



Wonderware
Information Model
Configuration Guide

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 Invensys Systems, Inc. No copyright or patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this documentation, the publisher and the author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein.

The information in this documentation is subject to change without notice and does not represent a commitment on the part of Invensys Systems, Inc. The software described in this documentation is furnished under a license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of these agreements.

© 2012-2013, 2014 by Invensys Systems, Inc. All rights reserved.

Invensys Systems, Inc.
26561 Rancho Parkway South
Lake Forest, CA 92630 U.S.A.
(949) 727-3200

<http://www.wonderware.com>

For comments or suggestions about the product documentation, send an e-mail message to ProductDocumentationComments@invensys.com.

All terms mentioned in this documentation that are known to be trademarks or service marks have been appropriately capitalized. Invensys Systems, Inc. cannot attest to the accuracy of this information. Use of a term in this documentation should not be regarded as affecting the validity of any trademark or service mark.

Alarm Logger, ActiveFactory, ArcestraA, Avantis, DBDump, DBLoad, DT Analyst, Factelligence, FactoryFocus, FactoryOffice, FactorySuite, FactorySuite A², InBatch, InControl, IndustrialRAD, IndustrialSQL Server, InTouch, MaintenanceSuite, MuniSuite, QI Analyst, SCADAAlarm, SCADASuite, SuiteLink, SuiteVoyager, WindowMaker, WindowViewer, Wonderware, Wonderware Factelligence, and Wonderware Logger are trademarks of Invensys plc, its subsidiaries and affiliates. All other brands may be trademarks of their respective owners.

CodeMirror © 2013 by Marijn Haverbeke <marijnh@gmail.com>.

Contents

	Welcome	5
	About This Guide	5
	Documentation Conventions	5
	Technical Support	6
Chapter 1	Configuring the Information Model	7
	Using the Information Model Configuration Tool	8
	Accessing the Information Model Configuration Tool	8
	Viewing the Relationship List	9
	Adding or Editing a Relationship	10
	Defining Data Sources	10
	Model.DataSources Table	11
	Model.DataSourceAttributes Table	12
	Defining Data Items	13
	Defining Root Data Items	14
	Defining Relationships	15
	Defining Rules	16
	Model.Rules	16
	Model.RuleAttributes Table	17

Defining Time Zones	20
Available Time Zones	20
Configuring Time Zones	20
Overriding the Data Source Time Zone in Rules	21
Parameters	21
Navigation Parameters	21
Special Parameters	24
Customizing Column Name Captions	26
Specifying Well-Known Data Items	28
Status Indicators	28
Time Series	30
Process Events	35
State Transitions	36
External Content	41
Securing the Information Model Configuration Tool	43
Chapter 2 Troubleshooting and Diagnostics	47
OverView Client Error Logging	47
Information Model Configuration Tool Error Logging	48
Repairing Information Model Configuration Tool Files	48
Re-configuration of the Information Model Feature	49
Chapter 3 Model Store Tables and Views	51
Model.Attributes	52
Model.DataSources	52
Model.DataSourceAttributes	53
Model.DataItems	54
Model.DataItemsAttributes	54
Model.Relationships	55
Model.Rules	55
Model.RuleAttributes	56
Model.DataItemColumns	57
Model.ResourceStrings	57
Model.TimeZones	58
Model Store Views	58
Chapter 4 Using Pre-defined Content	59
Index	61

Welcome

About This Guide

This guide describes the steps for configuring an information model and the specific rules that you need to follow during configuration.

Documentation Conventions

This documentation uses the following conventions:

Convention	Used for
Initial Capitals	Paths and file names.
Bold	Menus, commands, dialog box names, and dialog box options.
Monospace	Code samples and display text.

Technical Support

Wonderware Technical Support offers a variety of support options to answer any questions on Wonderware products and their implementation.

Before you contact Technical Support, refer to the relevant section(s) in this documentation for a possible solution to the problem. If you need to contact technical support for help, have the following information ready:

- The type and version of the operating system you are using.
- Details of how to recreate the problem.
- The exact wording of the error messages you saw.
- Any relevant output listing from the Log Viewer or any other diagnostic applications.
- Details of what you did to try to solve the problem(s) and your results.
- If known, the Wonderware Technical Support case number assigned to your problem, if this is an ongoing problem.

Chapter 1

Configuring the Information Model

You can configure an information model by performing the following actions:

- 1** Define the data sources, such as the MES or Alarm database, or Wonderware Historian. For more information, see "Defining Data Sources" on page 10.
- 2** Define the data items, such as Work Order, Jobs, Equipment, or Batch. For more information, see "Defining Data Items" on page 13.
- 3** Define the root data items, the starting point of your navigation. For more information, see "Defining Root Data Items" on page 14.
- 4** Establish relationships between the data items, such as Work Order Has Jobs. For more information, see "Defining Relationships" on page 15.
- 5** Define rules, which includes details on the action to be taken when a relationship is navigated. For example, an SQL script. For more information, see "Defining Rules" on page 16.
- 6** Create the Column Name Caption mapping for an optimal column header display in a client tool for a customized column header display in the OverView client. This is an optional step. For more information, see "Customizing Column Name Captions" on page 26.

The above actions need to be performed against the Model Store tables. You need to follow specific rules to make the content of the tables functional in the run-time environment and the OverView client.

Although starting the configuration of the information model from scratch is possible, a good starting point is to install the pre-defined content. To install the pre-defined content, see Chapter 4, "Using Pre-defined Content."

An easy way to configure the Model Store is to use the Information Model Configuration Tool. For more information, see "Using the Information Model Configuration Tool" on page 8. However, not everything can be configured with the Configuration Tool, and you will still need to refer to sections in this documentation to complete your configuration.

The following sections explain the process of configuring an information model, the behavior of special data items and the rules for navigation parameters.

Before configuring an information model, you must ensure that the product is properly installed on your computer. Verify that the OverView client does not report any errors and that the Information Server database contains the RootDataItems entry in the Model.DataItems table and that the Model.DataSources table contains an entry for the Model Store.

Using the Information Model Configuration Tool

Use the Information Model Configuration Tool to view, add, and edit relationships. Using the tool, you can create the model you want without having to know the underlying Model Store tables.

However, the tool does not currently support some operations, such as creating a root data item, deleting relationships, and managing data sources. For these operations, you must directly modify the content of the Model Store tables. For more information about the Model Store tables, see Chapter 3, "Model Store Tables and Views."

For more information about relationships, data items, root data items, or data sources, see the corresponding sections of this documentation before reading about the Information Model Configuration Tool.

Accessing the Information Model Configuration Tool

The Information Model Configuration Tool can be accessed under the Administration node of the Information Model Server Portal. To access this node, your user needs to have the Administrator role configured in Wonderware Information Server.

Before accessing the Information Model Configuration Tool, make sure your Windows user has access to the Model Store. For more information about granting access to the Model Store, see "Securing the Information Model Configuration Tool" on page 43.

The Information Model Configuration Tool is available from within the Wonderware Information Server portal. To use the tool, you must:

- Be assigned the Administrator role within Wonderware Information Server.
- Have security access to the Model Store. For more information about granting access to the Model Store, see "Securing the Information Model Configuration Tool" on page 43.

To access the tool

- 1 Expand the **Administration** node and then click **Information Model**.

The screenshot shows the Wonderware Information Server Administration interface. The left sidebar is titled 'Administration' and lists various management tools. The 'Information Model' link is highlighted with a red box. The main content area displays a description of the Administration page and a 'Links List' containing various management tools. The 'Information Model' link is also highlighted with a red box. A large globe graphic is visible in the background.

- 2 Click **Configuration Tool for the Information Model**.

Viewing the Relationship List

The Configuration Tool shows you all of the relationships from the Model Store. Each element of the list is a relationship. Most of the time, each relationship is represented as a sentence made up of a subject, predicate, and object. If a relationship is represented as a single word, this relationship is a root data item and can be accessed only from the OverView home screen.

If your model contains too many relationships, and you cannot find a specific one, you can filter them. Type any part of a relationship name and only those relationships containing the specified text are shown.

If you prefix your filter with the character "!", only the relationships that do NOT contain the specified text are shown. For example, the string "!Jobs" filters out any relationship that contains "Jobs" in its name.

Adding or Editing a Relationship

To add a relationship

- 1 Click the **Add Relationship** link at the top of the **Relationships** page.
- 2 Specify the subject, the predicate, and the object of the new relationship.
- 3 For the object data item, you can either select an existing data item or click **Add New** and type the name of a new data item to create.

To edit a relationship

- 1 Click a relationship in the relationship list.
- 2 Edit the relationship's script. This information is related to rules and scripts. For more information about relationship scripts, see "Defining Rules" on page 16.
- 3 Edit the relationship's data source. It is not possible to create a new data source from this page. For more information about data sources, see "Defining Data Sources" on page 10.

Defining Data Sources

The first step in configuring an information model is to define the data sources. The data source definition is contained in the following two tables:

- Model.DataSources
- Model.DataSourceAttributes

Model.DataSources Table

This table contains a high-level definition of the data sources used in the information model. For more information, see "Model.DataSources" on page 52.

The following is an example of content that can be found in the Model.DataSources table:

DataSourceID	Name	TimeZoneID	DataAdapterType
1	Model Store	NULL	Microsoft SQL Server Database
2	MESDB	2	Microsoft SQL Server Database
3	OSI PI	2	OSIsoft PI Server OLE DB

Note the following:

- The DataSourceID "1" for the Model Store connection must be present in the Model.DataSources table.
- The value "Null" in the TimeZoneID column for DataSource ID "1" indicates that the date/time values, if there are any in the Model Store, are stored in Coordinated Universal Time (UTC).
- The value "2" in the TimeZoneID column for DataSourceID "2" refers to the TimeZoneID, specified in the Model.TimeZones table.
- The values in the DataAdapterType column have to comply with the following data adapter definitions:
 - Microsoft SQL Server Database
 - Oracle Database
 - Text File
 - Historian
 - OSIsoft PI Server OLEDB

The data adapter definition files are available in the "<CommonFilesFolder>\Archestra\DataAdapters\bin" folder.

Model.DataSourceAttributes Table

This table contains detailed information, such as the connection string and the data adapter location, for every data source. For more information, see "Model.DataSourceAttributes" on page 53.

The following is an example of content that can be found in the Model.DataSourceAttributes table:

AttributeID	DataSourceID	Name	Value
1	1	ConnectionInfo	<encrypted connection string>
2	1	NodeName	Localhost
3	1	Port	8788
4	2	ConnectionInfo	Source Type=0;Data Source=localhost;Initial Catalog=MESDB;Integrated Security=True
5	2	NodeName	Localhost
6	2	Port	8788
7	3	ConnectionInfo	Provider=PIOLEDB;Data Source=myserver;Connection Type=Any;Initial Catalog=piarchive;Integrated Security=SSPI
8	3	NodeName	Localhost
9	3	Port	8788

Note the following:

- The NodeName and the Port attributes refer to the location (computer name or IP address) and the listening port of the data adapter communicating with the data source, respectively.
- The connection string will be automatically encrypted when it is edited using the Configurator.
- You can use the following as an alternative option for the user and password security connection (versus integrated security):

```
Data Source=MyHistorian;Initial Catalog=Runtime;Integrated Security=false;User Id = wwUser; password=wwUser
```

Defining Data Items

After defining the data sources, you need to define the data items. The `Model.DataItems` table contains the name and the unique ID of each data item. A data item can be used as a root data item that is initially shown in the `OverView` client or as a relationship link to another data item. For more information, see "Model.DataItems" on page 54.

The following is an example of content that can be found in the `Model.DataItems` table:

DataItemID	Name
1	RootDataItems
2	WorkOrder
3	Jobs

Note the following:

- The DataItemID "1" (RootDataItems) is reserved for the system.
- In this example, Work Order would be configured to be a root data item and would be initially shown in the `OverView` client. Work Order would have a relationship to Jobs.
- Data items can have any name. Some names have specific meanings when they are used in the context of the `OverView` client. The data item name cannot include a colon (:).
- You should avoid using invalid filename characters (`\ / * ? " < > |`) in the data item name. If you use any of these characters, the `OverView` client cannot persist the tabular display customizations such as column width for the data item.

