Optegrity

Heater Tutorial

Version 5.1 Rev. 0

Optegrity Heater Tutorial Version 5.1 Rev. 0
November 2015

The information in this publication is subject to change without notice and does not represent a commitment by Gensym Corporation.

Although this software has been extensively tested, Gensym cannot guarantee error-free performance in all applications. Accordingly, use of the software is at the customer's sole risk.

Gensym Corporation
52 Second Avenue
Burlington, MA 01803 USA
Telephone: (781) 265-7100
Fax: (781) 265-7101

Part Number: DOC128-510

Contents

Preface

Describes this tutorial and the conventions that it uses.

About this Tutorial

This tutorial provides an example of an end-to-end Optegrity application for abnormal condition management. It uses the heater application as a tutorial to describe:

- Optegrity modules and architecture.
- Running applications.
- Data configuration and data validation.
- Data logging.
- Data replay.
- Message and alarm management.
- Event detection.
- Diagnostic reasoning using generic and specific fault models.

Audience

This tutorial is for process modelers who want to learn how to use Optegrity for abnormal condition management. The tutorial shows you how to run the heater application, including simulating data, and interacting with messages and events. It also describes how the application is configured, including external data configuration and validation, data replay, data logging, event detection, and diagnostic reasoning.

The tutorial does *not* describe how to build an Optegrity application, which includes creating process maps, event detection diagrams, and diagnostic models. For information on how to build an Optegrity application, consult the additional Optegrity books listed in Related Documentation.

Users of this tutorial should be familiar with process modeling, as well as with DCS process control systems such as OPC and PI.

Users of this tutorial should also be familiar with the general concepts of object-oriented programming, using classes, class hierarchies, and inheritance.

In general, process modelers do not need to be familiar with G2, Optegrity's underlying software environment. However, familiarity with G2 allows modelers to create custom procedures that the application can execute when various events occur.

Conventions

This guide uses the following typographic conventions and conventions for defining system procedures.

Typographic

Convention Examples	Description
g2-window, g2-window-1, ws-top-level, sys-mod	User-defined and system-defined G2 class names, instance names, workspace names, and module names
history-keeping-spec, temperature	User-defined and system-defined G2 attribute names
true, 1.234, ok, "Burlington, MA"	G2 attribute values and values specified or viewed through dialogs

Convention Examples	Description
Main Menu > Start KB Workspace > New Object create subworkspace Start Procedure	G2 menu choices and button labels
conclude that the x of y ...	Text of G2 procedures, methods, functions, formulas, and expressions
new-argument	User-specified values in syntax descriptions
text-string	Return values of G2 procedures and methods in syntax descriptions
File Name, OK, Apply, Cancel, General, Edit Scroll Area	GUIDE and native dialog fields, button labels, tabs, and titles
File > Save Properties	GMS and native menu choices
workspace	Glossary terms
`c:\Program Files\Gensym\`	Windows pathnames
`/usr/gensym/g2/kbs`	UNIX pathnames
`spreadsh.kb`	File names
`g2 -kb top.kb`	Operating system commands

`public void main()`
`gsi_start`
	Java, C and all other external code

Note Syntax conventions are fully described in the *G2 Reference Manual*.

Procedure Signatures

A procedure signature is a complete syntactic summary of a procedure or method. A procedure signature shows values supplied by the user in *italics*, and the value (if any) returned by the procedure <u>*underlined*</u>. Each value is followed by its type:

```
g2-clone-and-transfer-objects
    (list: class item-list, to-workspace: class kb-workspace,
     delta-x: integer, delta-y: integer)
    ->  transferred-items: g2-list
```

Related Documentation

Optegrity

- *Optegrity Heater Tutorial*
- *Optegrity User's Guide*
- *SymCure User's Guide*

G2 Core Technology

- *G2 Bundle Release Notes*
- *Getting Started with G2 Tutorials*
- *G2 Reference Manual*
- *G2 Language Reference Card*
- *G2 Developer's Guide*
- *G2 System Procedures Reference Manual*
- *G2 System Procedures Reference Card*
- *G2 Class Reference Manual*
- *Telewindows User's Guide*
- *G2 Gateway Bridge Developer's Guide*

G2 Utilities

- *G2 ProTools User's Guide*
- *G2 Foundation Resources User's Guide*
- *G2 Menu System User's Guide*
- *G2 XL Spreadsheet User's Guide*
- *G2 Dynamic Displays User's Guide*
- *G2 Developer's Interface User's Guide*
- *G2 OnLine Documentation Developer's Guide*
- *G2 OnLine Documentation User's Guide*
- *G2 GUIDE User's Guide*
- *G2 GUIDE/UIL Procedures Reference Manual*

G2 Developers' Utilities

- *Business Process Management System Users' Guide*
- *Business Rules Management System User's Guide*
- *G2 Reporting Engine User's Guide*
- *G2 Web User's Guide*
- *G2 Event and Data Processing User's Guide*
- *G2 Run-Time Library User's Guide*
- *G2 Event Manager User's Guide*
- *G2 Dialog Utility User's Guide*
- *G2 Data Source Manager User's Guide*
- *G2 Data Point Manager User's Guide*
- *G2 Engineering Unit Conversion User's Guide*
- *G2 Error Handling Foundation User's Guide*
- *G2 Relation Browser User's Guide*

Bridges and External Systems

- *G2 ActiveXLink User's Guide*
- *G2 CORBALink User's Guide*
- *G2 Database Bridge User's Guide*
- *G2-ODBC Bridge Release Notes*

- *G2-Oracle Bridge Release Notes*
- *G2-Sybase Bridge Release Notes*
- *G2 JMail Bridge User's Guide*
- *G2 Java Socket Manager User's Guide*
- *G2 JMSLink User's Guide*
- *G2 OPCLink User's Guide*
- *G2-PI Bridge User's Guide*
- *G2-SNMP Bridge User's Guide*
- *G2-HLA Bridge User's Guide*
- *G2 WebLink User's Guide*

G2 JavaLink

- *G2 JavaLink User's Guide*
- *G2 DownloadInterfaces User's Guide*
- *G2 Bean Builder User's Guide*

G2 Diagnostic Assistant

- *GDA User's Guide*
- *GDA Reference Manual*
- *GDA API Reference*

Customer Support Services

You can obtain help with this or any Gensym product from Gensym Customer Support. Help is available online, by telephone, by fax, and by email.

To obtain customer support online:

➔ Access G2 HelpLink at `www.gensym-support.com`.

You will be asked to log in to an existing account or create a new account if necessary. G2 HelpLink allows you to:

- Register your question with Customer Support by creating an Issue.
- Query, link to, and review existing issues.
- Share issues with other users in your group.
- Query for Bugs, Suggestions, and Resolutions.

To obtain customer support by telephone, fax, or email:

➔ Use the following numbers and addresses:

	Americas	Europe, Middle-East, Africa (EMEA)
Phone	(781) 265-7301	+31-71-5682622
Fax	(781) 265-7255	+31-71-5682621
Email	`service@gensym.com`	`service-ema@gensym.com`

Part I

Getting Started with the Optegrity Heater Tutorial

Chapter 1: Introduction

Provides a conceptual overview of the Optegrity Heater Tutorial, including its modules, process maps and domain objects, data flow and module integration, terminology, and architecture.

Chapter 2: Running the Optegrity Heater Tutorial

Describes how to run the Optegrity Heater Tutorial, view the process map, simulate external process variable data, and view the resulting operator messages.

1

Introduction

Provides a conceptual overview of the Optegrity Heater Tutorial, including its modules, process maps and domain objects, data flow and module integration, terminology, and architecture.

Introduction

The Optegrity Heater Tutorial provides an example of an Optegrity solution for abnormal condition management, which demonstrates integration, modeling, validation, and diagnosis. The application:

- Monitors and detects out-of-limit values for process variable data.
- Monitors and detects an abnormal condition for heaters, excess coking.
- Performs diagnosis to determine root causes of detected symptoms.

Process Maps and Domain Objects

A **process map** provides a visual representation of your equipment and process. Each external DCS tag variable is embedded in a **domain object**, which is a graphical representation of your process equipment, such as a heater or a sensor.

Optegrity provides built-in event detection for domain objects in a process map and allows you to add your own event detection. Optegrity provides two types of domain objects: sensors, such as temperature, pressure, and flow sensors, and process equipment, such as heaters, pumps, and vessels. All sensors detect High and Low Limit, Projected High and Low Limit, Noisy, Flatline, and Change events. Heaters, for example, detect Heat Release Projected High, Low Efficiency, and Efficiency Severe Change events.

You can generate customized events by creating event detection diagrams and perform diagnostic reasoning on those events by creating graphical fault models for your domain objects.

External and Internal Datapoints

You represent each DCS tag variable whose data you want to monitor as an **external datapoint**. External datapoints obtain data from the external DCS via an **interface**, which provides connectivity through a bridge. Optegrity supports connectivity with OPC and PI DCS systems, using G2 OPCLink or the G2-PI Bridge, which are included with Optegrity.

External datapoints provide data to domain objects within a process map via **internal datapoints**. Internal datapoints are properties of domain objects, which monitor external process data such as sensor process values, controller output values and setpoints, or any other process data that your application requires.

Internal datapoints provide the link between domain objects and their event detection diagrams. When an internal datapoint changes, Optegrity automatically notifies any event detection diagram that refers to the datapoint. An event detection diagram uses the raw data provided by internal datapoints to determine when to generate an event. These events, in turn, can trigger diagnostic reasoning and generate operator messages.

Optegrity Modules

The Optegrity Heater Tutorial integrates the following Optegrity modules to build a complete application for abnormal condition management:

This module...	Performs these tasks...
Optegrity	Provides domain object classes for building process maps that integrate with GDSM, GDPM, GEVM, GEDP, and SymCure.
G2 Data Point Management (GDPM)	• Configures external datapoints for each DCS tag variable, using CSV (comma-separated value) files. • Provides the link between external and internal datapoints within a process map. • Configures and performs data validation, data replay, and data logging, all using CSV files.
G2 Data Source Management (GDSM)	Configures the interface object that provides connectivity through a bridge.
G2 Event Management (GEVM)	Manages low-level notification and operator messages, and displays them in various types of message browsers.
G2 Event and Data Processing (GEDP)	Defines generic logic for monitoring internal datapoint values and generating low-level notifications, operator messages, and SymCure events that triggers diagnostic reasoning.
SymCure Also known as CDG (Causal Directed Graphs)	Defines generic diagnostic models for domain object classes, which reason about events to determine root causes and to take corrective actions.
G2 Engineering Unit Conversion (GEUC)	Specifies engineering units to use for entering and displaying values, as well as a large number of synonyms for those conversions in both the English and metric systems
Optegrity Events	Contains support for built-in event detection for domain objects.
Intelligent Sensor	Defines logic for monitoring sensor events.

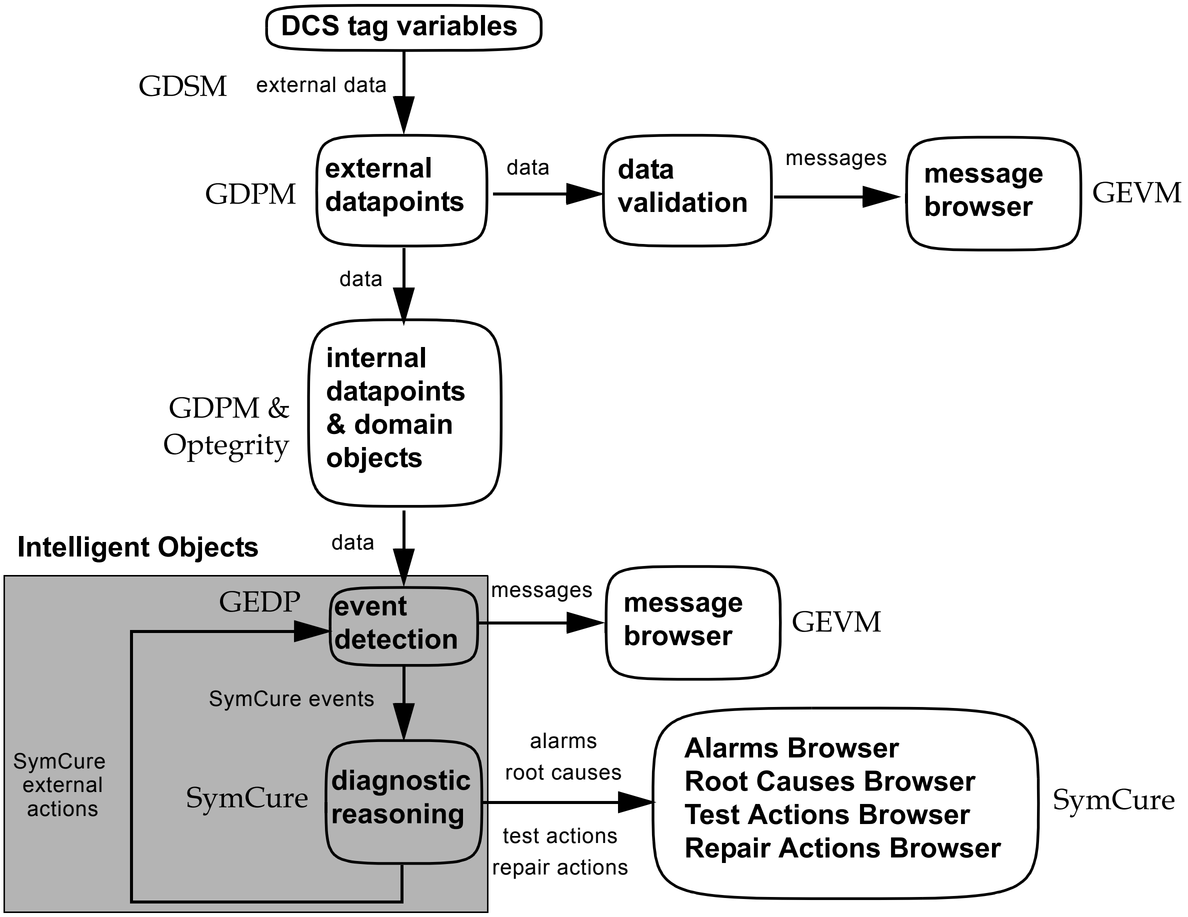
This module...	Performs these tasks...
Intelligent Controller	Defines logic for monitoring controller events.
Intelligent Heater	Defines logic for monitoring heater events.

Data Flow and Module Integration

The Optegrity Heater Tutorial integrates the Optegrity modules and manages data flow, as follows:

- The GDSM module provides communication between the external DCS tag variable and the external datapoints.
- The GDPM module creates and configures external datapoints to represent each DCS tag variable. GDPM performs low-level data validation on external datapoints. Messages that result from data validation appear in the pre-configured GEVM operator message browser.
- The GDPM module defines internal datapoints, which obtain their values from external datapoints. These internal datapoints are embedded in domain objects within a process map.
- The Intelligent Sensor module monitors sensor events such as increasing and decreasing flow, pressure, and temperature, which can be applied to internal datapoints.
- The Intelligent Controller module monitors controller events such as setpoint errors.
- The Intelligent Heater module monitors heater events such as efficiency severe change and low efficiency.
- GEDP monitors internal datapoint values and generates SymCure events for specific domain objects. GEDP also generates low-level notifications and operator messages, which appear in the GEVM message browser. GEDP templates apply generically to classes of domain objects.
- SymCure performs diagnostic reasoning on events for domain objects, based on specific fault models. SymCure fault models can generate correlated alarms and their underlying root causes, generate alarm and root cause messages, which appear in various browsers, and initiate external actions, including GEDP diagrams, to test and repair root causes. Similar to traditional methods in object-oriented programming, SymCure fault models are defined for a class of domain objects and can be inherited in the class hierarchy.

Here is a graphical representation of data flow and role of each module in the Optegrity Heater Tutorial:



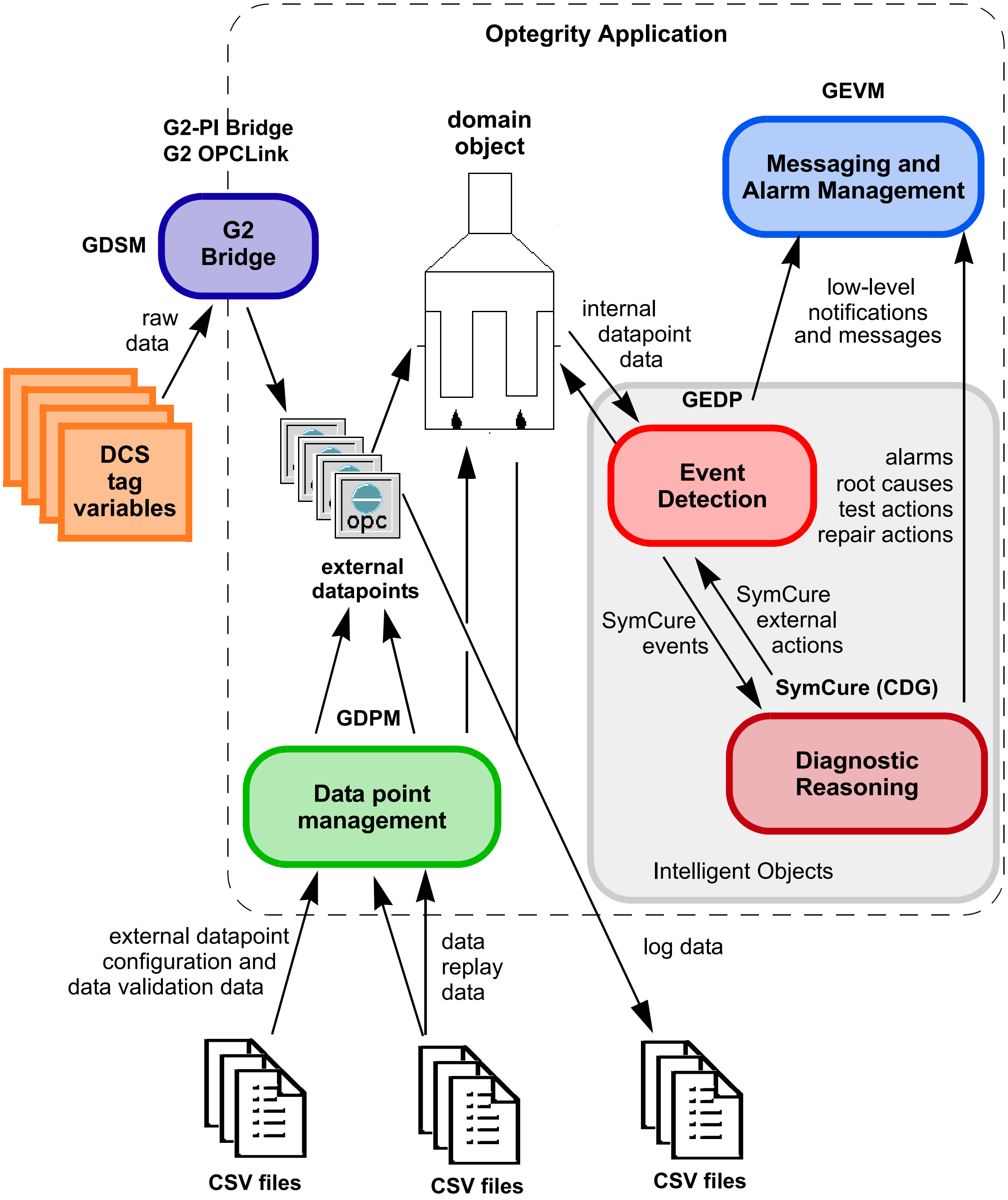
Terminology

This tutorial uses the following terminology in special ways:

Term	Definition
event	A SymCure alarm or any event that either causes an alarm or is an effect of an alarm. Event detection models generate events, and SymCure uses diagnostic models to reason about events.
alarm	A SymCure event that is either observed as true or predicted as true. Optegrity provides a diagnostic console browser that allows you to interact with alarms.
symptom	An alarm that is observed to be true.
fault	An event that is the root cause of an alarm. Optegrity provides a diagnostic console browser that allows you to interact with root causes.
external action	A SymCure test or repair action that is associated with an event and helps diagnose and fix root causes. Optegrity provides diagnostic console browsers for test and repair actions that allow you to interact with external actions.
low-level notification	An event that describes the state of an object, based on event detection. Optegrity provides a generic message browser for interacting with low-level notifications.
operator message	A message that describes an alarm condition for an object, based on event detection. Optegrity provides a generic message browser for interacting with operator messages.

Architecture

This figure shows the overall architecture of the Optegrity Heater Tutorial:



This table shows the progression of chapters in this tutorial, based on the architecture:

This chapter...	Explains how to...
Running the Optegrity Heater Tutorial	Run the Optegrity Heater Tutorial.
Viewing Data Configuration	• Configure the G2-PI Bridge and G2 OPCLink for obtaining raw data from external DCS systems. • Configure external datapoints. • Link external datapoints with internal datapoints, represented as domain objects. • Configure data validation.
Viewing Logging Configuration	Configure data logging.
Replaying Data	Replay data for simulation purposes.
Viewing Operator Messages	Interact with operator messages in the Message Browser.
Detecting Events	Detect events, based on internal datapoint values.
Interacting with SymCure Events	Interact with SymCure (CDG) events in the alarms and root causes browsers.
Performing External Actions	Create test actions and repair actions, and how to interact with those actions through browsers.
Viewing Generic Fault Models	Use SymCure generic fault models for diagnostic reasoning and root cause analysis.

Running the Optegrity Heater Tutorial

Describes how to run the Optegrity Heater Tutorial, view the process map, simulate external process variable data, and view the resulting operator messages.

Introduction

The Optegrity Heater Tutorial:

- Performs low-level data validation on external data.
- Processes the data and generates events.
- Performs diagnostic reasoning on those events.

The application consists of:

- A process map.
- Data configuration for providing external and simulated data to the model.
- Logging capabilities to track data values as they change.
- Data replay capabilities for replaying external and internal data.

- Generic event detection templates that are associated with domain object classes.
- Message browsers for viewing alarms, root causes, test actions, and repair actions.
- Generic SymCure fault models that define causal relationships between events for domain object classes.

The Optegrity Heater Tutorial shows how to integrate these Optegrity modules:

- GDPM for configuring internal and external datapoints from DCS tag variables.
- GEDP for processing data and detecting events.
- SymCure (CDG) for diagnosing events and taking actions.
- GEVM for browsing alarms and messages.

This chapter describes how to run the Optegrity Heater Tutorial, view the process map, simulate external datapoints to generate SymCure events, and view operator messages is the Message Browser.

Part II, Understanding the Optegrity Heater Tutorial describes how the Optegrity Heater Tutorial uses the various Optegrity modules to perform each modeling task.

Running the Optegrity Heater Tutorial

Optegrity is a client/server application. You start the Optegrity server from a batch file or from the Start menu. The Optegrity client is G2's Telewindows Next Generation.

To run the Optegrity Heater Tutorial, first, you start the Optegrity server with the tutorial model loaded, then you connect the client.

To run the Optegrity Heater Tutorial:

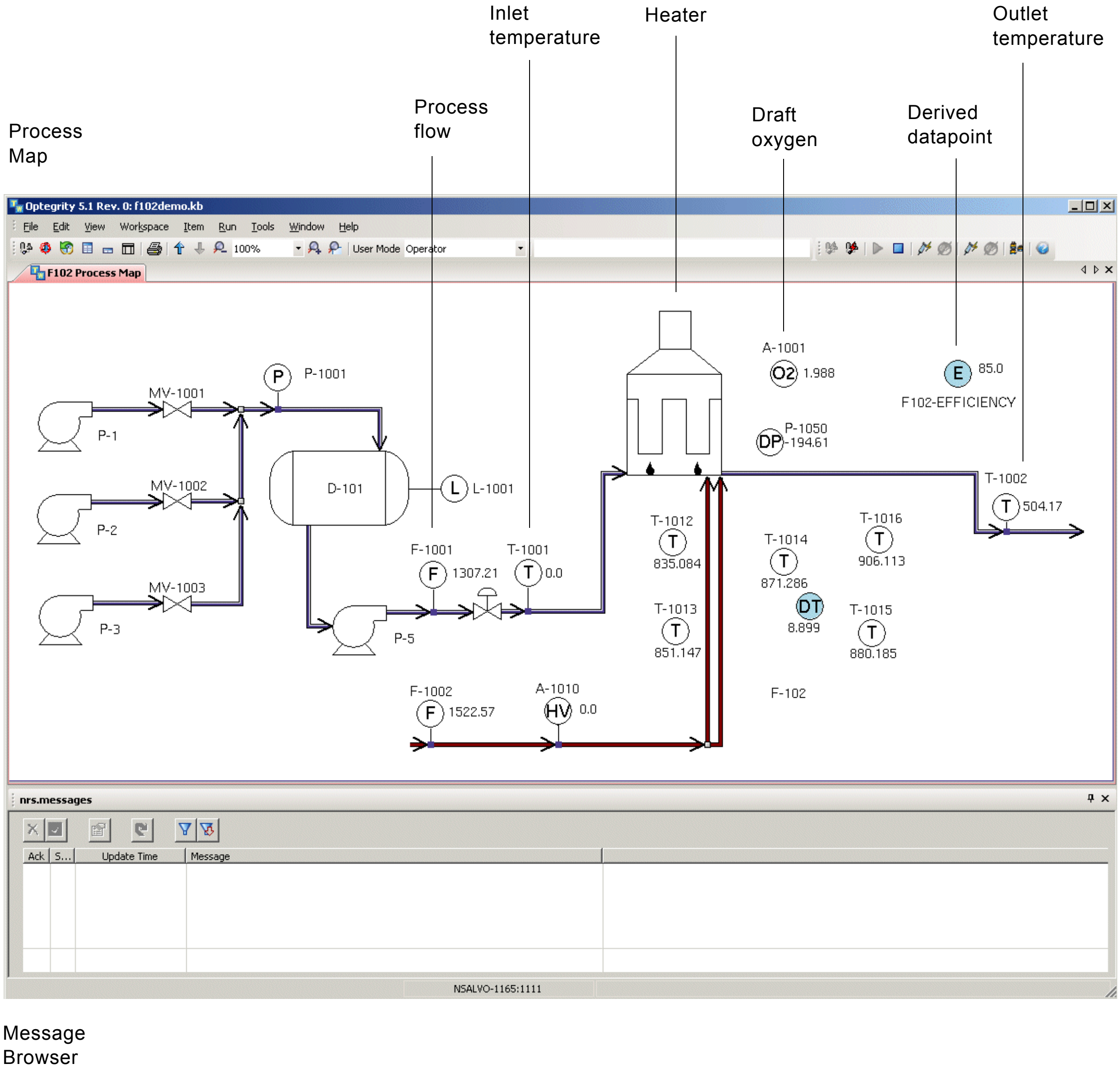
1 Choose Start > Programs > Gensym G2 2015 > Examples > G2 Optegrity > Heater Tutorial.

The Optegrity server is started with the tutorial loaded. You will see the G2 server icon in the system tray.

2 Choose Start > Programs > Gensym G2 2015 > Telewindows Next Generation.

You will see the Optegrity operator console, which is a restricted user interface for operators. It consists of a toolbar for interacting with a domain map, a default process map, and a message browser configured for the current user.

The process map provides a graphical representation of the equipment and process flow in the Heater Tutorial application, which consists of connected domain objects with internal datapoints. The focal point of the process map is the heater. The heater requires a number of related sensors, such as the process flow, inlet and outlet temperature, draft oxygen, and tube skin temperature sensors, and the tube skin delta temperature and heater efficiency derived sensors.



Each domain object and sensor in the application has knowledge about how to detect abnormal events and diagnose faults. For example, the heater has associated event detection diagrams that detect low efficiency, efficiency severe change, and heat release projected high events. A low efficiency event for the heater triggers SymCure diagnostic reasoning to determine the root cause of the low efficiency.

First, you will run the tutorial in Operator mode. Later, you will interact with the Optegrity Heater Tutorial in Modeler mode, which is the user mode you use for building domain models.

Simulating External Datapoints

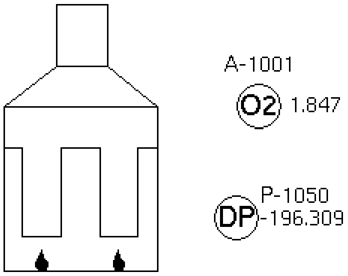
To run a simulation, you will generate values for each external datapoint in the process map. You can view values updating in the process map, in a summary of process map instruments, and in a graph.

To simulate external datapoints:

1 Click the Start Data Replay button (▶).

Clicking this button replays sample data for all external datapoints in the domain map from a CSV file.

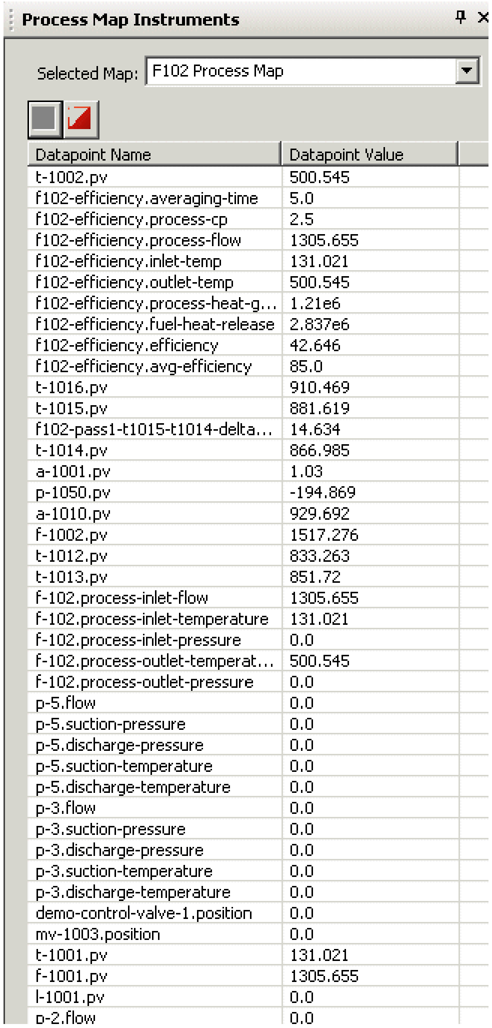
You will see values updating in the process map:



2 Click the Process Map Instruments button ( ) to display a list of all internal datapoints in the process map.

3 Click the Activate Update button (🟩) in the Process Map Instruments window to refresh the datapoint values.

Here is a partial list of the internal datapoints:



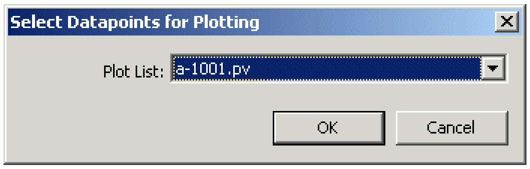
Selected Map: F102 Process Map

Datapoint Name	Datapoint Value
t-1002.pv	500.545
f102-efficiency.averaging-time	5.0
f102-efficiency.process-cp	2.5
f102-efficiency.process-flow	1305.655
f102-efficiency.inlet-temp	131.021
f102-efficiency.outlet-temp	500.545
f102-efficiency.process-heat-g...	1.21e6
f102-efficiency.fuel-heat-release	2.837e6
f102-efficiency.efficiency	42.646
f102-efficiency.avg-efficiency	85.0
t-1016.pv	910.469
t-1015.pv	881.619
f102-pass1-t1015-t1014-delta...	14.634
t-1014.pv	866.985
a-1001.pv	1.03
p-1050.pv	-194.869
a-1010.pv	929.692
f-1002.pv	1517.276
t-1012.pv	833.263
t-1013.pv	851.72
f-102.process-inlet-flow	1305.655
f-102.process-inlet-temperature	131.021
f-102.process-inlet-pressure	0.0
f-102.process-outlet-temperat...	500.545
f-102.process-outlet-pressure	0.0
p-5.flow	0.0
p-5.suction-pressure	0.0
p-5.discharge-pressure	0.0
p-5.suction-temperature	0.0
p-5.discharge-temperature	0.0
p-3.flow	0.0
p-3.suction-pressure	0.0
p-3.discharge-pressure	0.0
p-3.suction-temperature	0.0
p-3.discharge-temperature	0.0
demo-control-valve-1.position	0.0
mv-1003.position	0.0
t-1001.pv	131.021
f-1001.pv	1305.655
l-1001.pv	0.0
p-2.flow	0.0

4 Click the Deactivate Update button ( ) in the Process Map Instruments window to stop refreshing datapoint values, then close the window.

