngine startup after failover. When CPU usage by a core is close to 100%, other processes can be starved and startup times can extend as a result. When CPU load balancing is enabled, objects are loaded in discrete chunks, thus improving CPU utilization.

Application Server includes a registry key that you can use to configure CPU load balancing for redundant engines. CPU load balancing is disabled by default, and the configuration of each node is independent. That is, you can enable CPU load balancing on some nodes and not on others, and have different settings for each node.

---

**Note:** CPU load balancing will only work on run-time nodes with redundant engines.

---

- Registry Subtree – 64-bit:  
HKEY\_LOCAL\_MACHINE\SOFTWARE\Wow6432Node\ArchestrA\Framework
- Registry Subtree – 32-bit:  
HKEY\_LOCAL\_MACHINE\SOFTWARE\ArchestrA\Framework
- Registry Key:  
EngineFailoverStartup

Subkey entries used for enabling and tuning CPU load balancing are:

---

Subkey Name	Default Value	Description
EnableCPUBalancing	0	Turns CPU load balancing on or off (0 = OFF, 1 = ON)
LoadObjectsNumberPerGroup	50	Specifies the number of objects loaded at a time during startup. If set to 0, this function is disabled.
LoadObjectsTimeToSleep	200	Specifies in milliseconds, the time between loading groups of objects (end of load to start of next load). If LoadObjectsNumberPerGroup is 0, sleep is disabled.
FirstScanCycleNumberPerGroup	25	Specifies the number of objects to be initialized during the first scan cycle after failover. If set to 0, this function is disabled.
FirstScanCycleTimeToSleep	100	Specifies in milliseconds, the time between initializing groups of objects (end of first cycle to start of second). If FirstScanCycleNumberPerGroup is 0, sleep is disabled.

---

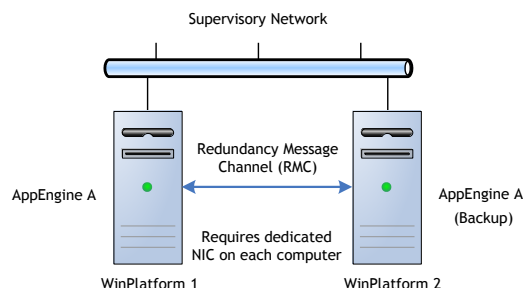
## Working with AppEngine Redundancy

You enable AppEngine redundancy in the Primary AppEngine. You must also configure two WinPlatforms for redundancy, one to host the Primary AppEngine and one to host the Backup AppEngine.

**Important:** WinPlatforms hosting redundancy-enabled AppEngines must be deployed to computers running the same operating system.

The configuration of both WinPlatforms should be the same. At a minimum, the store-forward directory configurations must be common to both WinPlatforms:

During configuration, you can assign Primary and Backup AppEngines to the same WinPlatform. To deploy, you must assign the Primary and Backup AppEngines to different WinPlatforms.



Each production computer hosting a redundancy-enabled AppEngine must have a minimum of two network cards. One NIC is for the supervisory network and PLC network, if the computer has only two network cards. The other must be for a dedicated Ethernet connection between computers for the redundancy message channel (RMC).

For information about distributed networks, see *Using Multiple Network Interface Cards* on page 539.

The RMC handles redundancy monitoring, message handling and data synchronization between redundant pairs.

## Configuring the Redundancy Message Channel

For the redundant pair of AppEngines to successfully communicate with each other, you must define the correct order of network connections in the network services of each computer.

We recommend that you name each card with a clearly identifiable function (for example, "Supervisory Net" and "Redundant Message Channel").

You must configure at least the following parameters:

- Redundancy Message Channel IP Address
- Redundancy Message Channel Port
- Primary Message Channel

### To configure network cards

1. Open the **Network Connections** dialog box in Windows. The specifics about opening this dialog box varies, depending on the version of Windows you are working on.
2. Click **Advanced Settings**. If the computer name is used in the platform's "node name" box, the first connection in the list must be the supervisory network card. Use the up and down arrows to define the correct order.
3. **Connections.**  
If a computer contains more than two network cards, the supervisory net must be listed first and the RMC connection can be listed in any other position. For example, your computer might have a supervisory connection, a field device connection, and a RMC connection.
4. Configure the DNS settings for the supervisory network card to function properly.
  - On the **DNS** page of the **Advanced TCP/IP Settings** dialog box, select the **Register this connection's addresses in DNS** check box.

- For the RMC network card to function properly, clear the **Register this connection's addresses in DNS** check box. For more details on these settings, see *Using Multiple Network Interface Cards* on page 539.
5. In Application Server, open the WinPlatform in the Object Editor for the computer you just configured.
  6. Change the **Redundancy Message Channel IP Address** to the IP address of the RMC connection. See the WinPlatform Help for more information about these parameters.
  7. Save and close.

## Configuring Redundancy

You configure redundancy in AppEngine/WinPlatforms objects using their Object Editors.

The redundancy-related parameters in the AppEngine Object Editor are located on the **Redundancy** and **General** tabs.

An AppEngine that is part of a redundancy pair has a deployment status indicating its own status and that of its partner object. These statuses are visually indicated in the **Application** views.

You can set the status of a redundancy pair to one of the following states:

Pair Deployed	Both Primary and Backup AppEngines are deployed.
Pair Undeployed	Both Primary and Backup AppEngines are undeployed.
Partial Deployed	Either the Primary or Backup AppEngine is deployed and its partner is not deployed. If an AppEngine has a Partial Deployed status, its partner has a Partial Undeployed status.
Partial Undeployed	Either the Primary or Backup AppEngine is undeployed and its partner is deployed. If an AppEngine has a Partial Undeployed status, its partner has a Partial Deployed status.

### To create AppEngine redundancy

1. In the Primary AppEngine, select the **Enable Redundancy** check box on the **Redundancy** tab.
2. Configure the remaining redundancy parameters as needed. See the help file for the AppEngine for specific information about these parameters.
3. On the **General** tab, set the **Engine Failure Timeout** option to 2000 milliseconds. You may need to tune this parameter between 2000 ms and the default 10000 ms, depending on the requirements of your application.



When the Active engine fails during a failover, it restarts automatically and becomes the Standby engine.

---

**Note:** The actual engine failure time-out is three times the value of this parameter. If you set the parameter to 2000 ms (2 seconds), a failover occurs if the AppEngine fails to communicate with the computer's Bootstrap for 6 seconds. A setting of 10000 ms (10 seconds) can be too long a wait period (30 seconds) for a well-functioning redundancy operation.

---

After you save the configuration of the object and check it into the Galaxy, the icon for the object changes. A Backup AppEngine is created with the same configuration as the Primary object.

Icon	Object
	Primary AppEngine
	Backup AppEngine

## Configuring Redundancy in Templates

You can create a redundancy-enabled AppEngine template. When you create an instance of the template, both the Primary and Backup instances are created.

The Backup AppEngine is hosted by the Unassigned Host or the default WinPlatform if you specified one in the **Configure User Information** dialog box.

If you change the configuration and check in the Primary AppEngine, the Backup AppEngine:

- Is checked out in the background
- Updates the configuration
- Is checked in without notification to connected clients

## Deleting Redundant AppEngines

You can disable redundancy in an AppEngine after the ArchestrA infrastructure creates the Backup object. If you disable redundancy and check in the Primary AppEngine, the Backup is deleted from the Galaxy. The exception is when the Backup is already deployed. In that case the newly configured Primary object cannot be checked in.

To delete a Backup AppEngine from the Galaxy, you must undeploy it first.

There is no redundancy-related configuration required on ApplicationObjects hosted by an AppEngine configured for redundancy. When a system failure occurs, the ApplicationObjects and their attribute values are duplicated on the Standby AppEngine, which becomes the Active AppEngine. For more about failover functionality, see *During Deployment* on page 552.

If the Platform hosting the redundant AppEngine is shutdown in SMC, the AppEngine cannot be flagged as "On failure mark as undeployed" when you undeploy the AppEngine. You see an engine communication error in the **Deploy** dialog box.

## Deploying AppEngine Objects

Primary and Backup AppEngines can be deployed together or individually.

- When they are deployed together, regardless of which object is actually selected for deployment, the Primary always becomes the Active and the Backup becomes the Standby.
- When they are deployed individually, the first one deployed becomes the Active.

Hosted ApplicationObjects are always deployed to the Active AppEngine. When deploying the first of a redundant pair of AppEngines, you can cascade deploy all objects it hosts. This operation can be paired with deploying both the Primary and Backup AppEngines at the same time.

---

**Important:** If you deploy the Backup AppEngine first and then deploy hosted objects to that AppEngine, make sure the network communication to both target computers is good before deploying the Primary AppEngine. Otherwise, errors occur.

---

In the run-time environment, either the Primary or Backup AppEngine can become the Active or Standby depending upon failure conditions on either computer.

## Configuration Requirements

Before deploying the Primary and Backup AppEngines, all configuration requirements must be met.

- Each AppEngine must be assigned to a separate WinPlatform.
- A valid redundancy message channel (RMC) must be configured for each WinPlatform.
- To deploy the Primary and Backup together, select **Include Redundant Partner** in the **Deploy** dialog box. This option is not available when doing the following operations:
  - Cascade deploy from the Galaxy
  - Multiple object selection deploy
  - Deploying the WinPlatform that hosts the Primary or Backup AppEngine

The following table shows a matrix of allowed operations based on specific conditions.

Condition	Deploy Both Primary and Backup Objects	Cascade Deploy Allowed
Backup AppEngine's host WinPlatform configured for failover and deployed	Yes	Yes
Backup AppEngine in error state	Yes	Yes
Backup AppEngine's host WinPlatform not deployed	No	Yes
Backup AppEngine's host WinPlatform not configured for failover and deployed	Yes	Yes
Deploy from Galaxy node	No	Yes
Deploy from WinPlatform hosting Primary AppEngine	No	Yes
Multiple object selection deploy	No	No
Backup AppEngine's host WinPlatform not configured for failover and not deployed	No	Yes
Deploy from Backup AppEngine	Yes	Yes
Deploy from Primary AppEngine	Yes	Yes

## Undeploying AppEngine Objects

Undeploying redundant pairs of AppEngines is just like undeploying any regular object.

You can undeploy the Active and Backup AppEngines separately or as a pair.

**Important:** Undeploying any objects, including redundant AppEngine pairs, does not uninstall code modules for that object from the hosting computer. Code modules are uninstalled only when the WinPlatform is undeployed.

### To undeploy as a pair

1. Select one of the objects in an Application view.
2. On the **Object** menu, click **Undeploy**. The **Undeploy** dialog box appears.
3. Select **Include Redundant Partner** in the **Undeploy** dialog box and click **OK**.

## During Deployment

During initial deployment of a redundant pair of AppEngines, files are deployed in the following order:

- Code modules and other files for the Primary AppEngine are deployed
- Those files for its assigned ApplicationObjects
- All of these files are deployed to the Standby AppEngine by the Active engine's WinPlatform using the redundancy message channel (RMC)

---

**Note:** If some or all of these files already exist on the Standby AppEngine's WinPlatform, perhaps, assigned to another AppEngine on that platform, only the delta files are deployed to the Standby AppEngine.

---

## Objects at Run Time

Objects are always assigned to the Primary AppEngine in the configuration environment.

During run time, objects are always deployed to the Active AppEngine whether or not it was initially configured as the Primary object.

All files are deployed by the Active AppEngine's WinPlatform to the Backup AppEngine as described in the previous section.

## During Run Time

In the run-time environment, the Active and Standby AppEngines first attempt to establish communication across the RMC. This occurs when an AppEngine belonging to a redundant pair first starts. Therefore, if one AppEngine is relocated later to a different WinPlatform, this communication between AppEngines can be reestablished.

During run time, the Active and Standby engines communicate with each other and monitor each other's status.

In the case of a hardware or software failure on the Active computer, the Standby AppEngine becomes the Active one. To move the new Standby AppEngine from its hosting computer, undeploy this AppEngine by selecting the **On failure mark as undeployed** option on the **Undeploy** dialog box. Reassign and redeploy it to a WinPlatform that is configured for redundancy on another computer.

