MICRO FOCUS

Orbix 3.3.13

Actional Integration with Orbix

Micro Focus
The Lawn
22-30 Old Bath Road
Newbury, Berkshire RG14 1QN
UK

http://www.microfocus.com

9/7/15

Contents

Preface

What is covered in this book

Orbix supports integration with the Aurea Actional® Application Performance Monitoring system. This guide explains how to enable Orbix applications and services to be monitored by Actional SOA management tools. This guide applies to Orbix applications and services written in both Java and C++.

Who should read this book

This guide is aimed at Orbix system administrators using Actional to monitor SOA environments, Orbix system architects, and Orbix application developers. System administrators do not require detailed knowledge of the technology used to create distributed enterprise applications.

Organization of this book

This book contains the following chapters:

- "Orbix–Actional Integration" describes the architecture of the Orbix integration with Actional.
- "Configuring Orbix for Actional Integration" explains how to configure integration between Orbix applications and services, and Actional.
- "Configuring Actional for Orbix Integration" provides some basic Actional configuration guidelines.
- "Monitoring Orbix Applications" shows examples of managing Orbix applications and services in Actional SOA management tools.

Document Conventions

This document uses the following typographical and keying conventions:

`Constant width`	Constant width words or characters represent source code or system values you must use literally, such as commands, options, and path names.
Italic	Italic words in normal text represent emphasis and new terms. Italic words or characters in code and commands represent variable values you must supply, such as arguments or commands or path names for your particular system.

This guide uses the following keying conventions:

... . . .	Horizontal or vertical ellipses in format and syntax descriptions indicate that material has been eliminated to simplify a discussion.
[ ]	Brackets enclose optional items in format and syntax descriptions.
{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.
|	A vertical bar separates items in a list of choices enclosed in { } (braces) in format and syntax descriptions.

Contacting Micro Focus

Our Web site gives up-to-date details of contact numbers and addresses.

Further Information and Product Support

Additional technical information or advice is available from several sources.

The product support pages contain a considerable amount of additional information, such as:

- The WebSync service, where you can download fixes and documentation updates.
- The Knowledge Base, a large collection of product tips and workarounds.
- Examples and Utilities, including demos and additional product documentation.

To connect, enter http://www.microfocus.com in your browser to go to the Micro Focus home page.

Note:
Some information may be available only to customers who have maintenance agreements.

If you obtained this product directly from Micro Focus, contact us as described on the Micro Focus Web site, http://www.microfocus.com. If you obtained the product from another source, such as an authorized distributor, contact them for help first. If they are unable to help, contact us.

Information We Need

However you contact us, please try to include the information below, if you have it. The more information you can give, the better Micro Focus SupportLine can help you. But if you don't know all the answers, or you think some are irrelevant to your problem, please give whatever information you have.

- The name and version number of all products that you think might be causing a problem.
- Your computer make and model.
- Your operating system version number and details of any networking software you are using.
- The amount of memory in your computer.
- The relevant page reference or section in the documentation.
- Your serial number. To find out these numbers, look in the subject line and body of your Electronic Product Delivery Notice email that you received from Micro Focus.

Contact information

Our Web site gives up-to-date details of contact numbers and addresses.

Additional technical information or advice is available from several sources.

The product support pages contain considerable additional information, including the WebSync service, where you can download fixes and documentation updates. To connect, enter http://www.microfocus.com in your browser to go to the Micro Focus home page.

If you are a Micro Focus SupportLine customer, please see your SupportLine Handbook for contact information. You can download it from our Web site or order it in printed form from your sales representative. Support from Micro Focus may be available only to customers who have maintenance agreements.

You may want to check these URLs in particular:

- http://www.microfocus.com/products/corba/orbix/orbix-3.aspx (trial software download and Micro Focus Community files)
- https://supportline.microfocus.com/productdoc.aspx. (documentation updates and PDFs)

To subscribe to Micro Focus electronic newsletters, use the online form at:

http://www.microfocus.com/Resources/Newsletters/infocus/newsletter-subscription.asp

Orbix–Actional Integration

Orbix provides support for integration with Actional SOA management products. This chapter explains the main components and concepts used in this integration.

Introduction

Aurea Actional® Application Performance Monitoring is an SOA management product that provides operational and business visibility, policy-based security, and control of services and business processes in a heterogeneous runtime environment. This section explains the main concepts and components used in the Orbix–Actional integration.

Note: Integration with Actional is not supported by Orbix for Microsoft Windows VC11 32-bit or VC11 64-bit editions. For more information on Orbix editions, see the Orbix ***Installation Guide***.

Orbix and Actional

Integration between Orbix and Actional enables Orbix applications to be monitored by Actional SOA management tools. For example, you can use Actional to perform discovery, monitoring, auditing, and reporting on Orbix applications. You can also correlate and track all messages through your SOA network to perform dependency mapping and root cause analysis.

The Orbix–Actional integration is deployed on Orbix systems to enable reporting of management data back to the Actional server. The data reported back to Actional includes system administration metrics such as response time, fault location, auditing, and alerts based on policies and rules. The Orbix–Actional integration can be used with Orbix applications written in both Java and C++.

Actional SOA management

The main components in the Actional SOA management system are the Actional server, Actional agents, and Actional intermediaries.

The Actional server is the central engine that correlates data received from Actional agents and distributes policies. The Actional agent collects data about service traffic from an application server and applies policies. The Actional intermediary acts as a proxy that brokers interaction between Web service applications and systems built on them.

All Actional components are Java applications. The Actional server uses the Jetty application server by default, while its web console uses JSP and Adobe Flash.

Figure 1 shows a high-level overview of the main Actional components.

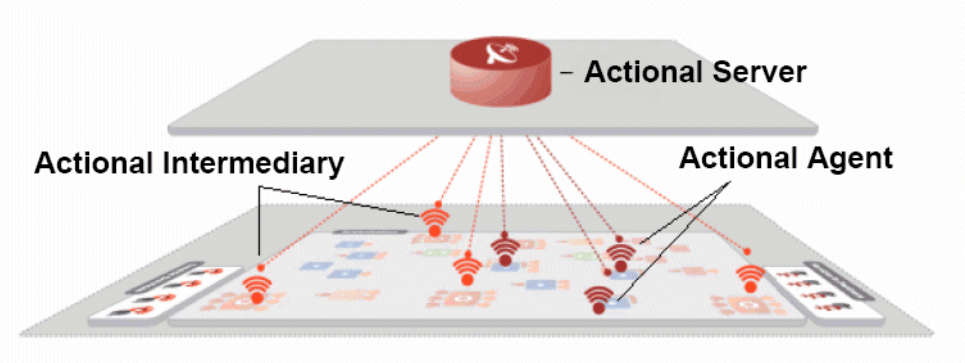

Figure 1: *High-Level Actional Overview*

Managed nodes

A node is defined as a system on the current network. A node with an Actional agent installed is referred to as an *instrumented node* or a *managed node*.

The managed node uses Actional's interceptor API to send monitoring data to the Actional agent. On any managed node, one Actional agent and one or more interceptors must be running.

Actional server

The Actional server is a central management server that manages nodes containing an Actional agent. The Actional server correlates the data it receives from each of its agents, and distributes policies to those agents. It enables an administrator to analyze service network data and create system-wide policies.

The Actional server hosts a database and pings Actional agents to obtain management data at configured time intervals. It analyzes the management data and displays it in a console—for example, the **Actional Management Server Administration Console**. This is a Web application deployed on Apache Tomcat, which provides runtime management and agent configuration. In addition, any alerts triggered at the Actional agent are sent immediately to the Actional server.

The default Actional server database is Apache Derby. Other supported databases include:

- PostgreSQL
- OpenEdge
- MSDE
- SQL Server
- Oracle
- DB2

By default, the Actional server uses port `4040` (for example, `http://`*`HostName`*`:4040/lgserver/`).

Actional agent

An Actional agent runs on each Orbix host that you wish to manage, and is used to provide instrumentation data back to the Actional server. The Actional agent includes two main components: an analyzer, and one or more interceptors. The analyser gathers and evaluates data such as records, statistics, and alerts. The interceptors collect data about service traffic from an application server, and apply policies to that traffic.

Actional agents are provisioned from the Actional server to establish initial contact and send configuration to the Actional agent. There is one Actional agent per managed node. By default, the Actional agent uses port `4041` (for example, `http://`*`HostName`*`:4041/lgagent/`).