The behavior of the `OverView` client varies according to the data to be shown. This behavior is based on the following well-known data items:

Data Item Name	Definition
RootDataItems	The first data item to be shown on the <code>OverView</code> client startup. RootDataItems are the first in a sequence of data item navigation. For more information, see "Defining Root Data Items" on page 14.
Status Indicators	A relationship to a Status Indicators data item enables the <code>OverView</code> client to display a visual cue, indicating that additional information exists for a record of the current data item. For more information, see "Status Indicators" on page 28.
Time Series	The Time Series data item contains Time Series data to be plotted on the trend display in the <code>OverView</code> client. For more information, see "Time Series" on page 30.

Data Item Name	Definition
Process Events	A relationship to a Process Events data item enables the OverView client to display a visual cue consisting of an icon and a tooltip. The cue provides additional information on the trend display in the OverView client. For more information, see "Process Events" on page 35.
State Transitions	A relationship to a State Transition data item enables the OverView client to display a visual cue consisting of a halo highlighting certain sections of the Time Series trend lines. The cue provides additional information on the trend display in the OverView client. For more information, see "State Transitions" on page 36.
External Content	A relationship to an External Content data item enables the Overview client to display links to external content in the relationships popup. These links allow the user to leverage other application capabilities using the default web browser to view the selected record data. For more information, see "External Content" on page 41.

After you navigate to a data item, the source data item name and the value of the first column of the record selected for navigation are updated in the context view pane of the OverView client. The context view pane always uses the first column value, even if the column is set to be hidden.

Defining Root Data Items

After defining data items, you need to set the root data items. The Model.DataItemsAttributes table contains a special attribute, indicating that a data item is the origin of navigation. For more information, see "Model.DataItemsAttributes" on page 54.

The following is an example in the Model.DataItemsAttributes table:

DataItemID	AttributeID
2	1

Note the following:

- The value in the DataItemID column corresponds to the DataItemID of the Work Order data item in the Model.DataItems table. For more information, see "Defining Data Items" on page 13.
- The value "1" in the AttributeID column sets "Work Order" as the root data item.

Defining Relationships

After defining the root data items, you need to define the relationships between the data items. The Model.Relationships table stores the information used to describe the relationships. For more information, see "Model.Relationships" on page 55.

The following is an example of content that can be found in the Model.Relationships table:

RelationshipID	ObjectID	Predicate	SubjectID
1	1	Null	Null
2	2	Null	Null
3	3	Contains	2

Note the following:

- RelationshipID "1" (ObjectID "1") is reserved for the system.
- RelationshipID "2" indicates that the Work Order data item is used as a root data item (Predicate and SubjectID are "Null").
- For RelationshipID "3", the values in the ObjectID and SubjectID columns refer to the DataItemID for the Jobs and Work Order data items in the Model.DataItems table. This means that the Work Order data item contains the Jobs data item.

There is no restriction on the predicate name. However, the Information Model Services pre-defines one predicate called "Has Metadata", which allows you to configure the relationship between the data items with metadata:

Predicate	Description
Has Metadata	<p>The relationship configured with the predicate provides the metadata to the data, retrieved for the subject involved in the relationship.</p> <p>For example, Alarm State Transitions can have color metadata that represents the different State Transitions.</p> <p>The client can use metadata to render a plot differently. For example, metadata can be used to draw halos to denote different State Transitions on a Time Series data trend. For more information, see "State Transitions" on page 36.</p>

Defining Rules

After defining the relationships between the data items, you need to define the rules. The rules definition is contained in the following two tables:

- Model.Rules
- Model.RuleAttributes

Model.Rules

The Model.Rules table contains a high-level definition of the action type to be executed when a relationship navigation is selected. For more information, see "Model.Rules" on page 55.

The following is an example of content that can be found in the Model.Rules table:

RuleID	RuleType	RelationshipId
1	Script	1
2	Script	2
3	Script	3

Note the following:

- RuleID "1" refers to the default system root data item (RootDataItems data item).
- RuleID "2" refers to the relationship used to retrieve the Work Order root data item.
- RuleID "3" refers to the relationship between Jobs and Work Order data items.

Model.RuleAttributes Table

The Model.RuleAttributes table contains the details of the script that will be executed to retrieve data. The script will be executed against the configured data source. For more information, see "Model.RuleAttributes" on page 56.

The following is an example of content that can be found in the Model.RuleAttributes table:

AttributeID	RuleID	Name	Value
1	1	DataSource	Model Store
2	1	SQLScript	SELECT * FROM (SELECT d.Name FROM Model.DataItems d INNER JOIN Model.DataItemsAttributes da on d.DataItemID=da.DataItemID INNER JOIN Model.Attributes a on da.AttributeID=a.AttributeID WHERE a.Name='RootDataItem') as RootDataItems
3	2	DataSource	MESDB
4	2	SQLScript	Select wo_id as WorkOrderId,wo_desc as WorkOrderDescription,item_id as Material,req_qty as RequiredQuantity from [dbo].wo
5	3	DataSource	MESDB

AttributeID	RuleID	Name	Value
6	3	SQLScript	<pre> select distinct wo_id +'.'+ oper_id +'.'+ cast(seq_no AS Nvarchar) AS Job, wo_id as WorkOrderId, oper_id as OperationId, seq_no as SequenceNumber, item_id as Material, job_desc as JobDescription, qty_prod as QuantityProduced, qty_reqd as QuantityRequired, qty_rejected as QuantityRejected, batch_size as Batch_Size, ent.ent_name as EquipmentName, edit_time as EditTime, act_start_time_UTC as ActualStartTimeUTC, act_finish_time_UTC as ActualFinishTimeUTC, sched_start_time_UTC as ScheduledStartTimeUTC, sched_finish_time_UTC as ScheduledFinishTimeUTC, uom.description as UOM FROM [dbo].job INNER JOIN [dbo].ent ON [dbo].job.run_ent_id= [dbo].ent.ent_id LEFT OUTER JOIN [dbo].uom ON [dbo].uom.uom_id = [dbo].job.prod_uom WHERE wo_id= @WorkOrderId </pre>
7	3	UseSourceTimeZone	False

Note the following:

- For each action, multiple rule attributes are required, as shown in the Model.RuleAttributes table: one for a data source target and one for a SQL script to be executed against the data source. Additional attributes may be available. For example, a Time Zone usage definition.
- In the Model.RuleAttributes table, AttributeID "1" and "2" are for the default handling of the RootDataItems. These rows must exist.

- AttributeID "3" and "4" refer to the data source and the SQL query to be executed for the first navigation item—Work Order list in this case.
- If UseSourceTimeZone is "Null" in the Model.RuleAttributes table, it means that you can use the time zone defined for data source. If the UseSourceTimeZone attribute is not present, it is set to "True" by default. This means that the value set of the TimeZoneID column in the Model.DataSources table is used.

There are limitations for the OSIPi OLEDB value type.

You need to use the CAST() function to cast the OSI PI value type to the Float64 type to view the trend and data in the OverView client. Check the sample script to plot a trend using the OSI PI Data. For more information, see "Time Series" on page 30.

For example, you have used the following query to set the Model.RuleAttributes table to get the trend data display for the tag "SINUSOID":

```
SELECT a.tag, b.descriptor, DIGSTRING(a.status)
as statusdesc,CAST(a.value as float64) as Value, a.time as
[Absolute Time] FROM piarchive.picomp2 a
INNER JOIN pipoint.pipoint2 b ON a.tag = b.tag where
a.tag='SINUSOID' and a.time >= @DayStart AND a.time < @DayEnd
```

The DateTime parameters need to be converted to DateTime values using the CAST() function in OSI PI when you pass the DateTime parameters across relationships in the OverView client using the OSI PI OLEDB data adapter. You can use the following sample scripts to plot the Time Series trend.

For the above scenario, the scripts in the Model.RuleAttributes table should be the following:

```
Rule1: Select DayStart,DayEnd,Weekday from calendar ( Data
source : non OSI PI DB)

Rule2( script for "Has Process variables" relationship) :
SELECT DISTINCT tag, DayStart=CAST(@DayStart as datetime),
DayEnd=CAST(@DayEnd as datetime) FROM pipoint..classic

Rule3 (script for "Process variables has time Series"
relationship):

SELECT tag, time as [Absolute Time], CAST(value as float64) as
value, status FROM piarchive..picomp2 WHERE tag = @tag and
time >= @DayStart and time <= @DayEnd
```

Defining Time Zones

The Information Model helps you visualize data that comes from different data sources. Each of these data sources could be located in different locations and could contain date/times in different time zones.

For example, a batch's start time could refer to the time in Houston, Texas (Central Time Zone), whereas a process variable's time could come from a different data source and refer to the time in Los Angeles, California (Pacific Time Zone).

In this example, if in OverView you want to correctly overlay together data from the batches and from the process variables, the Information Model must know the time zone of each data source.

Available Time Zones

The `Model.TimeZones` table stores all the time zones supported by the operating system on the node where the Information Model Services are running.

The following is an example of content that can be found in the `Model.TimeZones` table:

TimeZoneID	Name
1	Central Standard Time
2	GMT Standard Time
3	India Standard Time
4	UTC

For more information about the structure of the table, see "Model.TimeZones" on page 58.

Configuring Time Zones

To configure your data source time zone, you must reference the correct time zone in the "Model.DataSources" table. If you do not make this configuration, the Information Model assumes that your data source is using Coordinated Universal Time (UTC).

Overriding the Data Source Time Zone in Rules

As there can be many relationships using the same data source, what should be done if only one of these relationships needs to return a table with UTC time? In other words, what if you need to use the data source's time zone all the time, except for a specific case where UTC is needed?

In this case, you can override the data source's configuration for a specific rule. For this, you need to set the "UseSourceTimeZone" attribute to false. This attribute can be added in the "Model.RuleAttributes" table. If you do this, a relationship that uses this rule assumes that all date/times are in UTC, regardless of the configuration that was done for the source.

Parameters

You can use the following parameters in a query:

- Navigation parameters
- Special parameters

Navigation Parameters

Relationship navigation requires setting a context in which the next-level query can be executed. For example, when moving from a list of Work Orders to the list of Jobs for a specific Work Order, a Work Order parameter context is required.