5 Click the Instruments Plot button ( ) to display a dialog for choosing the internal datapoint value to plot.

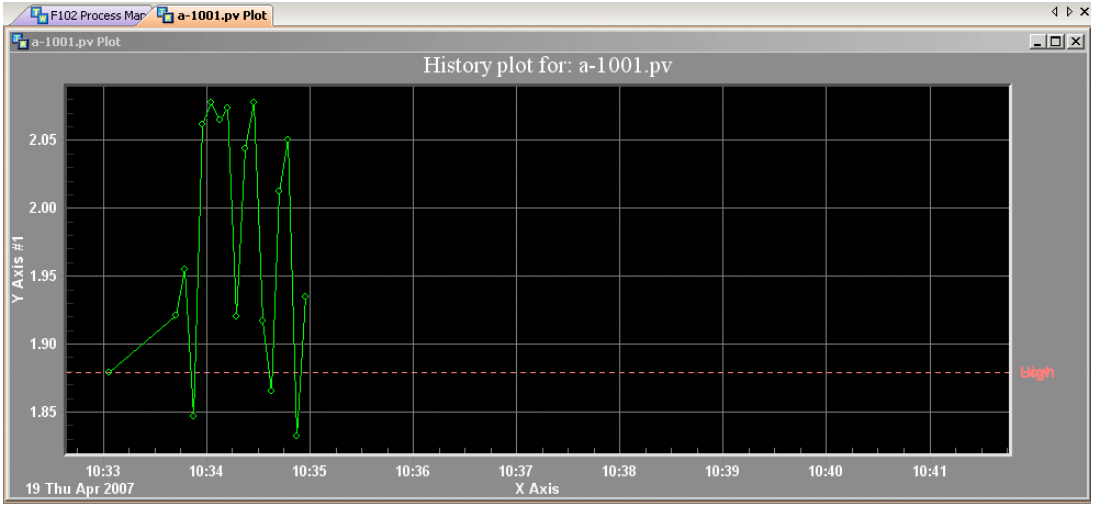
6 Choose the a-1001.pv internal datapoint, which is the Oxygen Analyzer for the heater, and click OK:



Select Datapoints for Plotting

Plot List: a-1001.pv

The plot appears in its own tab page:



7 Close the instrument plot window.

Now you will view and interact with the operator messages that are generated during the simulation.

Viewing Operator Messages

When you simulate data, event detection diagrams monitor data values and perform calculations to detect abnormal conditions for the various domain objects in the process map.

When abnormal conditions are detected, the event detection diagrams generate SymCure events. These events trigger SymCure diagnostic reasoning, which hypothesizes root causes for the detected alarms on specific domain objects. SymCure executes test actions to help diagnose faults and repair actions to help recover from faults. You view operator messages about alarms, root causes, test actions, and repair actions in the Message Browser.

An operator message provides system operators with real-time information about the status and availability of the underlying domain objects and the conclusions of the reasoning performed by the application. You can create event detection diagrams that generate operator messages, and you can configure operator messages for SymCure events.

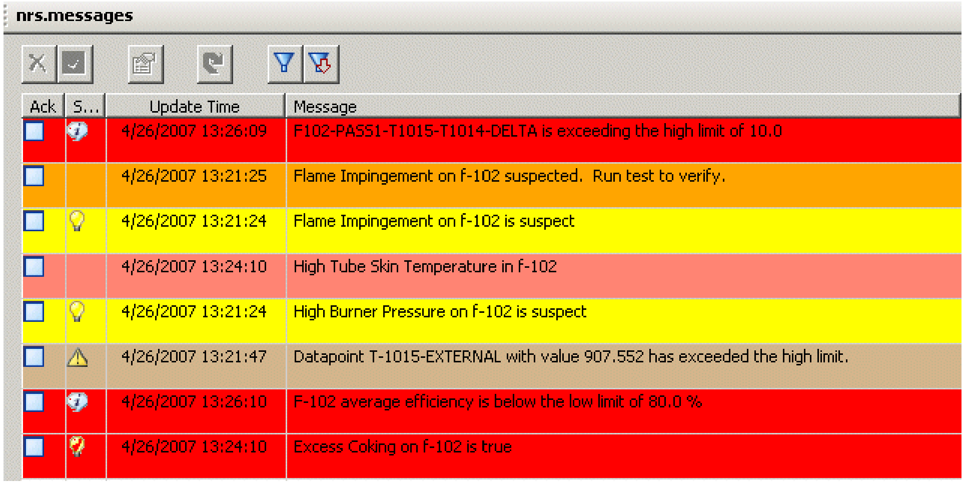
In addition to viewing messages about SymCure events in the Message Browser, you can interact with the events themselves in the SymCure diagnostic console browsers, which you will do later in this tutorial.

By default, each time a message occurs, the system beeps, which you control though your user preferences.

To view and interact with operator messages:

1 Click the Message Browser button ( ) or Mini Browser button ( ).

After running the simulation for several minutes, these messages appear in the mini browser:



nrs.messages

Ack	S...	Update Time	Message
□	ℹ	4/26/2007 13:26:09	F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0
□		4/26/2007 13:21:25	Flame Impingement on f-102 suspected. Run test to verify.
□	💡	4/26/2007 13:21:24	Flame Impingement on f-102 is suspect
□		4/26/2007 13:24:10	High Tube Skin Temperature in f-102
□	💡	4/26/2007 13:21:24	High Burner Pressure on f-102 is suspect
□	⚠	4/26/2007 13:21:47	Datapoint T-1015-EXTERNAL with value 907.552 has exceeded the high limit.
□	ℹ	4/26/2007 13:26:10	F-102 average efficiency is below the low limit of 80.0 %
□	💡	4/26/2007 13:24:10	Excess Coking on f-102 is true

You interact with messages by selecting the message in the browser and clicking a button in the toolbar. The toolbar buttons allow you to delete and acknowledge and delete messages, view details, go to the target and source of the message, filter messages, and lock the browser. If the message is generated by SymCure, you can also interact with the fault model through the message browser.

Messages are color coded according to priority, where red represents the highest priority and brown the lowest.

The red message regarding tube skin temperature is an operator message generated by an event detection diagram, which detects a high limit for one of the tube skin temperature sensors of the heater.

The red message regarding average efficiency of the heater is an operator message generated by an event detection diagram, which detects low efficiency for the efficiency derived sensor of the heater.

The red message regarding Excess Coking of the heater is a SymCure root cause message specified by the SymCure model that diagnoses the root cause of high tube skin temperature.

The two yellow messages are suspected root causes of the high tube skin temperature generated by the SymCure model that diagnoses high tube skin temperature events.

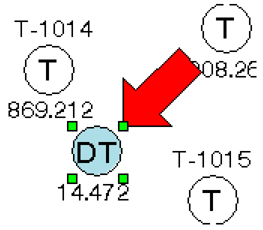
The pink message is a SymCure alarm regarding high tube skin temperature event.

The orange message is generated by the SymCure model that creates a test action for determining whether the Flame Impingement event is true.

The brown, low-priority message is a data validation message generated when the value of the external datapoint exceeds the configured limits.

2 Select the red message and click the Target button (◎).

An arrow appears next to the Tube Skin Delta Temperature derived sensor of the heater, indicating that the target of the message is this derived sensor:



You can click the red arrow to remove it.

3 Click the Stop Data Replay button (■).

You will learn more about how these messages were generated and how to interact with SymCure messages later in this tutorial.

Injecting Faults into the Simulation

You can run the simulation again, this time explicitly introducing faults into the data simulation. Before you run the simulation again, you must uninitialize, then initialize the process map. Initializing resets datapoint histories and clears all messages and diagnoses from the various browsers.

To initialize the process map:

➔ Click the Uninitialize Application button ( ), then click the Initialize Application button ( ).

To inject faults into the simulation:

1 Click the Start Data Replay button (▶).

2 Click the Inject High Delta T Fault button or the Inject Insufficient Air Fault button ( ).

3 Experiment with these faults to see the operator messages that occur and view the source of each operator message.

4 Stop the simulation, initialize the application, and close all tab pages.

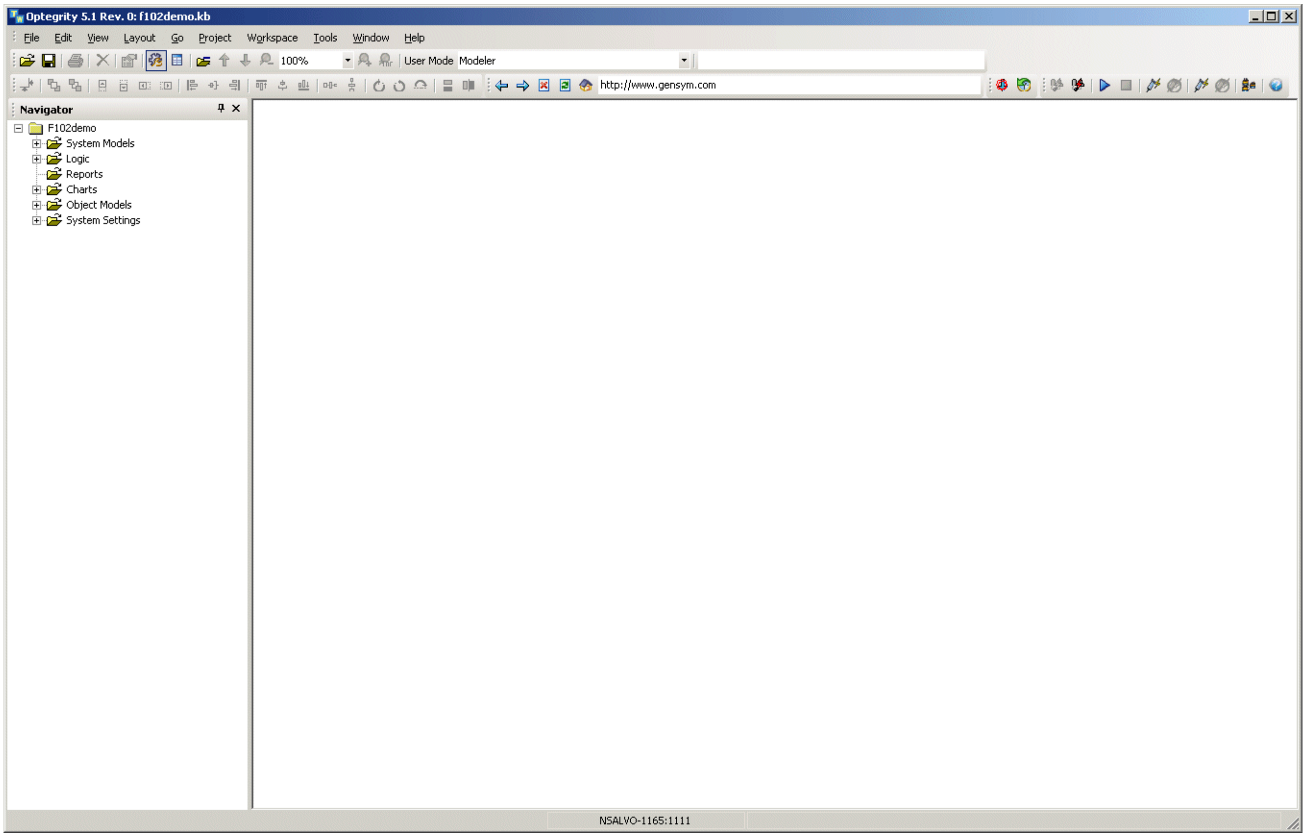
Switching to Modeler Mode

Now that you have run the simulation and viewed the resulting operator messages, you will explore how the model is configured in Modeler mode.

To switch to Modeler mode:

1 Choose Modeler from the dropdown list as the User Mode.

You will see the Optegrity modeling environment, which consists of a top-level menu bar and toolbars for building and running models:



Optegrity 5.1 Rev. 0: f102demo.kb

User Mode Modeler

http://www.gensym.com

You interact with models through the Project menu.

2 Choose Project > System Models > Manufacturing Processes > F102 Process Map to view the process map.

3 Choose Project > Initialize Application to initialize the process map.

Now that you have run a simulation and viewed the resulting operator messages, you will explore how the heater application is configured.

Part II

Understanding the Optegrity Heater Tutorial

Chapter 3: Viewing Data Configuration

Describes how the Optegrity Heater Tutorial configures the interface object that obtains external data through a bridge, configures external process variables, links those variables to internal process variables, and performs external datapoint validation.

Chapter 4: Viewing Logging Configuration

Describes how the Optegrity Heater Tutorial configures domain objects for logging.

Chapter 5: Replaying Data

Describes how to replay data from CSV files in the Optegrity Heater Tutorial.

Chapter 6: Viewing Operator Messages

Describes how to view operator messages in the default Message Browser for the Optegrity Heater Tutorial.

Chapter 7: Detecting Events

Describes how to view generic event detection templates and specific event detection diagrams for domain objects in the Optegrity Heater Tutorial.

Chapter 8: Interacting with SymCure Events

Describes how to view and interact with SymCure alarm and root cause events in the Alarm Browser and Root Cause Browser for the Optegrity Heater Tutorial.

Chapter 9: Performing External Actions

Describes how SymCure creates test actions and repair actions, and how you interact with those actions through browsers in the Optegrity Heater Tutorial.

Chapter 10: Viewing Generic Fault Models

Describes how to view the generic fault models for domain objects in the Optegrity Heater Tutorial.

Viewing Data Configuration

Describes how the Optegrity Heater Tutorial configures the interface object that obtains external data through a bridge, configures external process variables, links those variables to internal process variables, and performs external datapoint validation.

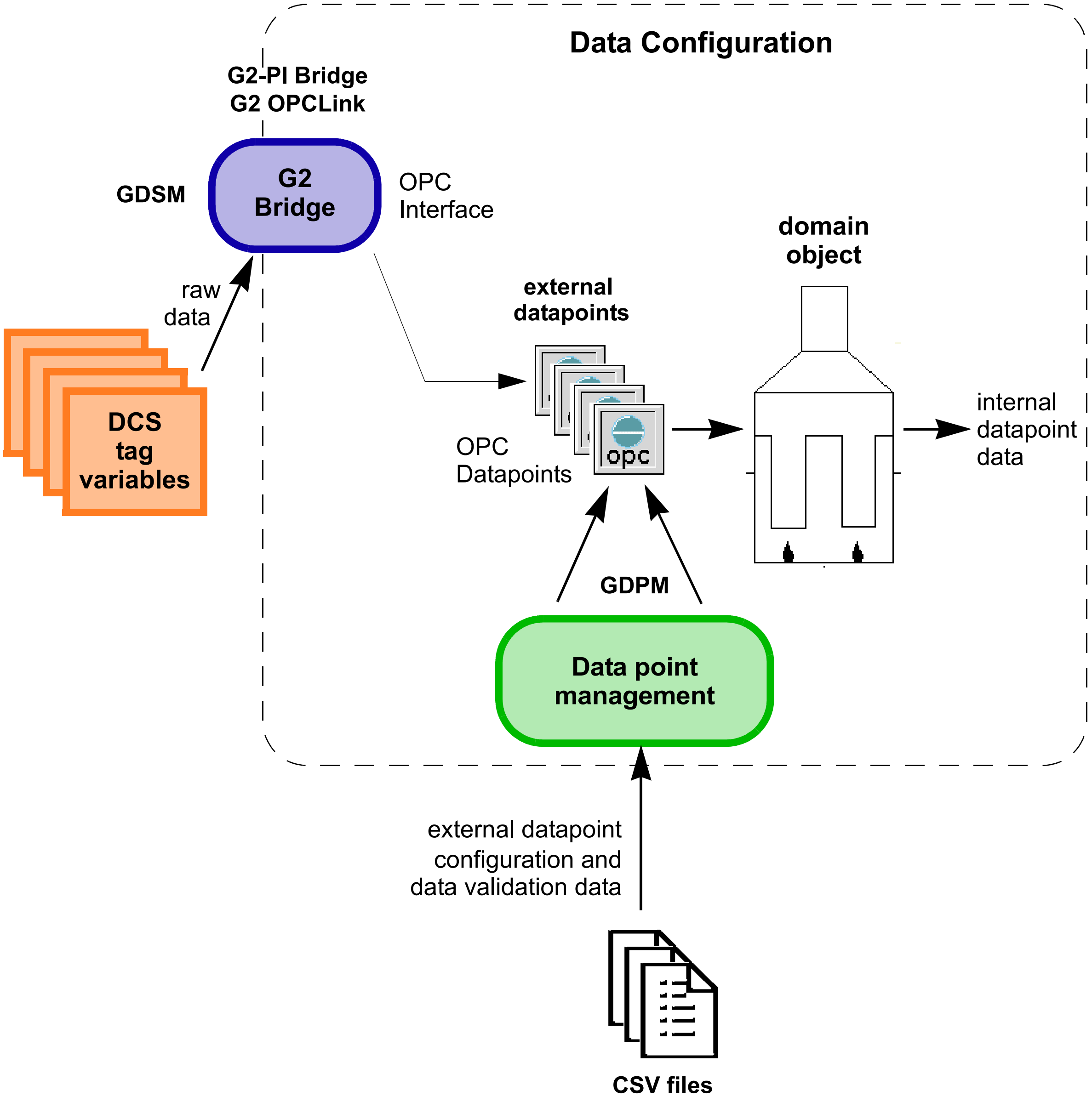
Introduction

Data configuration consists of:

- The bridge interface, which provides network communication between external systems and your application.
- External datapoints for each DCS tag variable, using CSV files. The CSV file links each external datapoint to an internal datapoint. In this application, each external datapoint is represented as a sensor object in a process map.
- Data validation limits, targets, and deviations for external datapoints, and data validation, which you configure in the same CSV file.

The G2 Data Source Management (GDSM) module handles network communication, and the G2 Data Point Management (GDPM) module handles datapoint configuration.

This figure shows the architecture of GDPM data configuration for the Optegrity Heater Tutorial. It includes the GDPM OPC Interface, External Datapoints container, and OPC Datapoints.



Object	Description
OPC Interface	Establishes the link between the OPC tag variables and the G2 OPCLink Bridge, which provides real-time data to the process map. Optegrity also supports the PI Interface, which communicates with PI tag variables through the G2-PI Bridge.
External Datapoints	Creates and configures external datapoints for each OPC tag variable from a CSV file. The external datapoints obtain external data through the specified OPC Interface.
OPC Datapoint	Represents an OPC tag variable. External datapoints provide communication between domain objects in a process map and the external OPC systems. External datapoints configure low-level data validation of external datapoints, using detected limits, targets, and rate of change.

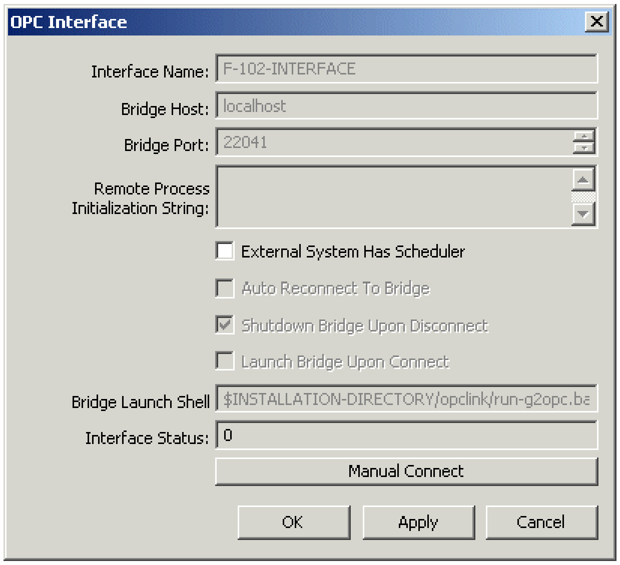
Viewing the OPC Interface

The OPC Interface provides communication between OPC tag variables and external datapoints, using the G2 OPCLink Bridge.

To view the OPC Interface:

➔ Choose Project > System Settings > Interfaces > OPC > F102-Interface to view its properties.

The properties dialog for a network interface allows you to connect to the specified bridge host and port. The properties are only available in Developer mode.



OPC Interface

Interface Name: F-102-INTERFACE

Bridge Host: localhost

Bridge Port: 22041

Remote Process Initialization String:

□ External System Has Scheduler

□ Auto Reconnect To Bridge

☑ Shutdown Bridge Upon Disconnect

□ Launch Bridge Upon Connect

Bridge Launch Shell $INSTALLATION-DIRECTORY/opclink/run-g2opc.ba

Interface Status: 0

If this application were communicating with an actual OPC system, you would configure the Host and TCP/IP Port of the computer on which the G2 bridge is running and click Connect to establish the connection. The other options in the dialog are advanced features.

For more information, see Configuring Network Interfaces in the *Optegrity User's Guide*.

Viewing the CSV File that Configures External Datapoints

The CSV file that configures external datapoints specifies the following information for each external datapoint, organized by function:

Column	Description
Datapoint	
Datapoint Name	The name of the external datapoint.
Datapoint Server Configuration	
Default Update Interval	How often to update data from the DCS system. The value is any time interval, such as 1 minute or 1 hour and 30 minutes.
Datapoint Tag Type	The type of datapoint the external datapoint represents. The options are: pv, sp, op, or mode. Pv represents the process value for a sensor or controller; sp represents the setpoint of a controller; op represents the controller output; and mode represents the controller mode. Once you configure the datapoint tag type, Optegrity shows you only datapoints of the required type when manually configuring the source datapoint of an internal datapoint.
Datapoint Type	The value type for the external datapoint. The options depend on the type of external system. For OPC variables, the options are: • float • integer • logical • text For PI variables, the options are: • real • integer • digital

Column	Description
Datapoint Units	The engineering units that the external datapoint uses, for example, C or deg C.
Related Process Map Datapoint Names	The name of the internal datapoint that gets its value from the external datapoint, expressed using dot notation.
Data Validation	
Min-Max Enabled High-High High Low-Low Low	Detected limit settings for data validation.
Target Enabled High-High High Low-Low Low Value	Detected target settings for data validation.
Rate Enabled High-High High Low-Low Low Interval	Detected deviation settings for data validation.
Animation	
Animation Enabled	Animation Enabled is currently not supported.
Animation Object Name	Animation Object Name is currently not supported.

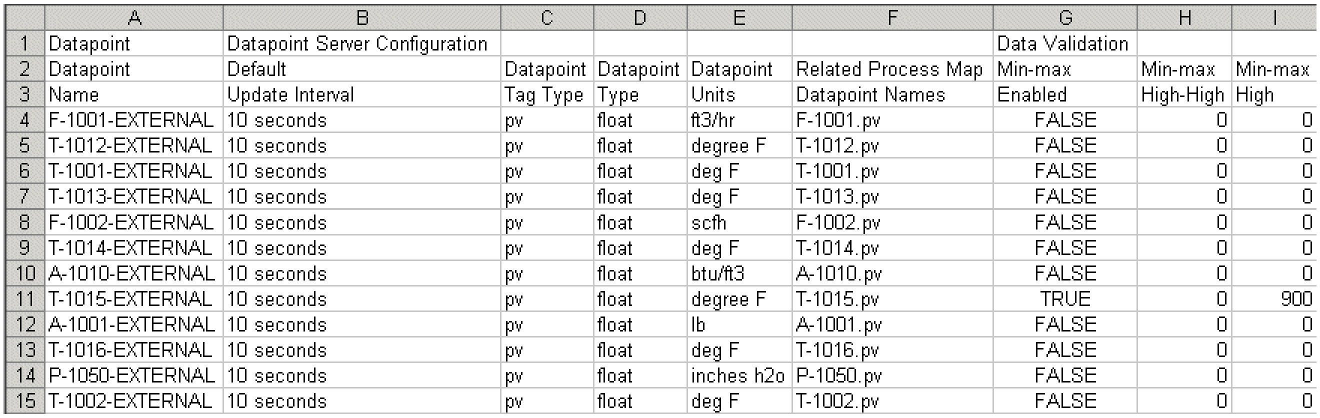
Column	Description
DCS Configuration	
OPC Item ID	The ID of the OPC tag variable from which the external variable gets its data.
OPC Access Path	The access path to the OPC tag variable in the external OPC system.
High Process Limit	The high limit for external data. Values above this limit are rejected.
Low Process Limit	The low limit for external data. Values below this limit are rejected.

To view the CSV file that configures external datapoints:

➔ Open the following file located in your installation directory:

```
\optegrity\data\
  F102-external-datapoint-configuration-OPC.csv
```

Each row defines a datapoint, and each column provides the configuration data. Here is part of the external datapoint configuration file:



	A	B	C	D	E	F	G	H	I
1	Datapoint	Datapoint Server Configuration					Data Validation		
2	Datapoint	Default	Datapoint	Datapoint	Datapoint	Related Process Map	Min-max	Min-max	Min-max
3	Name	Update Interval	Tag Type	Type	Units	Datapoint Names	Enabled	High-High	High
4	F-1001-EXTERNAL	10 seconds	pv	float	ft3/hr	F-1001.pv	FALSE	0	0
5	T-1012-EXTERNAL	10 seconds	pv	float	degree F	T-1012.pv	FALSE	0	0
6	T-1001-EXTERNAL	10 seconds	pv	float	deg F	T-1001.pv	FALSE	0	0
7	T-1013-EXTERNAL	10 seconds	pv	float	deg F	T-1013.pv	FALSE	0	0
8	F-1002-EXTERNAL	10 seconds	pv	float	scfh	F-1002.pv	FALSE	0	0
9	T-1014-EXTERNAL	10 seconds	pv	float	deg F	T-1014.pv	FALSE	0	0
10	A-1010-EXTERNAL	10 seconds	pv	float	btu/ft3	A-1010.pv	FALSE	0	0
11	T-1015-EXTERNAL	10 seconds	pv	float	degree F	T-1015.pv	TRUE	0	900
12	A-1001-EXTERNAL	10 seconds	pv	float	lb	A-1001.pv	FALSE	0	0
13	T-1016-EXTERNAL	10 seconds	pv	float	deg F	T-1016.pv	FALSE	0	0
14	P-1050-EXTERNAL	10 seconds	pv	float	inches h2o	P-1050.pv	FALSE	0	0
15	T-1002-EXTERNAL	10 seconds	pv	float	deg F	T-1002.pv	FALSE	0	0

The following table shows the data for the external datapoint in row 11, whose name is T-1015-EXTERNAL. The name of the related datapoint in the process map is T-1015.pv, which refers to the PV datapoint of the temperature sensor named T-1015. This technique of referring to internal datapoints of domain objects as *domain-object-name.datapoint-name* is known as **dot notation**.

Column	Value
Datapoint Name	*T-1015-EXTERNAL*
Default Update Interval	*10 seconds*
Datapoint Tag Type	*pv*
Datapoint Type	*float*
Datapoint Units	*degree F*
Related Process Map Datapoint Names	*T-1015.pv*
Min-Max	
Enabled	*TRUE*
High-High	*0*
High	*900*
Low-Low	*0*
Low	*0*
Target	
Enabled	*FALSE*
High-High	*0*
High	*0*
Low-Low	*0*
Low	*0*
Value	*0*
Rate	
Enabled	*FALSE*
High-High	*0*
High	*0*
Low-Low	*0*
Low	*0*
Interval	*0*
Animation Enabled	*FALSE (Currently not supported)*
Animation Object Name	*none (Currently not supported)*

Column	Value
OPC Item ID	*g2-v310a-ft/?DATA*
OPC Access Path	*v310a*
High Process Limit	*1000*
Low Process Limit	*850*

Viewing External Datapoints

External datapoints obtain their data from external data sources through an interface object. To create these datapoints, you must configure the External Datapoints container, then create the external datapoints from the specified CSV file. You can configure the external datapoints to propagate data when their values change (event-driven) or based on a schedule.

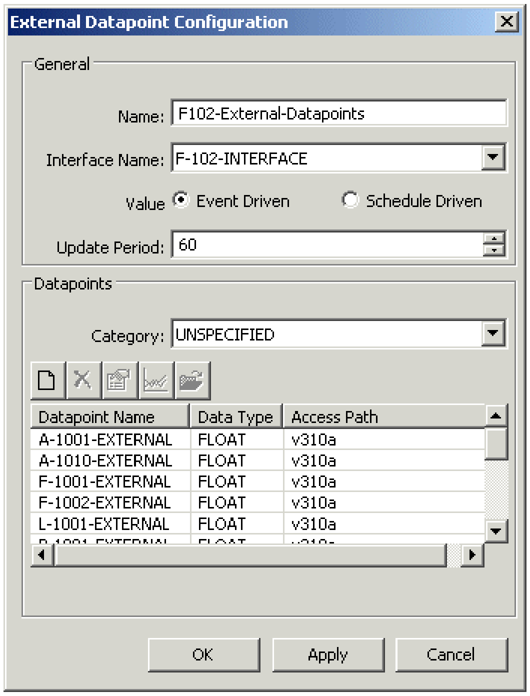
The External Datapoints properties dialog shows all external datapoints in the container. You can filter datapoints, based on a user-defined category. You can configure various properties for each external datapoint in the container.

To view the external datapoints:

1 Choose Project > System Settings > External Datapoints > F102-External-Datapoints to show its properties.

If no datapoints exist for the container, then external datapoints can be created manually or imported via the CSV file that contains external datapoint data and allows you to create the external datapoints automatically. Once the datapoints have been created, the dialog shows a listing of all external datapoints, their type, associated network interface, and connection status.

Here is the properties dialog for the f102-external-datapoints container:


External Datapoint Configuration

General

Name: F102-External-Datapoints

Interface Name: F-102-INTERFACE

Value ● Event Driven ○ Schedule Driven

Update Period: 60

Datapoints

Category: UNSPECIFIED

Datapoint Name	Data Type	Access Path
A-1001-EXTERNAL	FLOAT	v310a
A-1010-EXTERNAL	FLOAT	v310a
F-1001-EXTERNAL	FLOAT	v310a
F-1002-EXTERNAL	FLOAT	v310a
L-1001-EXTERNAL	FLOAT	v310a

Name is the name of the External Datapoints container.

Interface Name is the name of the OPC Interface object that obtains OPC tag data through the bridge.

Value Propagation determines how the external datapoint values are propagated. By default, datapoint values update when their values change in the external system, using event-driven propagation. You can also configure datapoint values to update based the Update Period, using schedule driven propagation.

Category allows you to filter external datapoints by their specified category.

The Datapoints section of the dialog allows you to add a new datapoint, delete a datapoint, display the properties of a datapoint, display a trend of a datapoint, and import datapoints from a CSV file.

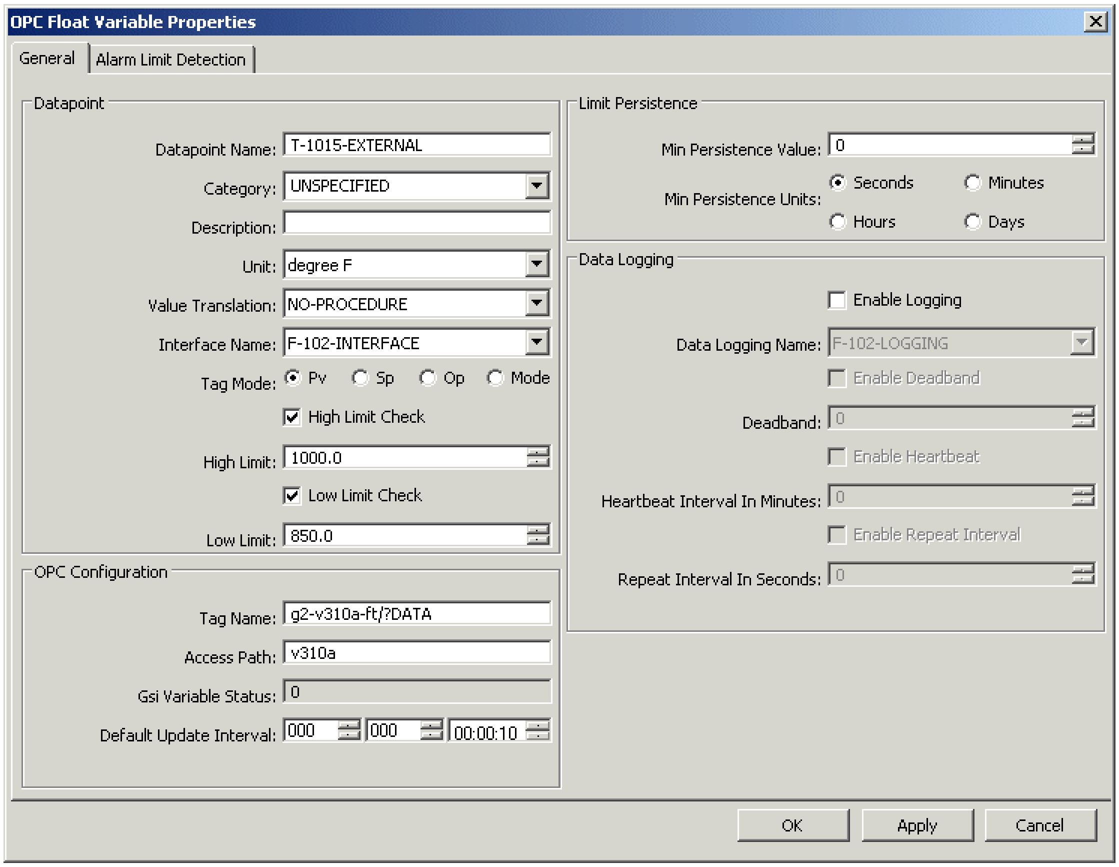
To delete a datapoint, display the properties of a datapoint, or display a trend of the datapoint, first choose the datapoint from the list and click the appropriate button. You can also create external datapoints by importing them from a CSV file.