Actional intermediary

An Actional intermediary is an in-network service broker that includes an integrated Actional agent. It serves as a proxy for Web service applications, providing features such as security, bridging, and activity tracking. The Actional intermediary supports application servers such as WebLogic, WebSphere, JBoss, and Oracle.

Actional agent interceptor SDK

The Actional Agent Interceptor Software Development Kit (SDK) is an Actional-specific API used to create custom interceptors. These can be used to send management instrumentation data from an application to the Actional agent.

Actional SOA management tools

In this guide, Actional is the general term used to describe the Actional SOA management system in which all data is stored and viewed. This simplifies the architecture of Actional for the sake of this discussion.

Figure 2 shows an example of the **Actional Management Server Administration Console**. Managed nodes are displayed as blue boxes, and unmanaged nodes are displayed as grey boxes. The green arrows indicate the message flow through various nodes. Clicking on each of the nodes shows more in-depth information regarding the response time, alerts and warnings, and so on.

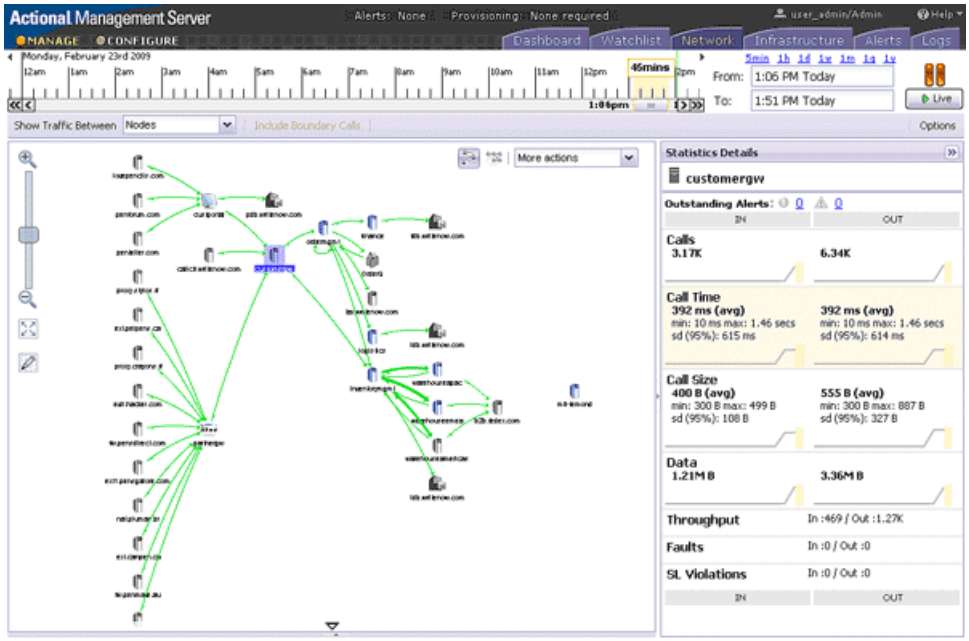

Figure 2: *Actional Management Server Administration Console*

NGSO mapping

When you click and drill down in the Actional **Path Explorer** view, the organization of the information displayed is *Node–Group–Service–Operation* (NGSO). In Orbix, this translates to *Host–Module–Interface–Operation*. Table 1 shows the mapping from Actional to Orbix.

Table 1: *NGSO Mapping*

Actional	Orbix
Node	Host
Group	Module
Service	Interface
Operation	Operation

For details on setting the configuration variables, see "Configuring Actional Monitoring".

Further information

For detailed information on all Actional features, see the Actional product documentation.

Orbix–Actional Integration Architecture

This section shows a basic Actional architecture, simplified for the purposes of this discussion. It explains how Actional interceptors provide data to the Actional agent, and how the Actional server manifest is used to correlate the origin and business flow of a request.

It then shows the Orbix–Actional integration architecture, and explains how Orbix plug-ins and Orbix interceptors are used to configure integration with Actional.

Basic Actional architecture

Figure 3 shows a high-level overview of a basic Actional architecture from the perspective of a consumer and service provider.

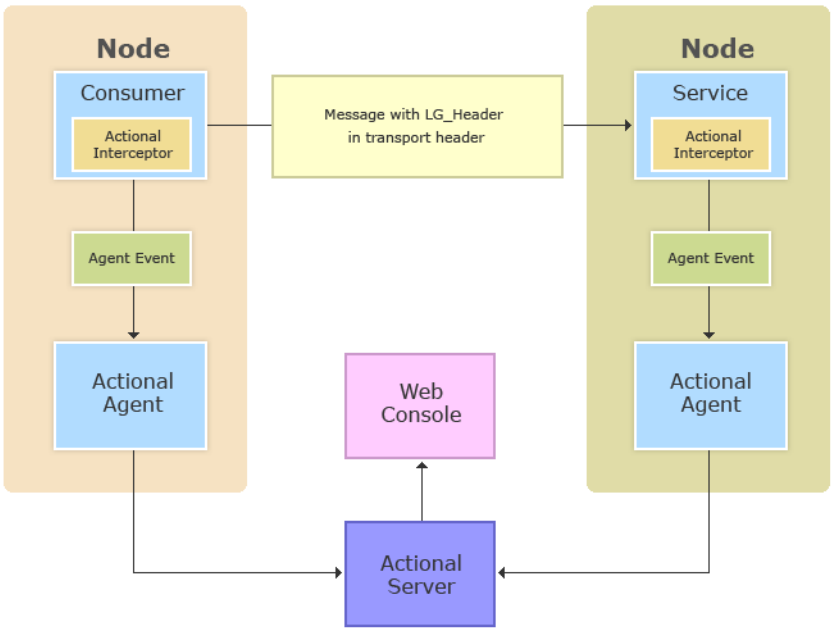

Figure 3: *Basic Actional Architecture*

In the interaction shown in Figure 3, the Actional interceptors sit in the flow between the application logic and the consumers and providers of other services. They intercept all inbound and outbound calls, and feed information about those calls to the Actional agent as asynchronous events.

The Actional agent is responsible for processing the event stream from the interceptors, computing and storing aggregate statistics, executing policies, and communicating with the Actional server.

The Actional server manifest (`LG_Header`) is a token that is sent in the transport header of the message to each participant in a call. This token identifies the origin and business flow of a request. For more details, see "Actional server manifest" on page 7.

Actional interceptors

Actional interceptors sit in the flow at the edge of an application, intercepting all incoming and outgoing messages. An Actional interceptor is designed as a lightweight component that imposes minimal overhead on the application (typically less than 100 microseconds per call).

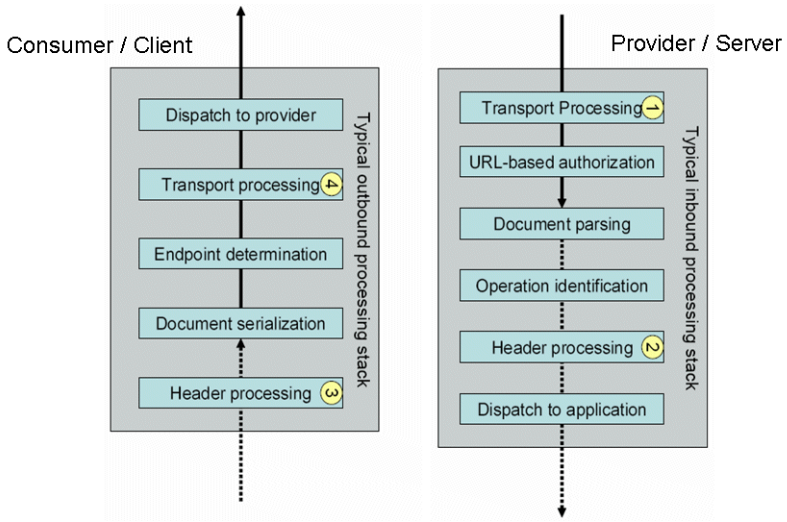

Figure 4: *Actional Interceptors*

The interceptor must perform the following tasks to gain the full functionality of the Actional server:

1. Extract an Actional server manifest (if any) from the incoming request document.
2. Insert an Actional server manifest into any outgoing request documents.
3. Transfer the interceptor context along the internal business flow, from the incoming interceptor, to any related outgoing interceptors.
4. Send the Actional agent an event for each incoming or outgoing document.

Actional server manifest

The Actional server sends an Actional server manifest (`LG_Header`) with a request document to provide information about the request's origin and the business flow that the request belongs to.

The Actional server manifest is used by the Actional server to correlate information it receives, from multiple agents, about interactions between different services. For this reason, the server manifest is sometimes referred to as a correlation ID.