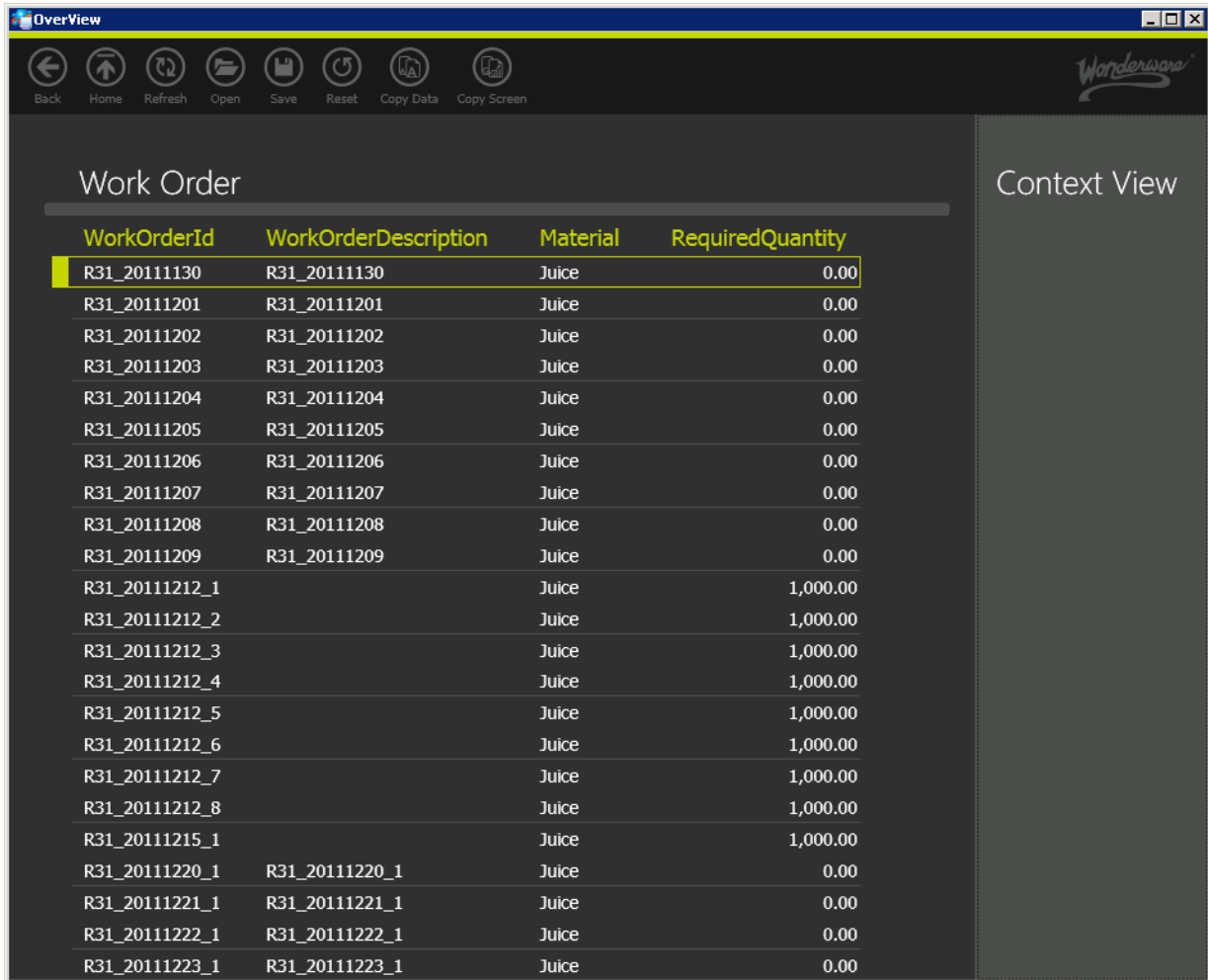
Any column from a query can be a parameter for the next query.

To pass a parameter, add "@" at the beginning of one or more column names returned in the preceding query.

For example, the following query returns the list of Work Orders from an MES database:

```
Select
    wo_id as WorkOrderId,
    wo_desc as WorkOrderDescription,
    item_id as Material,
    req_qty as RequiredQuantity
from [dbo].wo
```

The following graphic shows a list of Work Order items on the OverView tabular display.

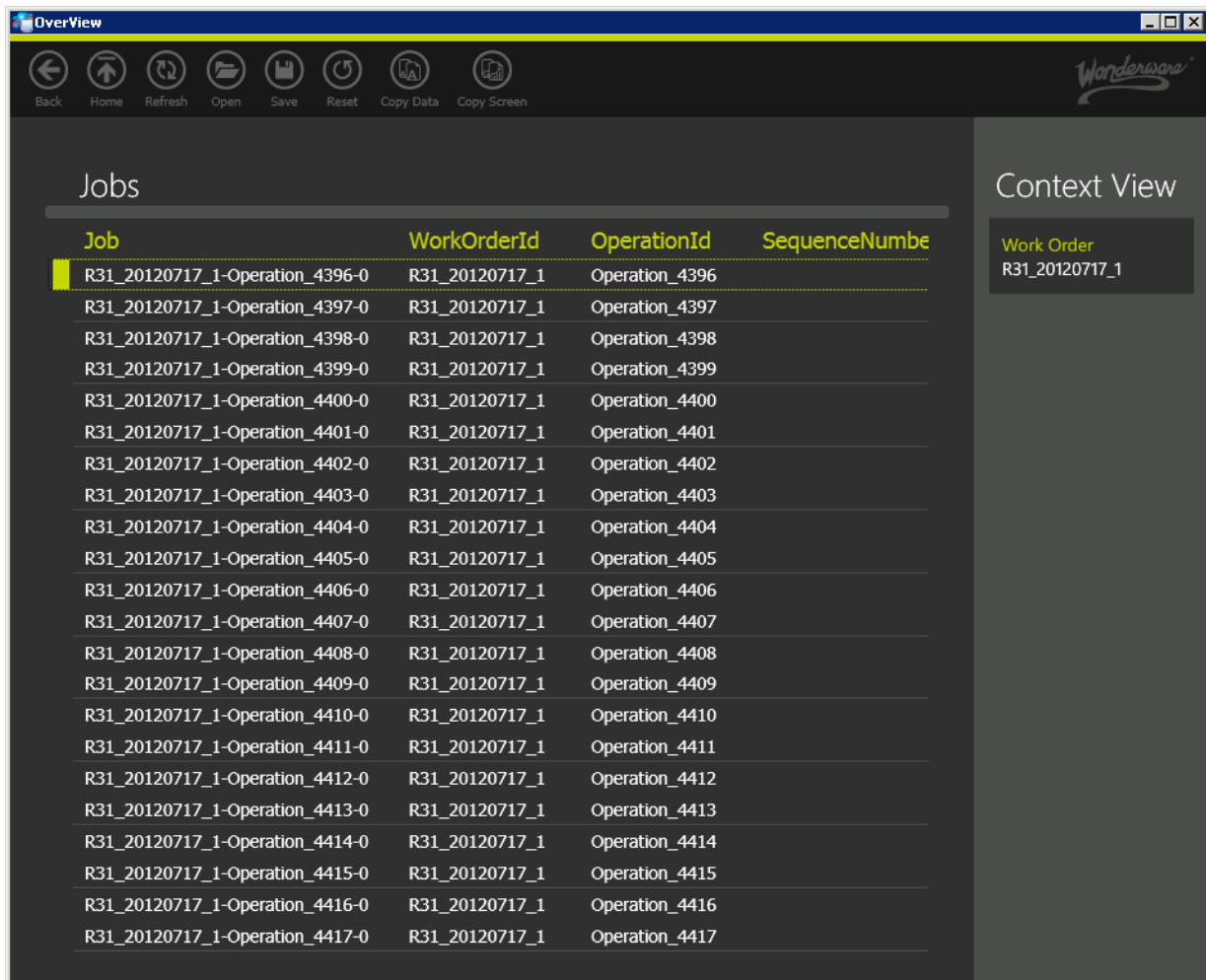


WorkOrderId	WorkOrderDescription	Material	RequiredQuantity
R31_20111130	R31_20111130	Juice	0.00
R31_20111201	R31_20111201	Juice	0.00
R31_20111202	R31_20111202	Juice	0.00
R31_20111203	R31_20111203	Juice	0.00
R31_20111204	R31_20111204	Juice	0.00
R31_20111205	R31_20111205	Juice	0.00
R31_20111206	R31_20111206	Juice	0.00
R31_20111207	R31_20111207	Juice	0.00
R31_20111208	R31_20111208	Juice	0.00
R31_20111209	R31_20111209	Juice	0.00
R31_20111212_1		Juice	1,000.00
R31_20111212_2		Juice	1,000.00
R31_20111212_3		Juice	1,000.00
R31_20111212_4		Juice	1,000.00
R31_20111212_5		Juice	1,000.00
R31_20111212_6		Juice	1,000.00
R31_20111212_7		Juice	1,000.00
R31_20111212_8		Juice	1,000.00
R31_20111215_1		Juice	1,000.00
R31_20111220_1	R31_20111220_1	Juice	0.00
R31_20111221_1	R31_20111221_1	Juice	0.00
R31_20111222_1	R31_20111222_1	Juice	0.00
R31_20111223_1	R31_20111223_1	Juice	0.00

If we assume that the next relationship navigation item is Work Order Has Jobs, the next query is:

```
select distinct
    wo_id + '-' + oper_id + '-' + cast(seq_no AS Nvarchar) AS Job,
    wo_id as WorkOrderId,
    ent.ent_name as EquipmentName,
    act_start_time_UTC as ActualStartTime,
    act_finish_time_UTC as ActualFinishTime,
from [dbo].job
    INNER JOIN [dbo].ent ON [dbo].job.run_ent_id=
        [dbo].ent.ent_id
where wo_id= @WorkOrderId
```

The following graphic shows a list of Jobs for the selected Work Order item.



Job	WorkOrderId	OperationId	SequenceNumber
R31_20120717_1-Operation_4396-0	R31_20120717_1	Operation_4396	
R31_20120717_1-Operation_4397-0	R31_20120717_1	Operation_4397	
R31_20120717_1-Operation_4398-0	R31_20120717_1	Operation_4398	
R31_20120717_1-Operation_4399-0	R31_20120717_1	Operation_4399	
R31_20120717_1-Operation_4400-0	R31_20120717_1	Operation_4400	
R31_20120717_1-Operation_4401-0	R31_20120717_1	Operation_4401	
R31_20120717_1-Operation_4402-0	R31_20120717_1	Operation_4402	
R31_20120717_1-Operation_4403-0	R31_20120717_1	Operation_4403	
R31_20120717_1-Operation_4404-0	R31_20120717_1	Operation_4404	
R31_20120717_1-Operation_4405-0	R31_20120717_1	Operation_4405	
R31_20120717_1-Operation_4406-0	R31_20120717_1	Operation_4406	
R31_20120717_1-Operation_4407-0	R31_20120717_1	Operation_4407	
R31_20120717_1-Operation_4408-0	R31_20120717_1	Operation_4408	
R31_20120717_1-Operation_4409-0	R31_20120717_1	Operation_4409	
R31_20120717_1-Operation_4410-0	R31_20120717_1	Operation_4410	
R31_20120717_1-Operation_4411-0	R31_20120717_1	Operation_4411	
R31_20120717_1-Operation_4412-0	R31_20120717_1	Operation_4412	
R31_20120717_1-Operation_4413-0	R31_20120717_1	Operation_4413	
R31_20120717_1-Operation_4414-0	R31_20120717_1	Operation_4414	
R31_20120717_1-Operation_4415-0	R31_20120717_1	Operation_4415	
R31_20120717_1-Operation_4416-0	R31_20120717_1	Operation_4416	
R31_20120717_1-Operation_4417-0	R31_20120717_1	Operation_4417	

Note: You should not use column aliasing [select x AS y from...] if you need to use the column name as a parameter in subsequent queries. For more information, see "Customizing Column Name Captions" on page 26.

You can use the Column Name Caption capacity to hide columns, if required. You can use it to hide navigation parameters, if they are not important, or if you do not want to clutter the client display with too much information.

Make sure to consider the associated time zone when using date/time values as parameters. For more information, see "Defining Time Zones" on page 20.

Special Parameters

In addition to the navigation parameters, you can also use special parameters in a query.