## AppEngine Redundancy States

Redundant pairs of AppEngines can have one of the following states at a time:

- **Active:** The state of an AppEngine when it has communication with its partner object, its partner is in Standby-Not Ready, Standby-Sync'ing with Active, or Standby-Ready state. A Standby AppEngine transitions into this state when a fail over condition is detected. In this state, an AppEngine schedules and runs deployed objects, sends checkpoint data and sends subscriber list updates to the Standby AppEngine.



- **Active - Standby not Available:** The state of an Active AppEngine when it determines it cannot achieve communications with its partner object. This could mean that checkpoint, subscription and alarm state changes are not successfully transmitted to the Standby object because the partner AppEngine is not deployed, number of heartbeats missed from standby objects exceeds the configured max consecutive number of heartbeats missed, or notification is received that the Standby AppEngine shutdown or is not running. If an AppEngine is in this state, it 1) continues normal execution of hosted objects, 2) cannot be manually switched to Standby state, and 3) while continuing to attempt communicate with the Standby, does not attempt to send data to the Standby object.
- **Determining Failover Status:** The initial state of a redundancy-enabled AppEngine when it is first started. It has not determined yet whether it is the Active or Standby AppEngine. Communication between the two AppEngines is attempted first over the RMC and then over the primary network to make this determination. If communication cannot be made after a certain time-out period, an AppEngine assumes the Active role if it has all of the code modules and checkpoint file data to do so. Continued attempts are made at communicating with its partner.
- **Standby - Missed Heartbeats:** The state of an AppEngine when 1) the heartbeat pings are not received from its Active partner through the RMC, but the number of missed beats haven't reached the maximum consecutive number of heartbeats missed, or 2) the heartbeats from active engine have been missed through the primary channel. When in this state, the Standby object attempts to determine whether or not the Active object failed. If a manual failover is started by using the ForceFailoverCmd attribute, it is processed only if the heartbeats were missed over the primary network and not missed over the RMC.
- **Standby - Not Ready:** The state of an AppEngine when one of several conditions occurs: 1) its lost communications with its partner object or it maintains communications with its partner but missed checkpoint updates or alarm state changes from the Active AppEngine, 2) new objects are deployed to the Active AppEngine and necessary files are not installed on the Standby AppEngine yet, or 3) the Standby AppEngine lost communications over the RMC before it completed synchronizing data. Typically, the AppEngine's partner is in one of the following states: Active-Standby not Available, or Active.
- **Standby - Ready:** The state of an AppEngine when it completed synchronizing code modules and checkpoint data with the Active AppEngine. In this state, the AppEngine monitors for Active AppEngine failure by verifying heartbeat pings received from the Active engine and checks that all files required for execution are in sync with the Active engine. It receives the following from the Active AppEngine: checkpoint change data, subscription-related notifications, alarm state changes, and history blocks.
- **Standby - Sync'ng with Active:** The state of an AppEngine when it is synchronizing code modules with the Active object. If code modules exist on the Standby computer that do not exist on the Active node, they are uninstalled, and likewise, any code modules that exist on the Active node but not on the Standby node are installed. After all code modules are synchronized, the AppEngine transitions to Standby-Sync'd Code state.
- **Standby - Syncing Code:** The state of a Standby AppEngine that successfully synchronized all code modules with the Active object.
- **Standby - Syncing Data:** The state of a Standby AppEngine when all object-related data, including checkpoint and subscriber information, are synchronized with the Active object. An object in this state typically transitions to Standby-Ready state.
- **Switching to Active:** A transitional state when a Standby AppEngine is commanded to become Active.
- **Switching to Standby:** A transitional state when an Active AppEngine is commanded to become Standby. When the active engine is switched to standby, the engine will be restarted. This transition state can be switched off by the user changing the attribute of the engine.

- **Failed:** The state of a redundant partner when its process crashes or is terminated by the user. The AppEngine process can be restarted using System Management Console/PlatformManager.
- **Unknown:** The state of a redundant partner when a communication loss occurs between AppEngines or when the partner AppEngine is stopped, shutdown, or undeployed.

For examples on redundant configuration, see the AVEVA Software Global Customer Support website.

## Troubleshooting

Most troubleshooting happens in the System Management Console. For more information about using the System Management Console, see the System Management Console User's Guide.

Certain requirements are validated by the system infrastructure. For example, the order in which you configure an object pair for redundancy is validated.

The following problems can occur when you are using redundancy:

- You can configure an AppEngine for redundancy before configuring its associated WinPlatform. If you do this, you see an error message that the Platform (specifically, the RMC) is not configured yet.
- If the **RMC IP Address** parameter is not configured in both hosting WinPlatforms, then the configuration state of both Primary and Backup AppEngines changes to Error. You also see a message indicating that the host WinPlatform is not configured with the network adapter required for redundant communications. When the RMC IP Address is configured and the WinPlatforms are checked in, the hosted AppEngines are automatically revalidated and the Error state is resolved. If hosted AppEngines are checked out, they are not revalidated.
- If both Primary and Backup AppEngines are assigned to the same WinPlatform and you try to deploy both engines, both the Primary and Backup fail to deploy. You see a message that the Primary and Backup objects must be hosted by different WinPlatforms. Reassign the Backup object to another WinPlatform and deploy it separately.
- If both the **Network Address** and **RMC IP Address** parameters in the WinPlatform's editor refer to the same network card, you get a warning message when you save the configuration. These parameters must refer to different network cards.
- Before restarting a computer that hosts one of a redundant pair of AppEngines (either the Active or Backup), ensure that the Primary Network is connected. A restart while the Primary Network is disconnected makes the Primary Network bind to the RMC's IP address. An incorrect redundancy state occurs, indicating that redundancy functionality is good. Any time you restart a redundancy-enabled computer, check for proper network connections afterwards. For more information, see *Using Multiple Network Interface Cards* on page 539.
- When scripting is used to activate attributes, attributes displayed in the watch window may appear to remain in an initializing state after deployment or engine failover. This happens because the script is capturing the attributes while they are still initializing on one of the redundant engines, before the engines are able to synchronize. The attributes will move to active state, but unless there is retry logic in the script, that information will not be returned. This situation can be avoided by ensuring that your scripts contain retry logic to capture the updated state of the attributes.

## Generating Alarms

When failover conditions occur, the ArchestrA system reports alarms to subscribed alarm clients like InTouch.

These alarms contain the following information:

- The name of the AppEngine reporting the alarm.
- The node name of the AppEngine reporting the alarm.
- The state of the AppEngine.

- The node name of the AppEngine's partner object.

Depending on what caused the failover, the Standby AppEngine can become the Active AppEngine in an Off scan state and alarms might not be generated.

If the Active AppEngine is shutdown off scan, the checkpointer can transfer that state to the Standby. When the Standby becomes the Active, it starts off scan. When the AppEngine is put on scan, alarms then are generated.

Reported alarms include the following:

Alarm	Previous State	Current State	Alarm Raised When	Alarm Cleared When	Alarm Reported By
Standby Not Ready*	Active	Standby Not Ready	Standby Not Ready	Entering Standby Ready	Active Engine
Standby Not Available	Active	Active Standby Not Available	Active Standby Not Available	Entering Active	Active Engine
Failover Occurred			Standby becomes Active	During the next scan of the Active engine	Active Engine

\* The Active AppEngine monitors the status of the Standby through the RMC to determine when to raise this alarm. Also, if the Active AppEngine is in Active-Standby not Available state, this alarm is not generated.

When a failover occurs, the Standby AppEngine that becomes active carries alarms outstanding from the old Active AppEngine. The state of those old alarms, though, will change to reflect the new partner's status.

Timestamps are preserved for alarms, including when:

- The alarm was acknowledged
- The alarm condition went true
- The alarm condition went false

Finally, the following information is preserved for alarms:

- An alarm was acknowledged
- The message input by the operator when the alarm was acknowledged

All alarm state information is collected and sent to the Standby AppEngine at the end of a scan cycle and before being sent to alarm clients.

The sequence of reporting alarms ensures that alarm clients do not report alarms in states that are different from those reported by the Standby AppEngine if the Active AppEngine fails.

## Generating History

All active objects (AppEngine and hosted objects) report history data as they normally do in the run-time environment.

Historical data is reported to the historian only from the Active AppEngine.

Losing connectivity with the historian does not cause a failover. The Active AppEngine goes into store-forward mode and caches data every 30 seconds. Store-forward data is synchronized with the Standby AppEngine.

When failover conditions occur, no more than 30 seconds of history data is lost in the transition from Standby to Active status. For more information, see the AVEVA Software Global Customer Support website.

## Working with Data Acquisition Redundancy

The RedundantDIOject monitors and controls the redundant DIOject data sources at the object level. Unlike redundant AppEngines, individual DIOject data sources do not have redundancy-related states. For all practical purposes, they function as standalone objects.

Only one DIOject data source provides field device data through the RedundantDIOject at a time. Both data sources must have commonly -configured DAGroups. These must be reflected in and channeled through the RedundantDIOject, which monitors the two DIOject data sources. It also determines which one is Active at any given time. Both data sources must also have the same item address space.

After deployment or engine failover, attributes may appear to remain in an initializing state when scripting is used to activate the attributes. The script may capture the attributes while still initializing, before the engines are able to synchronize with each other. When synchronization occurs, the attributes will move to active state, but unless there is retry logic in the script, that information may not be returned. This situation can be avoided by ensuring that your scripts contain retry logic to capture the updated state of the attributes.

## Configuring Data Acquisition Redundancy

Data acquisition redundancy objects involve two DIOjects and the RedundantDIOject. In data acquisition redundancy, you must configure all three components:

- Primary DIOject data source
- Backup DIOject data source
- Redundant DIOject data source

Because data acquisition redundant components are essentially standalone objects, all valid operations that apply to any other ApplicationObjects apply to the three objects.

All IDE commands, Galaxy Dump and Load functions, and import and export operations are valid on the two DIOject data sources and the RedundantDIOject.

See the online help associated with each DIOject for help in configuring its Object Editor. Also see the help associated with the RedundantDIOject.

Before you can deploy the RedundantDIOject, you must configure at least one scan group. Also, configure only scan, block read, and block write groups shared by the Primary and Backup DIOjects in the RedundantDIOject.

### To configure the Redundant DIOject

1. On the **General** tab of the Object Editor, set the **Primary DI Source** and **Backup DI Source**.
2. Set up the common scan groups.
3. Set up the common block read and block write groups on the tabs of the Object Editor.

## Deploying Redundant DIOjects

Deployment for data acquisition redundancy is the same as individually deploying unrelated objects. No special conditions apply to each DIOject data source and the RedundantDIOject.

See *Deploying and Running an Application* on page 317 for more information about deploying objects.

## What Happens in Run Time

The three objects in the data acquisition redundancy scheme (RedundantDIOobject and its two DIOobject data sources) work like any other ArcestrA object when deploying, alarming, and historizing. They have no special redundancy-related states or restrictions.

During run time, the RedundantDIOobject monitors the status of the DIOobject data sources, and handles the switching from Active to Standby object if failure conditions occur.

## RedundantDIOobject and PLC Connectivity

For the RedundantDIOobject, you can use the scan group PingItem attribute to monitor the connection status of the PLC at run time. If you are using the redundancy feature of the RedundantDIOobject to communicate with DIOobjects, you should configure the PingItem attribute for each scan group.



# APPENDIX A

## Managing Security for Application Server

### General Considerations for Security

Before you review the information in this section, it is recommended that you go through the following checklist to ensure you plan to cover the security areas that apply to your ICS and organization.

Security Area	Reference Section
Physical and virtual access to the host	<i>General Guidelines for Securing the Host</i> on page 560
Latest Windows patches applied	<i>Windows Updates</i> on page 561
Protecting the host from viruses and malware	<i>Scanning the Host</i> on page 561
Access to content on the host	<i>Protecting the Applications and Content on the Host</i> on page 562
Securing your network	<i>Securing the Network</i> on page 562
Configuring services and ports	<i>Managing Network Services and Ports</i> on page 563
Securing client/server communication	<i>Securing Communication between the Client and Server</i> on page 564
User and group management	<i>Securing Systems through Authentication and Authorization</i> on page 565
Planning for emergencies	<i>Contingency Planning</i> on page 566

For a list of security feature help topics refer to the table at the end of this section.

### Introduction

This appendix provides a general overview on how to securely deploy your AVEVA software product as an Industrial Control Systems (ICS) application.