The consumer and provider of the service must have an agreed mechanism (transport or protocol) for transferring the manifest. The following is an example `LG_Header`:

```
Interaction=CgJkcB+YlN0ZyBABdysAAA==;
Locus=ApM1eYBGBAR4LFJ1VvHOdg==;
Flow=CgJkcB+YlN0ZyBABdSsAAA==;
UpstreamOpID=FtfEJXM1nqJ0C995IBMkEQ==;
Path=7Qg2aVWCdwmP8gGebyLWYA==;
name=E_10-2-100-112-e0c7c3-110c80b4df0--7fdd-INITIATED;
CPTime=1171591682345;
FlowFields=MF1:1254;MF2:1589;
```

The main components in the server manifest are the `Interaction`, `Locus`, `Flow`, and `UpstreamOpID`. The other components are optional.

Orbix–Actional integration architecture

The Orbix–Actional integration is built using an extensible Orbix plug-in architecture. This means that Orbix–Actional integration can be enabled by adding a monitoring plug-in to your Orbix configuration. No code changes are necessary for Orbix client and server applications.

Figure 5 shows an overview of the Orbix–Actional integration architecture from an Orbix client-server perspective. This builds on the architecture shown in Figure 3, with the addition of Orbix monitoring plug-in. In Figure 5, the CORBA GIOP message also includes the `LG_Header` in a GIOP service context. A GIOP service context is a general mechanism for including out-of-band data in a GIOP request or reply message. Service contexts in GIOP are analogous to headers in other protocols such as HTTP.

Orbix interceptors

In the Orbix-Actional integration, an Orbix plug-in for Actional must also be added to your Orbix client and server processes. These plugins are loaded into the process via a config variable.

The Orbix monitoring plug-in is implemented using a proprietary feature of Orbix 3.3.x called *Filters and ServiceContext Handlers*. This allows us to intercept the Request messages at various points in the lifecycle of sending/receiving of requests in the ORB. This enables high-level request processing to be performed.

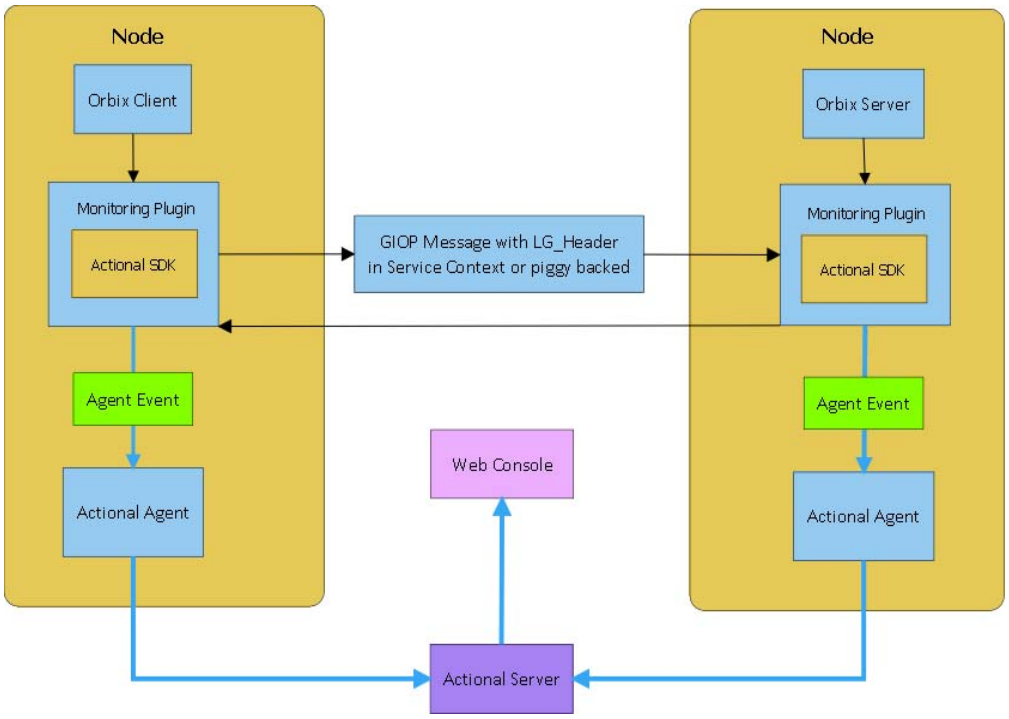

Figure 5: *Orbix-Actional Integration Architecture*

Configuring Orbix for Actional Integration

This chapter explains the steps required to configure Orbix for integration with Actional SOA management products.

This chapter includes the following sections:

Configuring Orbix Java Applications

This section explains how to configure Orbix Java applications for integration with Actional. It shows some examples from the Orbix Actional integration demo:

```
OrbixInstallDir/demos/common/monitoring
```

Update your Actional SDK

You must first update your Actional SDK JAR file as follows:

1. In the **Actional Agent Administration Console**, select **Getting Started|Interceptor SDK** (see Figure 6), and download the Windows (`.zip`) or UNIX (`.tar`) file. This includes the `actional-sdk.jar`, documentation, and samples.
2. Replace the existing `actional-sdk.jar` in the following location with the version that you downloaded:

```
OrbixInstallDir/lib
```

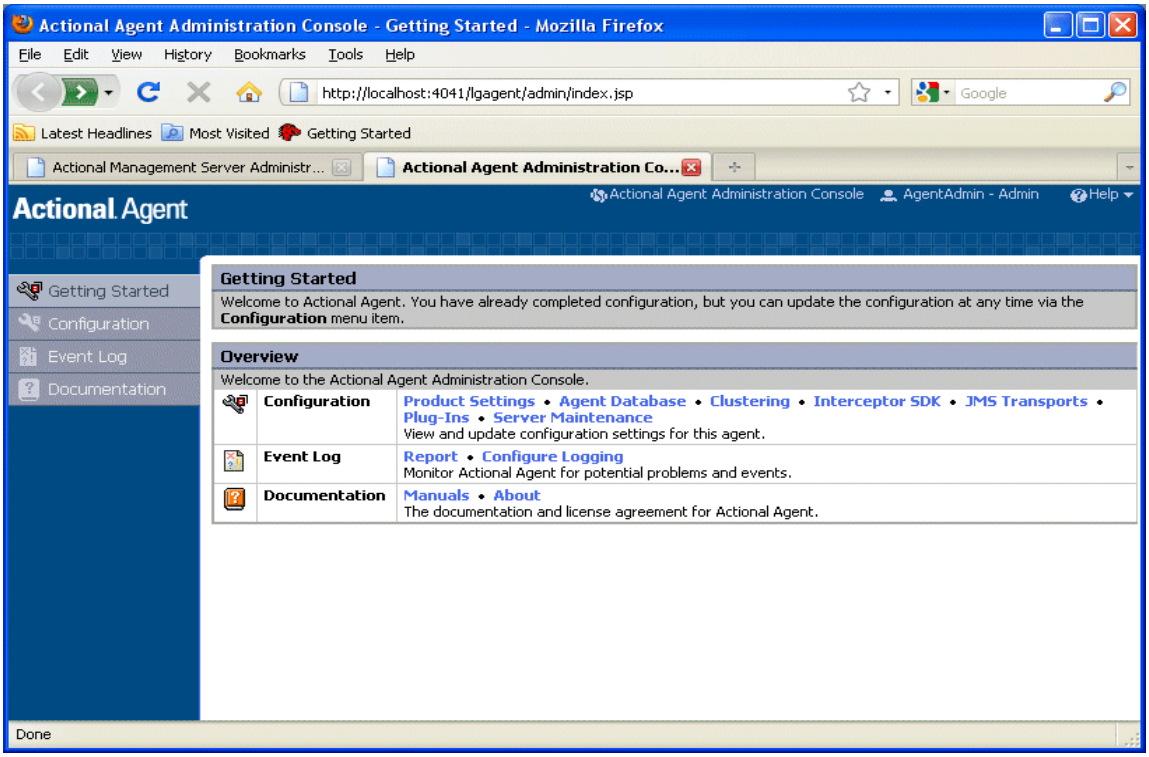

Figure 6: *Actional Agent Administration Console*

Configuring Actional Monitoring

There are two configuration variables that control the monitoring plugin:

- Specify the monitoring plug-in
- Specify the Uplink.cfg folder.