@aaServerHostName

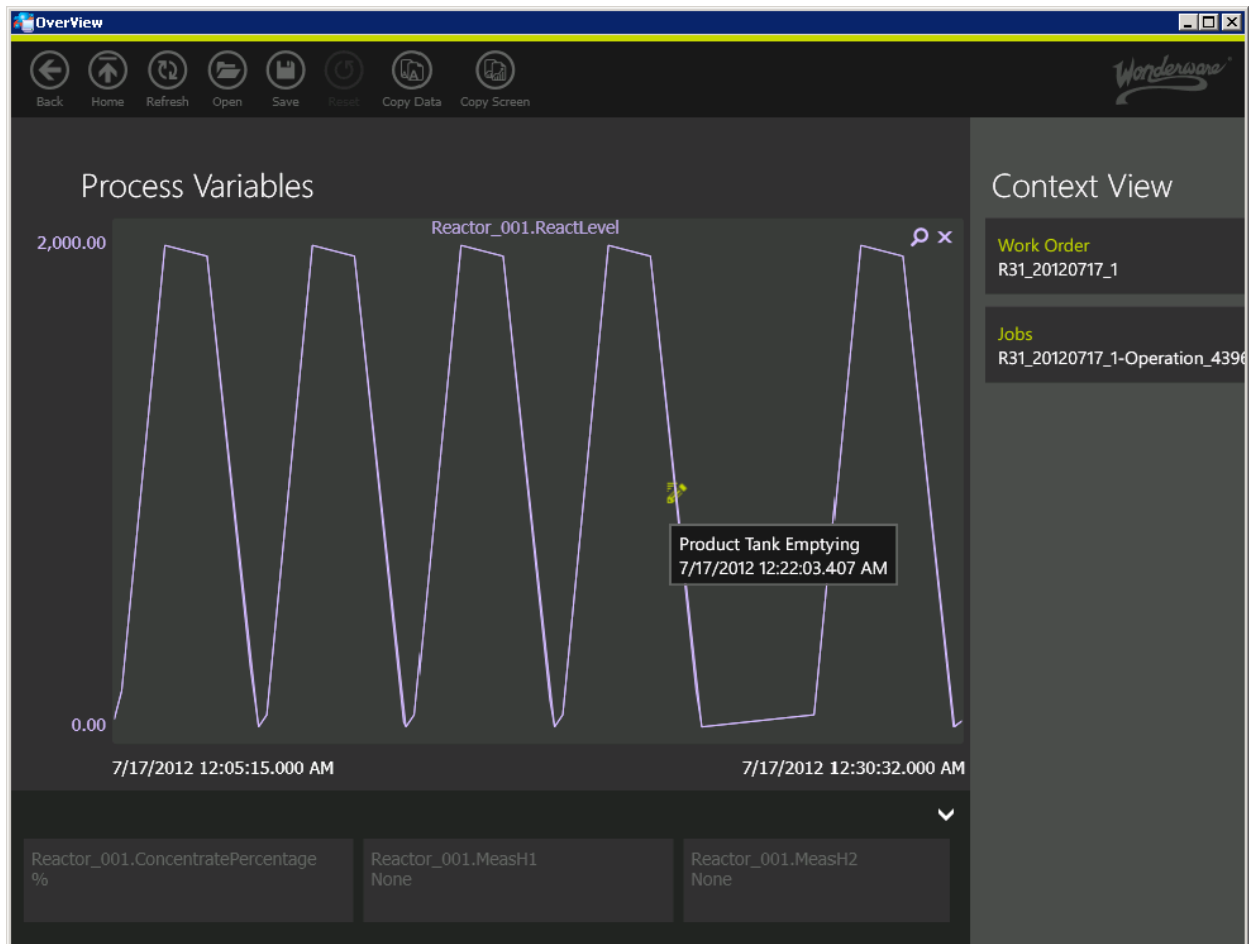
The value of the @aaServerHostName parameter is the host name or the IP address that was used to reach the Information Model Services. As an example, this parameter is useful when you are configuring a Status Indicators data item for the OverView client. You can use the @aaServerHostName parameter in a script to specify a relative URL to the image.

For example, the following query retrieves a Status Indicator whenever annotations exist for a TagName. In this case, the image is located in the "Images" sub-folder of the default virtual folder.

```
SELECT 'Annotation Count: ' + CONVERT(VARCHAR,AnnotationCount)
    as AnnotationHoverText,
    @ActualStartTimeUTC as ActualStartTimeUTC,
    @ActualFinishTimeUTC as ActualFinishTimeUTC,
    @TagName as TagName,
    CASE
        WHEN AnnotationCount > 0
            THEN N'http://' + CONVERT(NVARCHAR,@aaServerHostName)
            + N'/Images/AnnotationImage.png'
        ELSE NULL
    END as AnnotationImageURL

FROM (SELECT COUNT(*) As AnnotationCount FROM MyAnnotationTable
    WHERE [DateTime] >= @ActualStartTimeUTC
    AND [DateTime] <= @ActualFinishTimeUTC
    AND TagName = @TagName)
```


The following graphic shows annotations on the Time Series trend.



@aaClientHostName

The value of the @aaClientHostName parameter is the host name that the Overview client runs on. The parameter value is the fully qualified DNS hostname. For a workgroup configuration, it is only the computer name.

```
SELECT *,
      FROM MyTableName
      WHERE MachineNameColumn = @aaClientHostName
```

@aaClientUserName

The value of the @aaClientUserName parameter is the user name, including the domain, that the Overview client runs as. An example in which this parameter might be useful is configuring a data item such as "Work Orders Assigned to Me".

```
SELECT *,
      FROM MyTableName
      WHERE UserNameColumn = @aaClientUserName
```

@aaClientLCID

The value of the @aaClientLCID parameter is the Locale ID integer for the locale that the OverView client runs in. An example in which this parameter might be useful is configuring data items with locale-specific language and terminology.

```
SELECT *,
    CASE @aaClientLCID
        When 1031 Then 'Dies ist ein deutscher Text'
        When 1033 Then 'This is an English text'
        else 'Unknown' END as LocalizedTextColumn
FROM MyTableName
```

Customizing Column Name Captions

You can customize the table column names, if required. Though the model queries can include aliasing (Select ColumnName AS [My Column Name]...), it is not feasible for columns to be used as parameters, as the original column name has to be preserved.

The Information Model Services allow you to configure column captions, which are then used by a client display. You can also hide columns in a client display by using "#Hidden" as the column caption.

The Column Name Caption mapping details are stored in two Model Store tables—Model.DataItemColumns and Model.ResourceStrings. For more information, see "Model.DataItemColumns" on page 57 and "Model.ResourceStrings" on page 57.

The [Model].[sp_i_DataItemColumn] stored procedure can be used to customize column names.

For example, the syntax is as follows:

```
[Model].[sp_i_DataItemColumn] @dataItemName, @columnName,
    @displayName
```

For example, assuming the following query is used to retrieve work orders:

```
Select wo_id ,wo_desc ,item_id from [dbo].wo
```

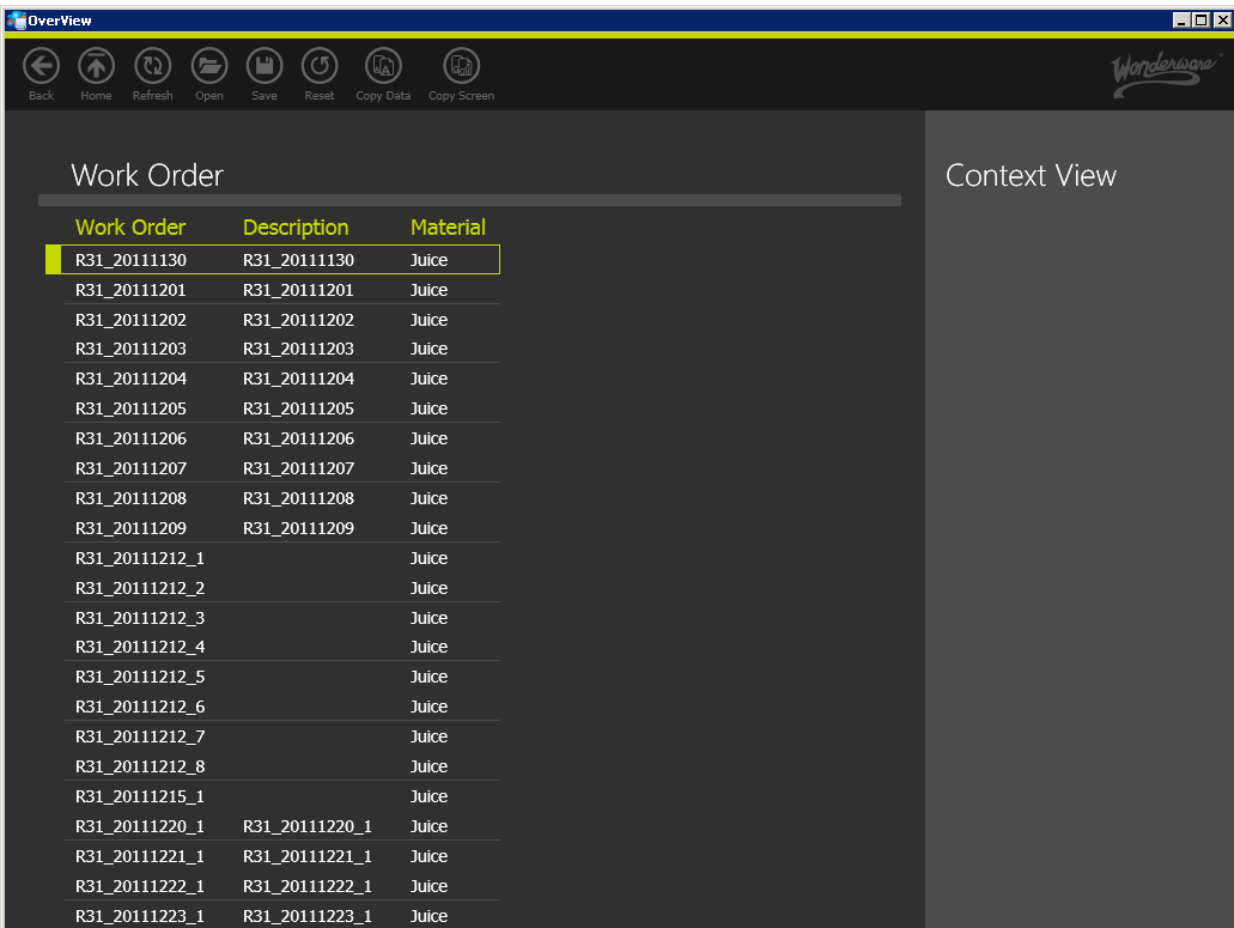
The column name caption can be customized using:

```
exec [Model].[sp_i_DataItemColumn]
    @dataItemName = 'Work Order',
    @columnName   = 'wo_id',
    @displayName  = 'Work Order ID'

exec [Model].[sp_i_DataItemColumn]
    @dataItemName = 'Work Order',
    @columnName   = 'wo_desc',
    @displayName  = 'Description'
```

```
exec [Model].[sp_i_DataItemColumn]
    @dataItemName = 'Work Order',
    @columnName   = 'item_id',
    @displayName  = 'Material'
```

The OverView client displays the following:



The following updates are made to the Model Store:

Model.ResourceStrings table:

StringID	DisplayString
1001	Work Order
1002	Description
1003	Material

Model.DataItemColumns table:

DataItemID	ColumnName	ResourceStringID
2	wo_id	1001
2	wo_desc	1002
2	item_id	1003

Specifying Well-Known Data Items

By using certain words when naming your data items, you can turn a regular data item into one of the following:

- Status Indicators
- Time Series
- Process Events
- State Transitions
- External Content

Status Indicators

The Status Indicators data items are well-known data items in the OverView client and consist of an image (icon) with a tooltip.

The OverView client recognizes the Status Indicators data items because their names end with "Status Indicators". For example:

- Alarm Status Indicators
- Annotation Status Indicators

To leverage the Status Indicators feature, the Rule Attribute script needs to return columns ending with the following names:

- ImageURL (type:String)—The path to the status indicator image to be shown in the OverView client.
- HoverText (type:String)—The text to be shown as the Status Indicators tooltip.

A relationship to a Status Indicators data item enables the display of a visual cue in the form of an additional column in the tabular display or an icon in the trend display's pen list.