AVEVA's approach to securing site networks and ICS software is driven by the following principles:

- View security from both Management and Technical perspectives
- Ensure that security is addressed from both IT and ICS perspectives.
- Design and develop multiple network, system and software security layers.
- Ensure industry, regulatory and international standards are taken into account.
- Aim to prevent security breaches, supported by detection and mitigation.

These principles are realized by implementing the following security recommendations:

- Prevent security breaches using the following components:
  - Firewalls
  - Network-based intrusion prevention/detection
  - Host-based intrusion prevention/detection

- Segregate IT and Plant networks
- Include a clearly defined and clearly communicated change management policy. For example, firewall configuration changes.

This appendix is not meant to be comprehensive, and it does not provide any detailed instructions. It is only a collection of basic concepts and recommendations that you can use as a checklist to secure your own systems. If you need help with a specific item in this guide, see the official documentation for that item -- for example, if you need help with your anti-virus software, see the documentation for that software.

---

**Note:** AVEVA strongly recommends following the guidelines prescribed by the U.S. Department of Commerce for securing ICS software. The document "Guide to Industrial Control Systems (ICS) Security" [NIST Special Publication 800-82 Revision 2](<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>) provides detailed information about ICS, typical system topologies, security threats and vulnerabilities, and recommendations for implementing security measures.

---

## Securing the Host

Given the sensitive nature of industrial control, it is important to secure not only the ICS software, but also:

- the host on which it runs
- the network to which it is connected
- the hardware used for the ICS software.

---

**Note:** The "host" is the Windows computer or Windows Embedded device on which your ICS software is installed and running.

---

There are several factors to consider for securing the host including:

- Access to the host
- Keeping track of and applying the latest Windows updates
- Keeping the host computer free of viruses and malware
- Protecting the applications and content on the host

Each of these factors is covered in the sections below.

## General Guidelines for Securing the Host

Here are a few guidelines to secure the host:

- Use an account with administrative privileges to install the ICS software, and one without administrative privileges to run the ICS software.
- Restrict configuration of ICS to a limited set of users.
- Consider running the ICS software as a Windows service, if that option is available. If the ICS software is run as a service, run it as a low privileged virtual service account.
- Once the host is fully configured and placed in its permanent location, restrict physical access and remote access to it so that only authorized personnel (for example, system administrators, application engineers, run-time operators) can use it.
- Consider disabling or removing physical ports (for example, USB, memory card) that might be used to connect external storage devices and then transfer data.



## Windows Updates

Check that the Windows operating system on the host is a version that is under what Microsoft calls "mainstream support", which means Microsoft actively maintains and releases updates for it. Older versions of Windows are under Microsoft "extended support", which means they are not actively maintained and therefore might become vulnerable without notice. For more information about the different versions of Windows and the different levels of support, see [Windows lifecycle fact sheet](<https://support.microsoft.com/en-us/help/13853/windows-lifecycle-fact-sheet>).

Automate Microsoft product updates using Microsoft Windows Server Update Services (WSUS), which enables you to manage and distribute updates to computers on your network. For more information about WSUS, see [Windows Server Update Services](<https://docs.microsoft.com/en-us/windows-server/administration/windows-server-update-services/get-started/windows-server-update-services-wsus>). If the host does not or will not have a reliable connection to the WSUS server, perhaps because it is located on a private network, you can either develop a procedure to manually apply updates or consider changing the operating system to a Long-Term Servicing Channel (LTSC) version of Windows, which is updated less frequently.

In addition, AVEVA ICS software is tested for compatibility with Microsoft updates the results of which are published on the *Security Central site* <https://softwaresupportsp.aveva.com/#/securitycentral>. Security advisories and bulletins are also published on this site.

The screenshot shows the AVEVA Security Central interface. At the top is the AVEVA logo and navigation icons. Below is a search bar and a menu icon. The main content area is titled "Security Central" and has three tabs: "Microsoft Security Updates Reports" (selected), "Product Cyber Security Updates", and "Policy & Guidelines". Under "Select Product Line:", "Wonderware" is selected. To the right, there is a link for "Archived Security Central Supported Products" with a document icon. Below this is a table of security updates.

Posted	Report	Status	MS Security	Description	Microsoft KB/OS
Feb 11, 2020	WW20-022	In Testing	<a href="#">Release Notes</a>	Security Update for Adobe Flash Player. (KB4537759)	<a href="#">View</a>
Feb 11, 2020	WW20-021	In Testing	<a href="#">Release Notes</a>	Security Only Update for SQL Server. (KB4532098, KB4532095, KB4535288, KB4535706, KB4532097)	<a href="#">View</a>
Feb 11, 2020	WW20-020	In Testing	<a href="#">Release Notes</a>	Microsoft Office (KB4484163, KB4484267, KB4484264, KB4484156, KB4484265, KB4484255, KB4484250, KB4484256)	<a href="#">View</a>
Feb 11, 2020	WW20-	In Testing	<a href="#">Release Notes</a>	Cumulative Security update for Internet Explorer	<a href="#">View</a>

## Scanning the Host

Use both anti-virus and anti-malware software and file integrity checking software to regularly scan the host.

Windows includes Windows Defender by default, but you may choose to install and use additional software that scans for more types of malware or performs other functions. If you do that, make sure the software is provided by a reputable company. And, as with the operating system, if the host does not or will not have reliable access to the software's update service, develop a procedure to manually apply updates. If you develop a manual update procedure, it should account for all devices on a network or at a site, because a single outdated device can leave the entire network or site vulnerable.

## Protecting the Applications and Content on the Host

To protect the applications and content on the host:

- Enable Windows Firewall, and configure it to close all ports that are not used by the ICS software. For more information about port usage, see *Managing Network Services and Ports*.
- Disable Windows features like remote desktop and file sharing, and remove unnecessary programs like games and social media.
- Restrict access to the files, databases, registry and other resources on the host.
- Use Windows BitLocker to encrypt the hard drive of computers that are either mobile or not located in a secure facility. However, BitLocker may impact the performance of computers.
- Consider using server-class storage (SANs) infrastructure to avoid storing sensitive data on mobile devices.

## Securing the Network

Usually the host computer will have some sort of network access; it is increasingly rare for an ICS device to run as an entirely standalone device. The host may use the network to communicate with other ICS components such as controllers, sensors, databases, remote clients, and even other hosts in peer-to-peer relationships. You may also use the network to manage several ICS devices from a development or supervisory workstation.

Once you determine that the host will have network access, decide how it will connect to the network. In recent years there has been a shift from wired networks (that is, "Ethernet") to wireless networks ("Wi-Fi"), even for business and industrial uses. We recommend against using Wi-Fi for your ICS network because you do not have physical control over who or what might access the network. Any computer or device within range of the Wireless Access Point (WAP) can try to access the network, and even if the network is ostensibly secure, an intruder can intercept and analyze network traffic and potentially discover a vulnerability.

Nevertheless, if you decide to use Wi-Fi for your ICS network, enable all access control features on the WAP including encryption (for example, WPA/WPA2), a strong password and a list of authorized MAC addresses. Do not try to "hide" the Wi-Fi network by disabling broadcast of the Service Set Identifier (SSID), because doing so actually generates more network traffic that can be intercepted and analyzed.

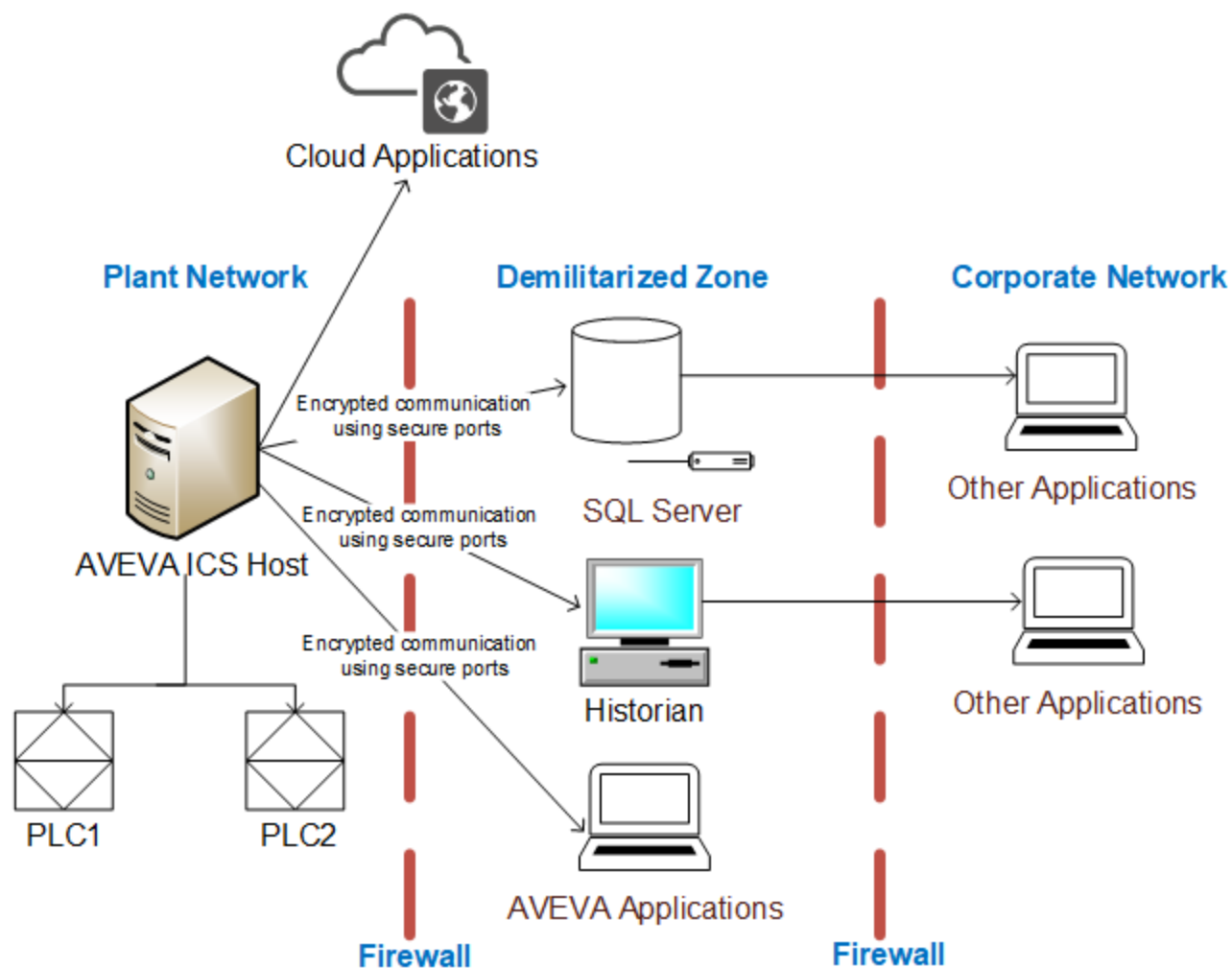
## Segmenting the ICS Network

The ICS network itself can be either physically or logically segmented from your other corporate networks. A physically segmented network is by definition the most secure. The network hardware and all computers and devices connected to it form a single closed network with no physical connection to any other network, so an intruder cannot access the network unless they also have access to the physical location.

In contrast, a logically segmented network is physically connected to your other corporate networks and/or the public internet, but it uses various methods to segregate ICS network traffic from other network traffic. This may include:

- Using a unidirectional gateway
- Implementing a Demilitarized Zone (DMZ) network architecture with firewalls to prevent network traffic from passing directly between the corporate and ICS networks
- Having different authentication mechanisms and credentials for users of the corporate and ICS networks.
- The ICS should also use a network topology that has multiple layers, with the most critical communications occurring in the most secure and reliable layer.

Given below is a sample deployment topology.



In no case should your ICS network and devices be directly accessible from the public internet. If there is some part of your ICS that you want to be accessible, (for example, if you want be able to view web-enabled HMI screens from a browser or smart phone), your ICS software should include features that securely pass the necessary traffic between your ICS network and a public-facing server.

## Managing Network Services and Ports

A network port is an endpoint of communication in an operating system. While the term is also used for hardware devices, in software it is a logical construct that identifies a specific process or a type of service. In other words, a network port is conceptually different from hardware ports like USB, memory card, and even the wired network connection.