2 Select the T-1015-EXTERNAL external datapoint from the list, then click the properties button.

Each external datapoint gets its data from a specific DCS tag variable in the external system via the specified interface object.

Each external datapoint provides data for an internal datapoint in the process map. For example, the external datapoint named T-1015-EXTERNAL provides data for the t-1015.pv internal datapoint in the process map. Not all external datapoints need to be associated with an internal datapoint. External datapoints can also be shared among multiple domain objects, in which case the external datapoint provides data to two internal datapoints.

Here is the General tab of the properties dialog for the T-1015-EXTERNAL external datapoint:



Datapoint

Datapoint Name: T-1015-EXTERNAL

Category: UNSPECIFIED

Description:

Unit: degree F

Value Translation: NO-PROCEDURE

Interface Name: F-102-INTERFACE

Tag Mode: ● Pv ○ Sp ○ Op ○ Mode

☑ High Limit Check

High Limit: 1000.0

☑ Low Limit Check

Low Limit: 850.0

OPC Configuration

Tag Name: g2-v310a-ft/?DATA

Access Path: v310a

Gsi Variable Status: 0

Default Update Interval: 000 000 00:00:10

Limit Persistence

Min Persistence Value: 0

Min Persistence Units: ● Seconds ○ Minutes ○ Hours ○ Days

Data Logging

□ Enable Logging

Data Logging Name: F-102-LOGGING

□ Enable Deadband

Deadband: 0

□ Enable Heartbeat

Heartbeat Interval In Minutes: 0

□ Enable Repeat Interval

Repeat Interval In Seconds: 0

Datapoint Name is the name for the external datapoint.

Category is a user-defined category for filtering external datapoints in the properties dialog for the External Datapoints container.

Description is a user-defined text for describing the external datapoint.

Unit is the engineering unit that the external datapoint uses, for example, C or deg C.

Value Translation is the name of a user-defined procedure for translating the external datapoint value.

Interface Name is the name of the OPC Interface object that provides OPC tag data to the external datapoint.

Tag Mode indicates the type of datapoint for the process data, as configured in the CSV file. The options are pv, sp, op, and mode, which represent the process value, setpoint, controller output, and controller mode of a DCS tag variable, respectively.

High Limit and Low Limit are limits for excluding the external datapoint value. Values above the high limit or below the low limit are not propagated when High Limit Check and Low Limit Check are enabled.

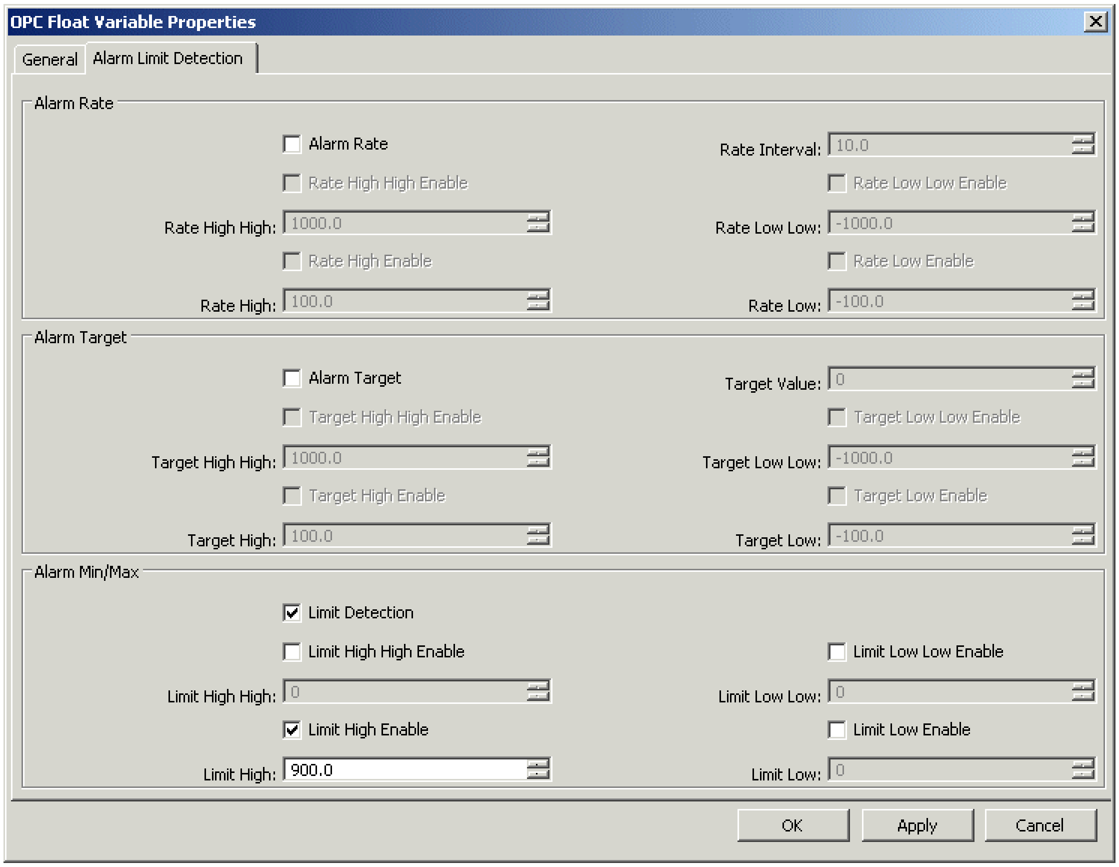
Tag Name and Access Path are OPC configuration values, specified in the CSV file. Default Update Interval indicates how often to update the OPC configuration data. GSI Variable Status is a read-only value that reports the status of the external DCS tag variable.

Min Persistence Value is the amount of time to display operator messages when an alarm limit is detected, for example, 3 hours. You also configure the Min Persistence Units. By default, they never expire.

You can also configure data logging to log external datapoint values as they are sent from the DCS system. Logging is described later in this tutorial.

External datapoints can define limits, targets, and/or deviations, as well as data validation. You often configure data validation for controllers, which define process data and setpoints. You configure data validation on the Alarm Limit Detection tab.

3 Click the Alarm Limit Detection tab to show the alarm limits for the external datapoint:



Alarm Rate

□ Alarm Rate

Rate Interval: 10.0

□ Rate High High Enable

□ Rate Low Low Enable

Rate High High: 1000.0

Rate Low Low: -1000.0

□ Rate High Enable

□ Rate Low Enable

Rate High: 100.0

Rate Low: -100.0

Alarm Target

□ Alarm Target

Target Value: 0

□ Target High High Enable

□ Target Low Low Enable

Target High High: 1000.0

Target Low Low: -1000.0

□ Target High Enable

□ Target Low Enable

Target High: 100.0

Target Low: -100.0

Alarm Min/Max

☑ Limit Detection

□ Limit High High Enable

□ Limit Low Low Enable

Limit High High: 0

Limit Low Low: 0

☑ Limit High Enable

□ Limit Low Enable

Limit High: 900.0

Limit Low: 0

Alarm Rate, Alarm Target, and Alarm Min/Max configure data validation for the external datapoint. These values correspond with the values you saw earlier in the CSV file for this external datapoint. When an alarm limit is detected, Optegrity sends a message to the Message Browser, which persists for the amount of time specified by the Limit Persistence attributes.

Viewing Internal Datapoints

Each internal datapoint has an associated source datapoint, which is the external datapoint that provides its data. You can configure the datapoint to keep a history, then plot the history in a trend chart. Similar to external datapoints, internal datapoints can have a logging specification, which you will learn about later in this tutorial.

A domain object's properties dialog shows all built-in and user-defined internal datapoints. You can filter the datapoints, based on a user-defined category. You can configure the properties and display a trend chart of historical data for each internal datapoint displayed in the domain object's properties dialog.

The Optegrity Heater Tutorial contains the following sensors, which represent the various related sensors of the heater:

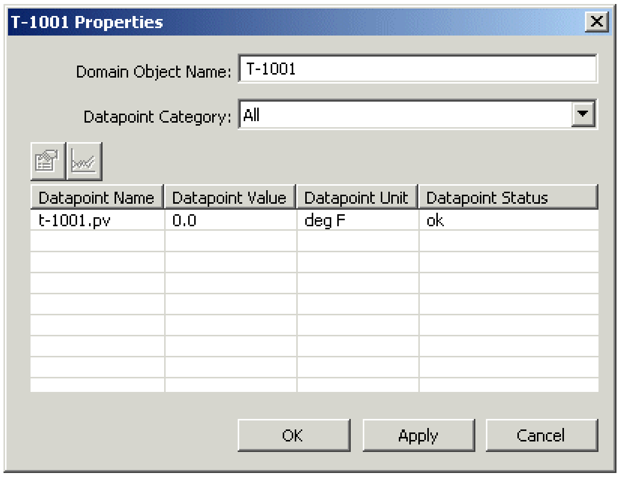
- T-1001 represents the inlet temperature sensor.
- T-1002 represents the outlet temperature sensor.
- F-1001 represents the inlet feed flow sensor.
- F-1002 represents the inlet fuel flow sensor.
- A-1001 represents the O2 sensor internal datapoint.
- P-1050 represents the DP sensor internal datapoint.
- T-1012, T-1014, T-1016, T-1013, and T-1015 represent the tube skin temperature sensors.
- F102-Pass1-T1015-T1014-Delta represents the derived delta temperature sensor.
- F102-Efficiency represents the derived efficiency sensor.
- A-1010-2nd-Sensor represents the derived 2nd sensor delta sensor.

To view internal datapoints:

1 On the process map, choose Properties on the T-1001 domain object.

The properties dialog for the sensor shows the name of the domain object and lists all its internal datapoints, using dot notation. Each datapoint shows its current value, units of measurement, and status. The dialog has a trend chart for plotting datapoint histories.

Here is the properties dialog for the T-1001 sensor:



Domain Object Name: T-1001

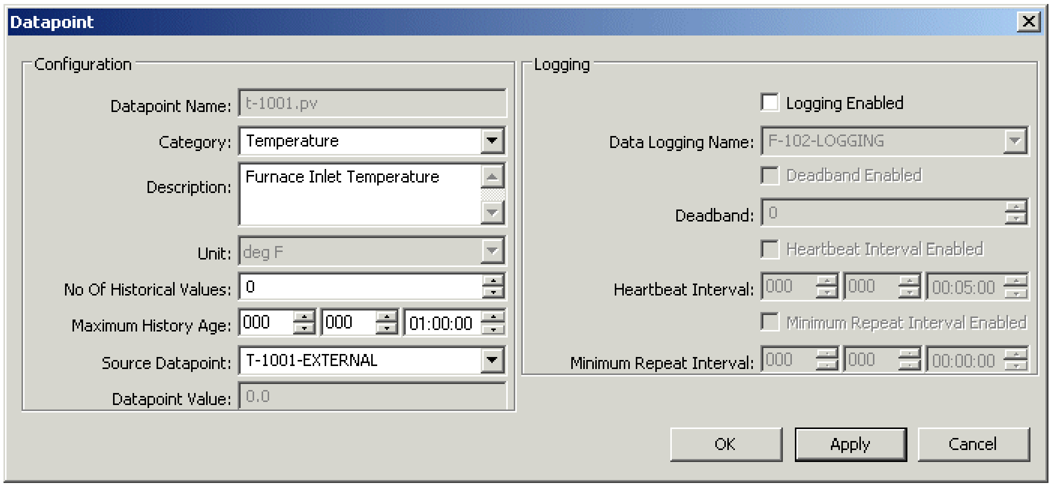
Datapoint Category: All

Datapoint Name	Datapoint Value	Datapoint Unit	Datapoint Status
t-1001.pv	0.0	deg F	ok

You can filter internal datapoints by choosing a category, which you can configure for each internal datapoint.

2 Click the t-1001.pv internal datapoint from the list, then click the properties button.

Here is the properties dialog for the t-1001.pv internal datapoint:



Configuration

Datapoint Name: t-1001.pv

Category: Temperature

Description: Furnace Inlet Temperature

Unit: deg F

No Of Historical Values: 0

Maximum History Age: 000 000 01:00:00

Source Datapoint: T-1001-EXTERNAL

Datapoint Value: 0.0

Logging

☐ Logging Enabled

Data Logging Name: F-102-LOGGING

☐ Deadband Enabled

Deadband: 0

☐ Heartbeat Interval Enabled

Heartbeat Interval: 000 000 00:05:00

☐ Minimum Repeat Interval Enabled

Minimum Repeat Interval: 000 000 00:00:00

Datapoint Name is the name of the internal datapoint embedded in the domain object. The datapoint name is derived from the domain object and the internal datapoint and, therefore, is read-only.

Category is a user-defined category for filtering internal datapoints in the properties dialog for the domain object.

Description is a user-defined description of the datapoint.

Unit is the engineering unit that the internal datapoint uses, for example, C or deg C.

Nb of Historical Values is the number of datapoint values to keep when plotting the history in a trend chart, for example, 100 points.

Maximum History Age is a time interval that specifies the amount of historical data to keep when plotting the history in a trend chart, for example, 10 minutes.

To plot data histories, you configure either Nb of Historical Values or Maximum History Age for the internal datapoint.

Source Datapoint defines the external datapoint that provides data for this internal datapoint. You can configure the source datapoint manually by using the combo box and choosing from the list of existing external datapoints. The source datapoint for the t-1001.pv internal datapoint is T-1001-EXTERNAL.

Datapoint Value is read-only when the Source Datapoint is configured; otherwise, you can configure the datapoint value manually.

You will learn about the logging attributes later in this tutorial.

Configuring Related Sensors for the Heater

When the intelligent object libraries are loaded, each piece of equipment in a process map can have a variety of related sensors that are used for event detection. For example, you can configure the following related sensors for a heater: process flow, inlet temperature and pressure, outlet temperature and pressure, draft oxygen, tube skin temperatures, and tube skin delta temperature, among others. The event detection diagrams use these related sensors to detect efficiency and heat release events.

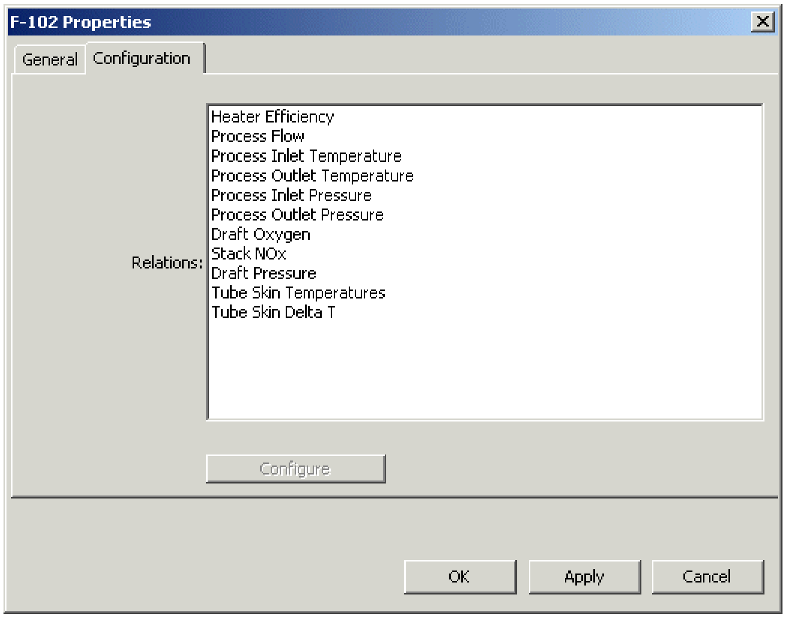
You configure related sensors of process equipment on the Configuration tab of the properties dialog. Only process equipment has related sensors.

To configure related sensors for the heater:

1 Show the properties dialog for the F-102 heater.

2 Click the Configuration tab.

Here is the list of related sensors that you can configure for a heater:

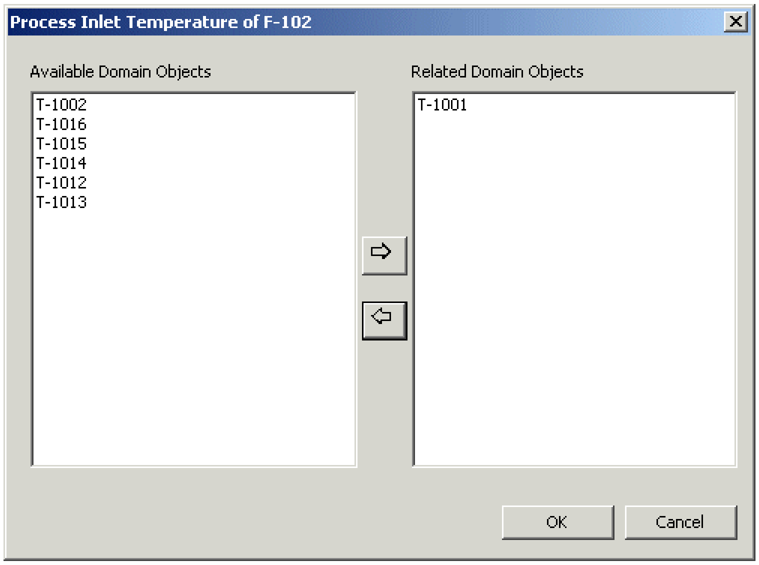


Relations:

Heater Efficiency
Process Flow
Process Inlet Temperature
Process Outlet Temperature
Process Inlet Pressure
Process Outlet Pressure
Draft Oxygen
Stack NOx
Draft Pressure
Tube Skin Temperatures
Tube Skin Delta T

3 Select Process Inlet Temperature from the Related Domain Objects list, then click the Configure button.

The Available Domain Objects list includes all sensors of the required type in the process map, that is, all temperature sensors. The Related Domain Objects list shows all domain objects that are currently assigned as the related sensors. In this example, the T-1001 sensor is assigned as the Process Inlet Temperature of the F-102 heater:



Process Inlet Temperature of F-102

Available Domain Objects

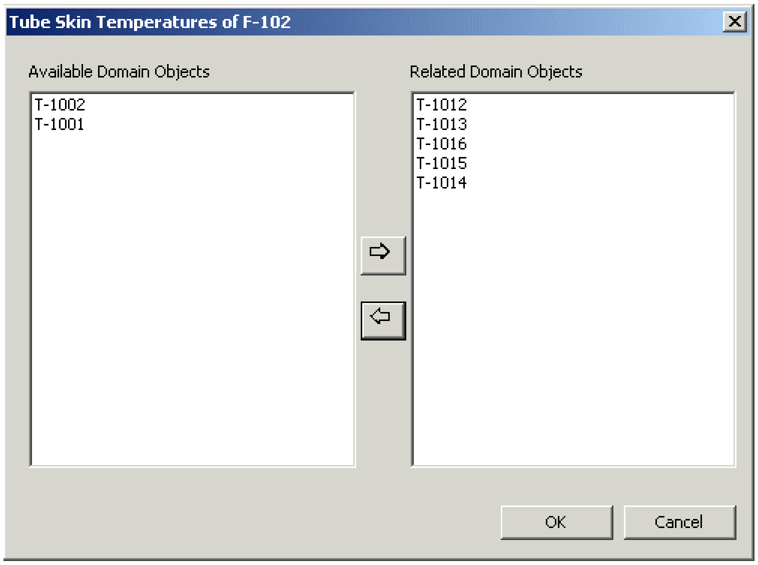
T-1002
T-1016
T-1015
T-1014
T-1012
T-1013

Related Domain Objects

T-1001

4 Now configure the Tube Skin Temperatures related sensors.

The Tube Skin Temperatures relation has five related sensors:



Tube Skin Temperatures of F-102

Available Domain Objects

T-1002
T-1001

Related Domain Objects

T-1012
T-1013
T-1016
T-1015
T-1014

Configuring Built-in Event Detection

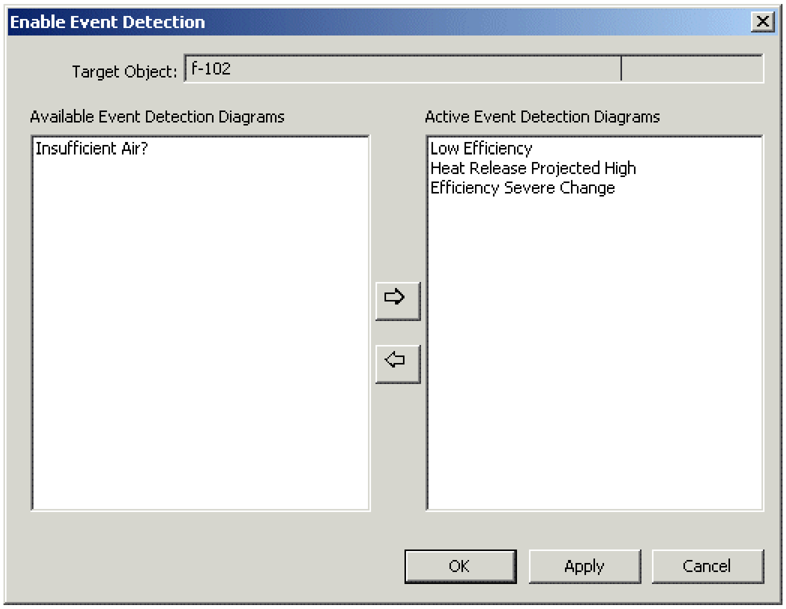
Optegrity provides built-in event detection for various types of process equipment and instruments. For process equipment, event detection rely on the related sensors that you have configured for the equipment. For instruments, event detection relies on the source datapoint that you have configured for the sensor.

Assuming you have the appropriate intelligent object libraries loaded, any subclass of opt-equipment or opt-sensor inherits the built-in generic event detection templates.

To configure built-in event detection for a domain object:

1 Choose Enable Dataflow Event Detections on the F-102 heater.

Here is the list of available event detection diagrams for a heater, three of which are active for the F-102 heater:



Target Object: f-102

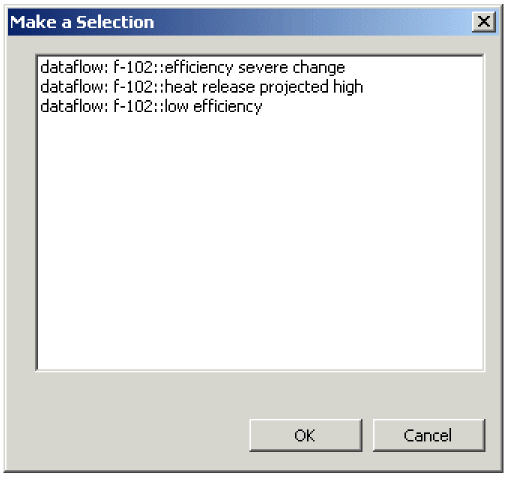
Available Event Detection Diagrams: Insufficient Air?

Active Event Detection Diagrams: Low Efficiency, Heat Release Projected High, Efficiency Severe Change

Once events have been activated for a domain object, you can view and configure the event detection logic for the specific domain object.

2 Choose Show Logic on the F-102 heater.

This dialog shows a list of all event detection diagrams that are active for the domain object:

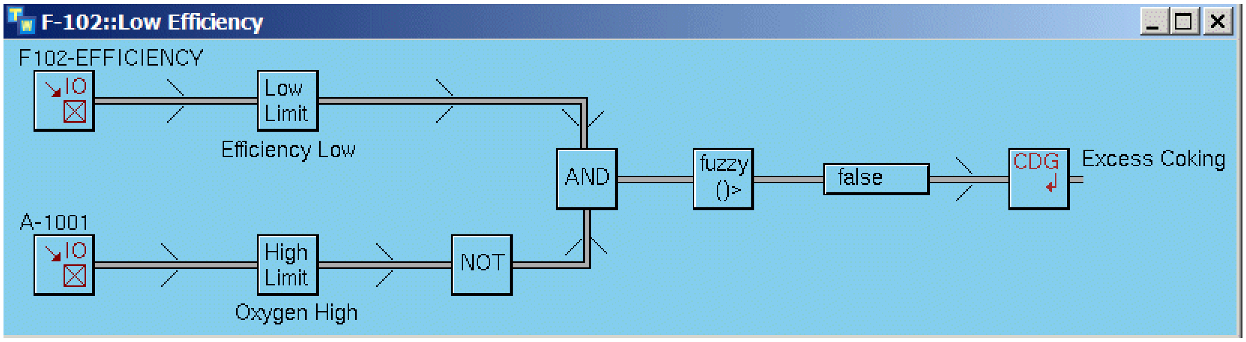


dataflow: f-102::efficiency severe change
dataflow: f-102::heat release projected high
dataflow: f-102::low efficiency

3 Select Low Efficiency and click OK.

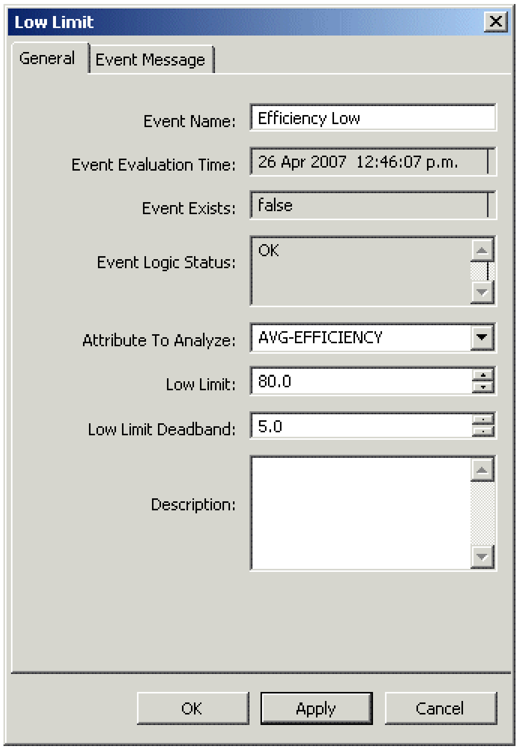
The specific event detection diagram for the F-102 heater appears. This specific diagram is generated from the OPT-HEATER::Low Efficiency generic template when the F-102 heater is initialized. Each heater that enables this event detection diagram has an associated Low Efficiency specific diagram. Optegrity allows you to modify any specific diagram independently of the generic template to customize your event detection logic for individual domain objects, as needed.

Here is the specific Low Efficiency event detection diagram for the F-102 heater. This diagram detects low heater efficiency and stack oxygen that is not high and generates a SymCure Excess Coking event when both conditions are true.



For the Low Efficiency event, you can configure the Low Limit block and the High Limit block. The Low Limit block monitors the value of the F102-EFFICIENCY derived sensor.

4 Show the properties of the Low Limit block:



Event Name: Efficiency Low

Event Evaluation Time: 26 Apr 2007 12:46:07 p.m.

Event Exists: false

Event Logic Status: OK

Attribute To Analyze: AVG-EFFICIENCY

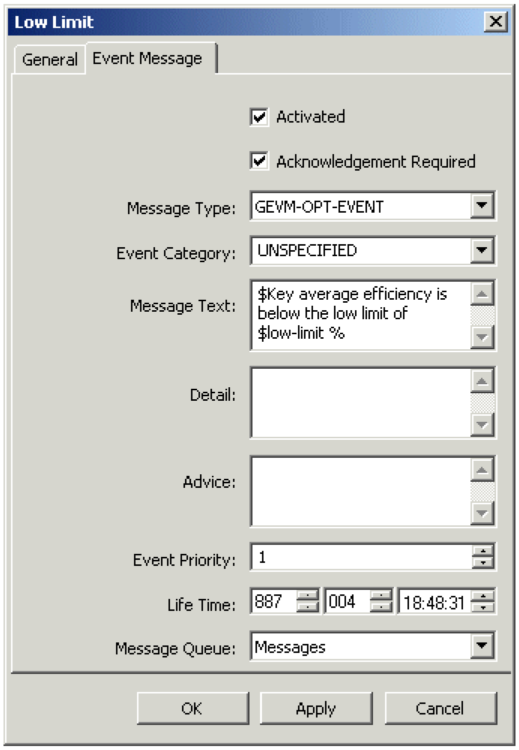
Low Limit: 80.0

Low Limit Deadband: 5.0

Description:

The General tab allows you to configure parameters related to how the block monitors the value. For example, the Attribute to Analyze is avg-efficiency, which is an internal datapoint defined by the F102-EFFICIENCY sensor, and the Low Limit is set to 80. The Low Limit block becomes true if the value of the avg-efficiency datapoint goes below 80.

5 Click the Event Message tab:



☑ Activated

☑ Acknowledgement Required

Message Type: GEVM-OPT-EVENT

Event Category: UNSPECIFIED

Message Text: $Key average efficiency is below the low limit of $low-limit %

Detail:

Advice:

Event Priority: 1

Life Time: 887 004 18:48:31

Message Queue: Messages

The Event Message tab allows you to create a specific message to be generated when the event becomes true. If you do not want the message to be generated when the block becomes true, disable the Activated check box. The Message Text uses special keywords preceded by a dollar sign ($) as text substitutions. See the *Optegrity User's Guide* for details.

Now that you have seen how the Optegrity Heater Tutorial configures external and internal datapoints to provide data to domain objects in a process map, and how to configure events, you will explore how to log internal and external datapoint values.

Viewing Logging Configuration

Describes how the Optegrity Heater Tutorial configures domain objects for logging.

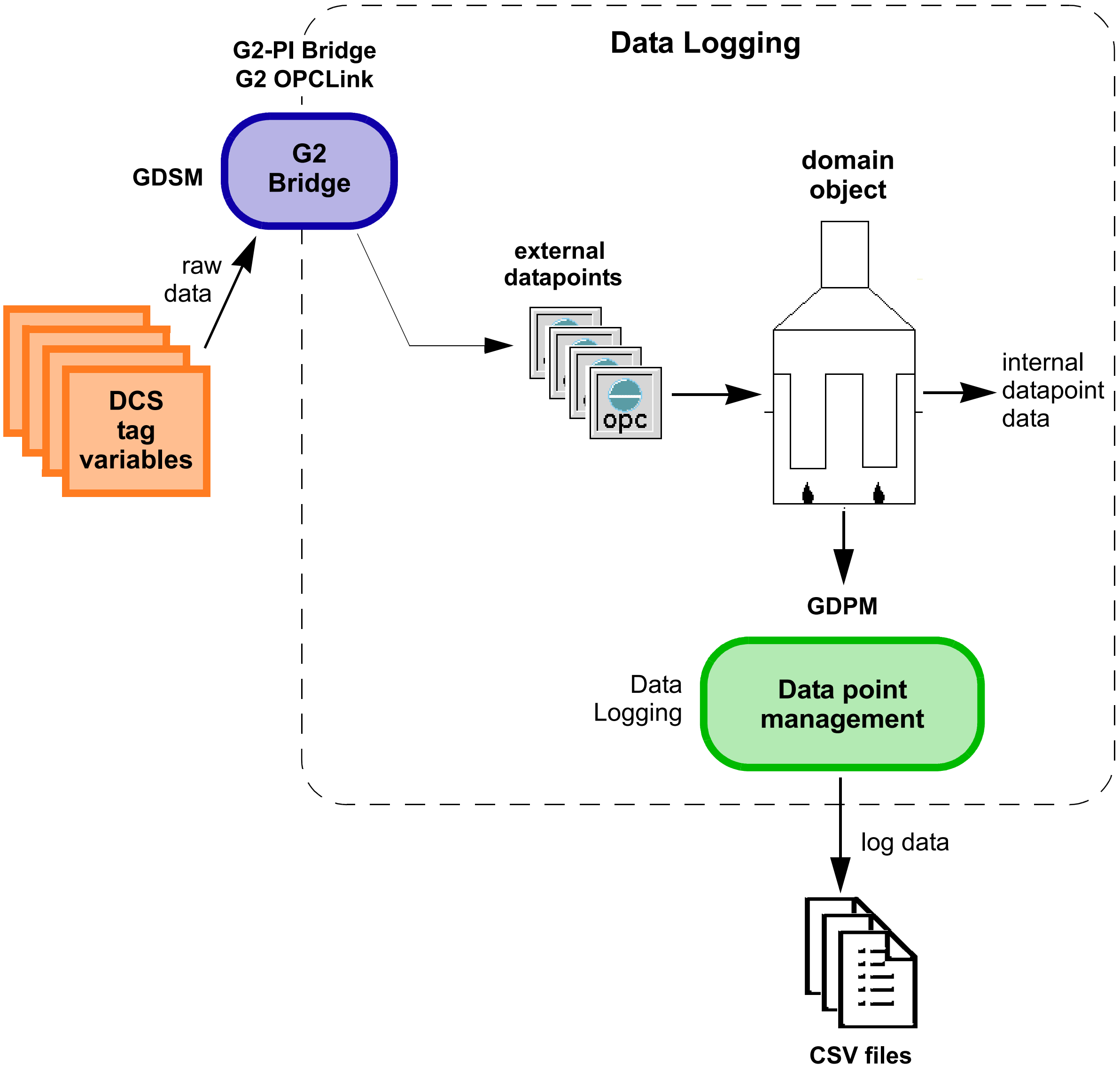
Introduction

Logging facilitates:

- The creation of external datapoints for data replay.
- A history of internal datapoint values for an audit trail or for off-line analysis.