Specifying the monitoring plug-in name

To tell the Orbix Java runtime to load the java monitoring plugin, add the following configuration variable to the OrbixWeb scope in your `orbixweb3.cfg` configuration file:

```
IT_ORB_INITIALISORS =
   "IE.Iona.OrbixWeb.Monitoring.MonitoringPI";
```

Specify the Uplink.cfg folder

If you install the Actional Agent's `uplink.cfg` into a non-default location this configuration variable may be useful as it will set the `com.actional.lg.interceptor.config` system property. Otherwise you need to add the property as `-Dcom.actional.lg.interceptor.config=<path-to-uplink-cfg>` to the Java command.

```
IT_ACTIONAL_LG_INTERCEPTOR = <path-to-uplink-cfg>
```

Any Java applications will need to modify their classpaths to ensure that both the `monitoring.jar` and `actional-sdk.jar` are present on the classpath.

Configuring Orbix C++ applications

This section explains how to configure Orbix C++ application for integration with Actional. It shows some examples from the Orbix Actional integration demo:

```
OrbixInstallDir/demos/monitoring
```

Setting your environment

No changes are necessary if the Actional `Uplink.cfg` configuration file is located in its default path:

UNIX `/var/opt/actional/LG.Interceptor`

Windows `%systemroot%\system32\LG.Interceptor`

On a 64-bit windows system, if the agent runs with a 32-bit Java VM, then the default location is:

`%systemroot%\SysWow64\LG.Interceptor`

Otherwise it is:

`%systemroot\system32\LG.Interceptor.`

The `Uplink.cfg` file is responsible for communication between the Actional interceptors and the analyzer in the Actional agent.

If the `Uplink.cfg` is not located in its default path, you must specify the path to this file as follows:

UNIX `export LG_INTERCEPTORCONFIG=`*`PathToFile`*

Windows `set LG_INTERCEPTORCONFIG=`*`PathToFile`*

Configuring the Orbix monitoring plug-in

You can configure the monitoring plug-in by editing the settings in your application configuration scope in your Orbix configuration file. This includes the following steps:

- Specify the monitoring plug-in
- Specify the monitoring log filter

Specifying the monitoring plug-in

You can tell Orbix to load the monitoring plugin by adding the following configuration variable to the Orbix scope: `orbix3.cfg`.

```
Orbix.IT_ORB_INITIALISORS = "it_monitoring";
```

You can also set the configuration variable in the environment, for example:

- On Windows: set `IT_ORB_INITIALISORS=it_monitoring` and `client.exe`
- On Unix: `IT_ORB_INITIALISORS=it_monitoring ./client`

Specifying the monitoring log filter

You can specify the following configuration variable and it will output various diagnostic information about the monitoring plugin, if you need more verbose information.:

```
Orbix.IT_LOGGING_FILTERS="monitoring";
```

This will create a logfile called `monitoring.log` in the current folder with all the logging from the plugin.

Troubleshooting Orbix

This section provides some tips to help troubleshoot your Orbix integration with Actional.

Ensure Actional Plugin is loaded

To verify that the Orbix monitoring plug-in is loaded and participating correctly.

C++

You can specify the IT_LOGGING_FILTERS config variable to ensure that the logging from the monitoring plug-in being outputted to the log file:

```
[17:42:16.080, pid: 20765 tid: -1424223984]: Client
   Interaction: url: Plugin/Tester
    Operation: find_name
    Group: Plugin
    Service: Tester
    HostName: jewel]
```

Also you can specify the environment variable IT_SHLIB_VERBOSE=1 in your environment, this will show a more verbose output of what the plug-in loader in Orbix is doing, or if it is loading the monitoring plugin.

```
[IT_Shlib_Manager, thread 1] loading it_monitoring.
[IT_Shlib_Manager, thread 1] Attempting to load library:
   libit_monitoring.3.3.gcc411.so.1
[IT_Shlib_Manager, thread 1] Located shlib:
   /vob/orbix/targets_orbix//lib/libit_monitoring.3.3.gcc411.so
   .1
[IT_Shlib_Manager, thread 1] About to load shlib:
   /vob/orbix/targets_orbix//lib/libit_monitoring.3.3.gcc411.so
   .1
[IT_Shlib_Manager, thread 1]
   /vob/orbix/targets_orbix//lib/libit_monitoring.3.3.gcc411.so
   .1 seems to be compatible with the shared libraries already
   loaded in this process.
[IT_Shlib_Manager, thread 1] Loading plugin:
   /vob/orbix/targets_orbix//lib/libit_monitoring.3.3.gcc411.so
   .1
```

Java

You can turn on diagnostics as normal with the `setDiagnostics` configuration variable, a level of 128 or higher will output the monitoring plug-in's information.

```
[MonitoringGeneric:group: Monitoring Service: Server2
   RepositoryID: Monitoring/Server2]
[MonitoringGeneric:ClientInteraction.setSelfAddr(10.2.2.141)]
[MonitoringGeneric:ClientInteraction.setPeerAddr(jewel)]
[MonitoringGeneric:ClientInteraction.setUrl(Monitoring/Server2)
   ]
[MonitoringGeneric:ClientInteraction.setGroupName(Monitoring)]
[MonitoringGeneric:ClientInteraction.setService(Server2)]
[MonitoringGeneric:ClientInteraction.setOpName(get_command)]
[MonitoringGeneric:ClientInteraction.requestAnalyzed()]
```

Configuring Actional for Orbix Integration

This chapter gives some basic guidelines on setting up Actional to run the Orbix Actional integration demo.

This chapter includes the following sections:

Prerequisites

This section describes prerequisites for integration between Actional SOA management products and Orbix.

Actional products

The following Actional products should be installed:

- Actional Management Server 8.0 (Actional server)
- Actional Flex Point 8.0 (Actional agent/intermediary)

Alternatively, the following Actional products can be installed separately:

- Actional Point of Operational Visibility 8.0 (Actional agent)
- Actional Client Security Enforcement 8.0 (Actional intermediary)

Actional agents

You must ensure that Actional agents are set up on each Orbix host node that you wish to manage. The provisioning of Actional agents is performed using the Actional server. For some basic details, see "Configuring Actional for Orbix Integration" on page 15.

For full details on how to set up Actional agents on managed nodes, see the Actional product documentation.

Further information

For information on installing Actional products, and the full range of platform and database versions supported by Actional, see the Actional product documentation.

This Orbix integration with Actional supports the full range of operating systems and compilers supported by Orbix. For more details, see the ***Orbix Installation Guide***.

Configuring Actional

This section provides some basic configuration guidelines on Actional agent and server configuration.

Actional agent configuration

No specific Actional agent configuration settings are required for integration with Orbix. For example, for the purposes of the Orbix–Actional integration demos, the Actional agent can be started with the default configuration settings.

Actional server configuration

The following sample configuration steps describe how to set up the Actional server to run an simple Orbix–Actional demo:

1. Install the Actional server with typical installation options, and select the Apache Derby database.
2. Specify the following URL in your browser:

   ```
   http://localhost:4040/lgserver
   ```
3. If this is a new installation click **Start**, and follow the new Actional server setup steps.

   Otherwise, if the Actional server is already installed, perform the following steps:

   i. In the Actional console Web interface, select the **Configure** radio button in the top left of the screen.

   ii. Select the **Platform** tab. This displays the general configuration settings, as shown in Figure 7.