Computers and devices can access many different network services at the same time by communicating on different network ports. Each network service or communication protocol has an associated port number. Some port numbers are specified by international standards, and therefore they are universally recognized. Other port numbers are claimed by proprietary software, and in most cases they can be changed in the software settings if there is a conflict with other software or services.

Firewalls control network traffic by either accepting or refusing communication on these network ports. If a port is open, it accepts communication, and if a port is closed, it refuses communication. Almost every layer of a network -- from the operating system on an individual computer or device, to the router that manages traffic within a network, to the gateway that manages traffic between networks -- has its own firewall.

The documentation for your ICS software should include a list of network ports that are commonly used by the software. Given the nature of ICS, the list typically includes services like web, email, file transfer, external databases, device drivers, and the ICS software itself for server-client communications. Configure the firewalls to open only those network ports that are actually used by your ICS. Disable all unused services and close all unused ports.

## Securing Communication between the Client and Server

Like most server-client applications, your ICS software should support secure communication between the server and client in order to prevent the messages sent between those two stations from being read by any other stations on the same network. Note that this is different from securing the network itself in order to prevent unauthorized access to the network.

This sort of communication is also sometimes known as "Encrypted Channel" because it uses the Transport Layer Security (TLS) standard to encrypt the server-client messages. The latest version of the standard is TLS 1.3 (released August 2018), but it is not yet in common use. The latest version of the standard in common use is TLS 1.2 (released August 2008). TLS supersedes the earlier Secure Sockets Layer (SSL) standard, although SSL is still used in older applications.

### Certificates

TLS and SSL use a system of certificates and keys to digitally "sign" the messages sent between the server and client. When the server establishes communication with the client (and vice versa), it presents its certificate which identifies its name, network address, organization, physical location, and so on. The client can then choose to either accept or refuse the certificate as presented. If it accepts the certificate, it agrees to accept messages encrypted with the same certificate, and it uses the associated key to decrypt those messages.

When you configure this sort of communication, you need to choose one of the following:

- Using self-signed certificate
- Using certificates signed by a Public Certificate Authority (CA)
- Using Domain-issued certificates or certificates signed by a Private Certificate Authority using systems like Microsoft Active Directory Certificate Service (AD CS)

A self-signed certificate is issued and signed by the same application that presents it. Self-signed certificates are easy to create and manage, but they are secure only if you control both the server and the client and therefore control which certificates are installed on each.

In contrast, CA-signed certificates are slightly difficult and expensive to acquire, but they are more flexible than self-signed certificates because you do not need to control both the server and the client. If you configure the server to present a CA-signed certificate, the client will accept the certificate because it recognizes the Certificate Authority.

Domain-issued certificates are internal certificates typically managed by your IT department. They are issued and validated by an Active Directory Certificate Authority. Domain-issued certificates are free and can be issued instantly.

You need to renew CA-signed and Domain-issued certificates at regular intervals.

For more information about how to enable Encrypted Channel features and manage self-signed certificates in your ICS software, see the documentation for that software. However, acquiring a CA-signed certificate and then using it to sign other certificates is typically beyond the scope of ICS software documentation.

---

**Note:** Encrypted and unencrypted communications typically use different network ports.

---

## Securing Systems through Authentication and Authorization

Typically, ICS software is comprised of a large number of systems, each accessed by a variety of users including engineers, operators and managers. The level of access that each type of user requires is different. So, it is necessary to manage user authentication and authorization to secure the system.

### Authentication

Authentication is the process of verifying a user's/system's identity. Authentication can be managed in the following ways:

- Within the ICS software through application accounts
- Through Windows accounts, which can be local to a single computer
- Through Authentication systems (see the next section for details)

While ICS software allows for user and role management, it can become cumbersome and complicated to manage a large number of user accounts as employees and roles change. Because of this, use of Windows accounts is generally preferred.

### Authentication Systems

Authentication systems such as Active Directory and Lightweight Directory Access Protocol (LDAP), referred to as authentication servers, are a repository of and provide centralized management for all system accounts and individual user accounts. An authentication protocol is used for all communication between authentication servers and the user or server requesting authentication.

Even though use of authentication systems provides improved scalability, the following factors must be considered depending upon the size and complexity of your operations:

- It is important that the authentication servers are highly secured.
- The authentication server system creates a single system for managing all system accounts. Therefore, it requires to be available at all times. To ensure minimal disruption during an emergency, redundancy must be considered.
- Permit caching of user credentials only for users who have authenticated their identity recently.
- Networks that support the authentication protocol must be reliable and secure to assist in trouble-free authentication.

It may also be worthwhile implementing two-factor authentication using additional applications such as PingID.

### Authorization

Authorization is the process of providing the correct level of privileges to users by applying access rules to authenticated users, systems (HMI's, field devices and SCADA servers) and networks (remote sites' LANs).

## Managing Users and Groups through ICS Software

Your ICS software should have a built-in security system that controls who may use the software and what privileges they have.

Users should be assigned permissions that determine what each user is authorized to do within the ICS system. Permissions can be managed either on a per-account basis or on a group basis by making use of roles. Group or role-based access control is preferred as it greatly simplifies management. Users can be moved from one role to another as the organization's needs change, and can also be members of multiple roles if required.

Each user should have their own user account with a unique user name and a strong password. The user account can then be assigned to one or more groups.

Accounts should always be assigned the least privileges necessary to perform their functions. Accounts with Windows Administrator permissions should be reduced to the minimum, and typically only used to install and configure the software. Likewise, accounts with SQL Server SysAdmin privileges should be reduced to the minimum, and typically only used to install and configure the software.

In most cases, the ICS software will allow associating Windows Groups with roles within the product. While defining and assigning roles, consider the following:

- Roles should be defined to have the least privileges necessary for their functionality.
- Roles should be limited to a single purpose in order to simplify the permissions assigned to them.
- Users can be members of multiple roles if necessary.

## Managing Users and Groups through Windows

When you configure security, you may choose to do one of the following:

- Keep the configuration local to a single application.
- Share the configuration between multiple applications.
- Manage the configuration as part of the network domain (for example, using Active Directory). This option typically allows users to have the same user account for the network, the host, and the ICS software. Using Active Directory gives you the following advantages:
  - A centralized repository for user and group data, enabling effective implementation of security policies and procedures.
  - Provides a single point of access to all network resources after the user is identified and authenticated.

To manage users and groups:

- First define a specific role for each group, and then configure the group privileges to fit that role.
- Groups may overlap, but it is often better to have clearly separate groups and then assign individual users to multiple groups, if necessary.
- Set or change the password for the ICS software's default user (e.g., "guest").
- Define stringent password policies to force users to create strong passwords. Enforce mandatory password updates on a regular basis.

## Contingency Planning

Incidents are inevitable. It is, therefore, important to develop a strategy to detect an incident quickly and respond to it in a timely manner in order to minimize loss and protect your system. An organization must consider contingencies arising from incidents such as fire, flood and so on, and those arising from failure of hardware or software components. Cyber attacks such as ransomware are becoming more common and must also be considered.

An organization should have contingency plans in place to cover the entire range of failures and eventualities. Employees should be trained and be familiar with the contents of the contingency plans.

As part of planning for contingencies, it is important to establish a site, physically separated from the central one, that has replication capability. Doing so will ensure the integrity of an operational system where the central site is at risk from fire, floods or other disasters. The replication capability includes having duplicated hardware, and requires software configuration and key state information to be periodically propagated from the central site to the recovery site. Each recovery scenario is unique, so it is important to consult with system integration experts regarding the design of communications equipment, hardware and the configuration of the software.

Protecting the data stored in your system is also of paramount importance. Full and incremental backups must be scheduled on a regular basis. Backups should be verified by running tests to restore from backed up data. Backups should be stored offline so that they are safe from cyber attacks such as ransomware.

Organizations should also have business continuity and disaster recovery plans that are similar to contingency plans. These plans are covered briefly in the sections to follow.

## Business Continuity Planning

Business continuity planning addresses strategies to maintain or re-establish production in the event of any disruption. These disruptions may be caused by a natural disaster (flood, earthquake, etc), by an intentional or unintentional man-made event (arson, operator error, power outage, etc), or by system failure.

Depending upon the duration of the potential ICS application outage caused by a disruption, operational recovery plans for short-term outages and disaster recovery plans for long-term outages must be formulated. It is also important to employ physical security for areas of a production site that house data acquisition and control systems that might have higher-level risks. Your business continuity plan should specify system and data recovery procedures for your systems. Once the recovery procedures are documented, a schedule should be developed to test the recovery procedures. Particular attention must be paid to the verification of backups of system configuration data and product or production data. The procedures should be reviewed periodically.

If you are accessing cloud-based solutions, ensure that systems are available at all times. In case of a disaster, services should switch to a new physical location to provide continued service.

## Disaster Recovery Planning

A disaster recovery plan (DRP) is a set of procedures to protect and recover an IT infrastructure in case of a disaster. It contains the procedures to follow before, during and after a disaster. Disasters can be natural, environmental or man-made (intentional or unintentional).

A DRP is essential for continued availability of the ICS, and should cover the following:

- When the DRP should be activated depending upon an event, its duration and its severity.
- Detailed course of action for operating the ICS manually until external connections are secured.
- Personnel responsible for each procedure.
- Processes for securely backing up data and restoring it. This should cover:
  - Requirements for building redundancy.
  - File backup procedures.
  - Frequency of backups.
  - Storage mechanism for full and incremental backups.
  - Safe storage of installation media, license keys and configuration information.
  - List of individuals responsible for performing, testing, maintaining and restoring backups.
- List of personnel with physical and virtual access to the ICS.

- Detailed configuration information about the components of the ICS.
- Schedule for testing the DRP.

## Security Configuration for InTouch HMI

The table below lists the security areas that you need to configure for InTouch HMI and the details of the section(s) in this guide that provide the corresponding instructions.

Security Area(s)	Topic(s) in this guide	Summary
Protecting the Applications and Content on the Host	Configuring an Inactivity Time-Out	Configure WindowViewer to automatically log off an inactive operator from an InTouch application.
Securing Systems through Authentication and Authorization	Authentication and Authorization Based Security	Users need to authenticate themselves before using an InTouch application. InTouch verifies whether the authenticated user is authorized to use specific functionality.
	Using Virtual Accounts	Using virtual accounts provides an additional layer of security when accessing alarm functions. See Using Virtual Accounts in the InTouch HMI Alarms and Events Guide.
Protecting the Applications and Content on the Host	Locking System Keys	Restrict operator access to standard Windows functions by disabling system keys on the computer running an InTouch application.
Securing Systems through Authentication and Authorization		
Managing Users and Groups through ICS Software	Using InTouch-Based Security	Restrict which functions an operator is allowed to perform by linking those functions to internal tags.
Managing Users and Groups through Windows	Using Operating System-Based Security	Inherit some user/group account policies from the Windows operating system.



## Security Configuration for AVEVA Application Server

The following table below lists the security areas that you need to configure for Application Server and the details of the section(s) in this guide that provide the corresponding instructions.

Security Area(s)	Topic(s) in this guide	Summary
Basic information about using and configuring the Application Server IDE.	<i>IDE Overview</i> on page 19	Description of the Application Server Galaxy and the objects that comprise it. Also describes how to create, connect to, and log into a Galaxy.
Configuring user permissions, credentials, and galaxy security.	<i>Configuring Security</i> on page 481	How to configure ArcestrA security to manage access to configuration and run-time aspects of your Application Server application.
Protecting objects and attributes at configuration time and setting object security.	<i>About the General Editor Layout</i> on page 70	How to use the Object Editor restrict the write access that application builders have while configuring objects through locking/unlocking template attributes, setting object security, and group locking.
Configuring alarm and event notifications, and setting user access and security at run time.	<i>Working with Alarms and Events</i> on page 359	How to configure alarm and event notifications and enable security for system operators, such as which individuals are allowed to acknowledge alarms.
Deploying objects and Galaxies to run time.	<i>Deploying and Running an Application</i> on page 317	How to deploy an entire Galaxy or individual application objects to run time, and how to view operational and security status at run-time.
Galaxy management and creating a new ArcestrA User account.	<i>Managing Galaxies</i> on page 529	Backing and restoring Galaxies, configuring multiple network cards, managing the ArcestrA User Account, and synchronizing Galaxy objects.
Configuring large Galaxies that span more than a single GR node.	<i>Working with Multiple GR Nodes and Galaxies</i> on page 403	How to pair Galaxy Repositories to configure multi-galaxy communication. Includes troubleshooting information.