You can log internal or external datapoint values.

This figure shows the architecture of data logging for the Optegrity Heater Tutorial:



Object	Description
Data Logging	Configures the default behavior for logging internal datapoint values to a CSV file.

The Optegrity Heater Tutorial logs external datapoints to a CSV file during simulation. To log data, you must:

- Create and configure a Data Logging object.
- Associate a Data Logging object with a process map or External Datapoints container.
- Optionally, configure data logging parameters for each logged datapoint.

The G2 Data Point Management (GDPM) module handles data logging.

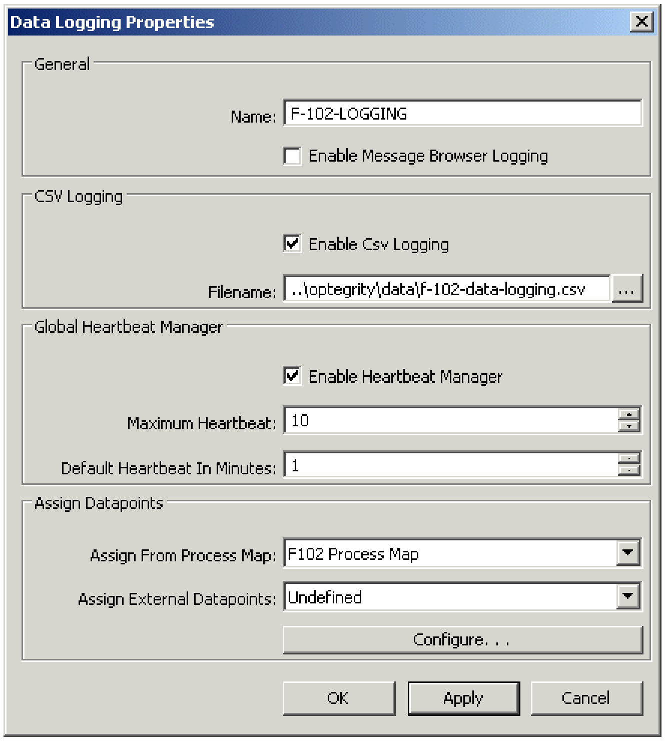
Viewing Data Logging Configuration

The Data Logging object configures the default behavior for logging internal and external datapoints in a process map.

To view data logging configuration:

➔ Choose Project > System Settings Datapoint Logs > F-102-Logging to view its properties.

Here is the properties dialog for Data Logging in the Heater Tutorial:



Name: F-102-LOGGING

□ Enable Message Browser Logging

☑ Enable Csv Logging

Filename: ..\optegrity\data\f-102-data-logging.csv

☑ Enable Heartbeat Manager

Maximum Heartbeat: 10

Default Heartbeat In Minutes: 1

Assign From Process Map: F102 Process Map

Assign External Datapoints: Undefined

Name is the name of the Data Logging configuration object, which domain objects refer to when configuring logging.

Enable Message Browser Logging creates log messages in the Message Browser each time the value changes.

Enable CSV Logging logs data to the specified CSV file each time the value changes. By default, logged data values are appended to the existing file. You can also create log files by specifying a new file name in the dialog or by deleting the existing file.

Enable Heartbeat Manager logs data at regular intervals. You can enable this option when values arrive infrequently.

Maximum Heartbeat is a maximum value that any datapoint can specify for heartbeat interval, in minutes.

Default Heartbeat in Minutes is the default heartbeat interval for all logged datapoints in the specified process map, in minutes.

Assign from Process Map allows you to assign a process map for data logging. By default, all internal datapoints in the assigned process map use the specified logging configuration.

Assign External Datapoints allows you to assign an External Datapoint container for logging data. By default, all external datapoints in the assigned container use the specified logging configuration.

You can override the logging behavior of individual internal or external datapoints by configuring its properties. You can also click the Configure button, to configure all logged datapoints in a single spreadsheet. Both approaches are described later in this chapter.

Configuring Domain Objects for Logging

To configure domain objects for logging, you must associate data logging with a process map. Each internal datapoint in the specified process map is automatically associated with the specified data logging configuration.

You can configure the logging specification on the properties dialog for individual internal datapoints. You can also assign one or more process maps to a data logging configuration to configure data logging for all internal datapoints in the assigned maps.

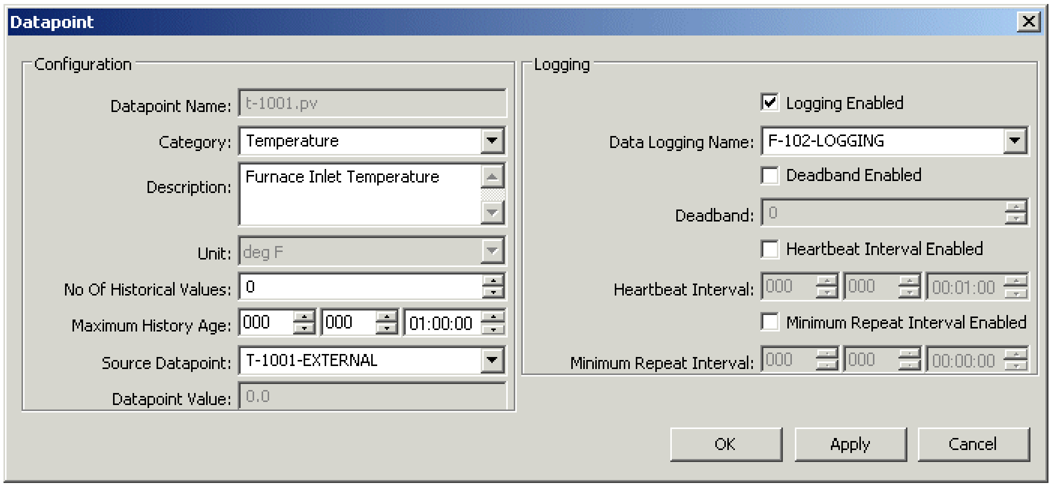
Similarly, you can configure logging for external datapoints by assigning one or more External Datapoint configurations to a Data Logging configuration, or by configuring data logging for individual external datapoints.

The Optegrity Heater Tutorial logs all internal datapoints in the process map.

To configure individual domain objects for logging:

1. On the process map, choose Properties on the T-1001 temperature sensor.
2. Select the t-1001.pv datapoint and select the Properties button to display its properties.

   The Logging group at the right of the dialog specifies that the internal datapoint logs its data, using the F-102-LOGGING data logging configuration:



Datapoint Name: t-1001.pv
Category: Temperature
Description: Furnace Inlet Temperature
Unit: deg F
No Of Historical Values: 0
Maximum History Age: 000 000 01:00:00
Source Datapoint: T-1001-EXTERNAL
Datapoint Value: 0.0

☑ Logging Enabled
Data Logging Name: F-102-LOGGING
□ Deadband Enabled
Deadband: 0
□ Heartbeat Interval Enabled
Heartbeat Interval: 000 000 00:01:00
□ Minimum Repeat Interval Enabled
Minimum Repeat Interval: 000 000 00:00:00

Logging Enabled indicates that logging is enabled for the internal datapoint. By default, logging is disabled. To configure the logging parameters, you must enable logging, as shown in the dialog.

Config Name is the name of the Data Logging configuration that specifies the default behavior for data logging.

When Deadband Enabled is enabled, Deadband is used to reduce the amount of noisy data that is logged. If the new value differs from the last logged value by less than the deadband, the new value is not logged.

When Heartbeat Interval Enabled is enabled, Heartbeat Interval is used to log data according to a schedule, rather than each time the value changes. The value cannot exceed the Maximum Heartbeat Interval given in the Data Logging configuration. The default value is the Default Heartbeat Interval given in the Data Logging configuration.

When Minimum Repeat Interval Enabled is enabled, Minimum Repeat Interval reduces the amount of data that is logged, based on the rate of incoming data. You configure this attribute when the rate of incoming data is greater than the rate at which you want to log data. The data is not logged if the elapsed time from the last logged value is less than the Minimum Repeat Interval.

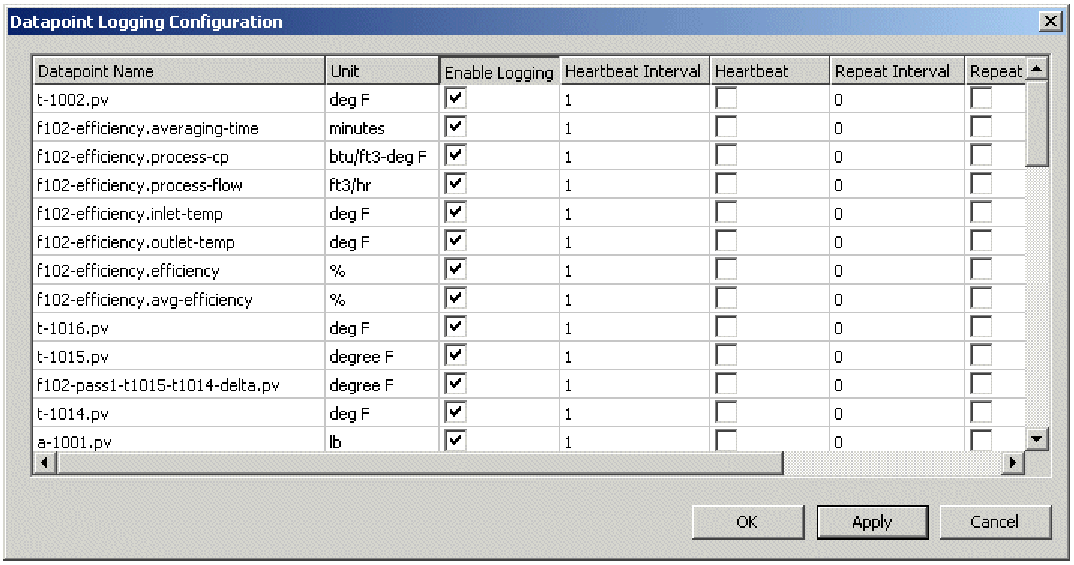
Configuring All Logged Datapoints

Once you associate Data Logging configuration with internal and external datapoints, you can configure data logging for all logged datapoints through a table.

To configure all logged datapoints:

1 Display the properties dialog for the f-102-logging Data Logging configuration, then click the Configure button.

Here is the spreadsheet for the Data Logging configuration, which enables logging for all datapoints. A check-mark in the Enable Logging column indicates that logging is enabled for that datapoint.



Datapoint Name	Unit	Enable Logging	Heartbeat Interval	Heartbeat	Repeat Interval	Repeat
t-1002.pv	deg F	☑	1	□	0	□
f102-efficiency.averaging-time	minutes	☑	1	□	0	□
f102-efficiency.process-cp	btu/ft3-deg F	☑	1	□	0	□
f102-efficiency.process-flow	ft3/hr	☑	1	□	0	□
f102-efficiency.inlet-temp	deg F	☑	1	□	0	□
f102-efficiency.outlet-temp	deg F	☑	1	□	0	□
f102-efficiency.efficiency	%	☑	1	□	0	□
f102-efficiency.avg-efficiency	%	☑	1	□	0	□
t-1016.pv	deg F	☑	1	□	0	□
t-1015.pv	degree F	☑	1	□	0	□
f102-pass1-t1015-t1014-delta.pv	degree F	☑	1	□	0	□
t-1014.pv	deg F	☑	1	□	0	□
a-1001.pv	lb	☑	1	□	0	□

You can enable and disable logging for each datapoint by clicking the check box labeled Enable Logging. You can enable and configure the Heartbeat Interval, Repeat Interval, and Deadband for each datapoint to filter logged data.

2 To filter logged data, click the Enable Heartbeat, Enable Repeat, and Enable Deadband cells for each logged datapoint to toggle the values to Enabled, as needed.

For example, to log only those values that vary by more than 1.0, you would configure a Repeat Interval of 2 and a Deadband of 1.0 for each logged datapoint.

Viewing the Log File

When data is simulated, the application generates a log file for domain objects that have been configured for logging.

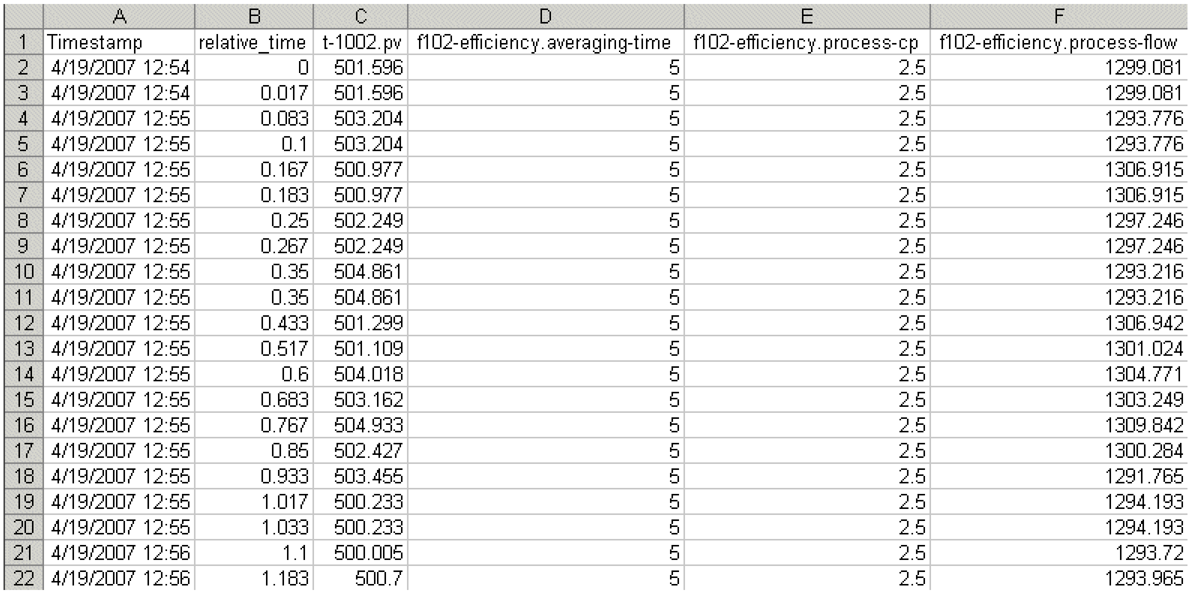
To view the log file:

1 Run the simulation again.

2 Open the following file in your installation directory:

\optegrity\data\F-102-logging.csv

Here is part of the sample log file with the logged data shown. The first row labels the logged datapoints. The first column is the time at which the data was logged, and the second column is the relative time from the start of the simulation.



	A	B	C	D	E	F
1	Timestamp	relative_time	t-1002.pv	f102-efficiency.averaging-time	f102-efficiency.process-cp	f102-efficiency.process-flow
2	4/19/2007 12:54	0	501.596	5	2.5	1299.081
3	4/19/2007 12:54	0.017	501.596	5	2.5	1299.081
4	4/19/2007 12:55	0.083	503.204	5	2.5	1293.776
5	4/19/2007 12:55	0.1	503.204	5	2.5	1293.776
6	4/19/2007 12:55	0.167	500.977	5	2.5	1306.915
7	4/19/2007 12:55	0.183	500.977	5	2.5	1306.915
8	4/19/2007 12:55	0.25	502.249	5	2.5	1297.246
9	4/19/2007 12:55	0.267	502.249	5	2.5	1297.246
10	4/19/2007 12:55	0.35	504.861	5	2.5	1293.216
11	4/19/2007 12:55	0.35	504.861	5	2.5	1293.216
12	4/19/2007 12:55	0.433	501.299	5	2.5	1306.942
13	4/19/2007 12:55	0.517	501.109	5	2.5	1301.024
14	4/19/2007 12:55	0.6	504.018	5	2.5	1304.771
15	4/19/2007 12:55	0.683	503.162	5	2.5	1303.249
16	4/19/2007 12:55	0.767	504.933	5	2.5	1309.842
17	4/19/2007 12:55	0.85	502.427	5	2.5	1300.284
18	4/19/2007 12:55	0.933	503.455	5	2.5	1291.765
19	4/19/2007 12:55	1.017	500.233	5	2.5	1294.193
20	4/19/2007 12:55	1.033	500.233	5	2.5	1294.193
21	4/19/2007 12:56	1.1	500.005	5	2.5	1293.72
22	4/19/2007 12:56	1.183	500.7	5	2.5	1293.965

Now that you have learned how to configure logging, you will learn how to replay data from CSV files to simulate data for OPC tag variables and their associated internal datapoints.

5

Replaying Data

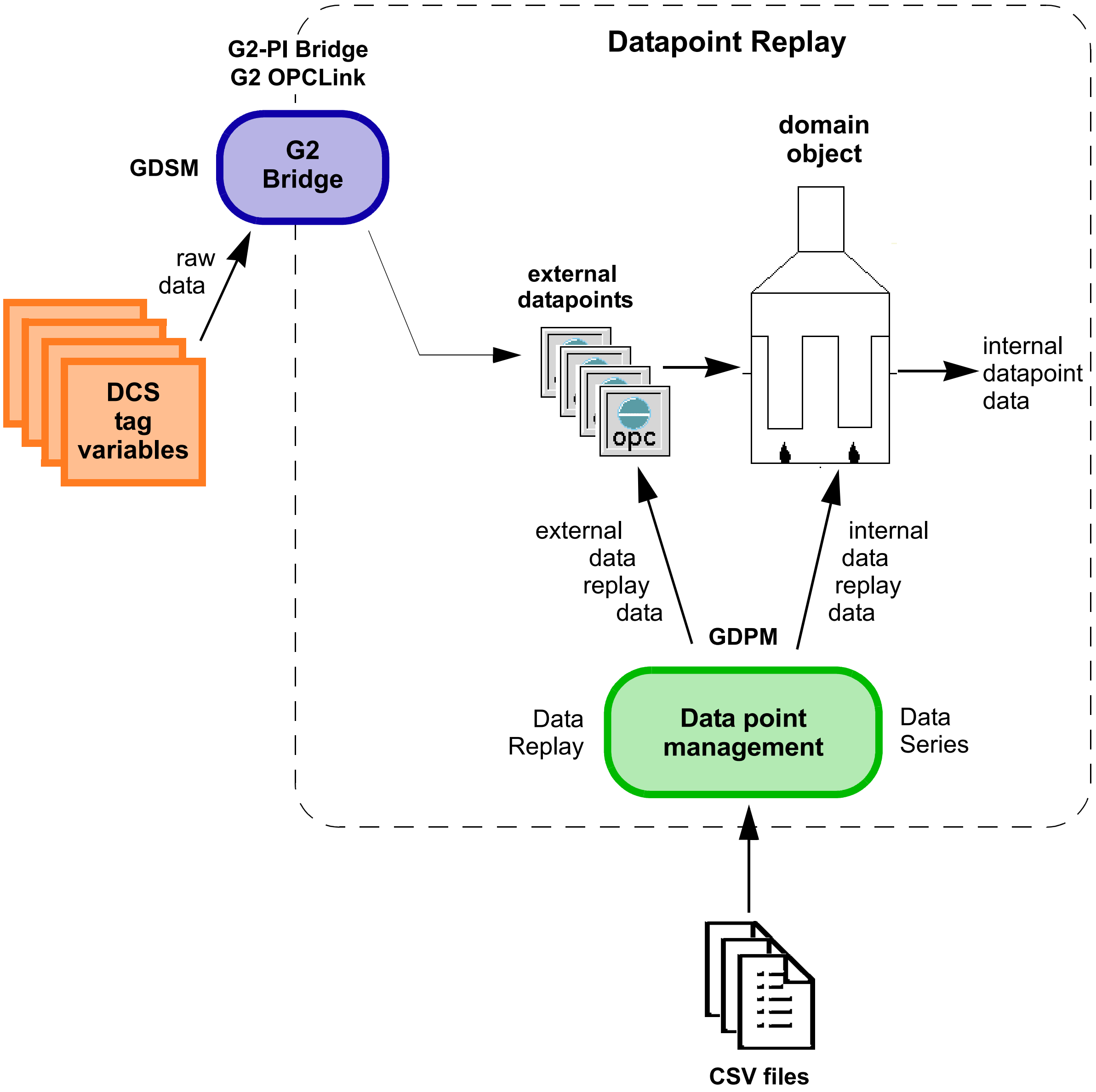
Describes how to replay data from CSV files in the Optegrity Heater Tutorial.

Introduction

You replay data to:

- Test an application in offline mode.
- Validate revised control algorithms prior to placing a system online.

This figure shows the architecture of data replay for the Optegrity Heater Tutorial:



Object	Description
Datapoint Replay	Configures datapoint series used for replaying data, and controls when to start and stop replaying data.
Datapoint Series	Configures various information about the CSV file used for datapoint replay.

You can run the Optegrity Heater Tutorial by replaying data from a CSV file. The CSV file contains columns for each datapoint whose value you want to simulate. Each row represents a set of new values for each datapoint.

You can simulate the data by providing values for:

- External datapoints, for example, T-1002-EXTERNAL.
- Internal datapoints, for example, T-1001.pv.

The Optegrity Heater Tutorial provides a CSV data file for replaying data for external datapoints.

To replay data from a CSV file, you create one of two types of datapoint series, depending on the type of process you want to simulate:

To simulate a...	Use this type of datapoint series...	Which sends new values...
Continuous process	Continuous	At regular time intervals, 10 seconds, by default.
Batch process	Differential	At a specified timestamp, based on an acceleration factor.

You configure Datapoint Replay to specify the data series to use for simulation. You can replay data from one or more CSV files at once. You start and stop data replay by using buttons in the Datapoint Replay dialog.

When simulating data, you can display a trend chart that plots data histories for individual internal datapoints.

The G2 Data Point Management (GDPM) module handles data replay.

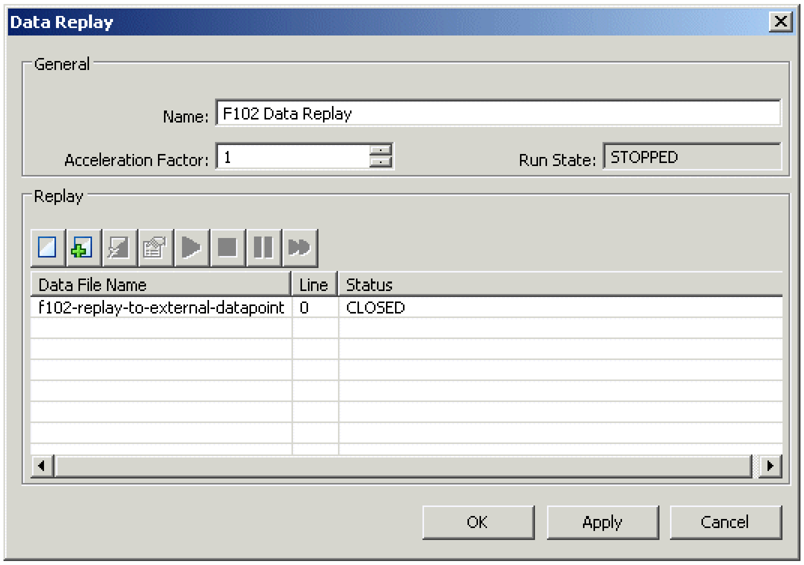
Configuring Datapoint Replay

The Optegrity Heater Tutorial allows you to replay data to external datapoints.

To configure data replay:

1 Choose Project > Logic > Datapoint Replay > F102 Data Replay to display its properties dialog.

Here is the Datapoint Replay properties dialog that replays data to external datapoints:



Name: F102 Data Replay

Acceleration Factor: 1

Run State: STOPPED

Data File Name	Line	Status
f102-replay-to-external-datapoint	0	CLOSED

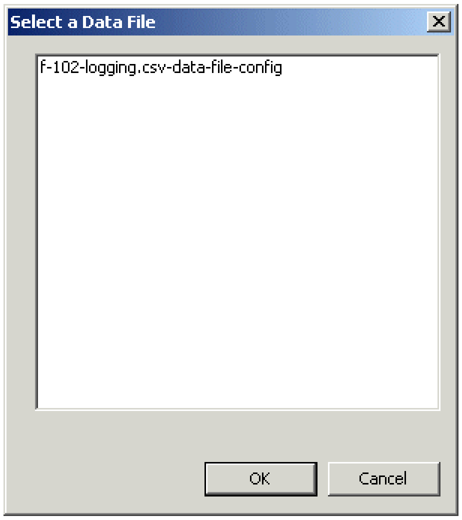
Name is a unique name for the Datapoint Replay configuration object.

Acceleration Factor allows you to speed up data replay for differential data files, based on the timestamp specified in the data file. The timestamp is divided by the Acceleration Factor to determine when the data is actually sent. For example, if the CSV file specifies a timestamp of 2 minutes (120 seconds) and the Acceleration Factor is 40, the data is actually sent at 3 seconds (120/40).

The Datapoint Replay dialog provides buttons for adding and removing data files replaying data, creating new data files, viewing data file properties, and controlling data replay. The data files that are currently being used appear in the dialog, along with the current line and the replay status. This Datapoint Replay configuration replays data from the f102-replay-to-external-datapoint data series.

2 Click the Add File button ( ) in the dialog.

All available data files in the application appear, including log files:



Select a Data File

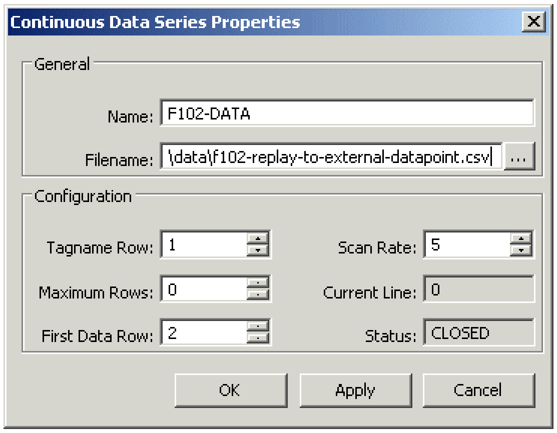
f-102-logging.csv-data-file-config

You can simulate data by using one or more data files.

3 Close this dialog.

4 In the Datapoint Replay dialog, select the f-102-replay-to-external-datapoint data series, then click the Properties button to show its properties.

The dialog title bar indicates that it is a Continuous Datapoint Series, which simulates data for batch processes:



Continuous Data Series Properties

General

Name: F102-DATA

Filename: \data\f102-replay-to-external-datapoint.csv

Configuration

Tagname Row: 1

Scan Rate: 5

Maximum Rows: 0

Current Line: 0

First Data Row: 2

Status: CLOSED

Name is a unique name for the data series.

Filename specifies the directory and file name of the CSV file to use as data, in this case, *f-102-replay-to-external-datapoint.csv*. You click the browser button to configure the file name.

Tagname Row is the CSV file row that contains the name of the datapoint whose value the file will simulate, which is the first row, by default.

Maximum Rows is the number of data rows contained in the file. A value of 0 means there is no maximum.

First Data Row is the CSV file row that contains the first row of data to replay, which is the second row, by default. You can use this field to skip rows of data in the CSV file.

Scan Rate is the rate at which the data file will be played, in seconds. For a scan rate of 5, one row will be read from the file every 5 seconds.

Current Line is a non-editable field displaying the current line of the file for the replay.

Status is a read-only field displaying the status of the file, showing either CLOSED or OPEN.

Viewing the CSV File that Provides External Datapoint Values

You can view the CSV file that provides data for the external datapoints. The CSV file was created by running a simulation and logging randomly generated data values.

The top row of the CSV file contains the name of each datapoint that whose data is to be simulated. Subsequent rows provide values for each external datapoint.

For differential data files, the first column is a timestamp that determines the rate at which to replay the data, expressed in decimal hours. The rate is determined by calculating the difference in time between consecutive rows. The time interval between rows does not need to be evenly spaced.

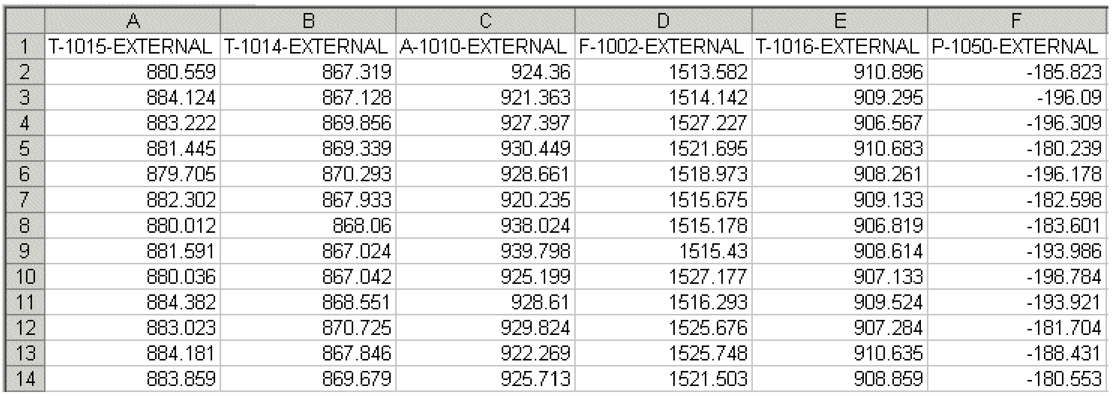
CSV files can specify comments, which are posted in the Message Browser.

To view the CSV file that provides external datapoint values:

➔ Open the following file located in your installation directory:

```
\optegrity\data\f102-replay-to-external-datapoint.csv
```

Here is part of the CSV file for replaying external datapoint values:



	A	B	C	D	E	F
1	T-1015-EXTERNAL	T-1014-EXTERNAL	A-1010-EXTERNAL	F-1002-EXTERNAL	T-1016-EXTERNAL	P-1050-EXTERNAL
2	880.559	867.319	924.36	1513.582	910.896	-185.823
3	884.124	867.128	921.363	1514.142	909.295	-196.09
4	883.222	869.856	927.397	1527.227	906.567	-196.309
5	881.445	869.339	930.449	1521.695	910.683	-180.239
6	879.705	870.293	928.661	1518.973	908.261	-196.178
7	882.302	867.933	920.235	1515.675	909.133	-182.598
8	880.012	868.06	938.024	1515.178	906.819	-183.601
9	881.591	867.024	939.798	1515.43	908.614	-193.986
10	880.036	867.042	925.199	1527.177	907.133	-198.784
11	884.382	868.551	928.61	1516.293	909.524	-193.921
12	883.023	870.725	929.824	1525.676	907.284	-181.704
13	884.181	867.846	922.269	1525.748	910.635	-188.431
14	883.859	869.679	925.713	1521.503	908.859	-180.553

Here is the list of external datapoints that appear across the top of the CSV file. These are the same external datapoints you saw earlier in the external datapoints configuration in Viewing External Datapoints.

```
T-1015-EXTERNAL
T-1014-EXTERNAL
A-1010-EXTERNAL
F-1002-EXTERNAL
T-1016-EXTERNAL
P-1050-EXTERNAL
T-1002-EXTERNAL
A-1001-EXTERNAL
T-1012-EXTERNAL
T-1013-EXTERNAL
T-1001-EXTERNAL
F-1001-EXTERNAL
```

Simulating External Datapoint Values

Now, you will simulate external datapoint values, using data from the CSV file. When you simulate the data, the Datapoint Replay configuration simulates the data at a rate given by the Scan Rate from the dialog.

You can watch the progress of the simulation by viewing the data series as it reads each line of the file. You can also view the values as they change in the displays on the process map.

Note Remember, before you simulate data, you must initialize the process map.