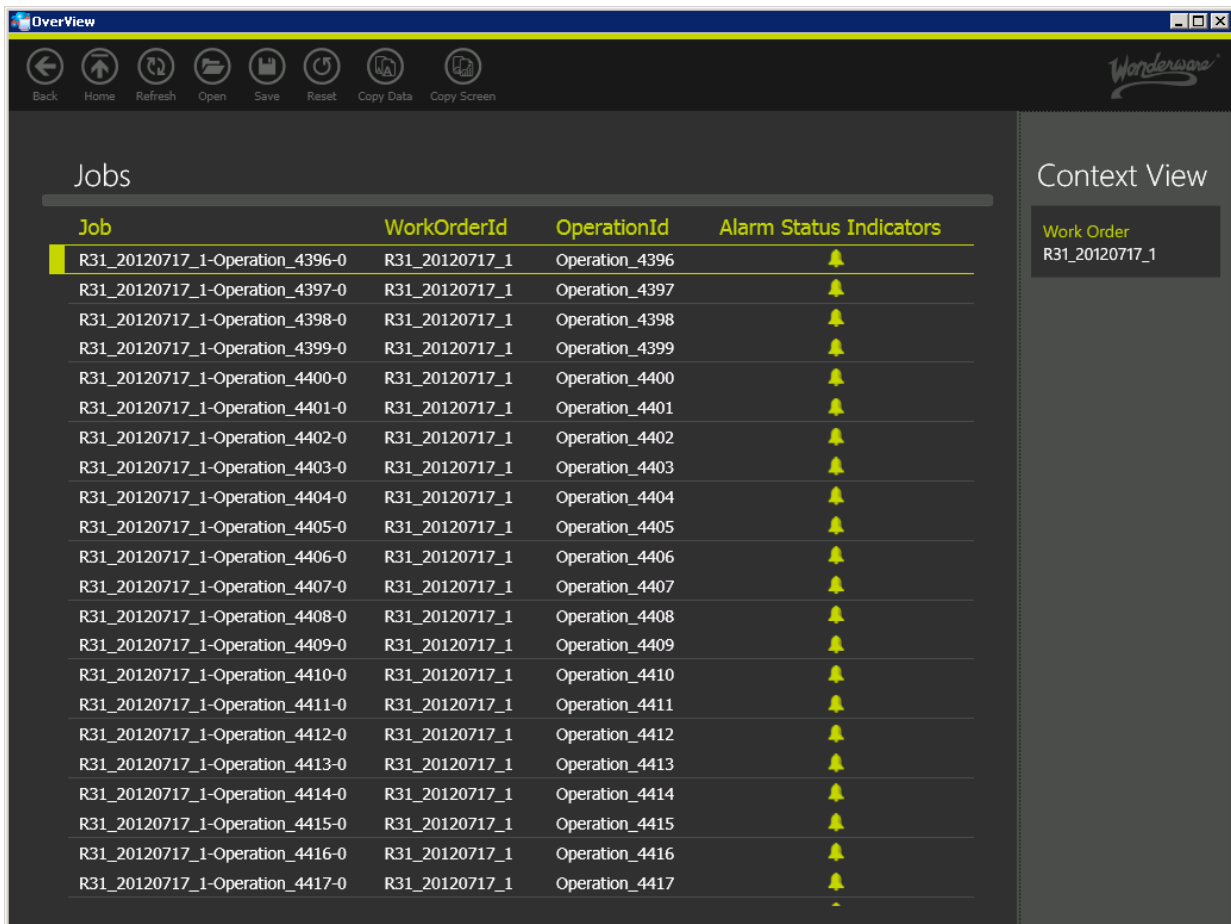
A relationship from a Status Indicators data item to another data item enables the ability to "drill through" for more information on a specific record.

Additional columns containing navigation parameters may be required to provide the context of the drill-through. These columns are not displayed in the OverView client.

The following is an example of a script used to retrieve status indicators:

```
SELECT
    'Alarm Count=' + CONVERT(VARCHAR,AlarmCount) as
    AlarmHoverText,
    CASE WHEN AlarmCount > 0
        THEN 'http://' + CONVERT(VARCHAR,@aaServerHostName)
        + '/Images/AlarmImage.png'
        ELSE NULL
    END as AlarmImageURL,
    @ActualStartTimeUTC as ActualStartTimeUTC,
    @ActualFinishTimeUTC as ActualFinishTimeUTC,
    @TagName as TagName
FROM
    (SELECT COUNT(*) As AlarmCount
     FROM v_AlarmEventHistory2
     WHERE AlarmState='UNACK_ALM'
     AND EventStampUTC >= @ActualStartTimeUTC
     AND EventStampUTC <= @ActualFinishTimeUTC
     AND TagName = @TagName) aaAlias
```

The following graphic shows an Alarm Status Indicator (bell) for the selected Job.



Time Series

The Time Series data items are well-known data items in the Overview client. The Overview client recognizes the Time Series data items because their names end with "Time Series". For example:

- Detailed Time Series
- Summary Time Series

Navigating to any data item that contains a relationship to a Time Series data item triggers the display of a trend in the Overview client.

The following example explains the navigation to a data item that contains a relationship to a Time Series data item:

Navigation Action	Result
Select Work Order from the Home display	A grid of Work Orders (tabular display)
Navigate the "Work Order Contains Jobs" relationship	A grid of Jobs for the selected work order (tabular display)
Navigate the "Job Has Process Variables" relationship (The "Process Variables Has Time Series" relationship exists)	A trend of the Time Series data for all process variables (trend display) If a data item has a relationship to a data item ending with "Time Series", the time series data will automatically be displayed in the trend display and the records for this data item will represent pens in the display.

To leverage the Time Series feature, the query needs to return the following columns:

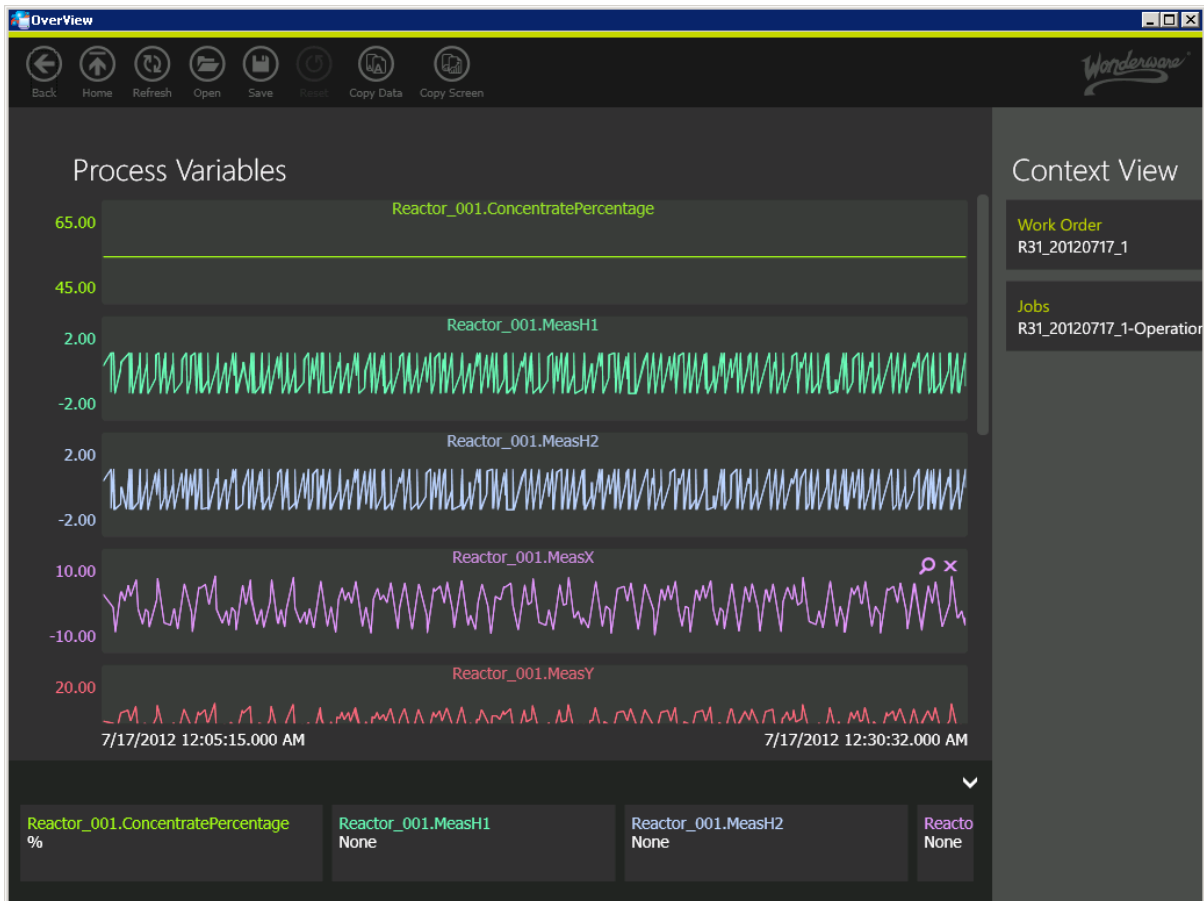
- Absolute Time (type: DateTime)—The timestamps of the values
- Value (type: float)—The values to be plotted on the trend

Note: The trend display does not use the other columns, such as OPCQuality. Only Absolute Time and Value are required.

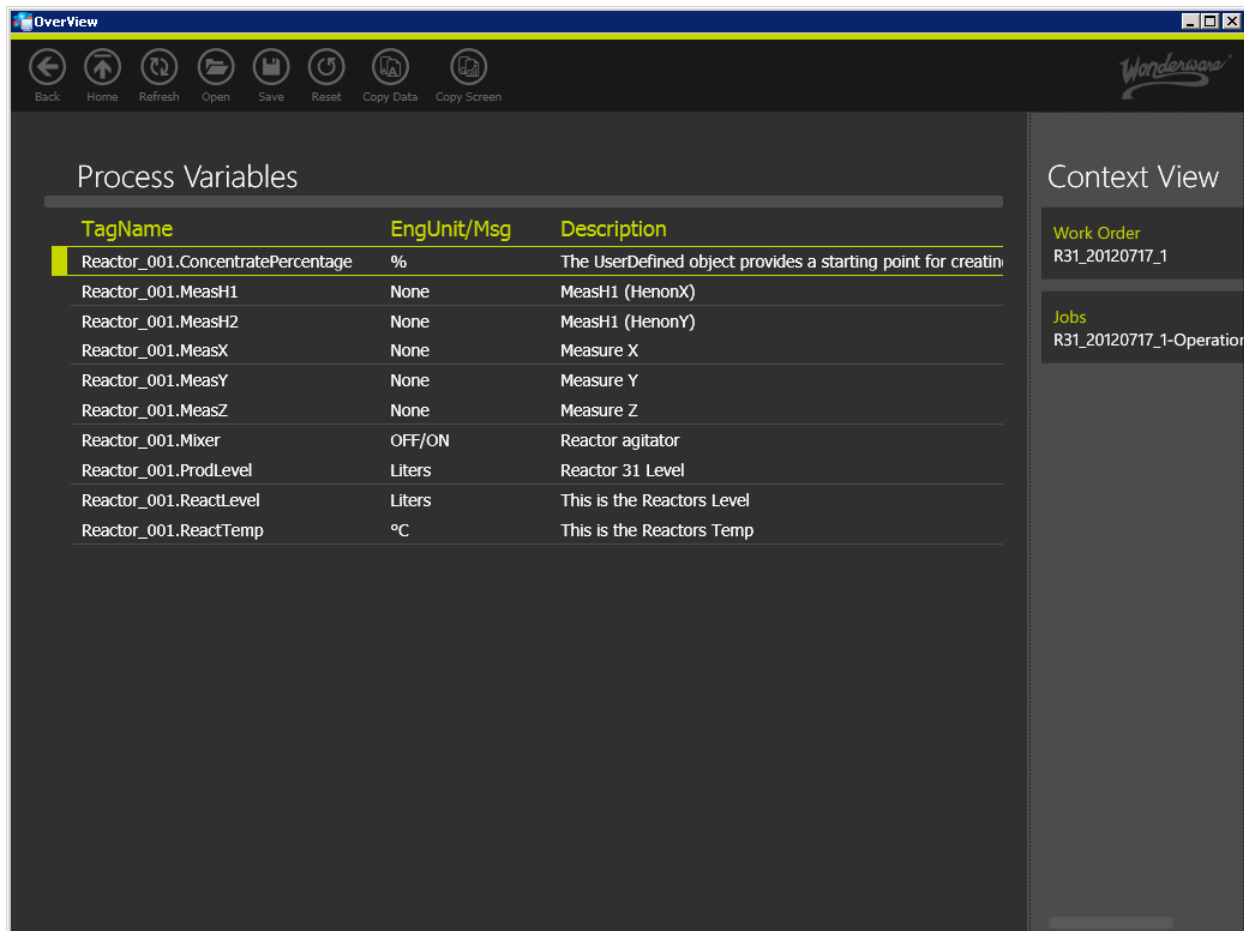
The following is a script example for the Wonderware Historian database:

```
SELECT
    [TagName],
    [Value],
    [DateTime] AS [Absolute Time],
    [OPCQuality]
FROM History
WHERE TagName = @TagName
    AND [DateTime] >= @ActualStartTimeUTC
    AND [DateTime] <= @ActualFinishTimeUTC
    AND wwTimeZone = 'UTC'
    AND wwRetrievalMode = 'BestFit'
```

The following graphic shows a Time Series trend in the OverView client.