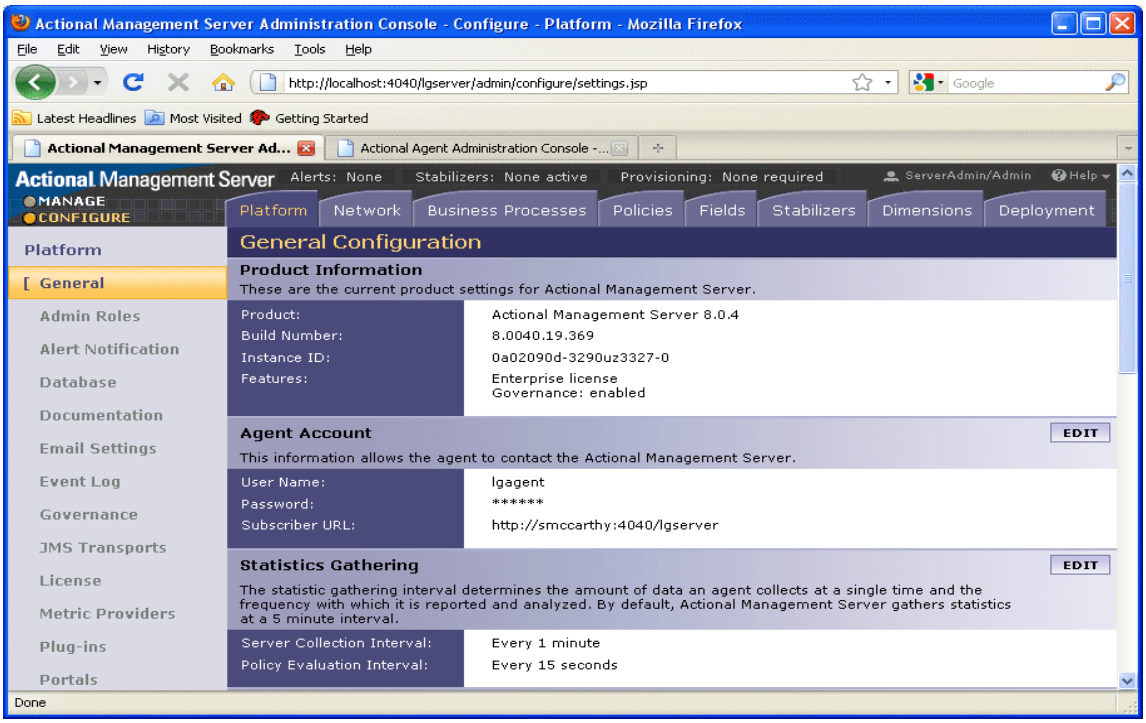

Figure 7: *Actional Server Configuration Settings*

Creating a managed node

To create a managed node for a simple Orbix demo, perform the following steps:

1. In the Actional **Configure** view menu bar, open the **Network** tab. This displays the **Network Nodes**.
2. Select **Add**. This displays **Node Creation / Managing Agents**.
3. Click **Managed Node**.

Configuring a new node

To configure a managed node for the demo, perform the following steps in the wizard:

Step 1: New Node - Identification

1. Specify the **Name** as `agent1`.
2. Specify the **Display icon** as `Auto Discover`.
3. Click **Next**.

Step 2: New Node - Management

1. Specify the **Transport** as `HTTP/S`.
2. Supply your Actional agent user name and password.
3. Ensure that **Override Agent Database** is checked.
4. Click **Next**.

Step 3: New Node - Agents

1. Specify the following URL:
   `http://`*`HostName`*`:4041/lgagent`
   You can specify a host name or an IP address in this URL.
2. Click **Add**. The agent URL is added.
3. Click **Next**.

Step 4: New Node - Endpoints

1. For **Endpoints**, add the hostname, fully qualified hostname, or IP address.
2. Click **Next**.

Step 5: New Node - Filters

1. Do not specify any filters for the demo.
2. Click **Next**.

Step 6: New Node - Trust Zone

1. Do not specify a trust zone for the demo.
2. Click **Finish**.

The newly created managed node now needs to be provisioned.

Provisioning a new node

To provision the new node to bring it under management, perform the following steps:

1. Select the **Configure** radio button at the top left of the screen.
2. Select the **Deployment** tab from the **Configure** menu bar.
3. The **Provisioning** page is displayed, and `agent1` is listed as not provisioned.
4. Select the `agent1` check box.
5. Click **Provision**. This displays a message when complete: `Successfully provisioned`.

6. Click the **Manage** radio button at the top left of the screen. You should see `agent1` added to the **Network** view as shown in Figure 8.

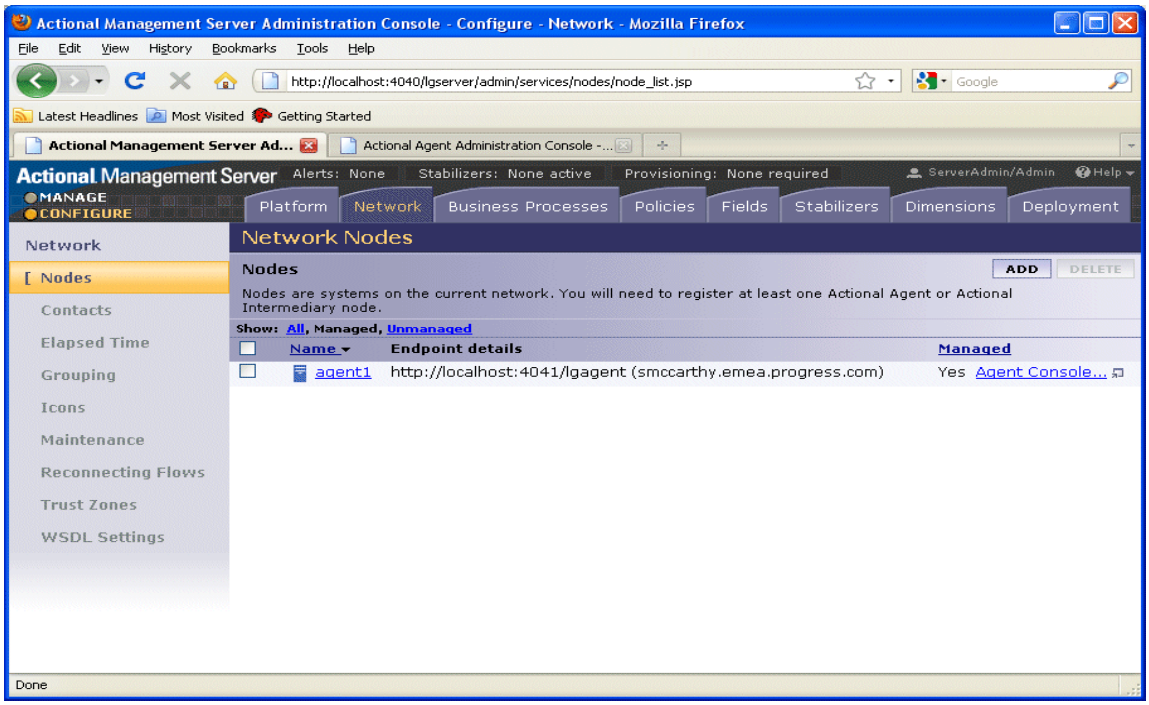

Figure 8: *Actional Server Provisioned Node*

Further information

For more details on setting up and running Actional SOA management tools, see the Actional product documentation.

Troubleshooting Actional

This section provides some tips to help troubleshoot your Actional integration with Orbix.

Setting default polling

For demonstration purposes, to update the display in your Actional server console more frequently, you can set the default polling to a shorter time span as follows:

1. Select the **Configure** radio button at the top left of the screen.
2. Select the **Platform** tab from the **Configure** menu bar.
3. In **Statistics Gathering** on the right, select **EDIT**.
4. Set the **Server Collection Interval** to 1 minute by using the drop down list.
5. Set the **Policy Evaluation Interval** to 15 seconds.

Ensuring events are reported to the Actional Agent

To ensure that Orbix monitoring events are being reported to your Actional agent, perform the following steps:

1. Ensure your Actional agent is running, and added as a managed node in your Actional server.
2. Verify that the agent generated the `Uplink.cfg` file in the directory specified during installation. If this file was not specified during the installation, it should be in the following default path (which should have write permission):

   UNIX `/var/opt/actional/LG.Interceptor`

   Windows `%systemroot%\system32\LG.Interceptor`

3. Open your Actional agent console and login:
   `http://`*`AgentHostName`*`:`*`Port`*`/lgagent/`
4. Specify the following URL to display the **Options** page shown in Figure 9:
   `http://`*`AgentHostName`*`:`*`Port`*`/lgagent/admin/options.js`
5. For **Audit agent events**, Click **On**.