To simulate external datapoint values:

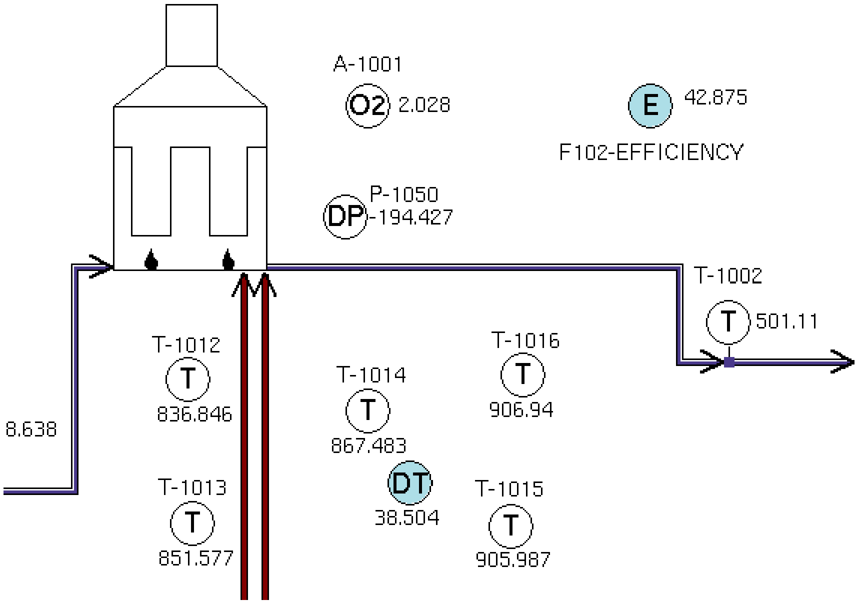
1. If you have not done so already, choose Project > Initialize Application.
2. Show the Datapoint Replay properties dialog for simulating external datapoint values.
3. Configure the Acceleration Factor to be 20, which will speed up the simulation by a factor of 20.
4. Click the Start Replay button (▶) to start replaying the data.

   Run State changes to Running, the Status of the data file is OPEN, and the Line indicator increments each time a row in the data file is read.

   You can pause, stop, and resume data replay at any time by clicking the appropriate buttons in the dialog.
5. Close the Datapoint Replay dialog.
6. View the process map and observe the data values in the datapoint displays as they change.

The simulation stops running when it reaches the end of the file.

Here is part of the process map during the simulation:



You can click the Process Map Instruments button ( ) or Instruments Plot button ( ) to view datapoint values and plots while the simulation is running.

Now that you have replayed data and viewed the results in the process map, domain map overview, and trend chart, you will learn how to view and interact with operator messages that are generated as a result of the simulation.

Viewing Operator Messages

Describes how to view operator messages in the default Message Browser for the Optegrity Heater Tutorial.

Introduction

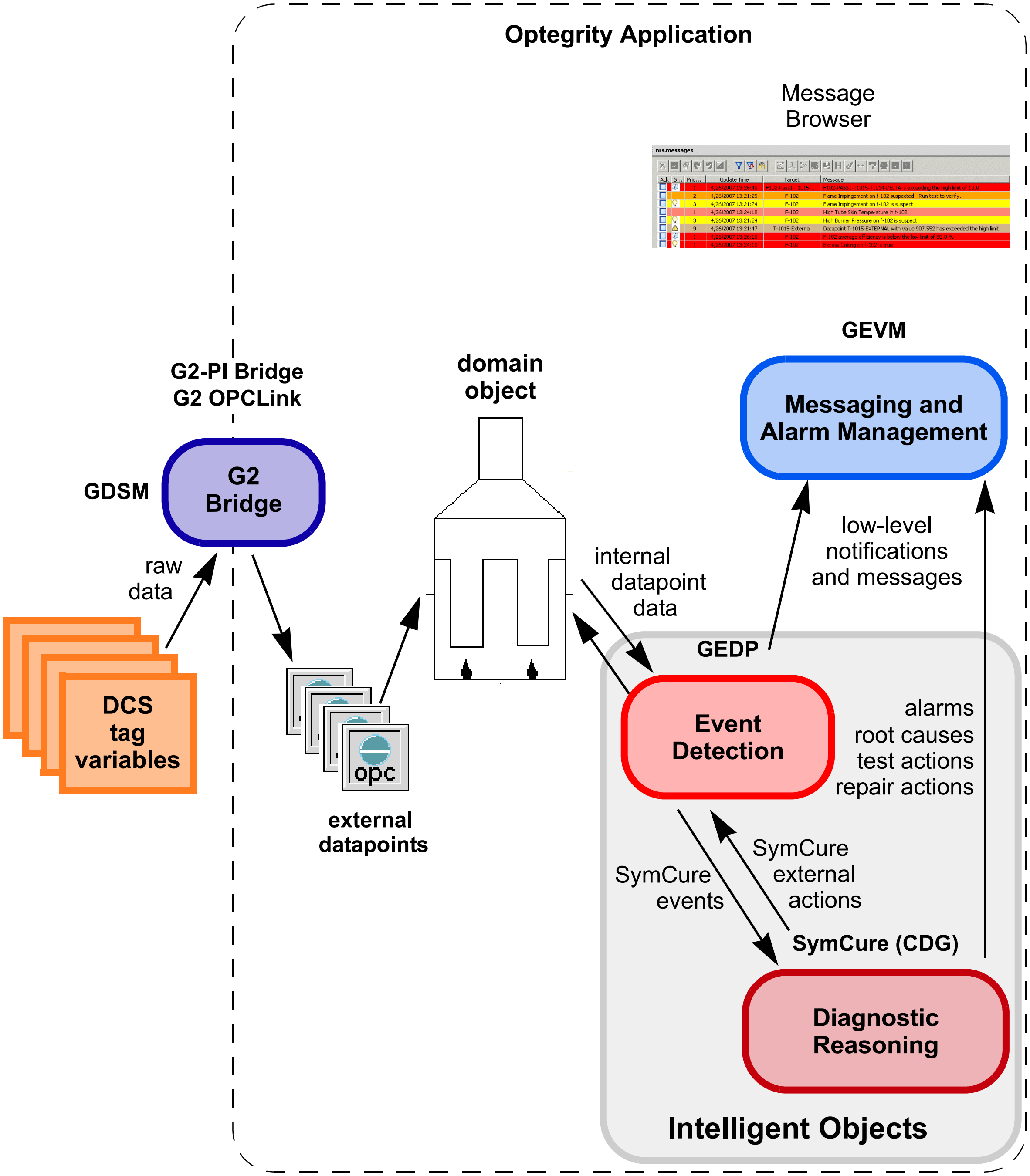
During simulation, Optegrity validates external process data against data validation limits, targets, and deviations associated with the external datapoints. If an external data value does not conform to its associated data validation specification, GDPM generates an operator message.

Optegrity also monitors datapoint values and generates operator messages, based on event detection diagrams associated with active events for the various domain objects in the process map. You can also define your own event detection diagrams and events for domain objects.

In addition, SymCure generates operator messages for alarms, root causes, test actions, and repair actions, based on fault models defined for the various domain objects in the process map.

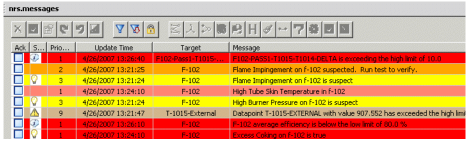
You can view operator messages in the default Message Browser. The G2 Event Management (GEVM) module handles message management.

This figure shows the architecture of GEVM message management for the Optegrity Heater Tutorial. It includes the Message Browser.



nrs.messages

Ack	S...	Prio...	Update Time	Target	Message
□		1	4/26/2007 13:26:40	F102-Pass1-T1015-...	F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0
□		2	4/26/2007 13:21:25	F-102	Flame Impingement on f-102 suspected. Run test to verify.
□		3	4/26/2007 13:21:24	F-102	Flame Impingement on f-102 is suspect
□		1	4/26/2007 13:24:10	F-102	High Tube Skin Temperature in f-102
□		3	4/26/2007 13:21:24	F-102	High Burner Pressure on f-102 is suspect
□	⚠	9	4/26/2007 13:21:47	T-1015-External	Datapoint T-1015-EXTERNAL with value 907.552 has exceeded the high limit.
□		1	4/26/2007 13:26:10	F-102	F-102 average efficiency is below the low limit of 80.0 %
□		1	4/26/2007 13:24:10	F-102	Excess Coking on f-102 is true

Browser	Browser Name	Description
	Message Browser	Provides a browser for viewing: • Low-level notifications and operator messages that originate from event detection diagrams. • Data validation messages. • Alarm, root cause, test action, and repair action messages generated by SymCure, based on fault model diagnosis. • Other operator messages that are generated at run time.

nrs.messages

Ack	S...	Prio...	Update Time	Target	Message
□		1	4/26/2007 13:26:40	F102-Pass1-T1015-...	F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0
□		2	4/26/2007 13:21:25	F-102	Flame Impingement on f-102 suspected. Run test to verify.
□		3	4/26/2007 13:21:24	F-102	Flame Impingement on f-102 is suspect
□		1	4/26/2007 13:24:10	F-102	High Tube Skin Temperature in f-102
□		3	4/26/2007 13:21:24	F-102	High Burner Pressure on f-102 is suspect
□	⚠	9	4/26/2007 13:21:47	T-1015-External	Datapoint T-1015-EXTERNAL with value 907.552 has exceeded the high limi
□		1	4/26/2007 13:26:10	F-102	F-102 average efficiency is below the low limit of 80.0 %
□		1	4/26/2007 13:24:10	F-102	Excess Coking on f-102 is true

Showing the Message Browser

When you simulate data in the Optegrity Heater Tutorial, a number of messages are generated. You can view these messages in the default Message Browser.

You can perform a number of actions on messages in the Message Browser, such as going to the message source, viewing message details, acknowledging a message, deleting a message, sorting messages, and filtering messages.

In the Optegrity Heater Tutorial, the following operator messages are generated:

- F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0.

  This message is generated by event detection and indicates that the tube skin delta temperature sensor has exceeded its high limit.

- F-102 average efficiency is below the low limit of 80%

  This message is generated by event detection and indicates that the calculated efficiency of the heater has exceeded its low limit.

- Excess Coking on f-102 is true

  This message is generated by SymCure diagnostics and indicates that Excess Coking is the root cause of the High Tube Skin Temperature event.

- High Tube Skin Temperature in f-102

  This message is generated by SymCure and indicates that the tube skin temperature alarm event is true.

- High Burner Pressure on f-102 is suspect.

  This message generated by SymCure and indicates that the High Burner Pressure root cause is suspect.

- Flame Impingement on f-102 is suspect.

  This message generated by SymCure and indicates that the Flame Impingement root cause is suspect.

- Flame Impingement on f-102 is suspect. Run test to verify.

  This message is generated by SymCure and provides a test action that the user can run to verify the suspected root cause.

- Datapoint T-1015-EXTERNAL with value 907.552 has exceeded the high limit.

  This message is generated as a result of data validation, which you learned about earlier in this tutorial.

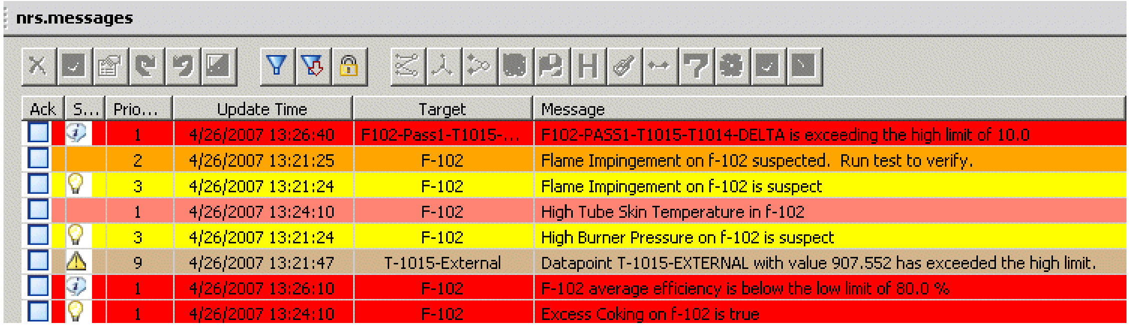
For information on what causes each of these messages to occur, see these references:

- For data validation messages, see Viewing External Datapoints.
- For event detection messages, see Viewing the Generic Event Detection Templates.
- For diagnostic fault model messages, see Viewing the Generic Fault Model for the Heater.

To show the Message Browser:

➔ Choose Project > Message Browser or click the equivalent toolbar button (▦).

Here is the Message Browser at the end of the simulation, which shows a data validation alarm message, a message generated by event detection, and a test action message and two root cause messages generated by SymCure diagnosis:

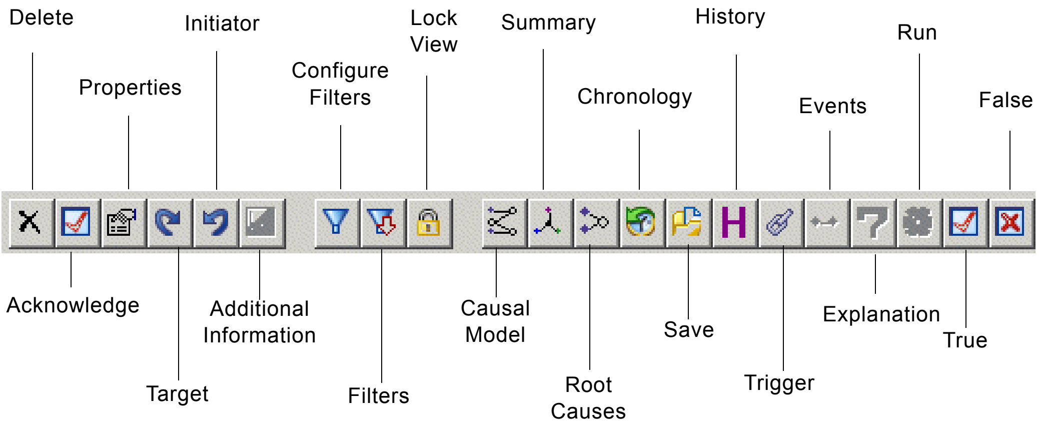


nrs.messages

Ack	S...	Prio...	Update Time	Target	Message
□	ⓘ	1	4/26/2007 13:26:40	F102-Pass1-T1015-...	F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0
□		2	4/26/2007 13:21:25	F-102	Flame Impingement on f-102 suspected. Run test to verify.
□	💡	3	4/26/2007 13:21:24	F-102	Flame Impingement on f-102 is suspect
□		1	4/26/2007 13:24:10	F-102	High Tube Skin Temperature in f-102
□	💡	3	4/26/2007 13:21:24	F-102	High Burner Pressure on f-102 is suspect
□	⚠	9	4/26/2007 13:21:47	T-1015-External	Datapoint T-1015-EXTERNAL with value 907.552 has exceeded the high limit.
□	ⓘ	1	4/26/2007 13:26:10	F-102	F-102 average efficiency is below the low limit of 80.0 %
□	💡	1	4/26/2007 13:24:10	F-102	Excess Coking on f-102 is true

Interacting with Messages

In Modeler mode, you interact with messages in the Message Browser, using the following toolbar buttons:

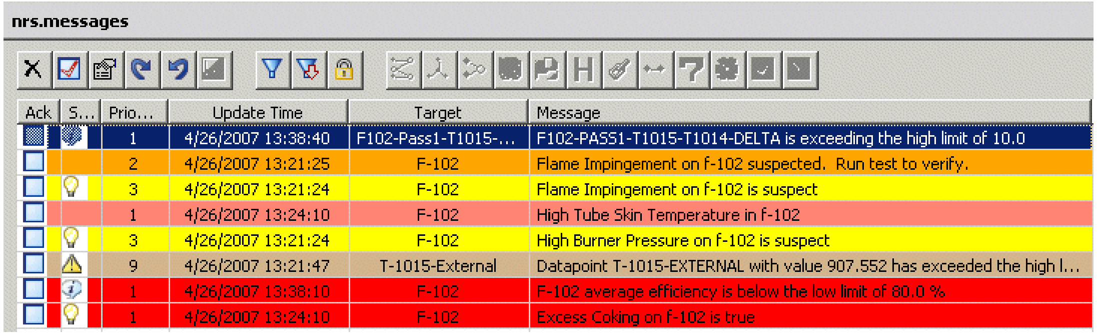


To interact with messages, you select one or more messages in the browser, then click the desired toolbar button. You can select multiple consecutive messages by selecting the starting message, then holding down the SHIFT key and selecting the ending message.

To interact with messages:

➔ Select the high-priority message about the tube skin delta temperature high limit being exceeded.

The message is highlighted and the relevant toolbar buttons become active. The toolbar buttons that are not active are not relevant for this type of message.


nrs.messages

Ack	S...	Prio...	Update Time	Target	Message
▨		1	4/26/2007 13:38:40	F102-Pass1-T1015-...	F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0
□		2	4/26/2007 13:21:25	F-102	Flame Impingement on f-102 suspected. Run test to verify.
□	💡	3	4/26/2007 13:21:24	F-102	Flame Impingement on f-102 is suspect
□		1	4/26/2007 13:24:10	F-102	High Tube Skin Temperature in f-102
□	💡	3	4/26/2007 13:21:24	F-102	High Burner Pressure on f-102 is suspect
□	⚠	9	4/26/2007 13:21:47	T-1015-External	Datapoint T-1015-EXTERNAL with value 907.552 has exceeded the high l...
□	ⓘ	1	4/26/2007 13:38:10	F-102	F-102 average efficiency is below the low limit of 80.0 %
□	💡	1	4/26/2007 13:24:10	F-102	Excess Coking on f-102 is true

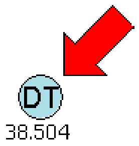
Showing the Message Target

You can go to the domain object that is the target of the message.

To show the message target:

➔ With the F102-PASS1-T1015-T1014-DELTA message selected, click the Target toolbar button:

A red arrow identifies the target in the process map. Here is the result of showing the target of the selected message:



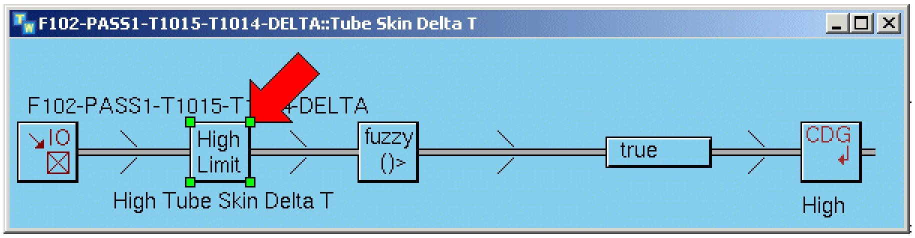
Showing the Message Initiator

You can go to the object that generated the message. The initiator depends on the type of message. For example, the F102-PASS1-T1015-T1014-DELTA message is the result of event detection, so the initiator is the GEDP block in the event detection diagram that posted the message. The initiator of the data validation message is the external datapoint that defines the data validation limits.

To show the message initiator:

➔ With the F102-PASS1-T1015-T1014-DELTA message selected, click the Initiator toolbar button:

Here is the event detection diagram that triggered the message. The arrow indicates the GEDP block that posted the message. You will learn more about this diagram later in this tutorial:


F102-PASS1-T1015-T1014-DELTA::Tube Skin Delta T
F102-PASS1-T1015-T1 -DELTA
IO
High
Limit
fuzzy
()>
true
CDG
High Tube Skin Delta T
High

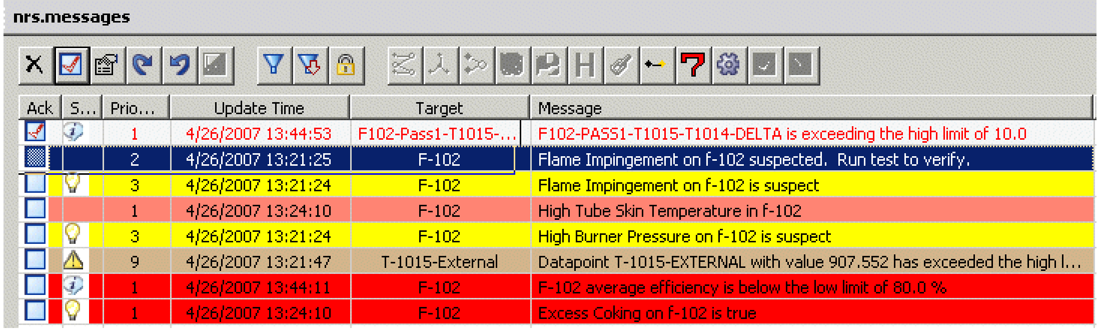
Acknowledging Messages

Typically, an operator acknowledges a message to indicate it has been read. The message keeps track of who acknowledged it and when it was acknowledged.

To acknowledge a message:

➔ With the F102-PASS1-T1015-T1014-DELTA message selected, click the Acknowledge toolbar button: ☑

An X appears in the Ack column to indicate it has been acknowledged, and the message turns white when not selected:



nrs.messages

Ack	S...	Prio...	Update Time	Target	Message
☑	ⓘ	1	4/26/2007 13:44:53	F102-Pass1-T1015-...	F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0
▨		2	4/26/2007 13:21:25	F-102	Flame Impingement on f-102 suspected. Run test to verify.
□	💡	3	4/26/2007 13:21:24	F-102	Flame Impingement on f-102 is suspect
□		1	4/26/2007 13:24:10	F-102	High Tube Skin Temperature in f-102
□	💡	3	4/26/2007 13:21:24	F-102	High Burner Pressure on f-102 is suspect
□	⚠	9	4/26/2007 13:21:47	T-1015-External	Datapoint T-1015-EXTERNAL with value 907.552 has exceeded the high l...
□	ⓘ	1	4/26/2007 13:44:11	F-102	F-102 average efficiency is below the low limit of 80.0 %
□	💡	1	4/26/2007 13:24:10	F-102	Excess Coking on f-102 is true

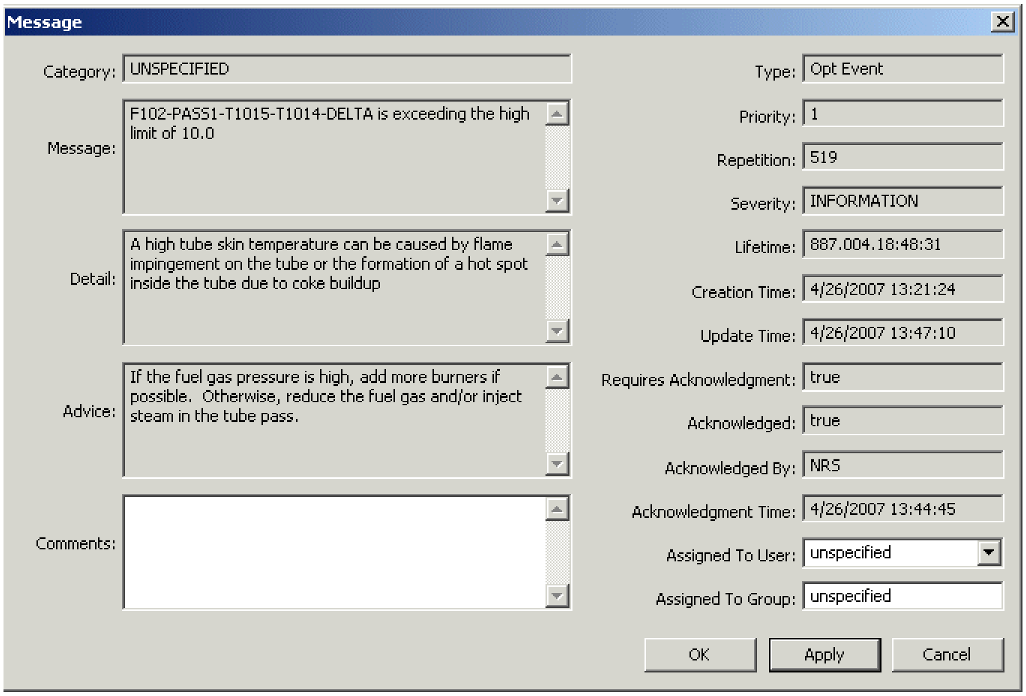
Showing Message Properties

Message properties provide a variety of information about the message, including the message category, text, detail, advice, priority, repetition, assignment, timestamps, and acknowledgement status. Some of the information you configure in the message initiator, such as the category, message text, detail, and advice. Other information is configured automatically, such as the timestamps. You can also configure comments. You will learn how to configure message details later in this tutorial.

To show message properties:

➔ With the tube skin delta temperature message selected, click the Properties toolbar button:

Here are the message properties after the message has been acknowledged:



Category:	UNSPECIFIED	Type:	Opt Event
Message:	F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 10.0	Priority:	1
		Repetition:	519
		Severity:	INFORMATION
Detail:	A high tube skin temperature can be caused by flame impingement on the tube or the formation of a hot spot inside the tube due to coke buildup	Lifetime:	887.004.18:48:31
		Creation Time:	4/26/2007 13:21:24
		Update Time:	4/26/2007 13:47:10
Advice:	If the fuel gas pressure is high, add more burners if possible. Otherwise, reduce the fuel gas and/or inject steam in the tube pass.	Requires Acknowledgment:	true
		Acknowledged:	true
		Acknowledged By:	NRS
Comments:		Acknowledgment Time:	4/26/2007 13:44:45
		Assigned To User:	unspecified
		Assigned To Group:	unspecified

Attribute	Description
Category	The message category, which you configure in the message initiator.
Message	The message text, which you configure in the message initiator.
Detail	Detailed information about the message, which you configure in the message initiator.
Advice	Operator advice for the message, which you configure in the message initiator.
Comments	User-defined comments, which the operator enters when viewing the message.
Type	The message class.
Priority	An integer that indicates the importance of the message, which is usually configured in the message initiator. The priority is a number from 1 - 9, where 1 is the highest priority.

Attribute	Description
Repetition	An integer that indicates the number of times the message has occurred on the same target, which is updated automatically each time the message is repeated.
Lifetime	The number of seconds that the message should exist before it is automatically deleted, which is usually configured in the message initiator.
Creation Time	The time at which the message was created, which is assigned automatically when the message is created.
Update Time	The time at which the message was last updated, either automatically or manually, which is assigned automatically when the message is updated.
Requires Acknowledgement	A truth value that indicates whether the message needs to be acknowledged, which is usually configured in the message initiator.
Acknowledged	A truth value that indicates whether the message has been acknowledged, which is set automatically when the message is acknowledged.
Acknowledged By	The user name of the operator who acknowledged the message, which is set automatically when the message is acknowledged.
Acknowledgement Time	The time at which the message was acknowledged, which is assigned automatically when the message is acknowledged.
Assigned to User	User-defined name, which the operator selects to indicate who is responsible.
Assigned to Group	The group to which the message is assigned.

Sorting Messages

By default, messages are sorted chronologically by creation time, with the newest messages on top. You can sort messages by any visible column in the browser, and you can reverse the sort order.

To sort messages by any visible column:

➔ Click the column header in the browser.

The column header includes an arrow to indicate that messages are sorted, based on that column’s data.

To reverse the sort order:

➔ Click the column header a second time.

The messages are sorted in reverse order to the current sort order and the arrow is reversed.

Filtering Messages

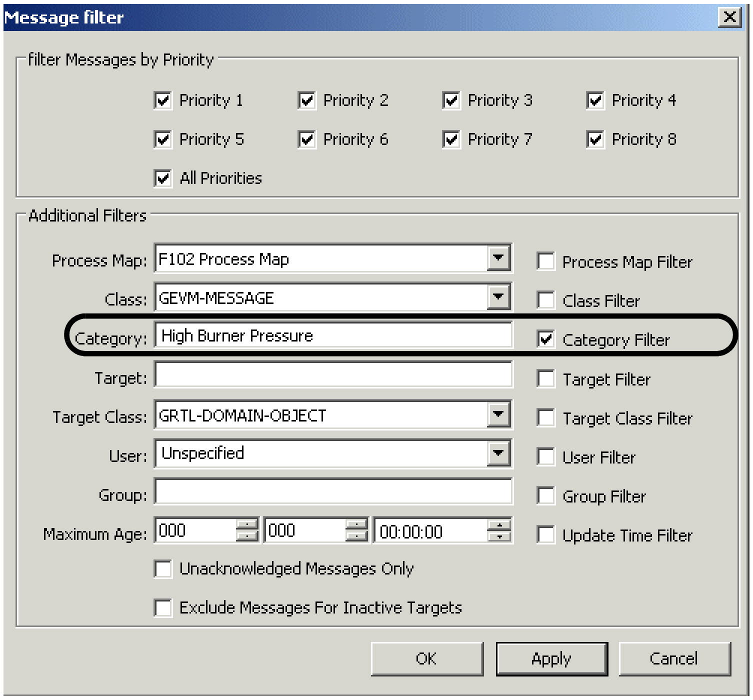
You can filter messages, based on criteria defined in the message browser. The criteria are defined in the browser and include priority, process map, message type, category, target, assigned to, update time, acknowledgement status, and subsumed messages. A message is displayed when the attributes of the message satisfy the filter criteria; the message is not displayed when the attributes fail to meet the criteria.

To filter messages:

1 Click the Configure Filters toolbar button:

This dialog lets you configure the filter criteria of the messages to show.

2 Configure the Category to show just messages in the High Burner Pressure category, then enable the Category Filter option:



Message filter

filter Messages by Priority

☑ Priority 1 ☑ Priority 2 ☑ Priority 3 ☑ Priority 4

☑ Priority 5 ☑ Priority 6 ☑ Priority 7 ☑ Priority 8

☑ All Priorities

Additional Filters

Process Map: F102 Process Map — □ Process Map Filter

Class: GEVM-MESSAGE — □ Class Filter

Category: High Burner Pressure — ☑ Category Filter

Target: — □ Target Filter

Target Class: GRTL-DOMAIN-OBJECT — □ Target Class Filter

User: Unspecified □ User Filter

Group: — □ Group Filter

Maximum Age: 000 000 00:00:00 — □ Update Time Filter

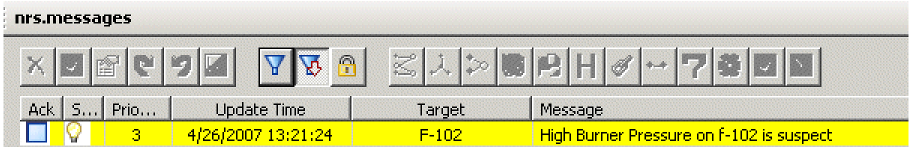
□ Unacknowledged Messages Only

□ Exclude Messages For Inactive Targets

Now, you must apply the filter.

3 Click the Filters toolbar button:

Only the messages in the specified filter category appear. The Filter button changes to indicate the filter is in effect: ( )



nrs.messages

Ack	S...	Prio...	Update Time	Target	Message
□		3	4/26/2007 13:21:24	F-102	High Burner Pressure on f-102 is suspect

4 Click the Filters toggle button again to show all the messages.

Deleting Messages

You can delete individual or multiple messages. To delete a range of messages, click the first message, then hold down the Shift key and click the last message. To select multiple messages that are not contiguous, use the Ctrl key.

To delete messages:

➔ Select the data validation message and click the Delete button:

Locking the Message Browser

You can lock the Message Browser to prevent additional messages from arriving or to prevent messages from being deleted. When you unlock the browser, all the messages that have been sent or deleted since the browser was locked now appear or disappear.

To lock the Message Browser:

1 Click the Lock View toolbar button:

2 To test the lock, delete a message in the browser.

The message remains in the browser because it is currently locked.

3 Click the Lock View toolbar button to unlock the view. The Lock View button changes to indicate the view is locked:

The message is now deleted because the browser is no longer locked.

Now that you have interacted with operator messages in the Message Browser, you will view the event detection diagram that generates the high tube skin delta temperature message.

Detecting Events

Describes how to view generic event detection templates and specific event detection diagrams for domain objects in the Optegrity Heater Tutorial.