In this example, if the "Process Variables" data item did not have a relationship to a Time Series data item, then the list of process variables would have been shown in the tabular display.



The screenshot shows the Wonderware Overview interface. The main area displays a table titled "Process Variables" with the following data:

TagName	EngUnit/Msg	Description
Reactor_001.ConcentratePercentage	%	The UserDefined object provides a starting point for creatin
Reactor_001.MeasH1	None	MeasH1 (HenonX)
Reactor_001.MeasH2	None	MeasH1 (HenonY)
Reactor_001.MeasX	None	Measure X
Reactor_001.MeasY	None	Measure Y
Reactor_001.MeasZ	None	Measure Z
Reactor_001.Mixer	OFF/ON	Reactor agitator
Reactor_001.ProdLevel	Liters	Reactor 31 Level
Reactor_001.ReactLevel	Liters	This is the Reactors Level
Reactor_001.ReactTemp	°C	This is the Reactors Temp

The right sidebar, titled "Context View", shows the following information:

- Work Order:** R31_20120717_1
- Jobs:** R31_20120717_1-Operati

In a trend display, each pen contains primary and secondary information in separate rows. The primary information shows the value of the first column from the pen record, and the secondary information shows the value of the second column from the pen record. If you click on the pen details button, the system shows a tooltip containing the pen record information in two columns—one with column captions and the other with column values.

The following graphic shows a trend pen with primary and secondary information with a tooltip.



If a pen contains a column name ending with "Interpolation Type", then its trend display is based on the value present in that column.

The Interpolation Type column can have the following values:

- Linear: This plots the trend with linear interpolation.
- Stair Step: This plots the trend with stair step interpolation.

If there is no column ending with "Interpolation Type", then linear interpolation is used to plot the trend.

The Interpolation Type column name and its values are case insensitive and must be defined on the data item that has a relationship with a data item whose name ends with "Time Series".

Process Events

The Process Events data items are well-known data items in the OverView client and consist of an image (icon) and a tooltip. Process Events are shown on the trend only if the current data item has related Time Series data. For instance, when you navigate to a particular data item, where relationships exist to both a Time Series and Process Events data item, the Process Event icons are shown on the related Time Series trend line.

The OverView client recognizes Process Events data items because their names end with "Process Events". For example:

- Alarm Process Events
- Annotation Process Events

To leverage the Process Events feature, the script needs to return columns ending with the following names:

- HoverText (type: string)—The text to be shown as the tooltip.
- ImageURL (type: string)—The absolute path of the image to be shown on the trend at the given timestamp.
- TimeStamp (type: DateTime)—The timestamp at which the process event occurred. This information will be added to the tooltip.

The following is an example of a script that returns process events:

```
SELECT
    'http://' + CONVERT(VARCHAR,@aaServerHostName)
    + '/Images/AcknowledgeAlarmImage.png' as AlarmImageURL,
    'Limit: ' + alm.LimitString + CHAR(13)
    + 'Value: ' + alm.ValueString as AlarmHoverText,
    m.UTCOriginationTime as Timestamp
FROM...
```

The following graphic shows the Process Events with a tooltip on the Time Series trend.



State Transitions

The State Transitions data items are well-known data items in the Overview client. State transitions define a period of time and are used to show a halo on certain sections of the Time Series trend lines.

The halo is shown on the trend only if the current data item has the following relationships:

- Time Series
- State Transitions

For instance, when a user navigates to a particular data item where relationships exist to both a Time Series and State Transitions data item, a halo can be shown on the related Time Series trend line.

The OverView client recognizes State Transitions data items because their names end with "State Transitions". For example:

- Alarm State Transitions
- Equipment State Transitions

The following example explains the navigation to a data item that has a relationship to a data item ending with "Time Series":

Navigation Action	Result
Select Work Order from the Home display	A grid of Work Orders (tabular display)
Navigate the "Work Order contains Jobs" relationship	A grid of Jobs for the selected Work Order (tabular display)
Without state transition halo: Navigate the "Job Has Process Variables" relationship (The "Process Variables Has Summary Time Series" relationship exists)	A trend of the Time Series data for all Process Variables. (trend display) If the Process Variables data item does not have a relationship to a data item ending with "Time Series", the Process Variables are displayed in a grid (tabular display)
With state transition halo: Navigate the "Job Has Process Variables" relationship (The "Process Variables Has Summary Time Series" relationship exists, along with "Alarm State Transitions")	A trend of Time Series data for all Process Variables (trend display), along with the halo trend indicating the "Alarm State" as defined in "State Transitions" relationship.
Navigate the "Process Variables Has Summary Time Series" relationship	A grid of the Time Series data for the selected Process Variable (tabular display)

To leverage the State Transitions feature, the script is required to return columns ending with the following names:

- StartTimeUTC (type: DateTime)—The timestamp at which the state transition began.
- EndTimeUTC (type: DateTime)—The timestamp at which the state transition ended.

Note: If the value of the State Transition column, StartTimeUTC, is "Null", then the halo is not displayed for this State Transition.

The following is an example of a script that returns alarm state transitions:

```
CREATE TABLE #AlarmsFinalList(StateName CHAR(6), StartTimeUTC
    DATETIME, EndTimeUTC DATETIME, Priority int)

INSERT INTO #AlarmsFinalList SELECT [StateName],
    [StartTimeUTC], [EndTimeUTC],[Priority]
    FROM
    (SELECT m.AlarmType AS [StateName], m.OriginationTime AS
    [StartTimeUTC],
    CASE m.AlarmType WHEN 'ROCL' THEN 1 WHEN 'ROCH' THEN 1
    WHEN 'DSC' THEN 1
    WHEN 'HiHi' THEN 2 WHEN 'LoLo' THEN 2
    WHEN 'Hi' THEN 3 WHEN 'Lo' THEN 3
    ELSE 10 END AS [Priority],
    (SELECT MIN(d.TransitionTime)FROM AlarmDetail d WHERE
    d.AlarmId = m.AlarmId
    AND d.AlarmState LIKE '%_RTN'
    AND d.TransitionTime >= @ActualStartTimeUTC
    -- User can extend this filter to include alarm records
    -- with later transition time
    AND d.TransitionTime <= DATEADD(HOUR, 1,
    @ActualFinishTimeUTC)
    ) AS [EndTimeUTC]
    FROM [WWALMDB].[dbo].[AlarmMaster] m
    WHERE TagName LIKE (CASE m.AlarmType
    WHEN 'DSC'
    THEN @TagName ELSE @TagName + '.*' END)
    AND m.OriginationTime <= @ActualFinishTimeUTC
    -- User can extend this filter to include alarm records
    -- with older origination time
    AND m.OriginationTime >= DATEADD(HOUR, -1,
    @ActualStartTimeUTC)
    ) aa

WHERE ...

SELECT [StateName], [StartTimeUTC], [EndTimeUTC] FROM
    #AlarmsFinalList
    ORDER BY Priority ...
```

The State Transition pre-defined content excludes the Alarms that have originated more than one hour before the Job start time. You can modify the script to extend the time range.

To provide the color metadata to draw halos to denote different State Transitions on the Time Series trend lines, the following items are required:

- The Model.Relationships table must have a "Has Metadata" relationship to the State Transitions data item.
- The script for this relationship must return the color in ARGB format along with columns matching columns found on the State Transitions data item to allow filtering.

ARGB represents different components of the color that will be used to display the halo: Alpha (transparency), Red, Green and Blue. Valid ARGB values are 0 to 255 for each component. An alpha value of 255 corresponds to an opaque color. Values from 1 to 254 create a semitransparent color and 0 creates a fully transparent (not visible) color. For the RGB components, a value of 0 does not contribute to the resulting color whereas a value of 255 provides a full contribution of that component to the resulting color.

Note: If no metadata is configured for the State Transition or there are no matching columns, then the halo will be shown with the default color.

If the StartTimeUTC value occurs before the first Time Series data point and the EndTimeUTC value occurs after the first Time Series data point, then the halo will begin from the left edge of the trend. The Y-axis starting value of the halo will be the first not-null Time Series data point value.

If the EndTimeUTC value occurs after the last Time Series data point and the StartTimeUTC value occurs before the last Time Series data point, then the halo will continue until the right edge of the trend and the end point value of the halo will be the last not-null Time Series data point value.

If the halo occurs between Time Series data points and within the "Null" value data point's duration, then the halo will be shown in a straight line and its value will be the previous not-null Time Series data point value.

If a State Transition rule returns multiple records, then multiple halos are shown using the start time from the StartTimeUTC column, end time from the EndTimeUTC column, and the color from the metadata of each record. If State Transitions overlap each other, the order in which state transitions are returned from the rule script dictates the priority of overlapping state transitions. For instance, the first State Transition record has the highest priority and will overlap lower priority State Transitions, while the second record has the next priority and will overlap lower priority State Transitions, and so on.

If an overlapping higher-priority State Transition is too small in comparison to the Trend Time Series range, the lower-priority State Transition may seem to "swallow up" the higher-priority State Transition in the trend display. The State Transition to Time Series range ratio is 10:1430.

For example, if the Time Series data set range is 24 hours, then a higher State Transition ≤ 10 minutes is not shown in the Trend. If the Time Series data set range is 7 days, then a higher State Transition ≤ 70 minutes will not be seen in the Trend.

The following is an example of a script that returns the color metadata that will be associated with alarm state transitions:

```
SELECT '150,255,120,120' AS Color, 'Hi' AS StateName UNION
SELECT '150,255,0,0' AS Color, 'HiHi' AS StateName UNION
SELECT '150,255,120,120' AS Color, 'Lo' AS StateName UNION
SELECT '150,255,0,0' AS Color, 'LoLo' AS StateName UNION
SELECT '150,175,0,0' AS Color, 'ROCL' AS StateName UNION
SELECT '150,175,0,0' AS Color, 'ROCH' AS StateName UNION
SELECT '150,175,0,130' AS Color, 'DSC' AS StateName
```

A tooltip is shown when the mouse pointer hovers over a halo. All information returned by the State Transition record, except hidden and metadata columns, is shown in two columns, one for the column caption and the other with the value. Content in the value column is localized as per the operating system being used. If two or more halos overlap each other, then the tooltip for the higher priority halo will be shown.

The following graphic shows State Transitions (with a tooltip) on a Time Series trend.



External Content

The "External Content" data items are well-known data items in the Overview client. When this type of data item is defined, links are added to the list of relationships. These links don't open an Overview display, but open URLs in the default browser.

External Content data items are recognized by Overview because their names end with "External Content". For example:

- AFWeb External Content

To leverage the External Content feature, the script is required to return columns ending with the following (case-insensitive) names.

- LinkURL (type:String): The URL link that will be opened for a specific record.
- DisplayName (type:String): The text to be shown in the relationships popup.