6. Click **Apply**.

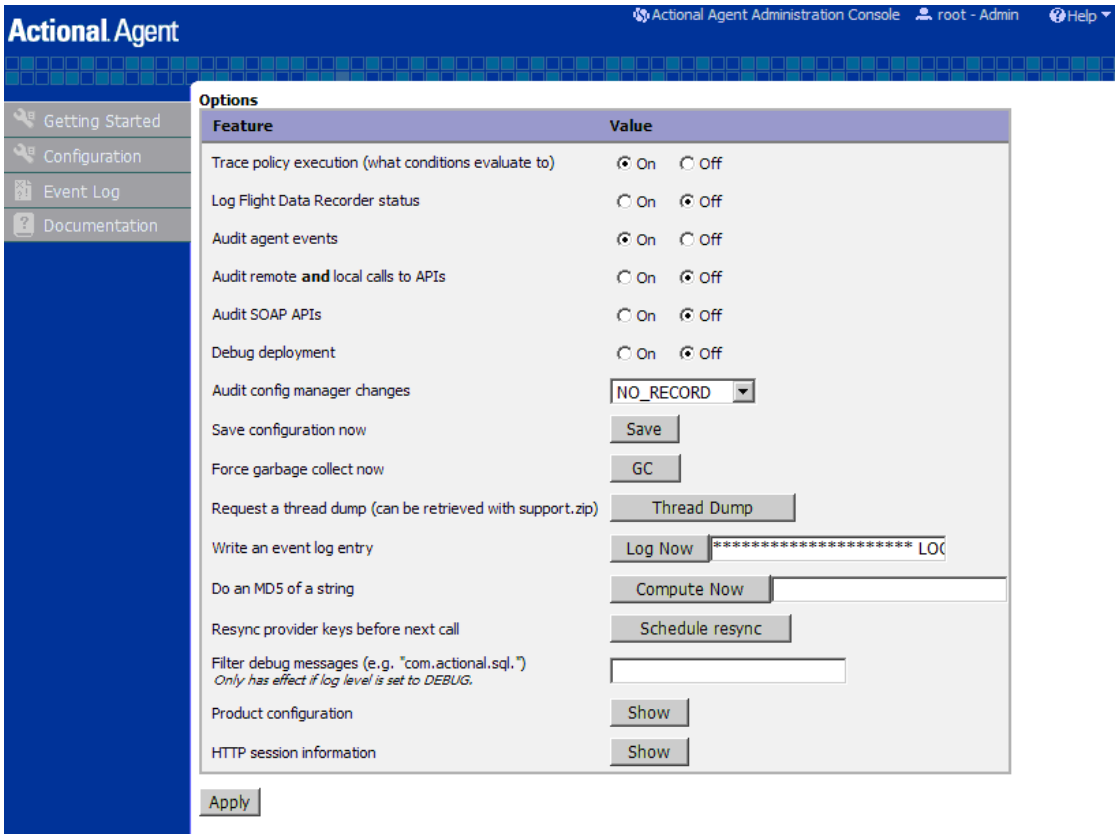

Figure 9: *Actional Agent Options*

Note: These settings are not persistent, and are reset when the Actional agent is restarted.

Viewing agent events

When **Audit agent events** is turned on, all external events coming from the Orbix monitoring plug-in can be reviewed in the Actional agent **Event Logs**, shown in Figure 10.

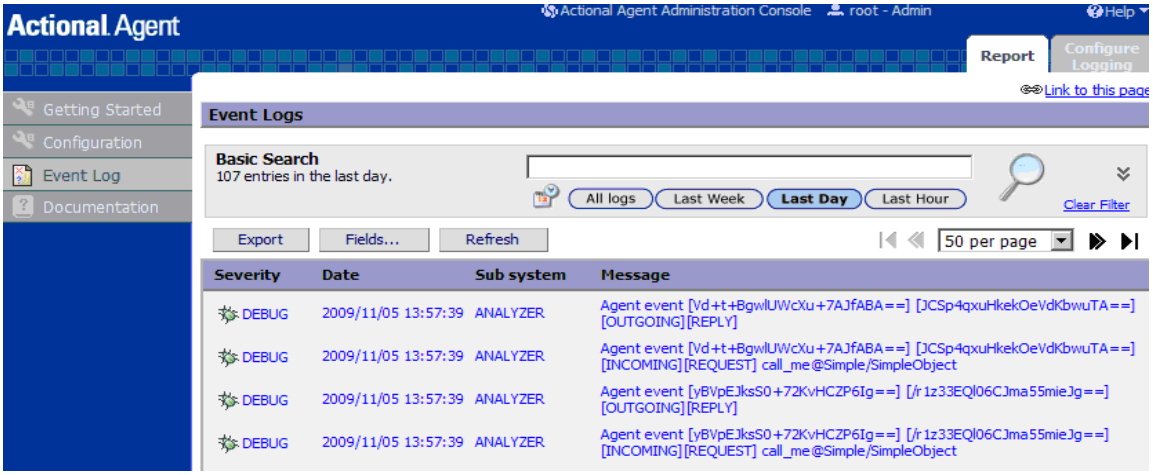

Figure 10: *Actional Agent Event Logs*

Figure 10 shows `INCOMING`, `OUTGOING`, `REQUEST`, and `REPLY` events reported from the monitoring plug-in. If these events are not reported, the path for the `uplink.cfg` may be incorrect, and the monitoring plug-in can not find the agent.

C++ applications

For C++ applications, verify that the `LG_INTERCEPTORCONFIG` environment variable is set correctly, and points to the directory where the agent has written the `uplink.cfg` file.

Java applications

For Java applications, verify that the `com.actional.lg.interceptor.config` property is passed on to the application correctly, and points to the directory where the agent has written the `uplink.cfg` file.

When incoming monitoring events are arriving at the agent, and the agent is configured correctly, you should see the calls displayed in the Actional server console **Network** view, as shown in the chapter "Configuring Orbix for Actional Integration".

Further information

For any problems with Actional agent configuration, please refer to the Actional product documentation.

Managing Orbix Applications in Actional

This chapter shows examples of managing a simple Orbix application and Orbix domain services in Actional SOA management tools.

This chapter includes the following sections:

Monitoring Orbix Applications

When your Orbix applications are configured for integration with Actional, they can be monitored using the Actional SOA management tools. No code changes are required for monitoring of Orbix applications.

For example, when you run the simple Orbix `actional_demo`, the **Actional Management Server Administration Console** displays the managed node that the demo is running on. Invocations are displayed as arrows flowing to and from managed components.

The Orbix `monitoring demo` illustrates the simple use of the ORB monitoring plug-in to report calls made between Orbix clients and servers to Actional. This demo is similar to `demos/corba/orb/simple`, and shows how to configure visibility of your application in Actional. For details on how to run this demo, see the `README` text files in the following directory:

```
OrbixInstallDir/demos/monitoring
```

Network view

The Actional network view displays the traffic between various components in your network environment. These include nodes, packages, services and operations.

Figure 11 shows the running Orbix `actional_demo` displayed in the **Network** tab of the **Actional Management Server Administration Console**. In this simple demo, the **Network** tab displays the Actional agent on the Orbix managed node that the demo is running on. This agent reports the monitoring data back to the Actional server. The single invocation is displayed as a green arrow flowing from the node and back to itself. In more complex examples with multiple nodes, the arrows flow between nodes.

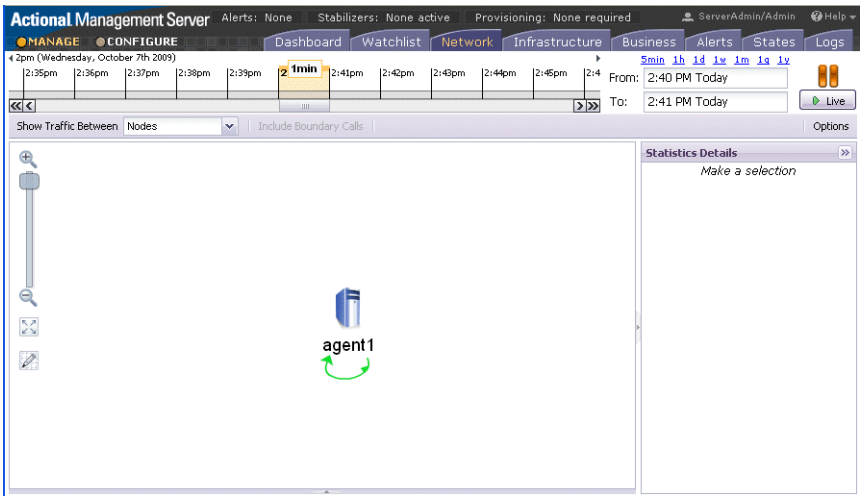

Figure 11: *Actional Server Network View*

By default, the **Network** view shows traffic between nodes. There is only one node in this case. You can also select to show traffic between packages in the top left of the screen. Figure 12 shows the traffic between the Orbix client and server packages.

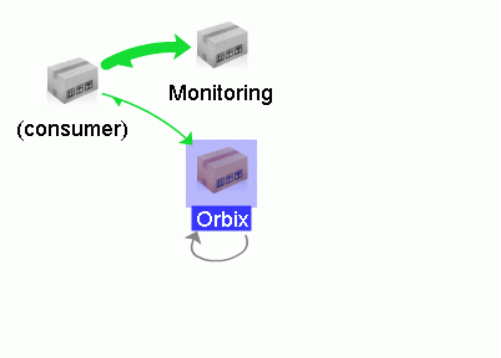

Figure 12: *Traffic Between Packages*

Path Explorer

Figure 13 shows the Orbix `actional_demo` displayed in the **Path Explorer** view of the **Actional Management Server Administration Console**.