Introduction

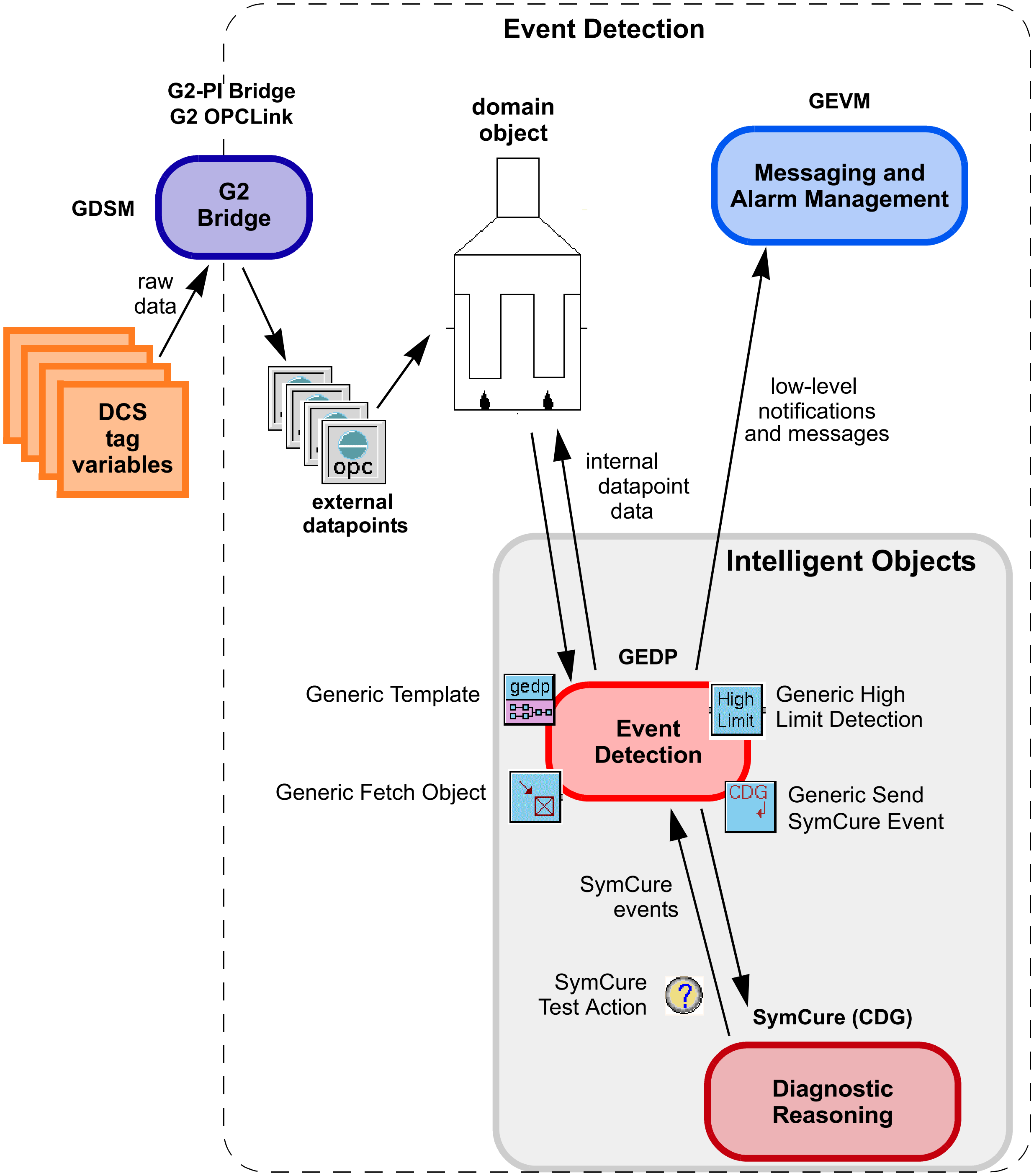
A domain object can have one or more associated event detection diagrams. These diagrams monitor internal datapoints, perform calculations on the data, draw conclusions based on the calculations, and generate events based on the conclusions. Event detection diagrams use a simple block language to perform these tasks. You create event detection diagrams by using the G2 Event and Data Processing (GEDP) module.

The event detection diagram for a domain object can generate SymCure events, which trigger SymCure diagnostic reasoning. An event detection diagram can also generate low-level notifications and operator messages, which are sent to the Message Browser. Thus, event detection diagrams integrate internal datapoints in a process map with SymCure diagnostic models and message browsers.

When the intelligent object libraries are loaded, domain objects provide a number of built-in generic event detection templates. When you initialize the application, Optegrity automatically enables event detection and creates specific event detection diagrams for each domain object in a process map.

For example, in the Optegrity Heater Tutorial, the F-102 heater defines three events, each of which has an associated specific event detection diagram. Similarly, each sensor in the process map has numerous events, each of which has an associated specific event detection diagram. You can configure parameters for various GEDP blocks in the specific event detection diagrams, for example, high and low limits.

This figure shows the architecture of event detection for the Optegrity Heater Tutorial. It includes the GEDP blocks that integrate event detection with domain objects, SymCure diagnostic reasoning, and GEVM messaging and alarm management.



Block	Block Name	Description
gedp	Generic Template	Defines a generic GEDP event detection template associated with a target domain object class.
↘☒	Generic Fetch Object Entry Point	Gets the domain object associated with the specific event detection diagram.
High Limit	Generic High Limit Detection	Compares incoming value against a limit. If the limit is violated, posts an operator message to the Message Browser.
CDG ↲	Generic Send SymCure Event	Sends a SymCure event to diagnose the causes of the detected event.

For detailed information about the GEDP block language, see the *G2 Event and Data Processing User's Guide*.

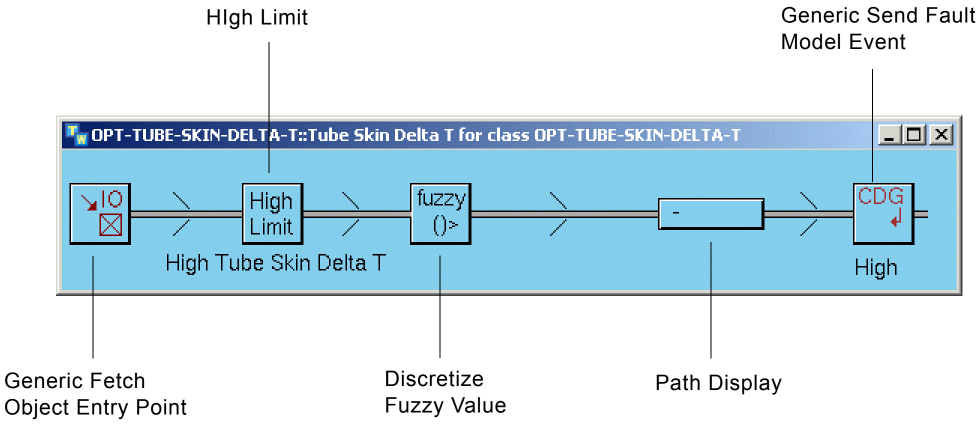
Viewing the Generic Event Detection Templates

Generic event detection templates are associated with a target class, which is a domain object class in the process map. Generic blocks in the template apply to instances of the target class.

To view the generic event detection template:

1 Choose Project > Logic > Detect > Dataflow Templates > Gensym Heater Analysis > OPT-TUBE-SKIN-DELTA-T::Tube Skin Delta T.

The Tube Skin Delta T generic event detection template monitors any opt-tube-skin-delta-t sensor for changes in the value of the PV internal datapoint of the sensor. If the value goes above the configured limit, the diagram generates a SymCure High event.



The Generic Fetch Object Entry Point block gets the specific instance of the opt-tube-skin-delta-t sensor class associated with the specific diagram.

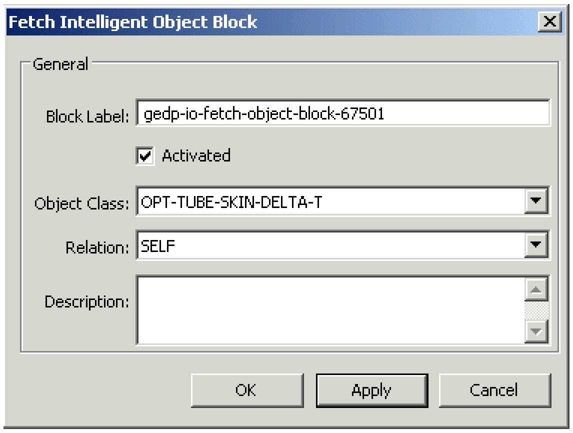
The High Limit block labeled High Tube Skin Delta T block monitor changes in the PV of any opt-tube-skin-delta-t sensor. It compare the value against a specified limit. If the limit is violated, the output value from the block is true and the block generates an operator message. This is the initiator for the operator message you saw in the Message Browser earlier: F102-PASS1-T1015-T1014-DELTA is exceeding the high limit of 20.0.

The Discrete Fuzzy Value block takes a numerical value and applies fuzzy truth thresholds to determine a discrete truth value.

The Path Display shows the current truth value.

The Generic Send Fault Model Event block generates a SymCure event for any opt-tube-skin-delta-t sensor. In this template, the block generates a High event whenever the value turns true or false.

2 Show the properties dialog for the IO Fetch Object block:



Block Label: gedp-io-fetch-object-block-67501

☑ Activated

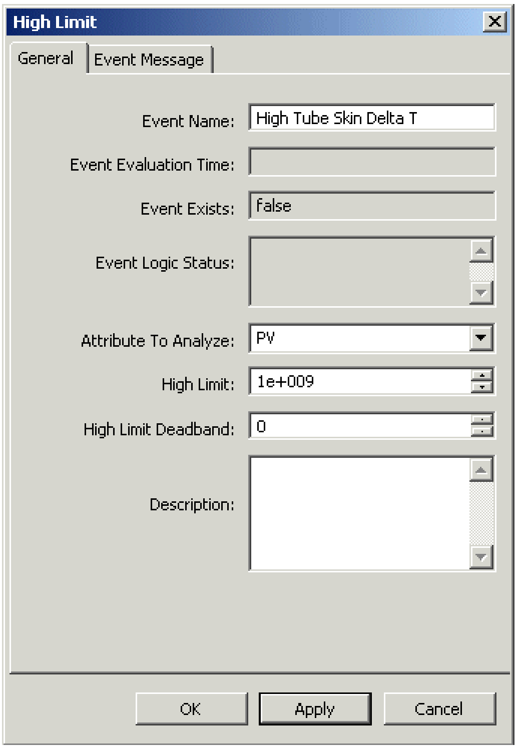
Object Class: OPT-TUBE-SKIN-DELTA-T

Relation: SELF

Description:

The block gets the specific instance of the opt-tube-skin-delta-t class associated with the specific diagram.

3 Show the properties for the High Limit block:



Event Name: High Tube Skin Delta T

Event Evaluation Time:

Event Exists: false

Event Logic Status:

Attribute To Analyze: PV

High Limit: 1e+009

High Limit Deadband: 0

Description:

Event Name is the name of the event and the label for the block.

Event Evaluation Time is the time at which the event was evaluated.

Event Exists indicates whether the event is currently true.

Event Logic Status indicates OK if the status of the block is good; otherwise, it contains a description of the error.

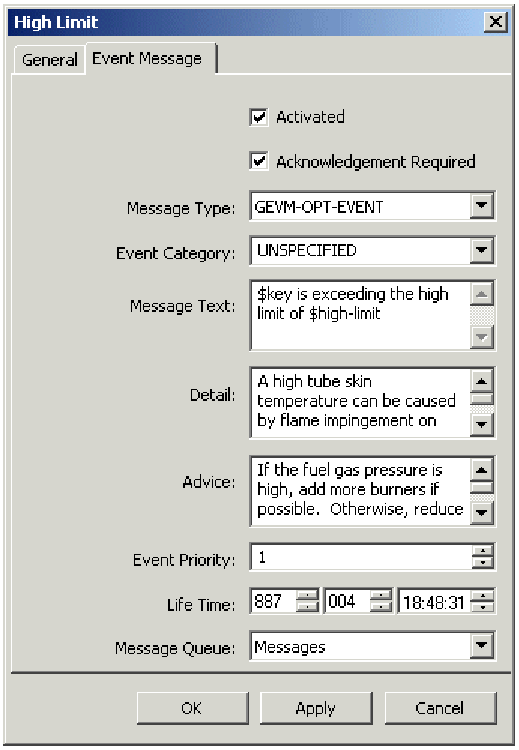
Attribute to Analyze specifies the attribute of the specific domain object to which to compare the block’s input value.

High Limit is the high limit for the event.

High Limit Deadband specifies a value around the High Limit value, which the block uses to create a fuzzy truth value.

Description allows the user to enter a textual description of the event.

4 Click the Event Message tab on the High Limit properties to see how the event’s message is configured:



☑ Activated

☑ Acknowledgement Required

Message Type: GEVM-OPT-EVENT

Event Category: UNSPECIFIED

Message Text: $key is exceeding the high limit of $high-limit

Detail: A high tube skin temperature can be caused by flame impingement on

Advice: If the fuel gas pressure is high, add more burners if possible. Otherwise, reduce

Event Priority: 1

Life Time: 887 004 18:48:31

Message Queue: Messages

Activated indicates whether a message should be sent when the event is true.

Acknowledgement Required determines whether the operator must acknowledge the message before it can be deleted.

Message Type is gevm-opt-event, which is the default message type for messages in the Message Browser.

Event Category is a user-defined category for the message, which you can use for filtering messages in the Message Browser.

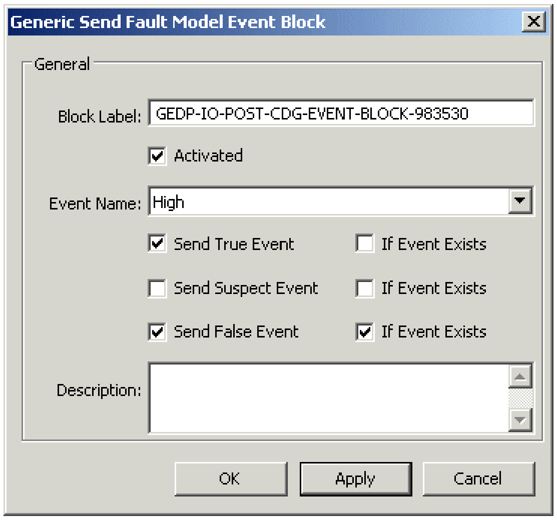
Message Text, Detail, and Advice are all text-based fields, which the operator can view in the Message Browser. Notice that the Message Text uses text substitution by using the keywords $key and $high-limit to refer to the target object and the high limit value of the event, respectively.

Event Priority is the priority of the message.

Life Time specifies how long the message exists. Once the duration of the specified life time expires, the message is automatically deleted.

Message Queue specifies that this event's messages are sent to the Messages queue, which are displayed in the default Message Browser. You can also send messages to custom message queues.

5 Show the properties dialog for the Generic Send Fault Model Event block:



Block Label: GEDP-IO-POST-CDG-EVENT-BLOCK-983530

☑ Activated

Event Name: High

☑ Send True Event ☐ If Event Exists

☐ Send Suspect Event ☐ If Event Exists

☑ Send False Event ☑ If Event Exists

Description:

The Event Name of the SymCure event that is posted is High. The High event is posted when the input value becomes either true or false.

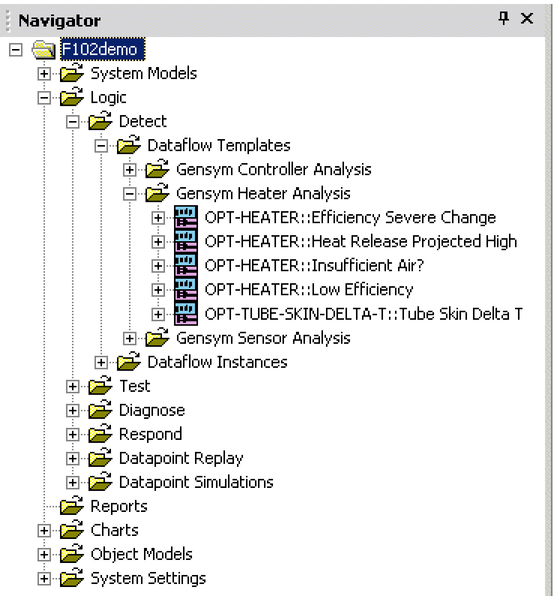
6 Choose Project > Logic > Detect > Dataflow Templates > Gensym Heater Analysis and explore the other generic event detection templates for the heater.

Notice that the OPT-HEATER::Low Efficiency generic event detection template also generates a SymCure event called Excess Coking, when the Efficiency sensor is too low and the Oxygen sensor is not too high. You will observe the effects of this event later in this tutorial.

You can also view the properties of the generic event detection template. One way to access this dialog is from the Project > Event Detection > Generic Event Detection > Manage dialog. Another way is from the Project window.

7 Choose View > Navigator or click the equivalent toolbar button to display a tree view of the overall project.

8 Expand the Logic > Detect > Dataflow Templates > Gensym Heater Analysis node:

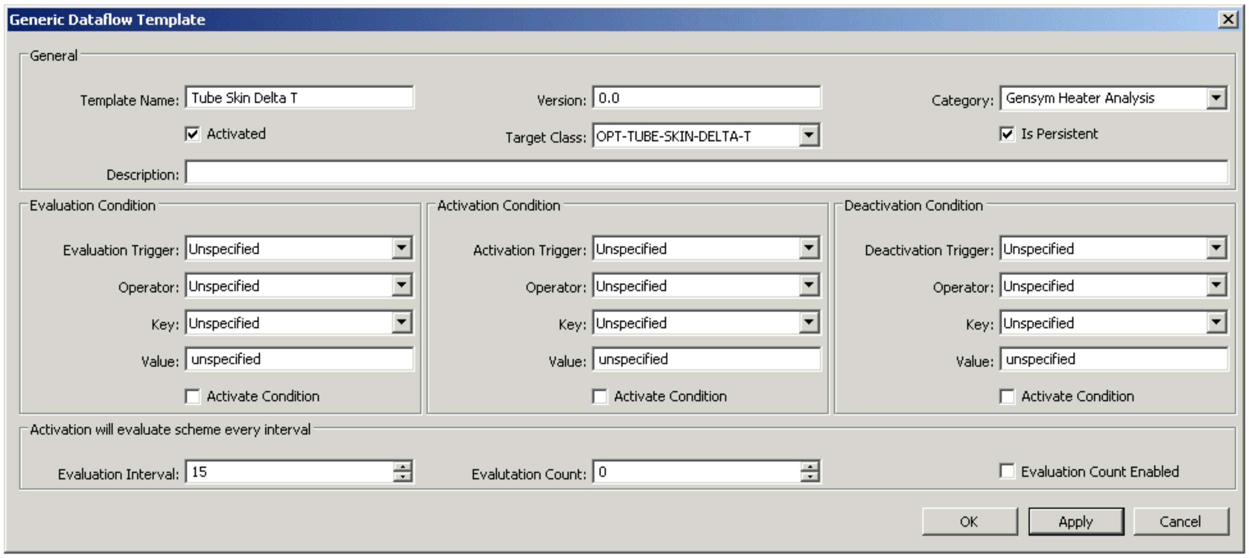


Navigator

- F102demo
  - System Models
  - Logic
    - Detect
      - Dataflow Templates
        - Gensym Controller Analysis
        - Gensym Heater Analysis
          - OPT-HEATER::Efficiency Severe Change
          - OPT-HEATER::Heat Release Projected High
          - OPT-HEATER::Insufficient Air?
          - OPT-HEATER::Low Efficiency
          - OPT-TUBE-SKIN-DELTA-T::Tube Skin Delta T
        - Gensym Sensor Analysis
      - Dataflow Instances
    - Test
    - Diagnose
    - Respond
    - Datapoint Replay
    - Datapoint Simulations
  - Reports
  - Charts
  - Object Models
  - System Settings

9 Choose Properties on the OPT-TUBE-SKIN-DELTA-T::Tube Skin Delta T generic template:

Here is the properties dialog for the generic template:



Generic Dataflow Template

General

Template Name: Tube Skin Delta T
Version: 0.0
Category: Gensym Heater Analysis
☑ Activated
Target Class: OPT-TUBE-SKIN-DELTA-T
☑ Is Persistent
Description:

Evaluation Condition

Evaluation Trigger: Unspecified
Operator: Unspecified
Key: Unspecified
Value: unspecified
☐ Activate Condition

Activation Condition

Activation Trigger: Unspecified
Operator: Unspecified
Key: Unspecified
Value: unspecified
☐ Activate Condition

Deactivation Condition

Deactivation Trigger: Unspecified
Operator: Unspecified
Key: Unspecified
Value: unspecified
☐ Activate Condition

Activation will evaluate scheme every interval

Evaluation Interval: 15
Evalutation Count: 0
☐ Evaluation Count Enabled

Template Name uniquely identifies the generic event detection template for the class. SymCure fault models can refer to this name to trigger event detection diagrams that perform test actions.

Version has no operational value and can be used for application documentation purposes.

Target Class is the domain object class to which the generic template is applied, in this case, the OPT-TUBE-SKIN-DELTA-T class.

Category provides a way of organizing templates in the Project tree view and the Project > Event Detection > Generic Event Detection menu.

Is Persistent determines whether the specific event detection diagrams created for instances of the target class persist through initialization. Is Persistent is enabled for all built-in generic event detection templates so you can configure limits in the specific diagrams that persist.

Evaluation, Activation, and De-Activation Conditions allow you to control when the specific event detection diagram evaluates. By default, the diagram is triggered when an event occurs.

For information on configuring these properties, see the *G2 Event and Data Processing User's Guide*.

Evaluation Interval specifies the frequency with which the specific diagram evaluates, in seconds. A value of 0 means the diagram evaluates when an event occurs.

Evaluation Count specifies the maximum number of times the specific diagram should evaluate, when Evaluation Interval is set.

Evaluation Count Enabled activates or deactivates the Evaluation Count.

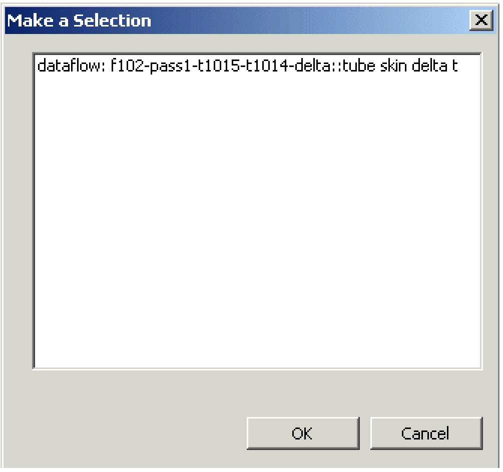
Viewing Specific Event Detection Diagrams

When you initialize the process map, GEDP creates specific event detection diagrams for each instance of the target class for which a generic template is defined. During simulation, the domain object uses the specific diagram for detecting events.

You can view specific event detection diagrams from the Project menu, the Project view, or from a specific domain object.

To view specific event detection diagrams:

1 Display the process map and choose Show Logic on the DT derived sensor:



dataflow: f102-pass1-t1015-t1014-delta::tube skin delta t

2 Choose the specific Tube Skin Delta T event detection diagram and click OK.

The specific event detection diagrams look identical to the generic templates you saw earlier.

3 Show the properties of the High Limit event.

The High Limit is 10 for the specific diagram.

Now that you have seen how the Optegrity Heater Tutorial detects events, you are ready to view and interact with SymCure events and diagnostic console.

8

Interacting with SymCure Events

Describes how to view and interact with SymCure alarm and root cause events in the Alarm Browser and Root Cause Browser for the Optegrity Heater Tutorial.

Introduction

When GEDP generates a SymCure event for a domain object, SymCure performs diagnostic reasoning about the event to determine root causes. SymCure uses **generic fault models** in the form of graphical causal models to describe causal relationships between events for domain object classes. SymCure fault models include:

- **Alarms**, which are events that are effects of underlying faults, about which operators need to be notified. They can be either observed directly or inferred through diagnostic reasoning.
- **Root causes**, which are the underlying faults that cause alarms and other events.

Any event that is observed to be true is called a **symptom**. Any event that is inferred through diagnostic reasoning is known as a **prediction**.

When an event occurs for a domain object, SymCure creates a **specific fault model**. SymCure's **fault propagation** algorithm uses the specific model to propagate event values to causally related events.

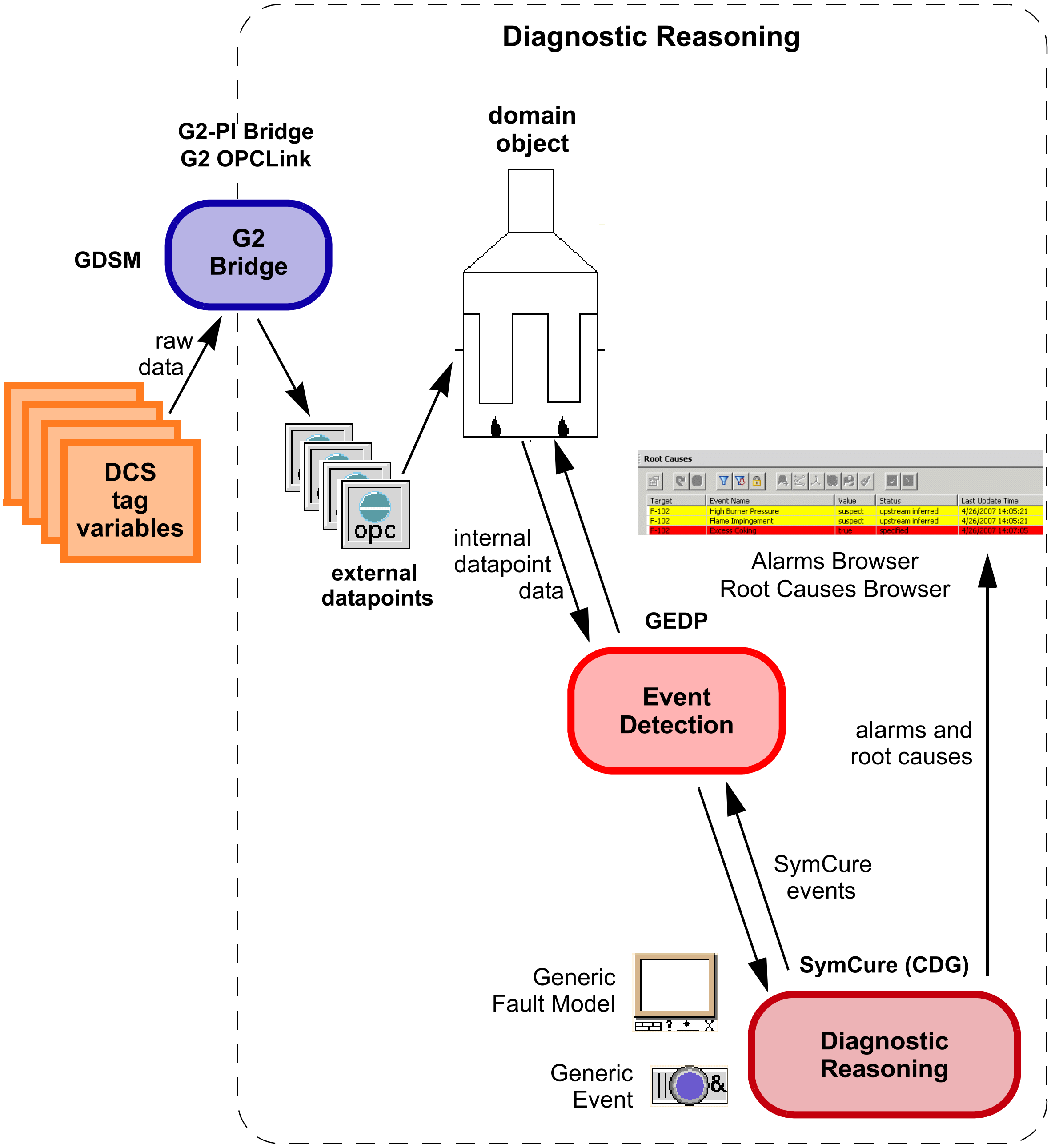
An upstream event can cause multiple downstream events to occur. Similarly, a downstream event might be caused by multiple upstream events. In the Optegrity Heater Tutorial, SymCure uses "or-or" logic to determine event status, whereby:

- An event is true if any one of its upstream events is true ("or" logic).
- When an event is true, at least one of its downstream events is true ("or" logic).

You view and interact with alarms and root causes for specific domain objects in the Alarms Browser and Root Causes Browser. These browsers provide a tabular view of specific events and additional toolbar buttons for interacting with alarms and root causes.

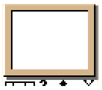
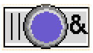
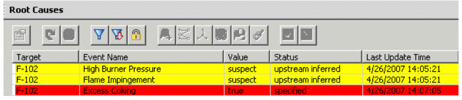
For detailed information about SymCure generic and specific fault models, as well as information on other types of event propagation logic, see the *SymCure User's Guide*.

This figure shows the architecture of the Optegrity Heater Tutorial from SymCure's diagnostic reasoning perspective. It includes a Generic Fault Model Folder, Generic Event, and Alarms Browser, and Root Causes Browser.



Root Causes

Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Flame Impingement	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Excess Coking	true	specified	4/26/2007 14:07:05

Object	Name	Description
	Generic Fault Model Folder	Defines a generic SymCure fault model container associated with a domain object class in which you place generic fault models.
	Generic Event	Defines a generic SymCure event associated with a domain object class. A generic event can be a type of alarm or root cause, or its type can be unspecified.
	Alarms Browser Root Causes Browser	Provides an interface for viewing and interacting with alarms and root causes.

Root Causes

Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Flame Impingement	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Excess Coking	true	specified	4/26/2007 14:07:05

Viewing Root Causes

Let's look more closely at the SymCure root cause events that occur in the Optegrity Heater Tutorial when you replay data from a CSV file.

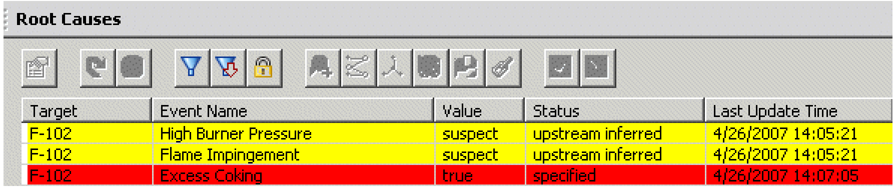
To view root causes:

1 Follow the steps under Simulating External Datapoint Values to run the simulation again.

2 Choose Project > Logic > Diagnose > Diagnostic Console > Root Causes or click the equivalent toolbar button: ( )

Tip Choose View > Toolbars > Fault Modeling to show the Fault Modeling toolbar.

When the simulation is complete, the Root Causes Browser has these messages:



Root Causes

Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Flame Impingement	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Excess Coking	true	specified	4/26/2007 14:07:05

When the High Tube Skin Delta T event in the F102-PASS1-T1025-T1014-DELTA::Tube Skin Delta T specific event detection diagram is true, these root causes appear in the Root Causes Browser:

- High Burner Pressure is suspect.
- Flame Impingement is suspect.
- Excess Coking is true.

Understanding Event Value and Event Status

Events can have one of these values:

This value...	Means...
true	The event is known or inferred to be true.
suspect	The event is suspected to be true, based on the value of upstream and/or downstream events.
false	The event is known or inferred to be false.
unknown	The value of the event is unknown until further diagnostics are run.

The event value can be specified, which means that it is obtained from a GEDP diagram or observed by an operator, or it can be inferred, which means it gets it value, based on upstream or downstream propagation. In this tutorial, if an event is true, all events downstream of that event are inferred to be suspect, based on downstream propagation. An upstream event is inferred to be true if it is the only event upstream of a true event. An upstream event is inferred to be suspect if it is one of multiple events upstream of a true event.

The color of the event indicates its value and status, as follows:

This color...	Indicates a value of...	And a status of...	Which means...
red	true	specified	The event is known to be true, because it was either observed or detected by an external source, such as an event detection diagram.
salmon	true	upstream inferred or downstream inferred	The event is inferred to be true, based on diagnostic reasoning.

This color...	Indicates a value of...	And a status of...	Which means...
yellow	suspect	upstream inferred, downstream inferred	The event is inferred to be suspect, based on diagnostic reasoning.
yellow-green	false	upstream inferred or downstream inferred	The event is inferred to be false, based on upstream or downstream propagation.
green	false	specified	The event is known to be false, because it was either observed or detected by an external source.
blue	unknown	unknown	The value and status of the event is unknown.

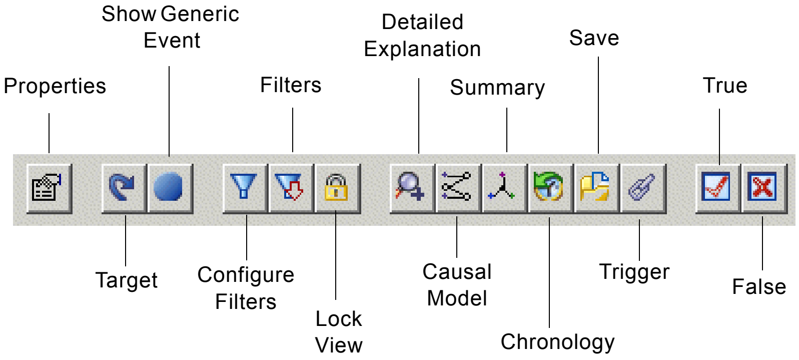
Based on these definitions, this table describes the value and status of the root causes that appear when you run the simulation:

This event...	Has a value of...	And a status of...	Which means it is...
Excess Coking	false	specified	Observed to be true.
High Burner Pressure	suspect	upstream inferred	Inferred to be true, based on upstream propagation.
Flame Impingement	suspect	upstream inferred	Inferred to be true, based on upstream propagation.

You will learn more about specified and inferred events later in this tutorial when you look at the SymCure generic fault models.

Interacting with the Root Causes Browser

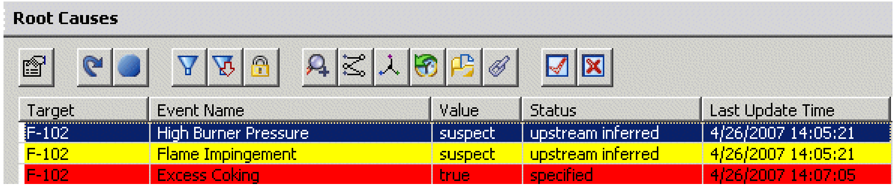
The Root Causes Browser shows the target, event name, event value, and last update time. You interact with root causes, using these toolbar buttons. The buttons are similar to those in the Alarms Browser.