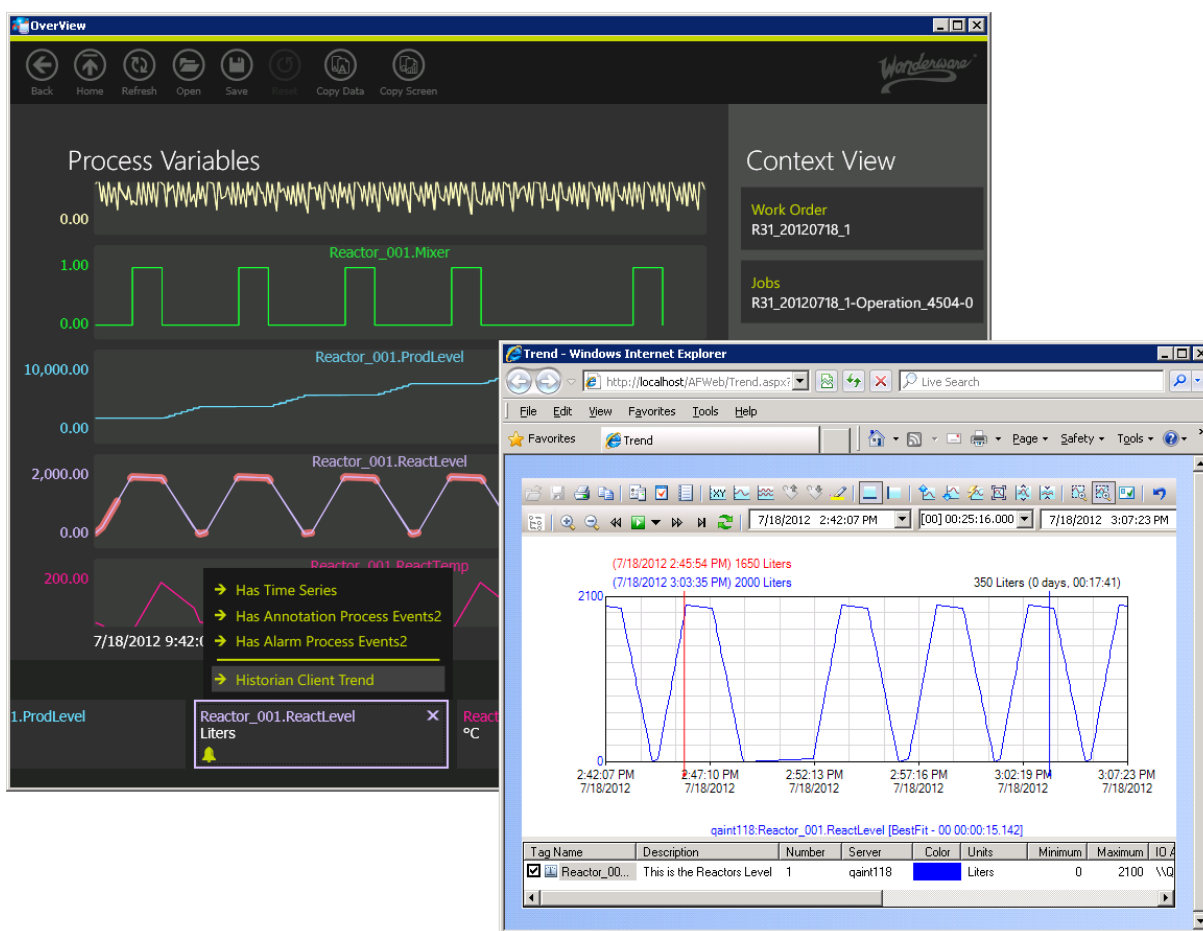
The URL link doesn't get modified by the Overview client. Only links having well-formed Http or Https URLs will be shown.

URL links are defined by the rule script. If the URL needs to open a web page with specific context, then the URL needs to be created with the right query string parameters.

For example, to enable the ActiveFactory Trend page opening in a browser, the table is as follows:

Link URL	DisplayName
Http://<Servername>/Afweb/Trend.aspx?SV=true&Tags=SysTime Sec&StartUTC=2012-07-15 07:01:01.001&EndUTC=2012-07-15 08:01:01.001	ActiveFactory Trend

The following graphic shows external content that was opened from the Trend display:

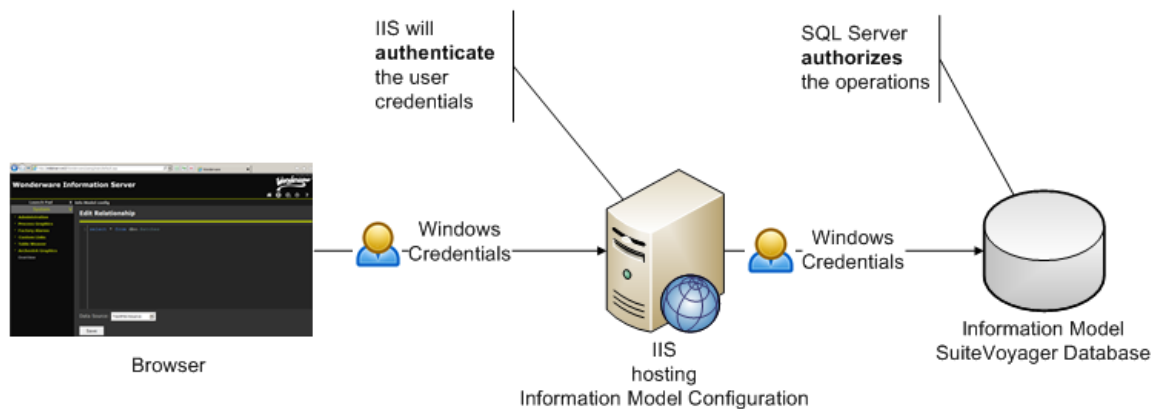


Securing the Information Model Configuration Tool

The Information Model Configuration Tool enables authorized users to add to or update the information model (that is, add relationships, configure scripts, and so on) using a web interface.

The installation of this tool does not set up any logins in SQL Server. Therefore, you must perform some additional steps to define logins and secure the model at the database level.

The Information Model Configuration Tool performs read/write operations on tables within the [Model] schema of the SuiteVoyager database. Windows credentials are passed from this web application down to the SQL Server to be authorized in the tables belonging to that [Model] schema.

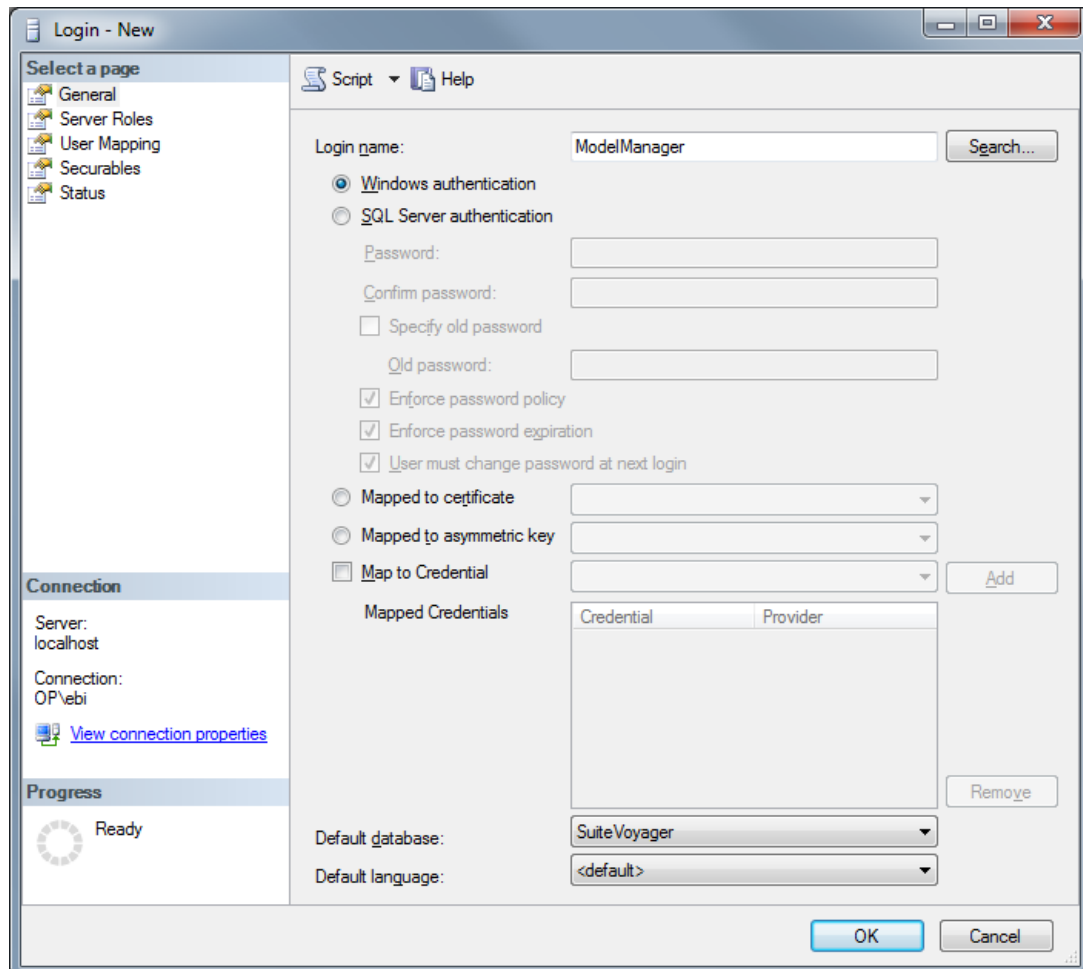


The following steps are one way to secure the information model. However, as long as the requirements stated above are satisfied, you can use other authorization techniques.

To secure the information model

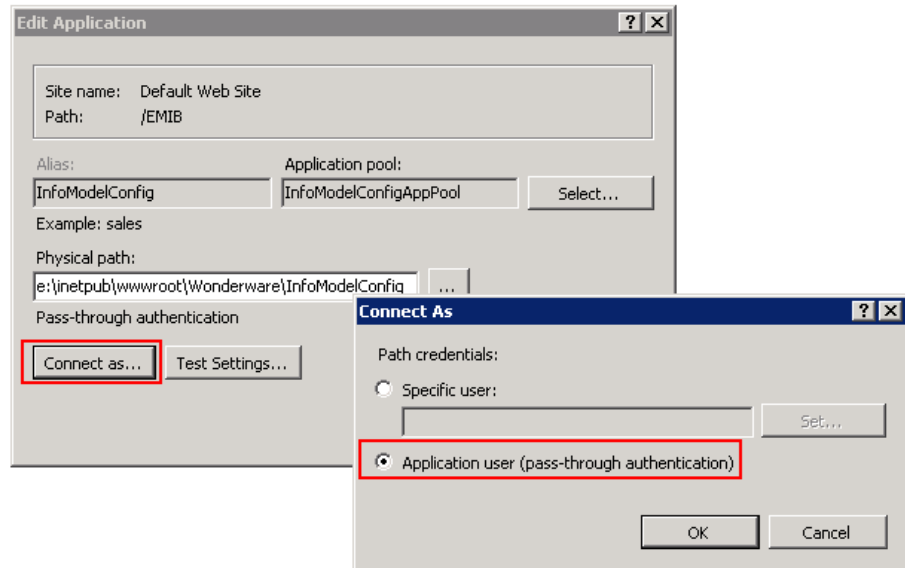
- 1 Define the Windows credentials of the information model manager(s) in SQL Server. Do the following:
 - a Add a new login to SQL Server.

- b** Map this login to the information model user using Windows authentication.



- 2** Set the User Mappings on the SuiteVoyager database holding the information model store.
- a** Enable the [InformationModelAdmin] membership roles.

Important: Windows credentials of the information model manager(s) are used to read/modify the content of the information model. By default, the 'InfoModelConfig' virtual directory's basic settings are configured with **Pass-through authentication**. Do not configure it to **Specific user**; this would open a security hole because everyone could have read/write access to the information model using credentials of this configured user. For more information, see <http://technet.microsoft.com/en-us/library/cc753653.aspx>. The following graphic shows the required configuration:



Chapter 2

Troubleshooting and Diagnostics

The information model provides you with the following diagnostic options to troubleshoot:

- The ArcestrA Logger in the System Management Console (SMC). The Information Model Services use the logger to output diagnostics information.
- The OverView client error logging
- The Information Model Configuration Tool error logging

OverView Client Error Logging

The OverView client does not provide output to the ArcestrA Logger, as it has its own log file. By default, the log file is not written to the disk. You can turn the logging on by modifying the `aaClient.exe.config` file. For more information, refer to the comment in the `<system.diagnostics>` section of the `aaClient.exe.config` file, which explains how to turn logging on or off. The file where the logs are written is specified in the same section.

You can also turn the logging on by pressing `Ctrl + F11` in the Overview client, and then using the **Save As** dialog box to create the log file. You can stop logging to the file by restarting the application.