To view this screen, double click on the managed node shown in Figure 11. Alternatively, click the **Display Path Explorer** button at the top right of the **Network** view.

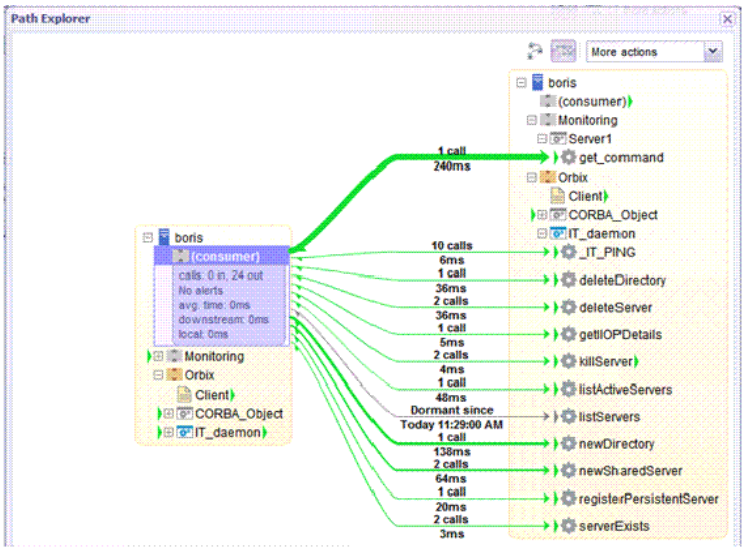

Figure 13: *Actional Server Path Explorer*

The **Path Explorer** view displays the relationships between different components in more detail. For example, you can view the call chain between services and consumers. Summary statistics are also displayed for the selected component.

Statistics details

The **Statistics Details** pane on the right displays statistics gathered by the selected component. These include the number of incoming and outgoing calls, call time, call size, and so on. Alerts, faults and violations are also displayed.

For example, Figure 14 shows the **Statistics Details** displayed for a client request to a server, when the operation is selected in the **Path Explorer**.

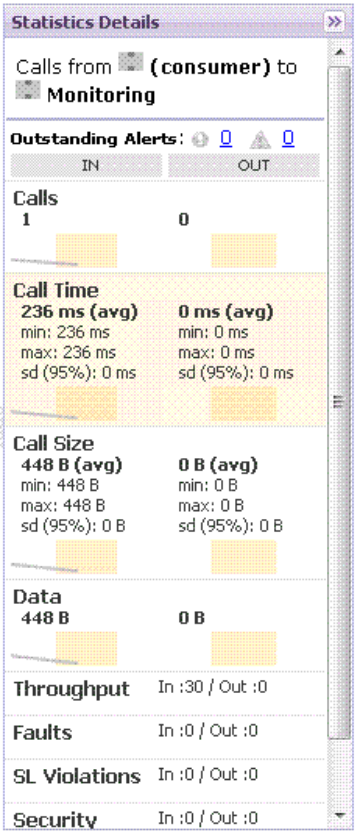

Figure 14: *Actional Server Statistics Details*

Double clicking on a particular statistic in this view (for example, **Call Size**) displays a summary chart. For example, Figure 15 shows a **Call Time** summary chart for the consumer.

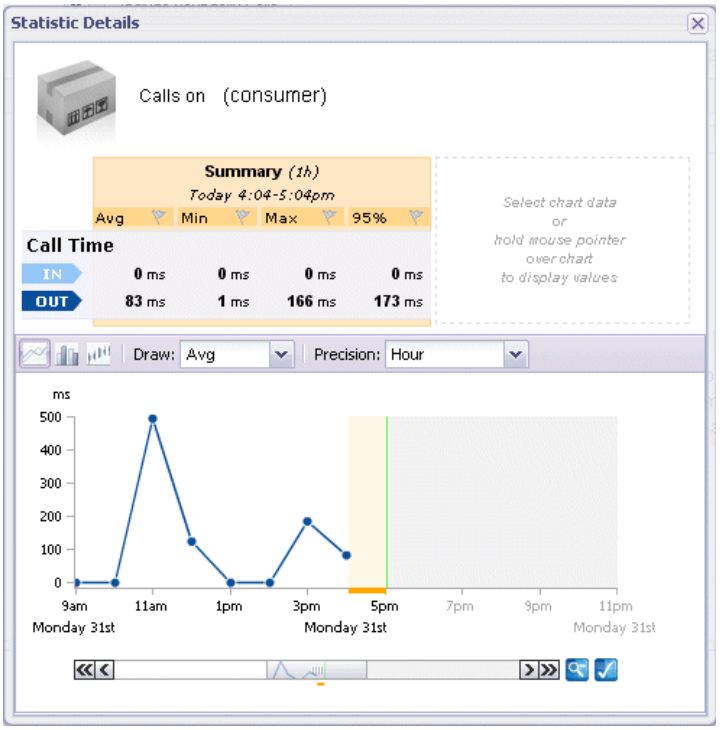

Figure 15: *Actional Server Statistics Chart*

Server manifest

The Actional server manifest (`LG_Header`) is a unique ID used by the Actional server to correlate information it receives from agents about interactions between different applications. For example, when you run the client application in the Orbix `actional_demo`, the following `LG_Header` is output on the command line:

```
Interaction=CgIEAUD6LU2sLiQBBwAAAA==;
Locus=4/LcwgqvldfxotEoegsSGg==;
Flow=CgIEAUD6LU2sLiQBBgAAAA==;
UpstreamOpID=xPnAfuwlTEV7QGYoGRBgYA==;
CallerAddress=10.2.4.1;
```

Further information

For detailed information on using Actional SOA management tools, see the Actional product documentation.

Auditing Orbix Applications

This section shows some simple examples of auditing the Orbix `actional_demo` and Orbix domain services.

Actional policy groups

Policy groups are used by Actional server to apply a set of policies and rules to managed items on your network. Policies and rules can be used to raise alerts on certain failure reasons. For example, when an Orbix operation takes too long to return, or when a specified IDL exception or fault is raised.

Figure 16 shows some example policy groups that have be defined in the **Policies** view. See configuring message fields section, for more detailed example on how to setup Policy Groups.

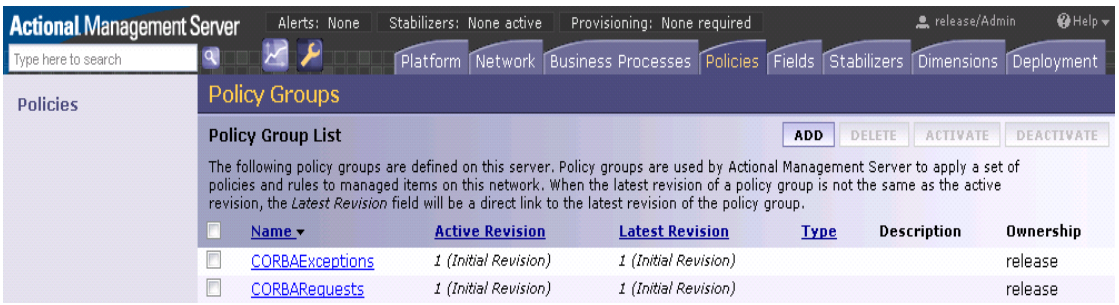

Figure 16: *Actional Policy Groups*

Viewing audit logs

When you have defined policies for your network, you can use them to audit and monitor alerts on certain failure reasons (for example, when a specified IDL exception or fault is raised).

Figure 17 shows some example audit logs for the Orbix application in the **Logs** view.