To interact with the Root Causes Browser:

➔ Select a root cause in the browser.

The root cause event is highlighted and the toolbar buttons become active:


Root Causes

Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Flame Impingement	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Excess Coking	true	specified	4/26/2007 14:07:05

Showing the Specific Fault Model for an Event

You can view the specific fault model for an alarm or root cause, which shows all upstream and downstream events for the particular event. The specific fault model is also called a **causal model** because it describes the causal relationships between events.

A causal model uses the same colors that the browsers use to represent event value and status:

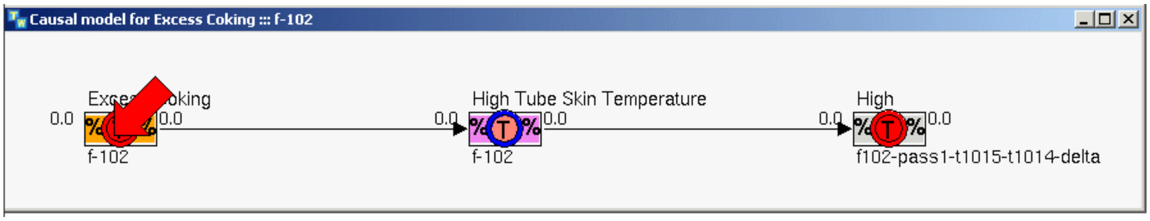
This color...	Represents this type of event...
Red	An event that is observed to be true.
Yellow	An event that is inferred to be suspect.
Salmon	An event that is inferred to be true, based on upstream or downstream propagation. If the predicted event is an alarm, it appears in the Alarms Browser. If the predicted event is a root cause, it appears in the Root Causes Browser. If the predicted event is unspecified, it does not appear in any browser.

Tip You can adjust the size of the causal model workspace and move the events, as needed, to view the workspace more clearly.

To show the specific fault model for an event:

➔ Select the Excess Coking event in the Root Causes Browser, then click the Causal Model toolbar button:

Here is the specific fault model for the Excess Coking event for the F-102 heater. It shows all upstream and downstream events, relative to the selected event.


Causal model for Excess Coking ::: f-102
Exce oking
0.0
%
%
0.0
f-102
High Tube Skin Temperature
0.0
%
T
%
0.0
f-102
High
0.0
%
T
%
0.0
f102-pass1-t1015-t1014-delta

Showing the Event Summary

You can view an **event summary** for alarms and root causes, which is an abstraction of the root causes, observed alarms, predicted alarms, and actions that are related to a particular event. Rather than showing all the causal relationships between events, the event summary shows just the endpoints.

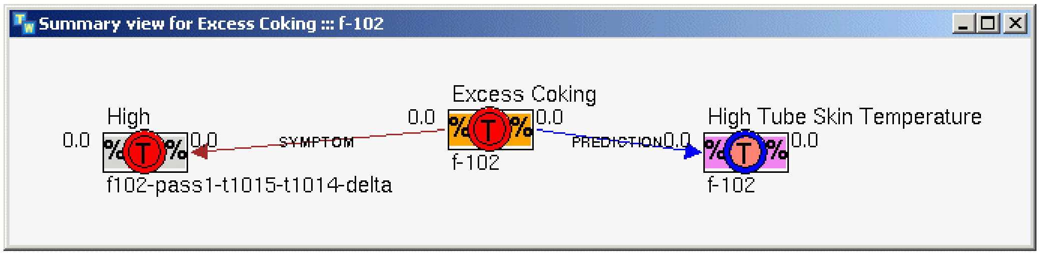
The event summary labels the type of relationship that exists between events, as follows:

This connection label...	Means...
Root Cause	One event is the root cause of another event.
Symptom	One event is an observed symptom that is caused by the other event.
Prediction	One event is a predicted alarm for the other event.
Action	The action is created as a result of the event.

To show the event summary for an event:

➔ Select the Excess Coking reading event in the Root Causes Browser, then click the Summary toolbar button:

Here is the event summary for the Excess Coking event for the F-102 heater. It shows the observed alarm (red) f102-pass1-t1015-t1014-delta, which is a symptom of the Excess Coking (red) root cause. It also shows the High Tube Skin Temperature event, which is related to the root cause.


Summary view for Excess Coking ::: f-102
High
0.0
0.0
f102-pass1-t1015-t1014-delta
SYMPTOM
Excess Coking
0.0
0.0
f-102
PREDICTION
High Tube Skin Temperature
0.0
0.0
f-102

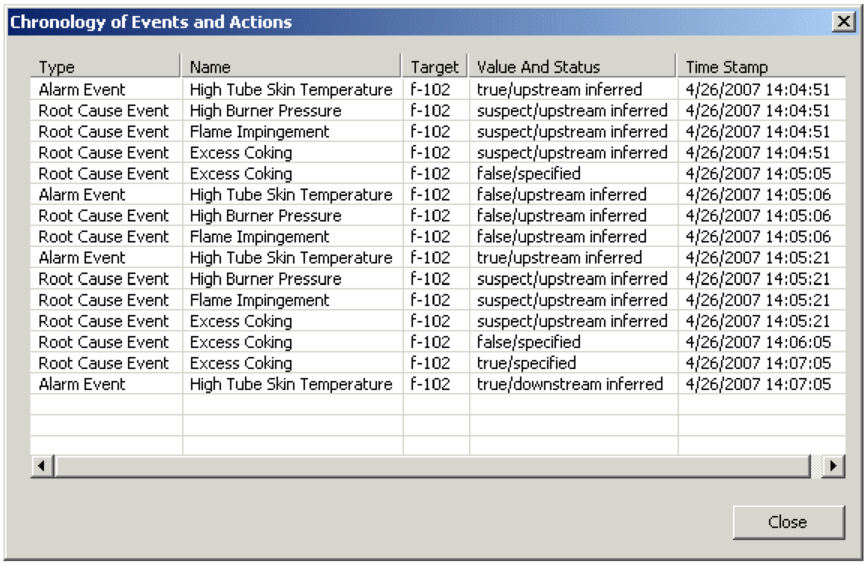
Showing and Saving the Event Chronology

You can view the chronology of events led up to a particular alarm or root cause becoming true. The chronology includes the type, which is either event or action, name, target object, value, status, and timestamp. You can also save the event chronology to a CSV file for further diagnosis. The event chronology is useful for logging purposes. Only alarm and root cause events, and actions appear in the event chronology.

To show the event chronology:

➔ Select the Excess Coking event in the Root Causes Browser, then click the Chronology toolbar button:

Here is the event chronology for the Excess Coking event:



Type	Name	Target	Value And Status	Time Stamp
Alarm Event	High Tube Skin Temperature	f-102	true/upstream inferred	4/26/2007 14:04:51
Root Cause Event	High Burner Pressure	f-102	suspect/upstream inferred	4/26/2007 14:04:51
Root Cause Event	Flame Impingement	f-102	suspect/upstream inferred	4/26/2007 14:04:51
Root Cause Event	Excess Coking	f-102	suspect/upstream inferred	4/26/2007 14:04:51
Root Cause Event	Excess Coking	f-102	false/specified	4/26/2007 14:05:05
Alarm Event	High Tube Skin Temperature	f-102	false/upstream inferred	4/26/2007 14:05:06
Root Cause Event	High Burner Pressure	f-102	false/upstream inferred	4/26/2007 14:05:06
Root Cause Event	Flame Impingement	f-102	false/upstream inferred	4/26/2007 14:05:06
Alarm Event	High Tube Skin Temperature	f-102	true/upstream inferred	4/26/2007 14:05:21
Root Cause Event	High Burner Pressure	f-102	suspect/upstream inferred	4/26/2007 14:05:21
Root Cause Event	Flame Impingement	f-102	suspect/upstream inferred	4/26/2007 14:05:21
Root Cause Event	Excess Coking	f-102	suspect/upstream inferred	4/26/2007 14:05:21
Root Cause Event	Excess Coking	f-102	false/specified	4/26/2007 14:06:05
Root Cause Event	Excess Coking	f-102	true/specified	4/26/2007 14:07:05
Alarm Event	High Tube Skin Temperature	f-102	true/downstream inferred	4/26/2007 14:07:05

To save the event chronology:

1 Select the Excess Coking event in the Root Causes Browser, then click the Save toolbar button:

2 Specify a CSV file in which to save the event chronology.

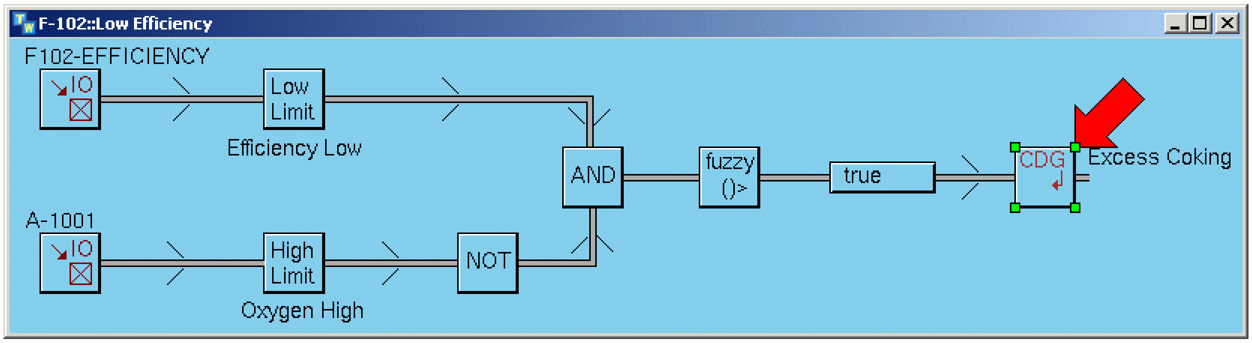
Showing the Trigger for an Event

You can view the specific event detection diagram that triggers an alarm or root cause, if one exists.

To show the event trigger for an event:

➔ Select the Excess Coking event in the Root Causes Browser, then click the Trigger toolbar button:

Here is specific event detection diagram for the Excess Coking event for the F-102 heater. The arrow points to the GEDP block that sends the SymCure event.



Simulating Event Value Changes

You might want to simulate the effect of changing the value of an alarm or root cause from true to false, or from false to true. You can do this from the Alarms or Root Causes Browser or from a specific fault model for an event.

To simulate event value changes:

➔ Select the High Burner Pressure event, and click the False toolbar button to change the value to false: ☒

You can also display the event summary, then choose False from the menu for a specific event.

Simulating Events for Domain Objects

The SymCure generic fault models define numerous events for the various domain objects in the process map. One way to trigger these events is by using an event detection diagram, as shown earlier.

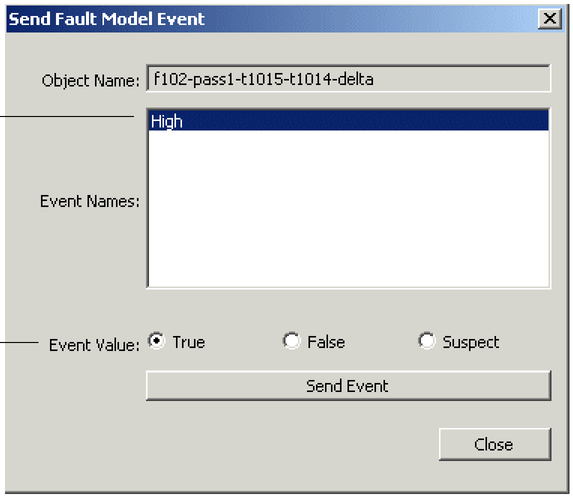
You might also want to simulate events to test your generic fault model. You can simulate events value to be true, false, or suspect.

To simulate an event for a domain object:

1. Choose Project > Initialize Application to clear all messages from the browsers.
2. Display the F-102 process map.
3. Show the popup menu for the F102-PASS1-T1015-T1014-DELTA Tube Skin Delta T sensor and choose Send Fault Model Event.

   This dialog shows all the events associated with the sensor.
4. Select High, choose an Event Value of true, then click Send Event to send the event.

   The dialog looks like this:



Send Fault Model Event

Object Name: f102-pass1-t1015-t1014-delta

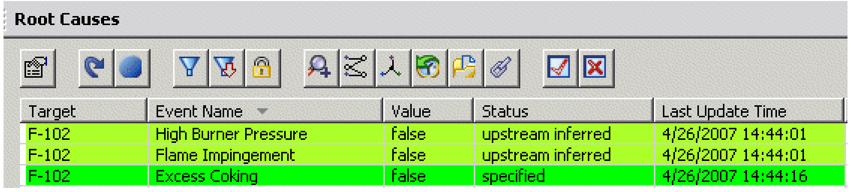
Event Names: High

Event Value: ● True ○ False ○ Suspect

Send this event...

...with this value.

Simulating this events triggers the same suspected root causes as triggered by the event detection diagram. However, because the event was simulated directly rather than being the result of event detection, the Message Browser only shows the suspected root causes. It does not show the message associated with the F102-PASS1-T1015-T1014-DELTA high limit.



Root Causes

Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	false	upstream inferred	4/26/2007 14:44:01
F-102	Flame Impingement	false	upstream inferred	4/26/2007 14:44:01
F-102	Excess Coking	false	specified	4/26/2007 14:44:16

Now that you have explored SymCure events, you will explore the test and repair actions.

Performing External Actions

Describes how SymCure creates test actions and repair actions, and how you interact with those actions through browsers in the Optegrity Heater Tutorial.

Introduction

As part of diagnosing root causes, SymCure can perform **actions**, which include:

- **Test actions**, which test the value of the underlying event and return a value of true or false, depending on the outcome of the test.
- **Repair actions**, which take corrective action or invoke some other type of proactive action, based on the underlying event.

For example, in the Optegrity Heater Tutorial, when the Flame Impingement root cause occurs on the F-102 heater, the specific fault model creates the Flame Impingement test action, which allows you to send a true, false, or unknown value for the Flame Impingement suspected root cause. In addition, when the High Burner Pressure root cause occurs on the F-102 heater, the specific fault model creates the Adjust Burners repair action, which runs automatically when the High Burner Pressure root cause becomes true.

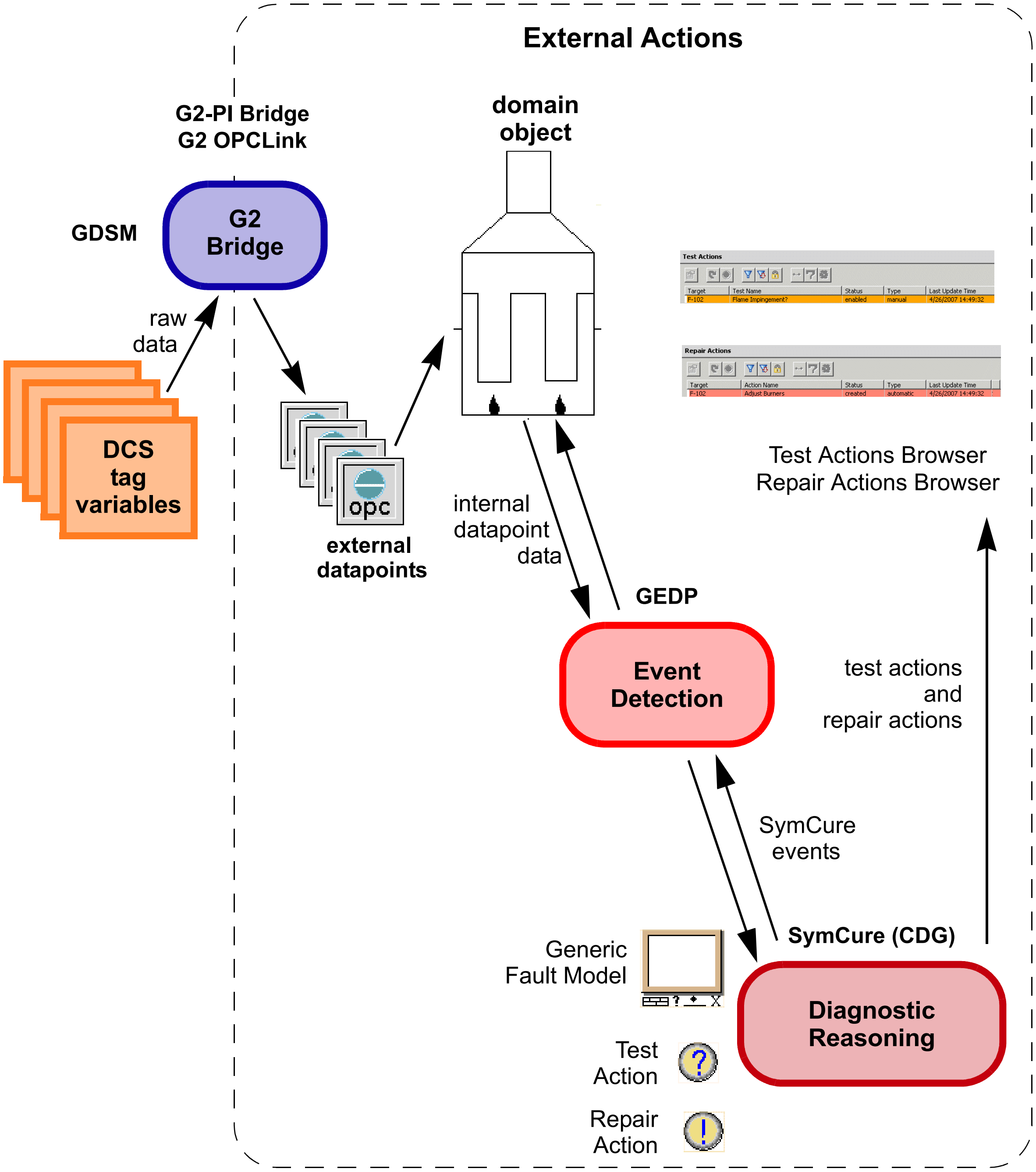
Before an action can execute, it must be enabled. SymCure enables actions, as follows:

SymCure enables this type of action...	When the underlying event becomes...
Test action	Suspect or unknown
Repair action	True

An action can be manual or automatic. When a manual action is enabled, you must execute it explicitly in the Test Action or Repair Action browser. When an automatic action is enabled, it executes automatically.

You view and interact with test actions and repair actions for specific domain objects in the Test Actions Browser and Repair Actions Browser. These browsers provide additional information relating to events and additional toolbar buttons for interacting with test and repair actions.

This figure shows the architecture of SymCure external actions for the Optegrity Heater Tutorial. It includes a Generic Fault Model Folder, Test Actions Browser, and Repair Actions Browser.

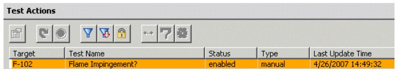
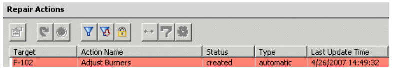


Test Actions

Target	Test Name	Status	Type	Last Update Time
F-102	Flame Impingement?	enabled	manual	4/26/2007 14:49:32

Repair Actions

Target	Action Name	Status	Type	Last Update Time
F-102	Adjust Burners	created	automatic	4/26/2007 14:49:32

Object	Name	Description
□	Generic Fault Model	Defines a container for a generic fault model, which can be associated with a domain object class.
?	Test Action	Defines a generic action associated with a domain object class and a generic event, which tests the value of an event and returns the value.
!	Repair Action	Defines a generic action associated with a domain object class and a generic event, which sets the value of an event.
 	Test Actions Browser Repair Actions Browser	Provides an interface for viewing and interacting with test actions and repair actions.

Test Actions

Target	Test Name	Status	Type	Last Update Time
F-102	Flame Impingement?	enabled	manual	4/26/2007 14:49:32

Repair Actions

Target	Action Name	Status	Type	Last Update Time
F-102	Adjust Burners	created	automatic	4/26/2007 14:49:32

Viewing Test and Repair Actions

Now, you will look more closely at the test and repair actions that occur in the Optegrity Heater Tutorial when you replay data from a CSV file.

To view test actions:

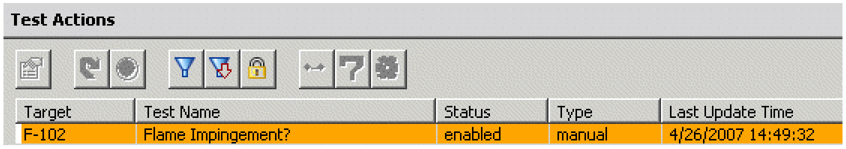
➔ After running the simulation, choose Project > Logic > Diagnose > Diagnostic Console > Test Actions or click the equivalent toolbar buttons: ( )

To view repair actions:

➔ After running the simulation, choose Project > Logic > Diagnose > Diagnostic Console > Repair Actions or click the equivalent toolbar buttons: ( )

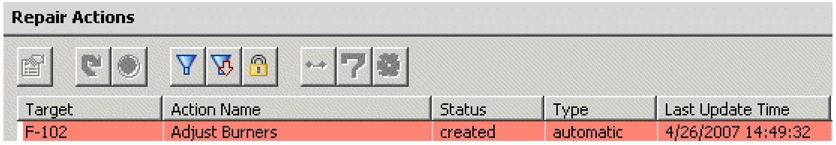
If you have restarted Optegrity, follow the steps under Simulating External Datapoint Values to run the simulation again.

Here are the Test Actions and Repair Actions Browsers that appear when the simulation completes:



Test Actions

Target	Test Name	Status	Type	Last Update Time
F-102	Flame Impingement?	enabled	manual	4/26/2007 14:49:32



Repair Actions

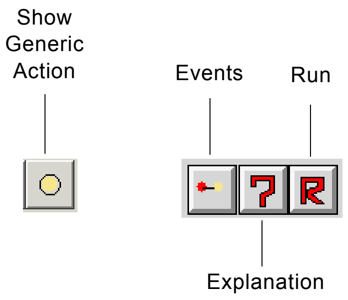
Target	Action Name	Status	Type	Last Update Time
F-102	Adjust Burners	created	automatic	4/26/2007 14:49:32

These test and repair actions appear:

- The Flame Impingement? manual test action on the F-102 heater is enabled.
- The Adjust Burners automatic repair action executes for the F-102 heater.

Interacting with the Test Actions Browser

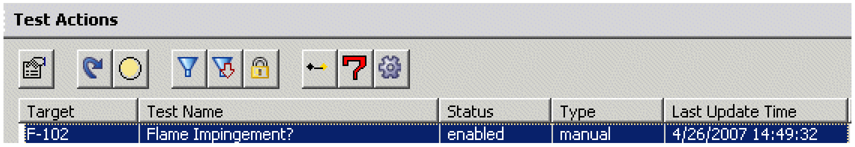
The Test Actions Browser shows the target, test name, status, type, and last update time. You interact with tests, using these toolbar buttons:



To interact with the Test Actions Browser:

➔ Select the test in the Test Actions browser.

The test action is highlighted and the toolbar buttons become active:



Test Actions

Target	Test Name	Status	Type	Last Update Time
F-102	Flame Impingement?	enabled	manual	4/26/2007 14:49:32

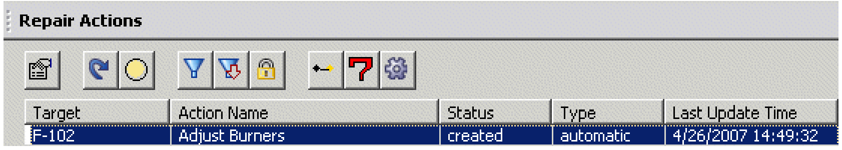
Interacting with the Repair Actions Browser

The Repair Actions Browser shows the target, test name, status, type, and last update time. You interact with the Repair Actions Browser, using the toolbar buttons, which are the same as in the Test Actions Browser.

To interact with the Repair Actions Browser:

➔ In the Repair Actions browser, select a repair action.

The repair action is highlighted and the toolbar buttons become active:



Target	Action Name	Status	Type	Last Update Time
F-102	Adjust Burners	created	automatic	4/26/2007 14:49:32

Interacting with Test Actions

The Optegrity Heater Tutorial generates a test action as the result of running the F102-Pass1-T1025-T1014-Delta event detection diagram. Recall that this model monitors the pv of the F102-PASS1-T1015-T1014-DELTA sensor and generates a High event if the tube skin delta temperature exceeds 20. The event detection diagram returns the value of the event to SymCure.

The test action is associated with the Flame Impingement root cause of the High Tube Skin Temperature event.

You can view the underlying event that triggers the test action. You can also run the test and explain the event that caused the test to be created.

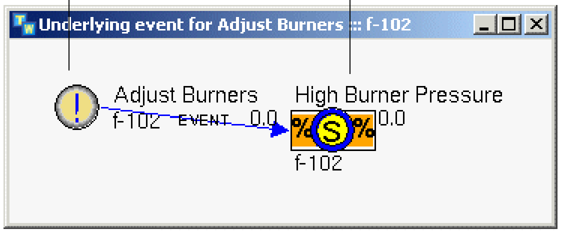
Viewing the Underlying Event for the Test Action

To view the underlying event for the test action:

1 Select the Flame Impingement test action in the Test Actions Browser:

2 Click the Events toolbar button:

Here is the underlying event for the test action. When the underlying event becomes suspect, the test action is created and activated.



The Flame Impingement test action is associated with the Flame Impingement event.

The test is created when the Flame Impingement event becomes suspect.

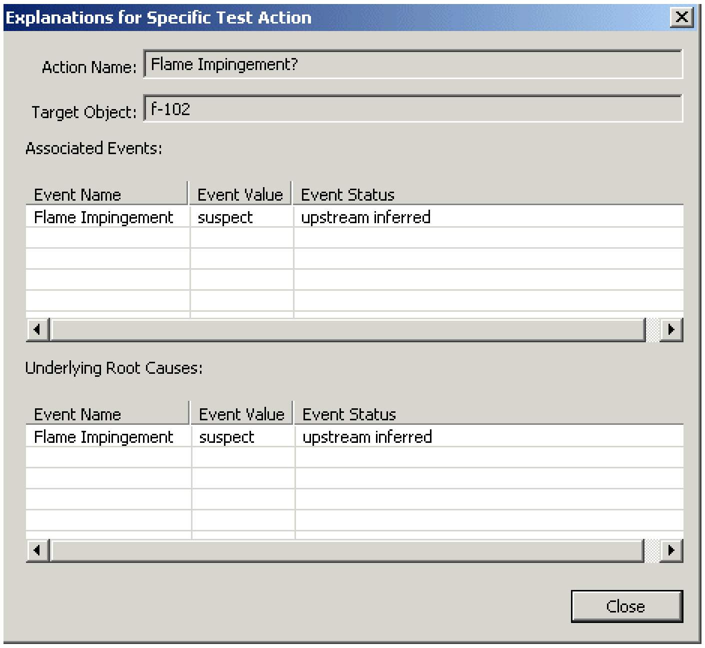
Explaining Test Actions

You can view an explanation of the event that caused the action to occur.

To explain the test action:

➔ Select the Flame Impingement test action and click the Explanation toolbar button: ?

The Flame Impingement test was generated when the upstream Flame Impingement event on the F-102 heater was inferred to be suspect. Its underlying root cause, Flame Impingement, is inferred to be suspect.



Action Name: Flame Impingement?

Target Object: f-102

Associated Events:

Event Name	Event Value	Event Status
Flame Impingement	suspect	upstream inferred

Underlying Root Causes:

Event Name	Event Value	Event Status
Flame Impingement	suspect	upstream inferred

Running the Test Action

The Flame Impingement test is a manual test, which means you must explicitly run it in the browser. In this case, the test simply displays a dialog that allows you to enter a value of true, false, or unknown for the underlying event.

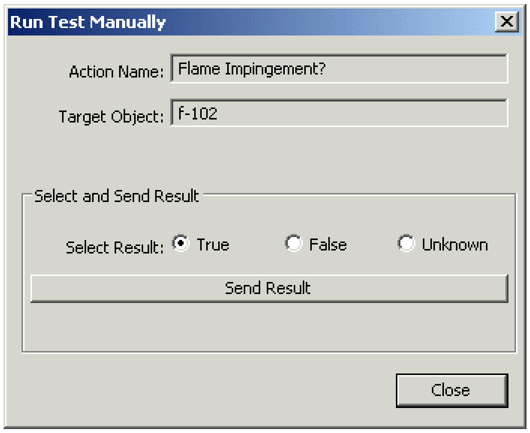
To run the test action:

1 Select the Flame Impingement test action in the Test Actions Browser.

2 Click the Run toolbar button to start the manual test: ⚙

A dialog appears for specifying the value of the underlying Flame Impingement event.

3 Configure the dialog to send a value of true for the underlying event:



Run Test Manually

Action Name: Flame Impingement?

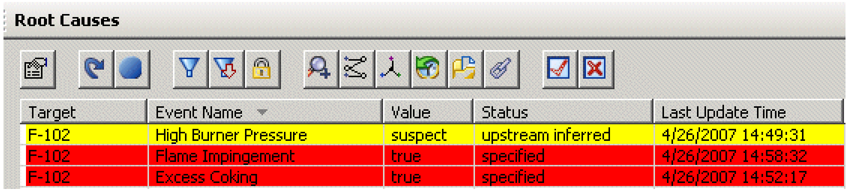
Target Object: F-102

Select and Send Result

Select Result: ● True ○ False ○ Unknown

4 Click the Send Result button.

As a result of running the test, the Flame Impingement root cause is specified as true and therefore turns red, as does the Excess Coking event. The High Burner Pressure root cause is still suspect.



Root Causes

Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	suspect	upstream inferred	4/26/2007 14:49:31
F-102	Flame Impingement	true	specified	4/26/2007 14:58:32
F-102	Excess Coking	true	specified	4/26/2007 14:52:17

Interacting with Repair Actions

When the High Burner Pressure root cause occurs, the specific propagation model creates the Adjust Burners repair action. You can view the underlying event that creates the repair action. The repair action has no associated event detection diagram.

Because the High Burner Pressure repair action is automatic, the action executes when the underlying action becomes true. Initially, the underlying event is suspect, so to execute it, you must manually set the High Burner Pressure root cause to true.

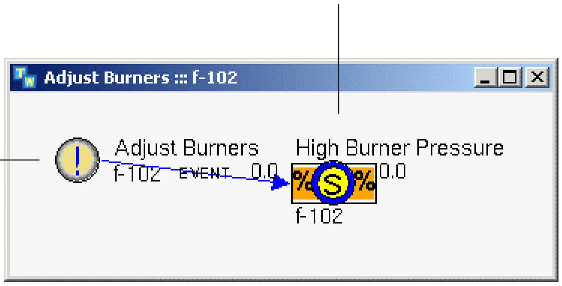
Viewing the Underlying Event for the Repair Action

To view the underlying event for the repair action:

1. Select the Adjust Burners repair action in the Repair Actions Browser:
2. Click the Events toolbar button:

Here is the underlying event for the Adjust Burners repair action. When the High Burner Pressure root cause becomes suspect, the Adjust Burners repair action is created. When the High Burner Pressure event becomes true, the Adjust Burners repair action is automatically executed.

Underlying event is suspect.



Repair action

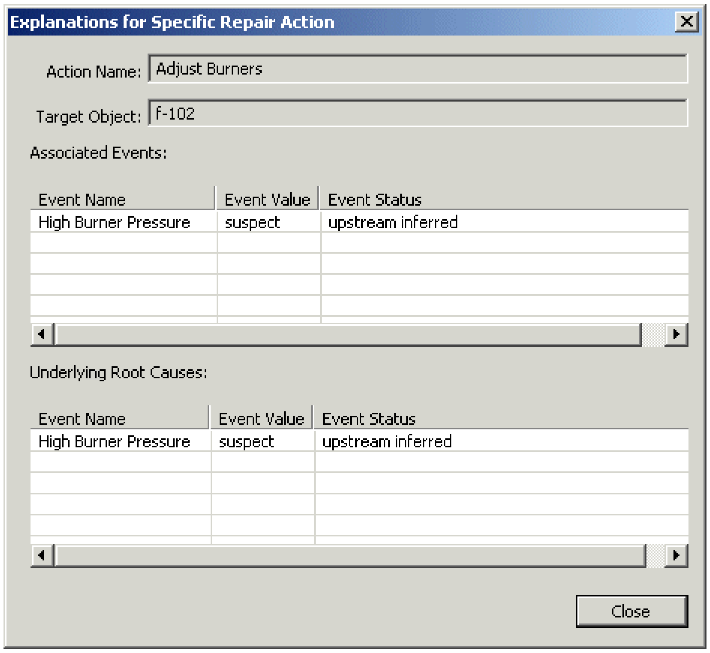
Explaining Repair Actions

You can view an explanation of the event that caused the repair action to occur.