Security Area(s)	Topic(s) in this guide	Summary
Scripting methods for security.	See 02Build_ScriptNet_Net Script InTouch Functions in the <i>Application Server Scripting Guide</i> .	<p>The following scripting methods can be used in run-time applications to enhance security:</p> <ul style="list-style-type: none"> <li>• AddPermission() Function</li> <li>• AttemptInvisibleLogon() Function</li> <li>• ChangePassword() Function</li> <li>• GetAccountStatus() Function</li> <li>• InvisibleVerifyCredentials() Function</li> <li>• IsAssignedRole() Function</li> <li>• LogonCurrentUser() Function</li> <li>• PostLogonDialog() Function</li> <li>• QueryGroupMembership() Function</li> <li>• SignedAlarmAck()</li> <li>• SignedWrite()</li> </ul>

# APPENDIX B

## Configuring a Multi-User Development Environment

### Best Practice Recommendations

The ability to have multiple users working concurrently on a single galaxy is often required, particularly for large and/or projects that often require multiple developers to work together. If multiple developers will be working with the IDE to build objects and applications, you can structure your environment to enhance team coordination and avoid disconnects and conflicts. Coordinating this effort can be difficult. Even before starting, there are IT infrastructure requirements and limitations that must be considered.

A multi-user environment can be set up with multiple concurrent IDE connections from either standalone computers within the galaxy or RDS client sessions. Advantages of a multi-user development system include:

- Multiple engineers can work simultaneously.
- Changes can always be rolled back.
- The Production Galaxy is isolated from the development system, and development work done on remote workstations does not interfere with or disrupt work being done on the Production Galaxy.
- Individual updates from remote workstations can be released to the Production Galaxy, without requiring check-in/check-out or long wait times.
- The IDE remains responsive, with no interruptions from security updates.

This section describes different ways to configure your development system, and also describes the recommended architecture that will support the most concurrent users. If you select one of the alternative architectures, keep in mind some basic considerations:

- SQL Server Express vs SQL Server: a full version of SQL Server is tolerant of more concurrent users than SQL Server Express (up to three for SQL Express vs. up to five for a full SQL Server version).
- Multi-galaxy pairing must be enabled whenever the development galaxy spans more than one GR.
- In a shared-GR development environment, the IDE may disconnect periodically from the galaxy as the number of users increases. You can avoid this if you use the recommended multi-user development architecture.
- Running InTouch OMI Preview may occasionally crash the the ASB services in shared GR development environments. Again, you can avoid this if you use the recommended multi-user development architecture.

### Development System Architecture

This section describes three basic system architectures for multi-user development, each with its own set of advantages and disadvantages. The recommended multi-user environment leverages AVEVA™ Integration Studio, a cloud-based development environment that provides high allows a virtually unlimited number of concurrent users, without negatively impacting performance.

In addition to the recommended architecture, two alternative basic system architectures for multi-user development are described in the following sections. Each has its own set of advantages and disadvantages, including the recommended architecture. The basic system architectures are:

- *Shared GR with Local IDE Nodes (Alternative Architecture)* on page 575
- *Shared GR and IDE Node (Alternative Architecture)* on page 576
- *Local GR Nodes plus Shared GR (Recommended)* on page 572

### **Object Wizards: Best Practices**

Regardless of the architecture that you implement, it is recommended that you have a set of hierarchical object wizards to use as the basis for your galaxy, and ensure that best practices for object wizard usage are observed. The most important of these are to develop a three-tier hierarchy of object wizards:

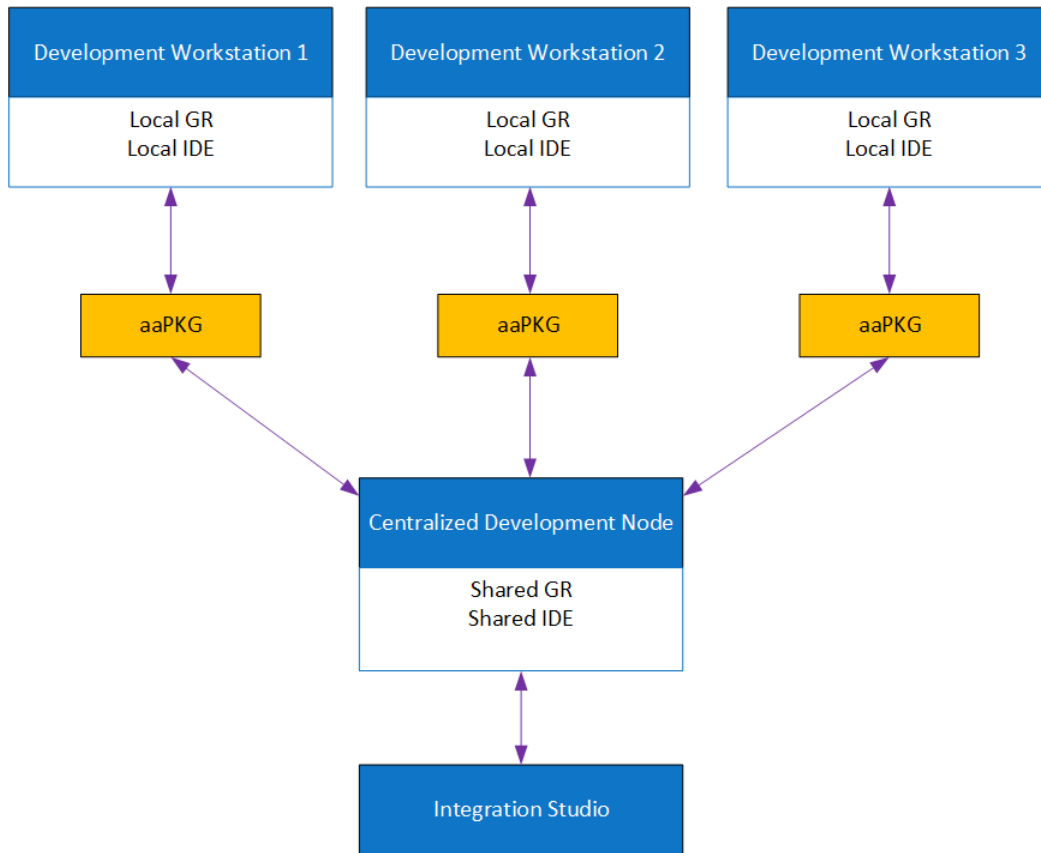
- Top-level object wizards define corporate standards, and serve as templates for derived object wizards.
- Second-level derived object wizards contain refinements that take into account, for example, regional differences or the requirements for different business units.
- Third-level object wizards contain additional refinements, as needed for a particular site or facility.

## **Local GR Nodes plus Shared GR (Recommended)**

Local GR nodes for each developer, combined with a shared, centralized GR node offers high performance and eliminates many of the challenges that can affect other development architectures. Most development work can be done on the local nodes, and then uploaded, via an aaPKG file, to the shared GR node for integration and/or testing. The shared GR node can be an on-premises system, or you can use Integration Studio, a cloud service hosted by AVEVA. With a local GR for each developer, most of the challenges that apply to other multi-user development environments are avoided.

[Click here for more information about AVEVA Integration Studio.](#)

Many concurrent users are supported (no upper limit has been identified).



This approach has the following advantages and disadvantages:

**Advantages:**

- High performance: with most development work done on local nodes, there are no slow downs associated with shared access and network traffic.
- Remote access: developers can work from different sites and geographic locations.
- IT infrastructure: with a shared GR and IDE, there are fewer potential bottlenecks between the two Application Server components.
- Multiple projects and versions: the centralized IDE allows better project/version management.

**Disadvantages:**

- Transferring work between the remote GR nodes and the shared GR node requires manually exporting and importing aaPKG files.

## Create a Multi-User Development System

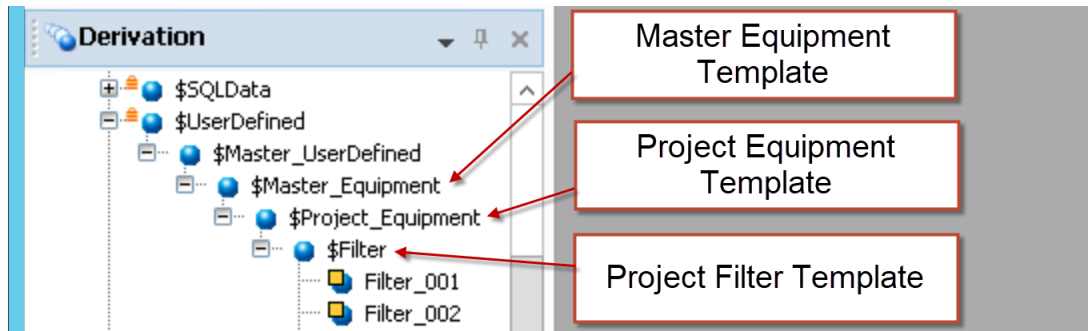
### Distributed Multi-User Development Solution

The following practices are highly recommended:

1. On the GR node that contains your production (functional) Galaxy, add a second Galaxy. We will call this second Galaxy the **Development Galaxy**. The Development Galaxy will act as a central storage location, and will be used to:
  - Maintain the current master templates, including top-level Object Wizards for all objects.

- Distribute master templates and Object Wizards to each Local GR for development (via .aaPkg export).
- Control master template updates from local developers (via .aaPkg import).
- Maintain ownership relations between developers and master templates via OS User Based or OS Group Based security.

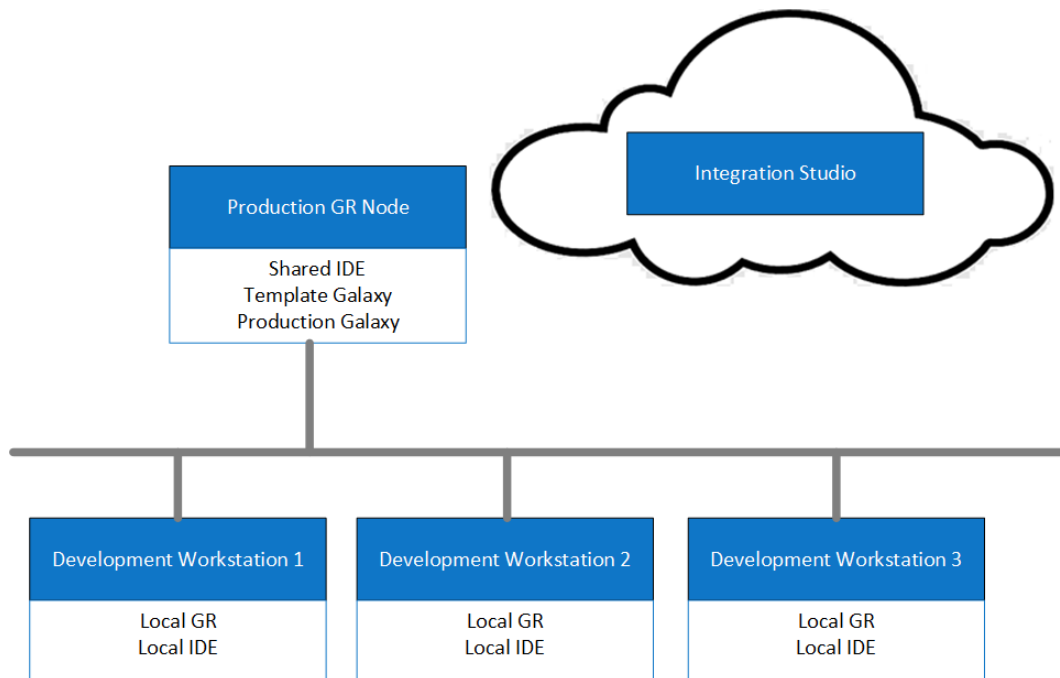
The figure below shows an example of template hierarchy levels. Set access and ownership within the team/organization to each template level to maintain control and security of the project.