Information Model Configuration Tool Error Logging

By default, error logging for the Information Model Configuration Tool is turned off. To turn logging on, edit the <system.diagnostics> section of the web.config file located in the InfoModelConfig virtual directory. The web.config file contains instructions on how to turn the logging on or off.

Error logging requires that the user who is accessing the tool have write privileges on the folder where the errors will be logged. By default, errors are logged in the following file:

C:\Windows\Temp\InformationModelConfigTool.log. On a Windows default installation all users have write privileges on the C:\Windows\Temp folder.

If the log file was not created, or you think error traces are not logged, you can do the following:

- Recycle the InfoModelConfigAppPool. This will free lock(s) on the log file from any process, and the next attempt to log error will start afresh.
- Make sure that the user accessing the Information Model Configuration Tool has:
 - Write privileges on the C:\Windows\Temp folder
 - Write privileges on the InformationModelConfigTool.log file

Or, you could configure in the web.config file a different location where this user has write privileges.

Repairing Information Model Configuration Tool Files

If the Information Model Configuration Tool files are missing or corrupted, doing a repair of Wonderware Information Server will not fix them. To repair these files, open the Configurator and configure the Information Model feature.

Re-configuration of the Information Model Feature

Re-configuration of the Information Model feature replaces all of the Information Model Configuration tool files and re-creates the virtual directory. Any changes you make to the InfoModelConfig virtual directory configuration or to any of the Information Model Configuration Tool files will be lost.

Chapter 3

Model Store Tables and Views

The Model Store contains the following tables and views:

- Model.Attributes table
- Model.DataSources table
- Model.DataSourceAttributes table
- Model.DataItems table
- Model.DataItemsAttributes table
- Model.Relationships table
- Model.Rules table
- Model.RuleAttributes table
- Model.DataItemColumns table
- Model.ResourceStrings table
- Model.TimeZones table
- Model.DataSourceSummary view
- Model.RuleSummary view

Model.Attributes

This table stores an attribute that is used when returning the list of root data items. This table should not be modified.

Model.DataSources

The Model.DataSources table stores the information about the data sources from where data is retrieved.

The following table explains the structure of the Model.DataSources table:

Column	Data Type	Description
DataSourceID	Integer	Contains the unique identifier for the data source, which is automatically generated.
Name	Character	Contains the unique name of the data source. The length of the text cannot exceed 100 characters. Null values are not allowed.
DataAdapterType	Character	Contains the type of data adapter. The length of the text cannot exceed 100 characters.
TimeZoneID	Integer	This is a foreign key. The value in this column refers to the TimeZoneID in the Model.TimeZones table. Null values are allowed.

Model.DataSourceAttributes

The Model.DataSourceAttributes table stores information on the data source connection and the ArcestrA Data Adapter Service.

The following table explains the structure of the Model.DataSourceAttributes table:

Column	Data Type	Description
AttributeID	Integer	Contains the unique identifier for the attribute, which is automatically generated.
DataSourceID	Integer	Contains the unique identifier for the data source, defined in the DataSources table.
Name	Character	This is case insensitive. Null values are not allowed. You can use the following attributes for this column: <ul style="list-style-type: none"> • ConnectionInfo: Stores the connection information, specified in the Value column. This is required to connect to the specified database. • NodeName: Stores the name of the node, specified in the Value column, on which the ArcestrA Data Adapter Service runs. • Port: Stores information on the port, specified in the Value column, used to run the ArcestrA Data Adapter Service.
Value	Character	Provides the values for the above attributes. The length of the text cannot exceed 4,000 characters.

Model.DataItems

The Model.DataItems table stores the name and the ID of the data items.

The following table explains the structure of the Model.DataItems table:

Column	Data Type	Description
DataItemID	Integer	Contains the unique identifier for the data item, which is automatically generated.
Name	Character	Contains the name of the data item. For example, "Work Order". This is case insensitive. The length of the text cannot be more than 400 characters. Null values are not allowed.

Model.DataItemsAttributes

The Model.DataItemsAttributes table stores information on attributes related to the data items.

The following table explains the structure of the Model.DataItemsAttributes table:

Column	Data Type	Description
DataItemID	Integer	Contains the unique identifier for the data item.
AttributeID	Integer	Contains the unique identifier for the attribute. The following attribute can be used for the AttributeID column: <ul style="list-style-type: none"> RootDataItem: The AttributeID is "1". You can use this attribute to mark a data item as a root data item. The data items marked with this attribute ID are displayed in the OverView client initially.

Model.Relationships

The Model.Relationships table stores information about the relationship between the data items.

The following table explains the structure of the Model.Relationships table:

Column	Data Type	Description
RelationshipID	Integer	Contains the unique identifier for the relationship, which is automatically generated.
SubjectID	Integer	Contains the unique identifier for the source data item.
ObjectID	Integer	Contains the unique identifier for the target data item.
Predicate	Character	Defines the relationship between the source and the target data items. For example, the "Has" relationship in Work Order Has Jobs. In this case, "Work Order" and "Jobs" are the source and target data items, respectively.

Model.Rules

The Model.Rules table stores information about the rules for navigating a relationship.

The following table explains the structure of the Model.Rules table:

Column	Data Type	Description
RuleID	Integer	Contains the unique identifier for the rule, which is automatically generated.
RuleType	Character	Contains the type of the rule. This is case insensitive. The supported rule type is the following: <ul style="list-style-type: none"> Script: This marks the rule type for the given relationship as a script. The script defined for this rule in the RuleAttributes table is executed for the relationship.
RelationshipID	Integer	Contains the unique identifier for the relationship defined in the Model.Relationships table.

Model.RuleAttributes

The Model.RuleAttributes table stores information about the attributes of the rules.

The following table explains the structure of the Model.RuleAttributes table:

Column	Data Type	Description
AttributeID	Integer	Contains the unique identifier for the attribute, which is automatically generated.
RuleID	Integer	Contains the unique identifier for the rule defined in the Model.Rules table for a relationship.
Name	Character	<p>This is case insensitive. Null values are not allowed. The attributes that can be used for this column are as follows:</p> <ul style="list-style-type: none"> • DataSource: Stores the name of the data source, specified in the Value column. This value is configured in the DataSources table. • SQLScript: Stores the script, specified in the Value column. This script needs to be executed to retrieve the required data from the configured data source when the relationship is executed. • UseSourceTimeZone: Stores Boolean values, "true" or "false". The default value is "true". The value "true" means that the date/time values retrieved, using the script, are in the time zone configured for the data source and need to be converted to UTC. The value "false" indicates that the date/time values are in UTC and no further conversion is required. This is an optional attribute.
Value	Character	Provides the value for the above attributes. The length of the text cannot exceed 4,000 characters.

Model.DataItemColumns

The Model.DataItemColumns table stores information about the data item columns.

The following table explains the structure of the Model.DataItemColumns table:

Column	Data Type	Description
DataItemID	Integer	Contains the value that maps to the DataItemID of the Model.DataItem table.
ColumnName	nvarchar(256)	Contains the name of the column used in the Model.RuleAttributes table. The system does not validate whether the entered column name exists in the query.
ResourceStringID	Integer	Contains the value that maps to the StringID of the Model.ResourceString table.

Model.ResourceStrings

The Model.ResourceStrings table stores information about the resource strings.

The following table explains the structure of the Model.ResourceStrings table:

Column	Data Type	Description
StringID	Integer	Contains the value that maps to the ResourceStringID of the Model.DataItemColumns table.
DisplayString	nvarchar(256)	Contains the grid header name that is used in a client tool.

Model.TimeZones

The Model.TimeZones table stores information about the different time zones.

The following table explains the structure of the Model.TimeZones table:

Column	Data Type	Description
TimeZoneID	Integer	Contains the value that is automatically generated. This is a primary key.
Name	nvarchar(256)	Defines the time zone. It contains the name of the time zone, such as UTC, and Pacific Standard Time.

Model Store Views

There are two views defined in the Model Store database to facilitate finding, reading, and updating information:

- Model.DataSourceSummary
- Model.RuleSummary

The Model.DataSourceSummary view into the Model schema facilitates accessing the data sources by joining relevant column information from the Model.DataSources table with the Model.DataSourceAttributes and Model.TimeZones tables into a consolidated view.

The Model.RuleSummary view provides an easily accessible view of the Model rules. The Model.RuleSummary view helps significantly in finding and updating a relationship between two data items. Most often you will modify the data in the Script column. The data in the Script column is the actual query that defines how two Data Items are related.

Columns with an "(ro)" suffix are not editable.

Columns ending with "ID" should not be edited as they identify key columns from the related tables.

Chapter 4

Using Pre-defined Content

The information model is delivered with a pre-defined content script, which helps you generate the model with the relevant information from MES, Alarms, and Historian data sources. You can use this feature with minimum actions.

Refer to the Wonderware Information Server installation documentation for information on how to use the post-installation Configurator to import the pre-defined content. Alternatively, you can manually execute pre-defined content scripts by connecting to the correct information model database and then performing the following actions in the same order:

- 1 Run "1 MES Model.sql".
- 2 Run "2 MES Alarms Model.sql".
- 3 Run "3 Historian Model.sql".
- 4 Run "4 Historian Alarms Model.sql".
- 5 Configure the data sources in the Model.DataSourceAttributes table.

The sequence of executing the scripts is very important. If you need to run any script again, you must run all the scripts in the defined order. You can import pre-defined content scripts to the Model Store on both case-sensitive and case-insensitive SQL Servers.

Index

Symbols

@aaClientHostName parameter 25
@aaClientLCID parameter 26
@aaClientUserName parameter 25
@aaServerHostName parameter 24

A

about
 guide 5

C

column name captions
 customize 26
configure
 information model 7
customize
 column name captions 26

D

data items
 define 13
data sources
 define 10
define
 data items 13

data sources 10
relationships 15
root data items 14
rules 16
time zones 20
definitions
 Model Store table 51
 Model Store view 51
diagnostics
 information model 47
documentation conventions 5

E

error logging
 Information Model Configuration Tool 48
 Overview 47

G

guide
 about 5

I

information model
 configure 7
 diagnostics 47

- troubleshoot 47
- Information Model Configuration Tool 8
 - accessing 8
 - corrupt or missing files 48
 - error logging 48
 - reconfiguring 49
 - relationship list 9
 - security 43
- L
- locale 26
- log file 47, 48
- M
- Model Store
 - views 58
- Model Store table
 - Model.DataItemColumns 57
 - Model.DataItems 54
 - Model.DataItemsAttributes 54
 - Model.DataSourceAttributes 53
 - Model.DataSources 52
 - Model.Relationships 55
 - Model.ResourceStrings 57
 - Model.RuleAttributes 56
 - Model.Rules 55
 - Model.TimeZones 58
- Model Store view
 - definitions 51
- Model.DataItemColumns
 - Model Store table 57
- Model.DataItems
 - Model Store table 54
- Model.DataItemsAttributes
 - Model Store table 54
- Model.DataSourceAttributes
 - Model Store table 53
- Model.DataSources
 - Model Store table 52
- Model.DataSourceSummary view 58
- Model.Relationships
 - Model Store table 55
- Model.ResourceStrings
 - Model Store table 57
- Model.RuleAttributes
 - Model Store table 56
- Model.Rules
 - Model Store table 55
- Model.RuleSummary view 58
- Model.TimeZones
 - Model Store table 58
- N
- navigation parameters 21
- O
- OverView
 - error logging 47
- P
- parameters 21
- process events
 - well-known data items 35
- R
- relationships
 - adding 10
 - define 15
 - editing 10
 - managing 8
 - viewing 9
- root data items
 - define 14
- rules
 - define 16
- S
- security 43
- special parameters 24
- specify
 - well-known data items 28
- state transitions
 - well-known data items 36
- status indicators
 - well-known data items 28
- T
- table
 - Model.DataSources 11
- technical support 6
- time series
 - well-known data items 30
- time zones
 - configuring 20
 - defining 20
 - overriding 21

- supported 20
- troubleshoot
 - information model 47

V

- views 58

W

- well-known data items
 - process events 35
 - specify 28
 - state transitions 36
 - status indicators 28
 - time series 30