To explain the repair action:

➔ Select the repair action and click the Explanation toolbar button:

The Adjust Burners repair action was created when the upstream High Burner Pressure root cause on the F-102 heater was inferred to be suspect. The underlying root cause, High Burner Pressure, is inferred to be suspect.



Action Name: Adjust Burners

Target Object: f-102

Associated Events:

Event Name	Event Value	Event Status
High Burner Pressure	suspect	upstream inferred

Underlying Root Causes:

Event Name	Event Value	Event Status
High Burner Pressure	suspect	upstream inferred

Triggering the Repair Action

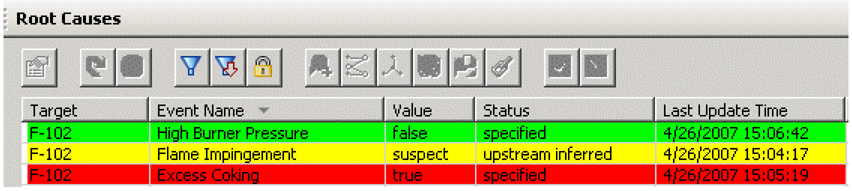
To trigger the automatic Adjust Burners repair action, the High Burner Pressure event must become true. You can simulate this event by manually setting the status of this event to true.

To trigger the repair action:

➔ In the Root Causes Browser, select the High Burner Pressure event and click the True toolbar button to change its status to true: ☑

Setting the status of the High Burner Pressure event to true automatically executes the Adjust Burners repair action. The status changes to running as it executes. After a period of time, the action completes and its status changes to inactive. As a result of the repair action, the value of the High Burner Pressure root cause becomes false, indicating no fault exists.

High Burner Pressure root cause becomes false as a result of the repair action.



Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	false	specified	4/26/2007 15:06:42
F-102	Flame Impingement	suspect	upstream inferred	4/26/2007 15:04:17
F-102	Excess Coking	true	specified	4/26/2007 15:05:19

Now that you have explored test actions and repair actions, you will view the generic fault models that define the causal relationships between events in the Optegrity Heater Tutorial.

10

Viewing Generic Fault Models

Describes how to view the generic fault models for domain objects in the Optegrity Heater Tutorial.

Introduction

For diagnostic applications such as the Optegrity Heater Tutorial, the SymCure generic fault models are at the heart of the application. Recall that:

- A generic fault model consists of events that are connected by causal relationships.
- The fault propagation algorithm uses the generic fault models to determine root causes from symptoms for specific domain objects.
- The SymCure browsers display alarms, root causes, test actions, and repair actions for specific domain objects.

In the previous chapter, you saw examples of specific fault models for events on specific domain objects. Now, you will look at the generic fault models for the various domain objects classes that are the basis for the specific events you encountered during your simulations in the previous chapter.

For detailed information about defining SymCure fault models, see the *SymCure User's Guide*.

Viewing the Generic Fault Model for the Heater

Now, you will investigate how the generic fault model for the Tube Skin Temperature of the heater triggers the root cause events that you saw in the previous chapter.

You can view generic fault models from the Project > Logic > Diagnose > Generic Fault Models menu. You can also view them directly from an event in one of the SymCure diagnostic console browsers.

To view the generic fault model for the heater:

1 Initialize the application and run the simulation again.

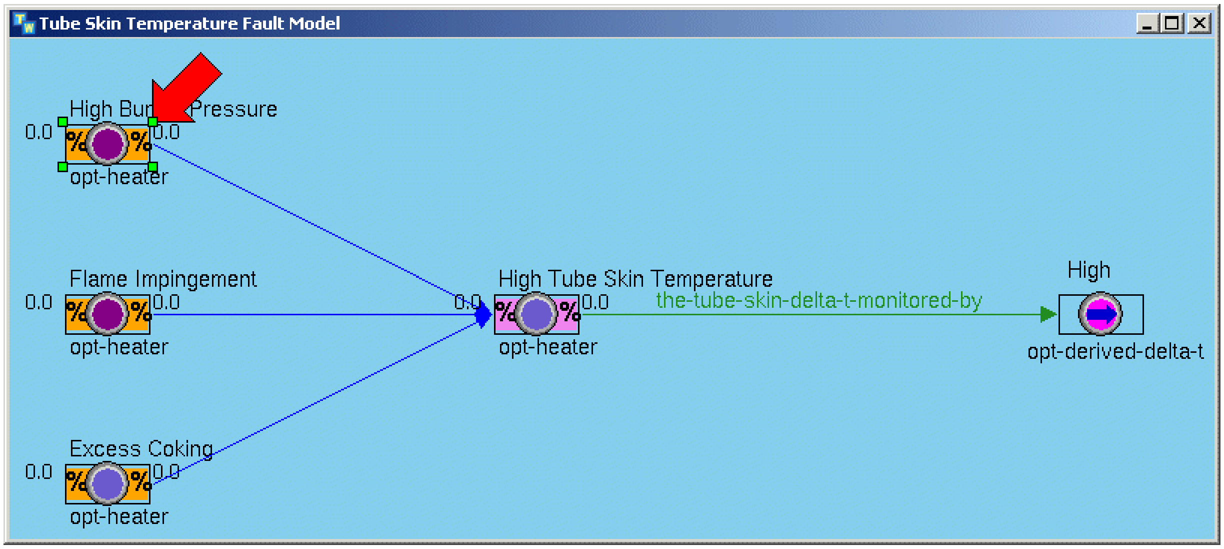
2 Display the Root Causes browser.

  You should see the three root cause events you saw earlier: Excess Coking is false, High Burner Pressure is suspect, and Flame Impingement is suspect.

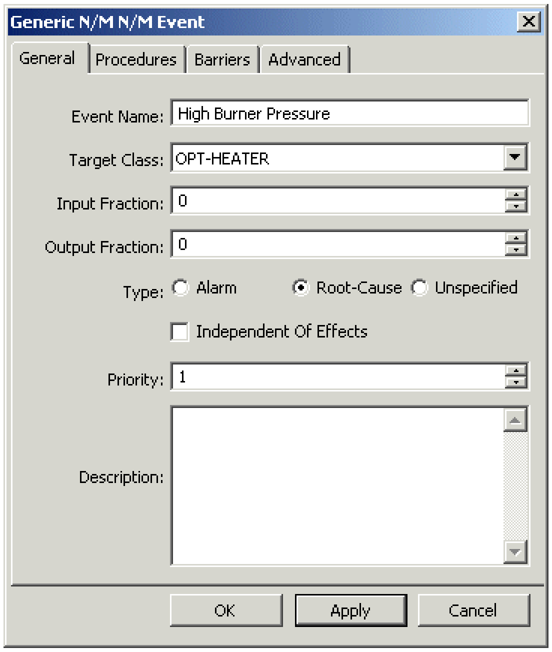
3 Select the High Burner Pressure root cause and click the Show Generic Event button (●).

You can also access the generic fault model by choosing Project > Fault Models > Generic Fault Models > Optegrity Heater > Tube Skin Temperature Fault Model.

Here is the generic fault model for the Tube Skin Temperature of the heater, which highlights the High Burner Pressure generic event as one of the three possible root causes of the High Tube Skin Temperature event. The other possible root causes are Flame Impingement and Excess Coking. These are the three events you saw earlier in the Root Causes browser. The High event, which is downstream of the High Tube Skin Temperature event, is the event you simulated for the Tube Skin Delta T sensor earlier.



4 Show the properties of the High Burner Pressure event:



Generic N/M N/M Event

Event Name: High Burner Pressure

Target Class: OPT-HEATER

Input Fraction: 0

Output Fraction: 0

Type: ○ Alarm ● Root-Cause ○ Unspecified

□ Independent Of Effects

Priority: 1

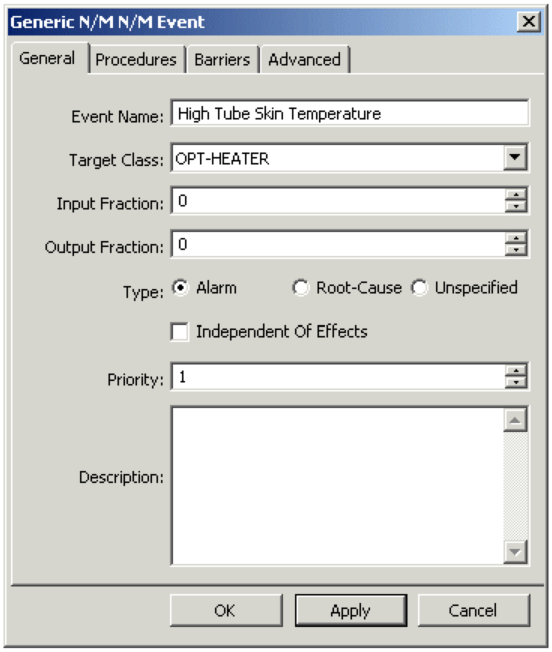
Description:

The High Burner Pressure event is an N/M N/M event, which is defined on the opt-heater target class. The event Type is Root-Cause, which means it appears in the Root Causes browser.

The Input Fraction and Output Fraction describe the percentage of inputs and outputs that must be true during upstream and downstream propagation, respectively, for the event to be true. The default values are both 0, which means the event uses OR logic on the input and output of the event.

5 View the properties of the Flame Impingement and Excess Coking root causes, which are configured just like the High Burner Pressure event.

6 Show the properties of the High Tube Skin Temperature event:



Event Name: High Tube Skin Temperature

Target Class: OPT-HEATER

Input Fraction: 0

Output Fraction: 0

Type: ● Alarm ○ Root-Cause ○ Unspecified

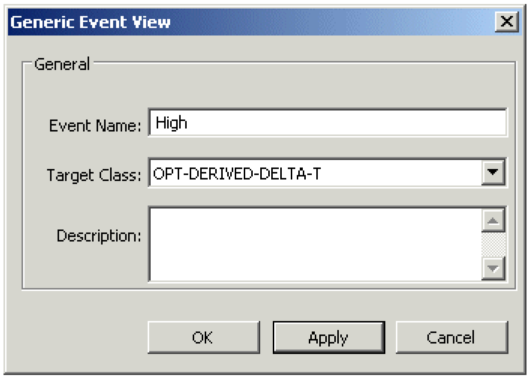
□ Independent Of Effects

Priority: 1

Description:

The event Type is Alarm, which means it appears in the Alarms browser.

7 Show the properties of the High event:



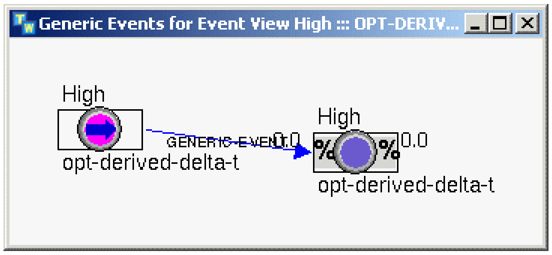
Event Name: High

Target Class: OPT-DERIVED-DELTA-T

Description:

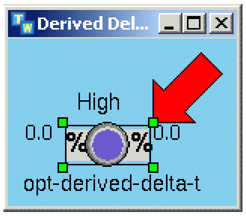
As you can see from the properties dialog, as well as from the icon, this event is a generic event view, which means it is simply a reference to a generic event. You create generic event views to refer to an event in multiple models.

8 Choose Show Generic Events on the event view:



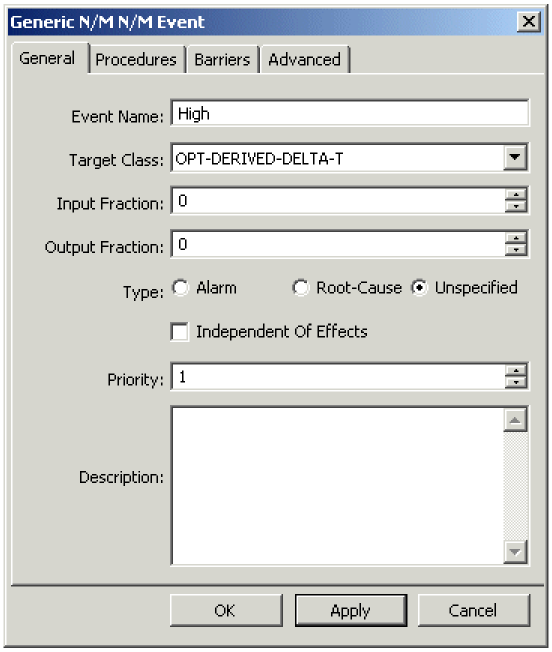
The generic event that is associated with the generic event view appears.

9 Choose Show Generic Event on the High event in the diagram to go to the generic event in the Derived Delta T generic fault model:



This generic fault model has a single event; the event has no upstream causes or downstream effects.

10 Show the properties of the High generic event:


Generic N/M N/M Event

General | Procedures | Barriers | Advanced

Event Name: High

Target Class: OPT-DERIVED-DELTA-T

Input Fraction: 0

Output Fraction: 0

Type: ○ Alarm ○ Root-Cause ● Unspecified

□ Independent Of Effects

Priority: 1

Description:

The event Type is also Undefined, which means it does not appear in any of the diagnostic consoles; it is used for diagnostic reasoning only.

Now you will look more closely to see how SymCure uses the generic fault model to create specific fault models and generate the root cause messages that you saw earlier.

Viewing the Specific Fault Model

SymCure creates a specific fault model, which consists of specific events and actions that occur on specific domain objects. SymCure creates diagnosis managers to manage specific fault models by:

- Diagnosing root causes from known symptoms by tracing upstream along the causal pathways from the symptoms to the faults.
- Predicting the impact of root causes by propagating downstream from causes to effects.

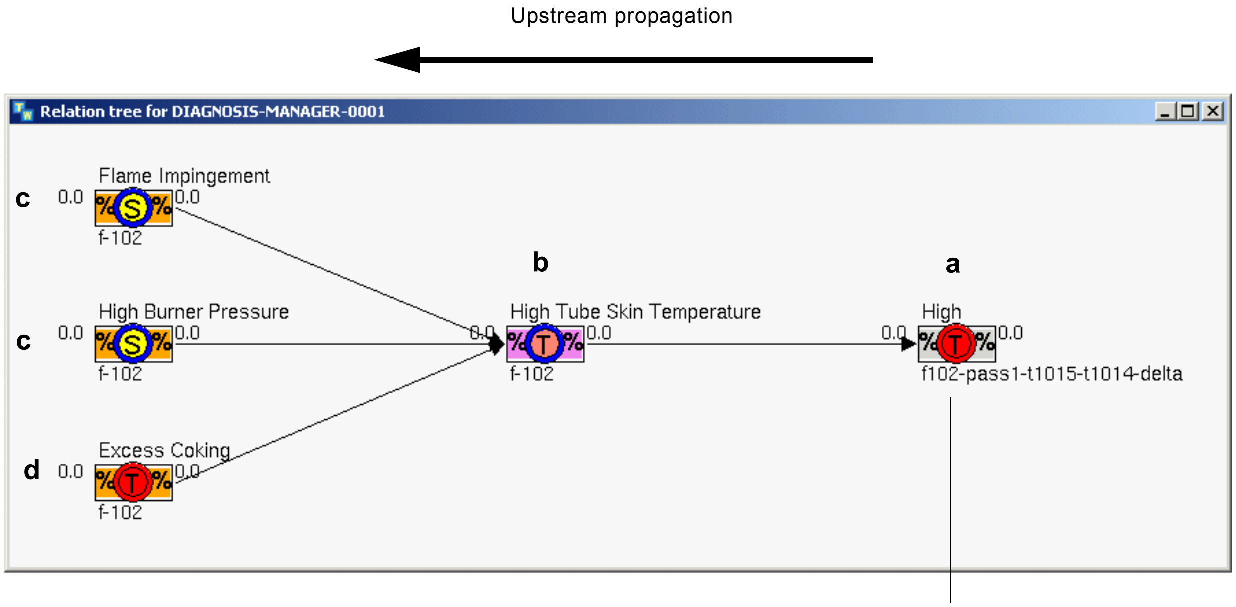
Specific events have a value and status, as described earlier. This table summarizes the event name, event type, target, value, and status of each specific event when you run the simulation in the Optegrity Heater Tutorial:

Event Name	Event Type	Target	Value	Status
High Burner Pressure	root cause	f-102	suspect	upstream inferred
Flame Impingement	root cause	f-102	suspect	upstream inferred
Excess Coking	root cause	f-102	true	specified

To view the specific fault model:

➔ Choose Project > Fault Models > Specific Fault Models > Diagnosis-Manager-0001.

This figure shows how events are propagated upstream to determine suspected root causes:



When the specific High event on f-102 is observed to be true, upstream fault propagation is triggered to determine suspected root causes.

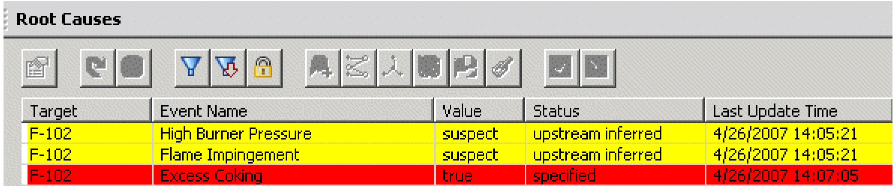
a The High event on the F-102 heater is specified as true, based on the F102-PASS1-T1015-T1014-DELTA::Tube Skin Delta T specific event detection diagram. Thus, the High event is an observed symptom of its suspected root causes.

b When the High event on the heater is observed to be true, the High Tube Skin Temperature event automatically becomes true, based on upstream propagation, because it is the only upstream event that causes the High event.

c When the High Tube Skin Temperature event becomes true, the Flame Impingement and High Burner Pressure root causes become suspect, based on upstream propagation, because they are upstream of the High Tube Skin Temperature event.

d Initially, the Excess Coking event is also suspect, then, it is specified as false, then it is specified as true, based on the Low Efficiency event becoming true.

These events correspond with the events you saw in the Root Causes browser earlier:



Root Causes

Target	Event Name	Value	Status	Last Update Time
F-102	High Burner Pressure	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Flame Impingement	suspect	upstream inferred	4/26/2007 14:05:21
F-102	Excess Coking	true	specified	4/26/2007 14:07:05

SymCure provides a debugger that you can use to step through the diagnosis. For more information, see the *SymCure User's Guide*.

Viewing External Actions

Generic fault models can include test actions, which are enabled when the underlying events turn suspect or unknown. They can also include repair actions, which are enabled when the underlying events turn true.

When a test or repair action is activated, it can execute a G2 procedure, which you specify as a property of the action. A test or repair action can also execute a specific event detection diagram when it activates. When no activation procedure is specified, the action executes the specific event detection diagram with the same name as the action.

As you saw earlier, the Optegrity Heater Tutorial defines two actions:

- The Flame Impingement test action is enabled when the Flame Impingement root cause event becomes suspect or unknown. You execute the test manually to set the value of the underlying event.
- The Adjust Burners repair action is enabled when the High Burner Pressure root cause event becomes true. The test executes automatically and invokes a G2 procedure, which waits for a period of time, then sets the High Burner Pressure root cause event to false.

Note Creating external action procedures requires knowledge of G2. For more information, see the *G2 Reference Manual*.

These procedures call API procedures that perform SymCure tasks programmatically. For more information about the SymCure API, see the *SymCure User's Guide*.

To view the external actions:

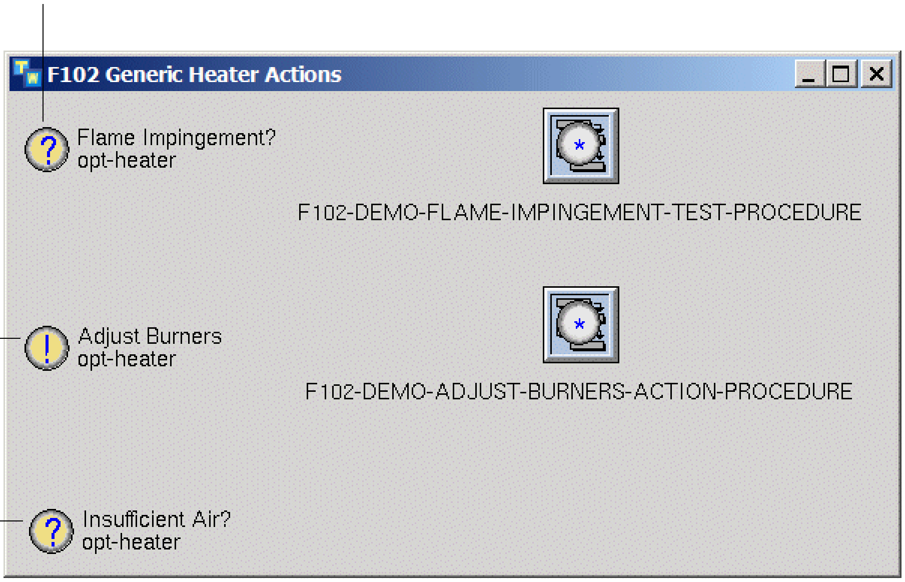
1 Choose Project > Fault Models > Generic Fault Models > Optegrity Heater > F102 Generic Heater Actions.

There are three external actions for the F-102 heater:

When the Insufficient Air test action activates, it executes the Insufficient Air event detection diagram.

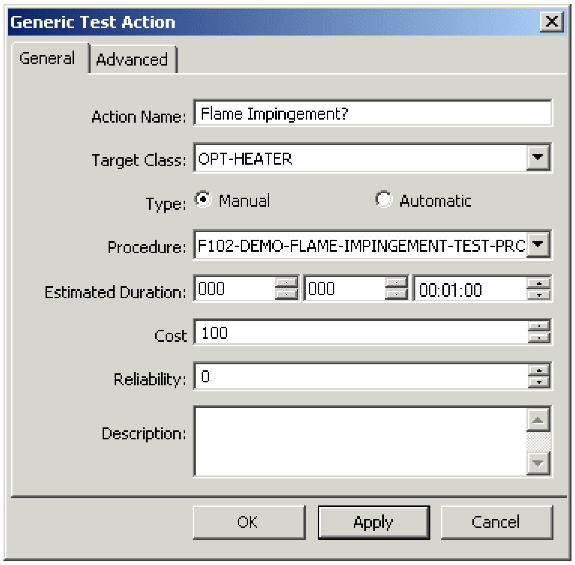
When the Adjust Burners repair action activates, it executes the f102-demo-adjust-burners-action-procedure.

When the Flame Impingement test action activates, it executes the f102-demo-flame-impingement-test-procedure.


F102 Generic Heater Actions
Flame Impingement?
opt-heater
F102-DEMO-FLAME-IMPINGEMENT-TEST-PROCEDURE
Adjust Burners
opt-heater
F102-DEMO-ADJUST-BURNERS-ACTION-PROCEDURE
Insufficient Air?
opt-heater

2 Choose Properties on the Flame Impingement repair action to view the Activation Procedure, which specifies the procedure to run when the repair action activates.

Here is the properties dialog for the repair action:



Generic Test Action

Action Name: Flame Impingement?

Target Class: OPT-HEATER

Type: ● Manual ○ Automatic

Procedure: F102-DEMO-FLAME-IMPINGEMENT-TEST-PRC

Estimated Duration: 000 000 00:01:00

Cost 100

Reliability: 0

Description:

Action Name is Flame Impingement, which is associated with the High Burner Pressure root cause event.

Target Class is opt-heater, which associates the repair action with the opt-heater class.

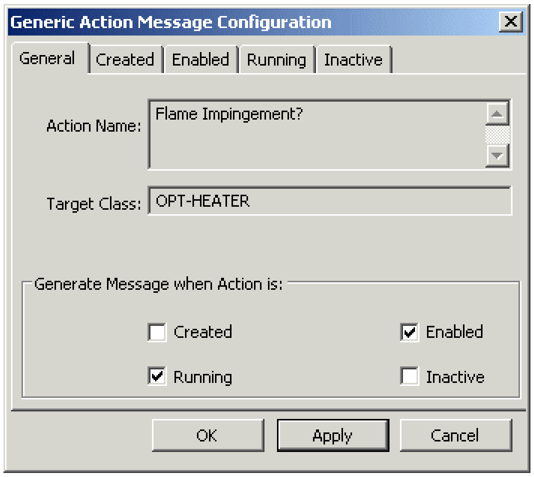
Activation Type is Manual, which means the repair action executes manually.

Activation Procedure is f102-demo-flame-impingement-test-procedure, which is the procedure to execute upon activation.

Estimated Duration is the estimate duration for executing the activation procedure.

Cost and Reliability allow you to assign user-defined measures for the cost and reliability of performing the action.

3 Click the Advanced tab, then click the Configure Messages button to display this dialog:



Action Name: Flame Impingement?

Target Class: OPT-HEATER

Generate Message when Action is:

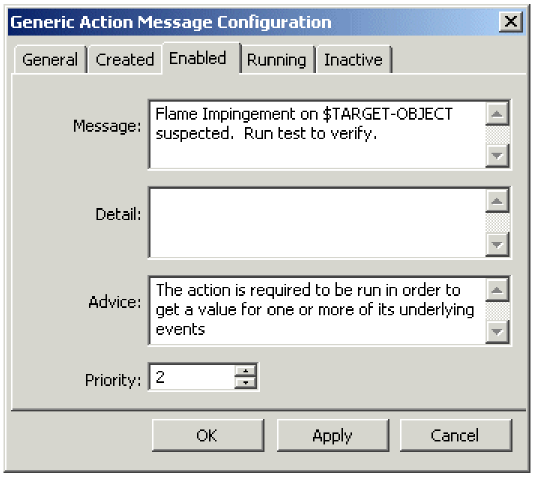
☐ Created ☑ Enabled

☑ Running ☐ Inactive

Tip You can also choose Configure Messages on the generic event to display this dialog.

The General tab indicates that an operator message should be generated when the action is enabled or running. These messages appear in the Messages Browser.

4 Click the Enabled tab to display this dialog:



Message: Flame Impingement on $TARGET-OBJECT suspected. Run test to verify.

Detail:

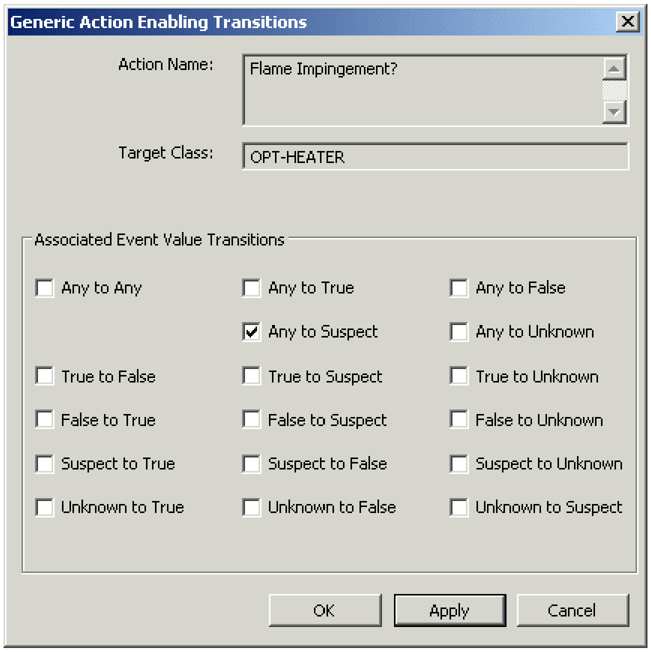
Advice: The action is required to be run in order to get a value for one or more of its underlying events

Priority: 2

Message uses text substitution to refer to the $TARGET-OBJECT. For example, the message text that you saw in the Message Browser looks like this for the specific action: Flame Impingement on F102 suspected. Run test to verify.

5 On the Advanced tab, click the Enabling Transitions button.

Enabling Transitions is Any to Suspect, which means the test action is enabled when the value of the underlying High Burner Pressure event becomes suspect:



Action Name: Flame Impingement?

Target Class: OPT-HEATER

Associated Event Value Transitions

☐ Any to Any ☐ Any to True ☐ Any to False

☑ Any to Suspect ☐ Any to Unknown

☐ True to False ☐ True to Suspect ☐ True to Unknown

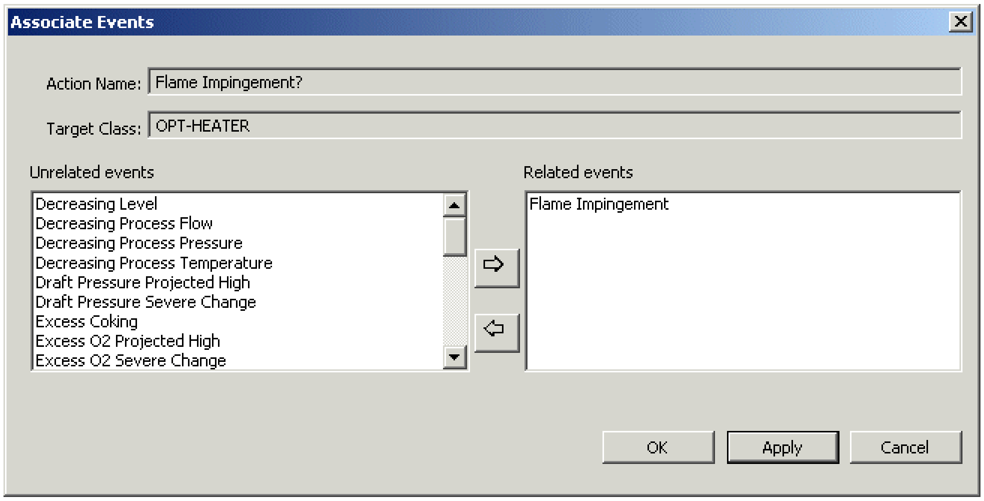
☐ False to True ☐ False to Suspect ☐ False to Unknown

☐ Suspect to True ☐ Suspect to False ☐ Suspect to Unknown

☐ Unknown to True ☐ Unknown to False ☐ Unknown to Suspect

6 On the Advanced tab, click the Associate Events button.

The related event is Flame Impingement:



Action Name: Flame Impingement?

Target Class: OPT-HEATER

Unrelated events

Decreasing Level
Decreasing Process Flow
Decreasing Process Pressure
Decreasing Process Temperature
Draft Pressure Projected High
Draft Pressure Severe Change
Excess Coking
Excess O2 Projected High
Excess O2 Severe Change

Related events

Flame Impingement

7. On the F102 Generic Heater Actions fault model, choose edit on the procedure to display its text.

   Here is the procedure associated with the Flame Impingement external action, which calls the cdg-default-run-test-manually-procedure API procedure, which is part of SymCure:

```
F102-Demo-Flame-Impingement-Test-Procedure (Target: class grtl-domain-
object {Target object}, SpecificAction: class cdg-specific-action {Specific action},
TriggeringEvent: item-or-value {The triggering event for the action or the symbol
none}, AssociatedEvents: sequence {All events associated with the action},
TimeStamp: integer {Time stamp}, Win: class object {The client window})

Begin
    Call cdg-default-run-test-manually-procedure (Target, SpecificAction,
        TriggeringEvent, AssociatedEvents, TimeStamp, Win)
End
```

8. On the F102 Generic Heater Actions fault model, view the properties, associated events, enabling transitions, message configuration, and associated procedure and event detection diagram of the Adjust Burners repair action.

   The Adjust Burners repair action is associated with the High Burner Pressure event, is automatic, and is activated when the value of the underlying event turns from any value to true. The associated procedure waits, then calls cdg-send-action-result to send a value of false for the underlying event.

9. On the F102 Generic Heater Actions fault model, view the properties, associated events, enabling transitions, message configuration, and associated procedure and event detection diagram of the Insufficient Air repair action.

   The Insufficient Air test action is associated with the Insufficient Air event, is manual, and is activated when the value of the underlying event turns from any value to suspect. The activation procedure is unspecified, which means the test action looks for an event detection diagram with the same name as the test action and executes it upon activation. The Insufficient Air event detection diagram sends the Insufficient Air event when low draft oxygen and high draft pressure both occur.

10. Simulate the Insufficient Air event by choosing Send Fault Model Event on the F-102 heater, then view the results in the Message Browser.

Summary

Congratulations! You have finished the *Optegrity Furnace Tutorial*. You should now know all about using Optegrity for abnormal condition management, including:

- Data configuration, data validation, data logging, and data replay.
- Message and alarm management.
- Event detection.
- Diagnostic reasoning.

For additional information on using Optegrity to build applications, consult these guides:

For information on...	See...
Using Optegrity to create domain object class definitions, to configure data sources, logging, and data replay, and to interact with and configure message browsers	*Optegrity User's Guide*
Using GEDP to create generic event detection templates	*G2 Event and Data Processing User's Guide*
Using SymCure to create generic fault models and perform diagnostic reasoning on events	*SymCure User's Guide*

Index

@ A B C D E F G H I J K L M
# N O P Q R S T U V W X Y Z