2. Install a temporary, local GR at each IDE development node. The local GR allows each developer to work independently, without interfering with or interference from other developers that may be checking in/out objects, security updates, and deployment actions.

The local GR can be used for both development and testing.

**Note:** The local GR should not require an additional platform count license because it is temporary, and it is not used for run time.



## Development Process

1. Log in to the Template Galaxy.
2. Select and check out template(s) you need for development.
3. Export the selected templates as an aaPKG file from the Template Galaxy.
4. Log in your local GR.
5. Import the aaPKG file to the local GR.
6. Edit, deploy, and test the new galaxy you develop from the imported aaPKG objects.
7. Export the new galaxy as aaPKG file.
8. Log in to the Template Galaxy and undo the checkout on the templates you exported (see step 2).
9. Import the aaPKG file from the local GR (see step 7) to the Template Galaxy.
10. To document your development efforts, check out and then check in (with comments) the changed objects .

## Production Galaxy Updates

Before updating the Production Galaxy:

- Back up the existing Galaxy before making any changes.
- Transfer templates to the Production Galaxy when the project is ready to be updated.
- Create new instances/assets in the Production Galaxy as needed.

## Benefits of Using a Distributed Multi-User Development System

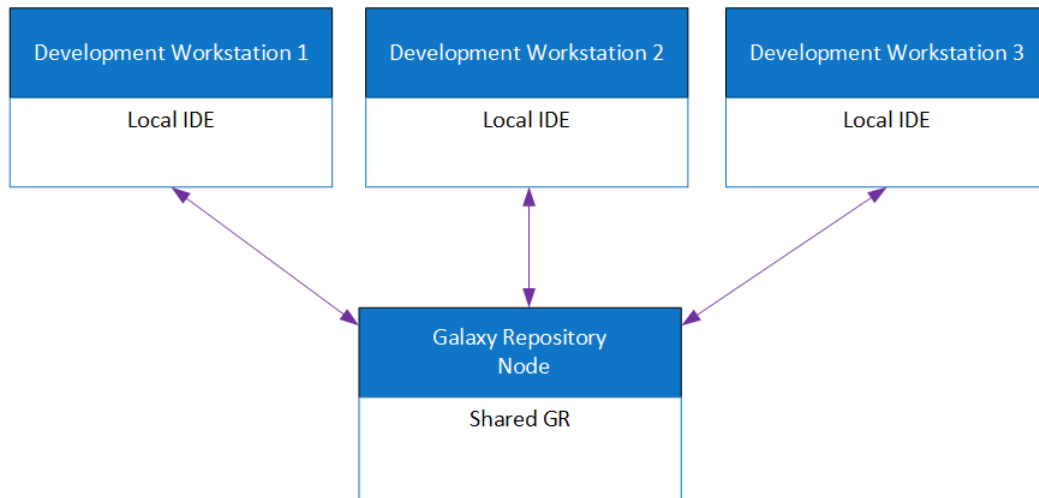
- Several engineers can work simultaneously.
- Unwanted changes can be easily rolled back.
- The Production Galaxy is isolated. This eliminates disturbances and disruptions due to development/updates.
- Updates and new releases to the Production Galaxy can be managed and controlled.
- Eliminates check-in/check-out, long waits, and time-outs.
- Eliminates non-responsive IDE and aaGR service hanging.
- Eliminates IDE connection interruptions due to security updates.

## Shared GR with Local IDE Nodes (Alternative Architecture)

As an alternative to using the recommended multi-user development architecture, you can utilize a centralized GR in combination with local IDE nodes for each developer. Note that this approach has a number of disadvantages, which are listed below.

The GR is shared among users, and each user works on objects within the repository, thus ensuring that all developers can see the work that others have done. An object that a user is currently working on is checked out, and unavailable to other users, thus preventing one user overwriting another user's concurrent work.

Using a centralized GR supports up to five concurrent users.



This approach has the following advantages and disadvantages.

**Advantages:**

- Single repository: All work is stored in, and accessible from, a centralized repository.

**Disadvantages:**

- Congestion: since all users are accessing the same GR, system response can slow down. A limit of five concurrent users is recommended.
- Team coordination: with multiple users working within the same GR, planning and communication between users is essential.
- Project coordination: managing multiple projects, as is the case with multiple users, requires planning and coordination.
- Versioning: an effective way to maintain versioning of a project is difficult to implement.
- IT infrastructure: adequate bandwidth and resource provisioning must be in place to enable adequate system response time.

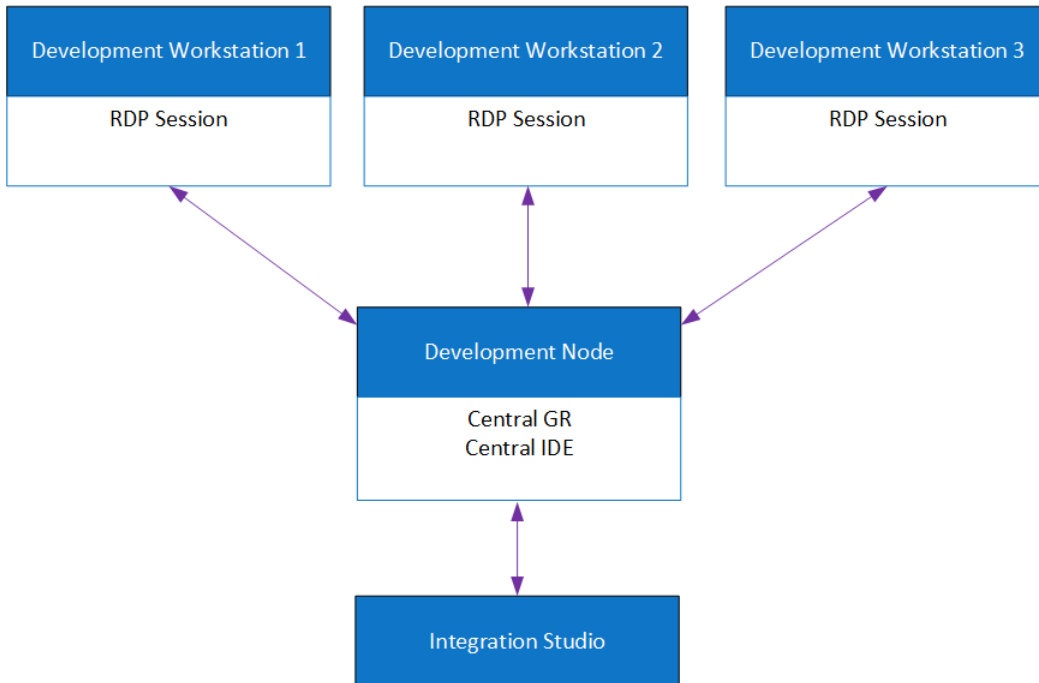
## Shared GR and IDE Node (Alternative Architecture)

As a second alternative to using the recommended multi-user development architecture, you can utilize a centralized GR in combination with centralized IDE. Developers then use RDS to access these resources.

This arrangement, with both the GR and IDE shared by developers, allows better coordination between team members than having a local IDE for each developer, although like the other alternative architecture described above, it also has its share of disadvantages including cost and congestion, as listed below.



Sharing both the GR and IDE requires more resources than simply sharing the GR however. The maximum number of concurrent users will vary between two and five, depending on computer hardware resources, whether or not you are using SQL Server Express (which will support fewer users) or a full version of SQL Server, and the size of the galaxy.



This approach has the following advantages and disadvantages:

**Advantages:**

- Single repository: All work is stored in, and accessible from, a centralized repository.
- IT infrastructure: with a shared GR and IDE, there are fewer potential bottlenecks between the two Application Server components.
- Multiple projects and versions: the centralized IDE allows better project/version management.

**Disadvantages:**

- Congestion: since all users are accessing the same GR and IDE, system response can slow down. A limit of five concurrent users is recommended.
- Cost: RDP CALs (client access licenses)



# APPENDIX C

## Glossary

**application**

A collection of objects in a Galaxy Repository that performs automation tasks. Synonymous with Galaxy. There can be one or more applications within a Galaxy Repository.

**Application Engine (AppEngine)**

A scan-based engine that hosts and executes the run-time logic contained within AutomationObjects.

**ApplicationObject**

An AutomationObject that represents some element of your production environment. This can include things like

- An automation process component. For example, a thermocouple, pump, motor, valve, reactor, or tank
- An associated application component. For example, function block, PID loop, Sequential Function Chart, Ladder Logic program, batch phase, or SPC data sheet

**Application Server**

The supervisory control platform. Application Server uses System Platform products such as InTouch HMI or AVEVA OMI for visualization, the Historian for data storage, and AVEVA Communications Drivers for device communications.

An Application Server can be distributed across multiple computers as part of a single Galaxy namespace.

**Application view**

The Applications view shows the object-related contents of the Galaxy in four different ways: Model view, Deployment view, Derivation view, and Operations view. The Model view appears when the IDE is opened for the first time.

**ArchestrA**

The distributed architecture for supervisory control and manufacturing information systems. It is an open and extensible technology based on a distributed, object-based design.

**ArchestrA Object Toolkit**

A programmer's tool to create new ApplicationObjects and Device Integration Object (DIOjects) templates, including configuration and run-time implementations. Includes a tool to build DI Objects and create unique Domain Objects that interact with DIOjects in the ArchestrA environment.

**area**

A logical grouping of AutomationObjects that represents an area or unit of a plant. It is used to group related objects for alarm, history, and security purposes. It is represented by an area AutomationObject.

**area object**

The system object that represents an area of your plant within a Galaxy. The Area Object acts as an alarm concentrator, and places other Automation Objects into proper context with respect to the actual physical automation layout.

**assignment**

The designation of a host for an AutomationObject. For example, an AppEngine object is assigned to a WinPlatform object.

**attribute**

An externally accessible data item of an AutomationObject.

**attribute reference string**

A text string that references an attribute of an AutomationObject.

**AutomationObject**

An object type that represents permanent things in your plant, such as ApplicationObject or Device Integration Object (DIObjects), with user-defined, unique names within the Galaxy. It provides a standard way to create, name, download, execute, and monitor the represented component.

**Backup Application Engine**

The object created by the ArchestrA infrastructure when the Primary object is enabled for redundancy. See redundancy for further details.

**base template**

A root template at the top of a derived hierarchy. Unlike other templates, a base template is not derived from another template but developed with the ApplicationObject Toolkit and imported into a Galaxy. All base templates names start with a dollar sign (\$).

**Block Read Group**

A DAGroup that is triggered by the user or another object. It reads a block of data from the external data source and indicates the completion status.

**Block Write Group**

A DAGroup that is triggered by the user or another object after all the required data items are set. The block of data is sent to the external data device. When the block write is complete, it indicates the completion status.

**Bootstrap**

The base ArchestrA service which is required on all ArchestrA computers. It provides the base software environment to enable a platform and allows a computer to be included in the Galaxy Namespace.

**Buffered Data**

Data captured and stored locally on a remote device for later transfer to a supervisory system for processing, analysis, and long-term storage. Can be configured within the I/O feature in the Attributes page.

**change log**

The revision history that tracks the life cycle activities of ArchestrA Objects, such as object creation, check in/check out, deployment, and import/export.

**change propagation**

The ability to create templates which allows each component template to support changes such that a change in one of the elements can be automatically propagated to all — or select, related — object instances.

**check in**

IDE operation for making a configured object available for other users to check out and use.

**check out**

IDE operation for the purpose of editing an object. It makes the item unavailable for other users to check out.

**Checkpoint**

The act of saving to disk the configuration, state, and all associated data necessary to support automatic restart of a running AutomationObject. The restarted object has the same configuration, state, and associated data as the last checkpoint image on disk.

**compound object**

An ApplicationObject that contains at least one other ApplicationObject.

**contained name**

An alternate naming convention that when combined with the `TagName` of the root container object, results in the hierarchical name. For example, for a given object, its Hierarchical Name = `Line1.Tank1.InletValve` and its Contained Name = `InletValve`.

**containment**

A hierarchical grouping that allows one or more `AutomationObject` to exist within the name space of a parent object and be treated like parts of the parent object. Allows for relative referencing to be defined at the template and instance level.

**DAGroup**

A data access group associated with Device Integration Object (DIObjects). It defines how communications are achieved with external data sources. It can be a Scan Group, Block Read Group or Block Write Group.

**DAServer Manager (DAS Manager)**

The System Management Console (SMC) snap-in supplied by the Operations Integration Server (OI Server) that provides the required interface for activation, configuration, and diagnosis of the OI Server.

**Data Access Server (DAServer)**

See Operations Integration Server (OI Server).

**Data Access Server Toolkit (DAS Toolkit)**

A developer tool that can build an Operations Integration Server (OI Server).

**Deployment**

The operation which instantiates an `AutomationObject` instance in the ArchestrA run time. This action involves installing all the necessary software and instantiating the object on the target platform with the object's default attribute data from Galaxy Repository.

**Deployment view**

The part of the Application view in the IDE that shows how objects are physically dispersed across Platforms, areas and Engines. This is a view of how the application is spread across computing resources.

**derivation**

The creation of a new template based on an existing Template.

**Derivation view**

The part of the Application view in the IDE that shows the parent-child relationship between base templates, derived templates and derived instances. A view into the genealogy of the application.

**derived template**

Any template with a parent template. Derived templates inherit the attributes of the parent template. You can change these attributes in the derived template.

**Device Integration Object (DIObjects)**

An `AutomationObject` that represents the communication with external devices or software. DIObjects run on an Application Engine (AppEngine), and include DINetwork Objects and DIDevice Objects.

**DHCP**

Dynamic Host Configuration Protocol. A network configuration protocol for hosts on IP networks.

**DIDevice Object**

An object that represents the actual external device (for example, a PLC or RTU) that is associated with a DINetwork Object. It can diagnose and browse data registers of the DAGroups for that device.

**DINetwork Object**

An object that represents the network interface port to the device through the Data Access Server (DAServer) or the object that represents the communications path to another software application. It provides diagnostics and configuration for that specific network card.

**element**

Basic shapes, such as rectangles, lines, and text elements, and controls you can use to create an Industrial Graphic to your specifications.

**Element Style**

An Element Style defines one or more visual fill, line, text, blink, outline, and status properties of Industrial Graphic elements. You can apply an Element Style to set preconfigured values to the visual properties of a graphic element.

**Engine Object**

An ArchedrA system-enabled object that contains Local Message Exchange and provides a host for ApplicationObjects, Device Integration Object (DIObjects) and area objects.

**event record**

The data that is transferred about the system and logged when a defined event changes state. For example, an analog crosses its high level limit, an acknowledgement is made, or an operator logs in to the system.

**export**

The act of generating a package file (.aaPKG) extension from persisted data in the Galaxy database. You can import the resulting .aaPKG file into another Galaxy.

**FactorySuite Gateway**

A Microsoft Windows application program that acts as a communications protocol converter. Built with the ArchedrA DAS Toolkit, FS Gateway links clients and data sources that communicate using different data access protocols.

**Galaxy**

The entire application. The complete ArchedrA system consisting of a single logical name space (defined by the Galaxy database) and a collection of platform objects, Engine Objects and other objects. One or more networked computers that constitute an automation system. This is referred to as the Galaxy Namespace.

**Galaxy database**

The relational database containing all persistent configuration information like templates, instances, and security in a Galaxy Repository.

**Galaxy Database Manager**

A utility to manage your Galaxy. It can back up and restore Galaxies if they become corrupt or to reproduce a Galaxy on another computer. The Galaxy Database Manager is part of the System Management Console (SMC).

**GalaxyObject**

The object that represents a Galaxy.

**Galaxy Repository**

The software sub-system consisting of one or more Galaxy databases.

**Graphic Toolbox**

The part of the IDE main window that shows a hierarchy of graphic toolsets, which contain Industrial Graphics, Situational Awareness Library symbols, apps, and client controls.

**hierarchical name**

The name of the object in the context of its container object. For example, Tank1.OutletValve, where an object called Tank1 contains the OutletValve object.

**Historical Storage System (Historian)**

The time series data storage system that compresses and stores high volumes of time series data for later retrieval.

**host**

The parent of a child instance in the deployment view. Example: a Platform instance is a Host for an Application Engine (AppEngine) instance.

**import**

The act of reading a package file (.aaPKG) and using it to create AutomationObject instances and templates in the Galaxy Repository.

**Industrial Graphic**

Formerly called an ArchestrA symbol or graphic, an Industrial Graphic is a symbol you create and use to visualize data in an HMI system. You use the Industrial Graphic Editor to create Industrial Graphics from basic elements, such as rectangles, lines, and text elements.

**instance**

An object derived from a template. You deploy instances to the run-time environment.

**instantiation**

The creation of a new instance based on a corresponding template.

**Integrated Development Environment (IDE)**

The Integrated Development Environment (IDE) is the interface for the configuration side of Application Server. In the IDE, you manage templates, create instances, deploy and un-deploy objects, and other functions associated with the development and maintenance of the system.

**InTouch View**

InTouch View clients are InTouch run-time clients that solely use of the Application Server for its data source. In addition, standard InTouch run times can also leverage Application Server.

**InTouchViewApp object**

Represents an InTouch application in the Application Server environment. The InTouchViewApp object manages the check-in, check-out, and deployment of an InTouch application.

**I/O count**

Number of I/O points being accessed within a Galaxy. I/O points are real I/O and are not equivalent to InTouch tags. I/O count is based on the number of I/O points that are configured through an OPC Server, I/O Server, Data Access Server (DAServer) or InTouch Proxy Object, over the whole Application Server namespace, regardless of how many computers are in the system.

**Message Exchange**

The object to object communications protocol used by Application Server. Message Exchange includes

LMX	communication between objects
NMX	communication between Galaxy nodes.

**Model view**

The area in the Application view in the IDE that shows how objects are arranged to describe the physical layout of the plant and supervisory process being controlled.

**object**

Any template or instance in a Galaxy database. A common characteristic of all objects is they are stored as separate components in the Galaxy Repository.

**Object Viewer**

A utility in which you can view the attribute values of the selected object in run time. This utility is only available when an object is deployed. Object Viewer shows you diagnostic information on ApplicationObjects so you can see performance parameters, resource consumption and reliability measurements. In addition to viewing an object's data value, data quality and the communication status of the object, you can also modify some of its attributes for diagnostic testing. Modifications can include adjusting timing parameters and setting objects in an execution or idle mode.

**OffScan**

The state of an object that indicates it is idle and not ready to execute its normal run-time processing.

**Operations Integration Server (OI Server)**

The server executable that handles all communications between field devices of a certain type and client applications. Similar to I/O Servers but with more advanced capabilities. Also called a DAServer.

**OnScan**

The state of an object in which it is performing its normal run-time processing based on a configured schedule.

**Operations view**

The area in the IDE that shows the results of validating the configuration of objects.

**package definition file (.aaPDF)**

The standard description file that contains the configuration data and implementation code for a base template. File extension is .aaPDF.

**package file (.aaPKG)**

The standard description file that contains the configuration data and implementation code for one or more objects or templates. File extension is .aaPKG.

**Platform Count**

Number of computers in the Galaxy. Each Workstation or Server communicating directly with the Application Server requires a platform to be part of the Galaxy Namespace. This includes each InTouch and InTouch View client.

**Platform Manager**

This utility is an extension snap-in to the ArchestrA System Management Console (SMC). Provides Galaxy application diagnostics by allowing you to view the run-time status of some system objects and to perform actions upon those objects. Actions include setting platforms and engines in an executable or idle mode and starting and stopping platforms and engines.

**platform object**

An object that represents a single computer in a Galaxy, consisting of a system wide message exchange component and a set of basic services. This object hosts all Application Engines.

**Primary Application Engine**

The object created by the ArchestrA infrastructure when the Backup object is created through redundancy. See redundancy for further details.

**properties**

Data common to all attributes of objects, such as name, value, quality, and data type.

**proxy object**

An AutomationObject that represents an actual product for the purpose of device integration with the Application Server or InTouch HMI. For example, a Proxy object enables the Application Server to access an OPC server.

**redundancy**

Two computers: One executes objects. The other is a stand by.

**RedundantDIOobject**

Monitors and controls the redundant Device Integration Object (DIOobjects) data sources. Unlike redundant AppEngines, individual DIOobject data sources do not have redundancy-related states. They function as stand-alone objects.

**Redundant Message Channel**

A dedicated Ethernet connection which is required between the platforms hosting redundant engines. The RMC is vital to keep both engines synchronized with alarms, history, and checkpoint items from the engine that is in the Active Role. Each engine also uses this Message Channel to provide its health and status information to the other.

**reference**

A string that refers to an object or to data within one of its attributes.



**relative reference**

Objects may refer to themselves, containers, hosts or to child objects elsewhere in the parent/child hierarchy using special reserved keywords such as "Me" or "MyContainer". Relative references continue to work properly even if the object that is referenced is renamed. Examples of relative references are "Me", "MyArea", "MyContainer", "MyHost", and "MyPlatform".

**remote reference**

The ability to redirect ArchestrA object references or references to remote InTouch tags. The new script function that redirects remote references at run time is `IOSetRemoteReferences`.

**SCADA**

Supervisory Control and Data Acquisition (SCADA) is a centralized industrial control system that controls and monitors sites or systems which are spread over large areas - an entire industrial plant or a country.

**Scan Group**

A DAGroup that requires only the update interval be defined. The data is retrieved at the requested rate.

**Scan State**

The Scan State of an object in run time. This can be either OffScan or OnScan.

**security**

Application Server security is applied to IDE, System Management Console (SMC), and the run-time data level. At the run-time data level which centralizes the definition of all permissions to the ApplicationObjects. These ApplicationObjects can be accessed by a variety of clients but the security is centrally defined, allowing ease of maintenance. Users that are allowed to modify these ApplicationObjects at run time are mapped to the objects by user-defined roles. These roles can be mapped directly to existing groups in a Microsoft Domain or workgroup.

**System Management Console (SMC)**

The central run-time system administration/management product where you perform all required run-time administration functions.

**system object**

An object that represents an area, platform or engine.

**TagName**

The unique name given to an object. For example, for a given object, its TagName = V1101 and its HierarchicalName = Line1.Tank1.InletValve.

**template**

An object containing configuration information and software templates used to create a derived template or instance.

**Template Toolbox**

The part of the IDE main window that shows Toolsets containing templates. The Template Toolbox shows a tree view of template categories in the Galaxy.

**Toolset**

A named collection of templates shown together in the IDE Template Toolbox.

**ViewEngine object**

Hosts InTouchViewApp objects. The ViewEngine object supports common engine features such as deployment, undeployment, startup, and shutdown.

**WinPlatform object**

An object that represents a single computer in a Galaxy. A WinPlatform object consists of a system-wide message exchange component and a set of basic ArchestrA services. The WinPlatform object hosts the Application Engine (AppEngine).



# Index

▪

## .csv files

- characters in • 534
- editing • 532
- importing • 536
- structure • 533
- time formats • 535

## A

- aaAdministrators group • 494
- aaConfig SQL utility • 494
- aaGalaxyOwner user account • 494
- active object • 546
- active object, redundancy • 546
- adding custom help to objects • 81
- Advanced Communication Management
  - configuring scan modes • 326
  - description • 324
  - saving historical data • 348
  - setting for Galaxy • 326
- Alarm Comment Language Switching • 507
- Alarm Comments Language File • 521
- Alarm configuration • 392
- alarms
  - aggregating • 394
  - AlarmInhibit attribute • 366
  - AlarmMode attribute • 366
  - Area AutomationObjects • 382
  - definition • 359
  - disabled • 368
  - extensions • 139, 140, 141, 143, 146
  - grouping • 382
  - plant state • 387
  - setting categories • 381

- severity • 393
- statistical type • 365
- subscribing • 383

## aliases

- rules for locking in scripts • 164
- using in scripts • 162

## analog device objects, output extensions • 134

## AppEngine objects

- description • 47
- history configuration • 351
- redundancy • 547

## applications

- ArchestrA architecture, ArchestrA • 453
- templates • 47

## ArchestrA

- messaging system • 453

## ArchestrA user account • 494

## Area object, description • 47

## Attribute naming conventions • 123

## attributes

- adding to objects • 121
- and scripting • 125
- dynamic • 460
- hardware register • 460
- hidden • 123
- historizing • 344
- locking in child objects • 121
- locking in instances • 70
- locking in templates • 70
- locking locking attributes • 70
- property, finding • 88
- runtime • 460
- search • 74
- security • 73

- security locked in parent • 73
  - unlocking in instances • 70
  - unlocking in templates • 70
  - Attributes page • 126
  - auto-bound reference • 148
- B**
- backup object
    - description • 545
    - redundancy • 545
  - backup server, redundancy • 545
  - base templates
    - creating derived templates • 45
    - description • 44
    - importing • 105
    - modifying • 28
  - bit field access • 90
  - Boolean
    - alarm extensions • 139, 140, 141, 143, 146
    - data types • 134
  - Buffered Data and History • 447
- C**
- changing, users • 41
  - check-in comments, setting • 38
  - components in Galaxies • 19
  - configure user information • 38
  - Configuring
    - history • 344
    - objects to store history • 355
    - WinPlatforms and AppEngines for history • 351
  - configuring bit field access • 90
  - configuring computers for redundancy • 545
  - configuring redundancy
    - AppEngine • 549
    - ordering network connections • 548
  - connecting
    - existing Galaxy • 22
  - contained
    - names • 58, 64, 100
    - templates, creating new • 57
  - contained objects
    - naming conventions • 64, 67
    - viewing • 67
  - containment
    - definition • 58
    - names, scripting • 60
    - naming conventions • 64, 67
    - relationships, viewing • 30, 64
    - templates • 58
  - creating
    - contained templates • 58
    - Galaxy, security restrictions • 20
    - object extensions • 126
    - scripts • 162
  - creating user accounts • 538
  - customizing
    - derived templates • 57
    - information for a Galaxy • 38
    - object help • 81
- D**
- data
    - types and bit field access • 90
  - data types, configure history • 344
  - default, templates • 44, 45
  - deploy host objects first • 333
  - deploying
    - history • 348
    - imported objects • 119
    - instances • 43
    - templates • 43
  - deployment
    - redundancy • 556
  - deployment status
    - pair undeployed • 549
    - partial deployed • 549
    - partial undeployed • 549
    - redundancy objects • 556
  - Deployment view
    - assignment relationships • 32
  - Derivation view

- parental relationships • 33
- derivation, viewing objects • 33
- derived locked scripts • 164
- derived templates • 45, 48
  - contained names • 64
  - creating • 57
  - customizing • 57
  - nesting • 57
- Deviation Alarms Feature Parameters • 141
- device integration templates • 47
- DI Device objects, definition • 47
- DI Network objects
  - configuration limits • 32
  - definition • 47
  - deploying • 333
- DI Object data sources • 556
- disabled alarms • 369
- disabling
  - redundancy • 550
- disk space requirements • 537
- DNS settings, configuring • 539
- Domain local groups (DLGs) • 504
- dynamic attributes • 460

## E

- editing .csv files • 533
- editing .csv files, characters in • 534
- enabled alarms • 368, 369
- event
  - Configuration • 392
  - distributors • 382
- events
  - definition • 359
  - types • 359
- Events, turn on/off historization • 392
- execution order of objects, setting • 80
- expanding tabs • 37
- exporting
  - Galaxy dump file • 532
  - instances • 102, 532
  - object help • 532
  - objects • 102, 532

- script function libraries • 104
- script libraries • 532
  - scripts • 102
- exporting templates • 532
- extension inheritance • 126
- extensions, deleting • 126

## F

- failover and redundancy • 545
- field reference object • 134
- finding
  - object attributes • 88
- floating
  - areas • 37
  - views • 37
- fonts, languages • 510

## G

- Galaxies
  - adding a language • 508
  - changing • 530
  - changing the default language • 510
  - components of • 19
  - default users • 481
  - definition • 19
  - deploying components • 19
  - disk space requirements • 537
  - Galaxy Repository • 19
  - language settings • 507, 508
  - location on the network • 19
  - logging in to • 41
  - managing multiple • 530
  - naming conventions • 20
  - opening with security • 22
  - removing a language • 509
  - reserved names • 20
  - restoring • 529
  - specifying GR node name • 20
  - validating • 99
- Galaxy Browser • 88

Galaxy Dump File, editing • 533

Galaxy dump files

characters • 534

importing • 536

time formats • 535

group

lock icons • 74

## H

hardware register attributes • 460

HCAL

historizing data • 343

using • 343

heartbeats • 537

hidden attributes, Attributes

hidden attributes • 123

hiding

areas • 37

views • 37

hierarchical

names • 58, 64, 100

names, viewing • 30

Historization

alarm • 392

configuration • 392

Event • 392

Events, turn on/off • 392

historized data

store forward • 348

when data is stored • 348

historizing

attributes • 344

data, installing Wonderware Historian • 343

history

deploying • 348

undeploying • 348

History Feature Attributes • 136

history, data types, configuring • 344

HTML editors for customizing object help • 81

## I

I/O Advanced Property • 127

I/O Auto Assignment

Upload Runtime Changes • 157

I/O auto-binding • 148

icons, deployment status • 333

IDE objects in runtime • 318

IDE, views • 24

importing

.csv files • 536

base templates • 105

objects • 105

objects and deploying • 119

objects and security groups • 119

script function libraries • 109

individual alarms • 367

inheriting attributes, parent and child relationships • 43

Input extensions, data quality • 135

InputOutput

attributes in scripts • 132

extension • 130

InputOutput extensions, data quality • 135

installing Wonderware Historian • 343

instances

creating • 99

defined • 44

deploying • 43

exporting • 102, 532

locking attributes • 70

naming conventions • 99

on other computers • 32

reserved names • 100

unlocking attributes • 70

IP address, network settings • 539

## L

Language Features • 507

languages

changing default for a Galaxy • 510

- configuring for a Galaxy • 507, 508
  - configuring for Galaxy • 508
  - dictionary files • 511
  - exporting data for all symbols • 512
  - exporting data for specific objects • 512
  - exporting for managed InTouch application • 514
  - fonts • 510
  - removing from a Galaxy • 509
  - translating symbol text • 511
  - Limit Alarm Parameters • 139
  - limit alarms • 363
  - locked scripts in child objects • 164
  - locking
    - aliases in scripts • 164
    - attributes • 121
    - scripts • 164
    - template attributes • 70
  - logging
    - in to disconnected networks • 503
    - in to Galaxies • 41
    - in to Galaxies, security enabled • 41
- M**
- managed InTouch application, exporting language data • 514
  - manually
    - initializing scripts • 125
    - validating objects • 99
  - Message Exchange and attributes • 453
  - moving
    - objects, undeploying • 30
    - views • 37
  - multiple Galaxies • 530
- N**
- naming conventions
    - Attributes • 123
    - containment • 64, 67
    - Galaxies • 20
    - instances • 99
    - scripts • 162
    - templates • 57
    - toolsets • 51
  - nesting templates • 44, 58
  - network
    - configuring DNS settings • 539
    - configuring IP address • 539
  - network cards
    - redundancy • 547
    - setting • 539
    - specifying the order • 539
  - Network drives, mapping • 537
- O**
- object
    - derivation, viewing • 33
    - properties, viewing • 49
    - relationships, viewing • 30
  - object attributes, finding • 88
  - Object Editor
    - getting help • 69
    - opening • 68
  - object help
    - adding images • 81
    - exporting • 532
    - HTML editors for customizing • 81
  - Object Information page • 80
  - objects
    - checking in • 97
    - checking out • 97
    - deleting • 101
    - exporting • 102, 532
    - exporting language data • 512
    - importing • 105
    - in runtime • 318
    - inheriting extensions • 126
    - opening Object Editor • 68
    - specific information • 134
    - undeploying • 338
    - validating manually • 99

Operations view, opening • 36

ordering network cards • 539

Output

    functionality • 134

Output extensions, data quality • 135

## P

parent and child, relationships, viewing • 33

passwords, changing • 41

Primary AppEngine, viewing attributes • 90

primary object, redundancy • 545

primary server, redundancy • 545

propagating changes, templates • 45

published InTouch application, exporting

    language data • 514

## R

rate of change alarms • 365

reassigning objects, undeploying • 30

recreating ArchedrA user account • 538

redundancy

    alarm generation • 554

    ApplicationObjects configuration • 550

    configuring templates • 550

    during deployment • 552

    history generation • 555

    pair • 545

    runtime • 546

    setting network cards • 547

Redundancy, page • 549

redundant engines, undeploying • 551

RedundantDIObject • 545

reference

    auto-bound • 148

references in scripts, validating • 98

relationships, viewing parent/child • 33

renaming

    contained objects, updating references • 67

reorganizing the workspace • 37

reserved names

    instances • 100

    list of • 20

    templates • 57

resetting

    the workspace • 38

restoring Galaxies • 529

return views to the default locations • 38

reusing existing objects • 105

ROC Alarms Feature Parameters • 140

rules for locking

    aliases in scripts • 164

    scripts • 164

runtime

    and bit field access • 90

    attributes • 460

    how objects deploy from the IDE • 318

runtime environment, scripting • 161

## S

script function libraries

    exporting • 104

    importing • 109

script libraries, exporting • 532

scripting

    Attributes • 125

    containment names • 60

scripts

    aliases, using • 162

    automatically starting • 125

    execution types • 162

    exporting • 102

    initializing Startup scripts • 125

    InputOutput attributes • 132

    locked in child objects • 164

    locking rules • 164

    manually initializing • 125

    naming conventions • 162

    timing • 162

    validating • 98, 161

Scripts page • 79

Search • 74

Search Current Attributes • 74

security

    Galaxy users • 481



- groups, default • 481
  - groups, importing objects • 119
  - icon not available • 73
  - icons • 74
  - opening Galaxies • 22
  - options in attributes • 74
  - restricting access to attributes • 73
  - restrictions, creating new Galaxies • 20
  - security roles • 481
  - Security Mode • 494
  - setting network cards • 539
  - setting prompts for check-in comments • 38
  - silenced alarms • 368, 369
  - single scan cycles • 134
  - Situational Awareness Library symbol • 170
  - SmartSymbols, exporting language data • 514
  - standby object, redundancy • 546
  - Startup scripts, initializing • 125
  - state alarms • 362
  - Statistics Feature Parameters • 147
  - store forward, historized data • 348
  - storing history, configuring • 355
  - subscribing to alarms • 383
  - suppressed alarms • 368
  - switch objects • 134
  - Symbol Wizard
    - Consumer workflow • 170
    - creating multiple configuration symbols • 171
    - Designer workflow • 170
    - embedding symbols into an InTouch application • 172
  - symbols
    - exporting language data • 512
    - translating text • 511
  - system templates • 47
- T**
- tagnames • 58, 64, 100
  - tagnames, containment • 60
  - target deviation alarms • 364
  - Template Toolbox, location • 24
  - templates
    - alarm extensions • 139, 140, 141, 143, 146
    - application • 47
    - base • 44, 45
    - classes • 45
    - configuring redundancy • 550
    - contained names • 64
    - containment • 58
    - creating derived • 57
    - default • 44, 45
    - defined • 44
    - deploying • 43
    - derived • 45, 48
    - device integration • 47
    - exporting • 532
    - inheriting extensions • 126
    - naming conventions • 57
    - nesting • 44
    - nesting levels • 58
    - propagating changes • 45
    - reserved names • 57
    - system • 47
    - time formats, .csv files • 535
    - timestamps
      - saving as historical data • 344
    - toolsets
      - creating • 52
      - naming conventions • 51
- U**
- undeploying
    - history • 348
    - objects before moving • 30
  - undeploying redundant engines • 551
  - unlocking, template attributes • 70
  - Upload Runtime Changes
    - I/O Auto Assignment • 157
  - User Account Control • 537
  - user accounts • 538
  - user defaults, setting • 38
  - users

- default • 481
- deleting • 502

## V

### validating

- Galaxies • 99
- object manually • 99
- objects • 98
- objects, viewing results • 36
- references in scripts • 98
- scripts • 98, 161

### viewing

- object properties • 49
  - object relationships • 30
  - objects, assignment relationships • 32
- views in the IDE • 24

## W

### Windows

- user accounts • 538

### WinPlatform object, configuring for redundancy • 547

### WinPlatform, object • 47

### WinPlatforms, history configuration • 351

### Wonderware Historian, installing • 343

### writing

- to attributes in runtime, attributes  
in runtime, writing to attributes • 70