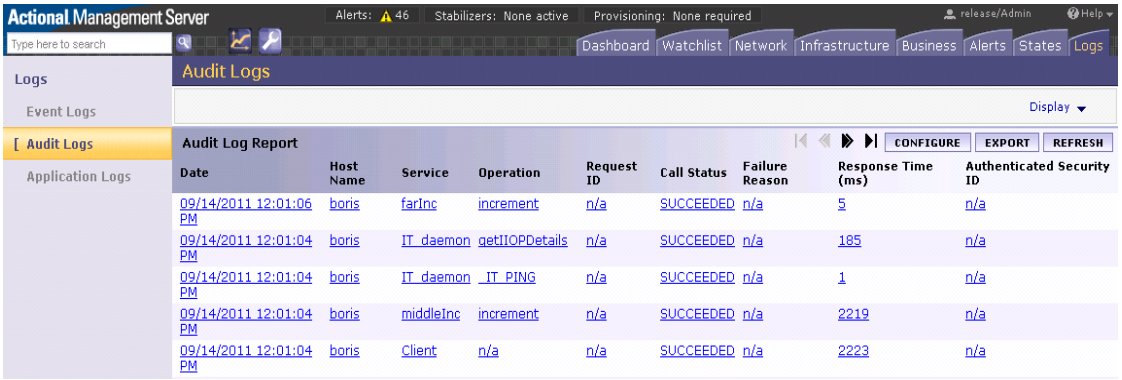

Figure 17: *Audit Logs from instrumented application*

Figure 18 shows an example audit log record displayed on clicking on an entry for the Orbix application in Figure 17.

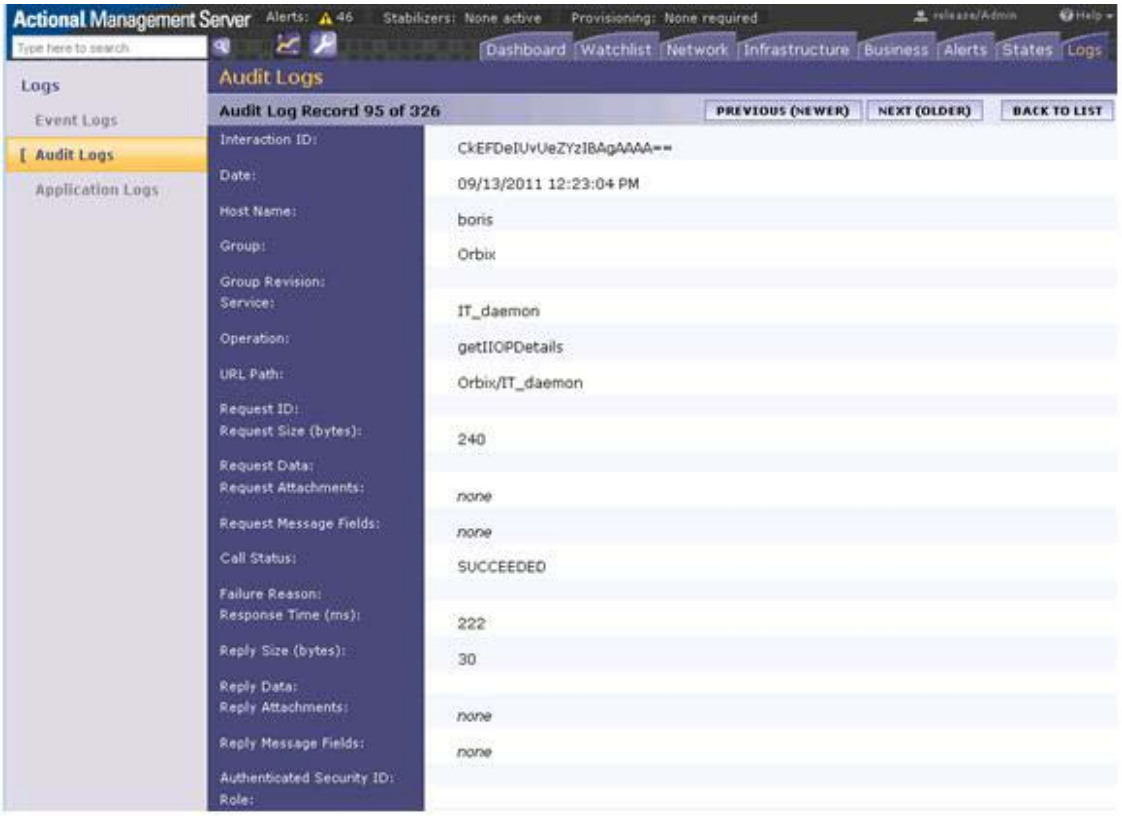

Figure 18: *Orbix Daemon call getIIOPDetails Audit Log Record*

The **Interaction ID** displayed at the top of the screen is used by the Actional server to correlate information it receives, from multiple agents, about interactions between different services.

Figure 19 shows some example audit logs for Orbix configuration domain services in the **Logs** view. The Orbix service displayed in this example is the Orbix node daemon.

Figure 19: *Audit Logs from application*

Figure 20 shows an example audit log record displayed on clicking an entry for the farInc server in Figure 19.

Figure 20: *Orbix server farInc's Log Record*

Message Fields

Message fields are pieces of textual data that are reported, such as the TCP port.

For C++ applications, only the remote port is actually reported:

- For Client C++ applications, this is the port to which the client is connected. This is also called the server port.
- For Server C++ applications this is the local port of the application, or the port where the server is listening on.

For Java applications, both the clientport and serverport are reported, which means both the local and remote ports of both sides of client and server.

Message fields are turned on by default and are immediately available once the Actional Management Server is configured to look for the message fields.

Configuring Actional to report Message Fields

The Actional Management Server console allows you to configure policies and alerts in configuration mode (click **States** tab, and on the **Stabilizers** page click the **configuration page** link).

1. To create a Policy Group, click the **Policies** tab.
2. Click **Add** to create a new policy group.
3. Enter the **Name, Type,** and **Description.**
4. Click **FINISH**.

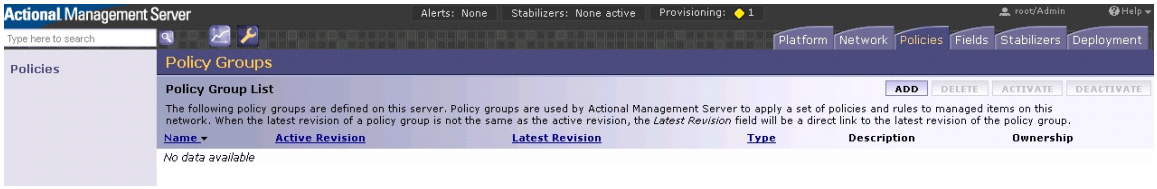

Figure 21: *Creating a Policy*

Figure 22: *Creating a Policy Group*

5. For creating a Rule Set, click the **Policies** tab, and click the name of the policy group to which you want to add a rule set.
6. In the **Rule Set** section, click **Add**.
7. Enter the **Name** and **Description** on the **Rule Set - Identification** page.
8. Click **OK**.

Figure 23: *Creating a new RuleSet*

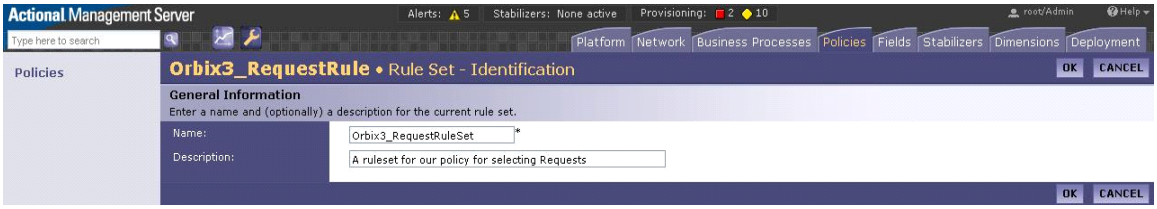

Figure 24: *Creating a RuleSet*

9. After creating the Rule Set, click **Add** on the Rule Set Summary.

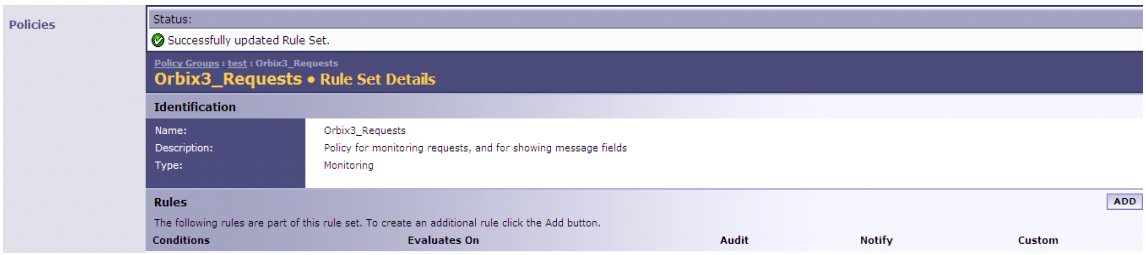

Figure 25: *Add Rule evaluation*

10. On the **New Monitoring Rule - Specify Evaluation** page, select **Always on Request**, and click **Next**.

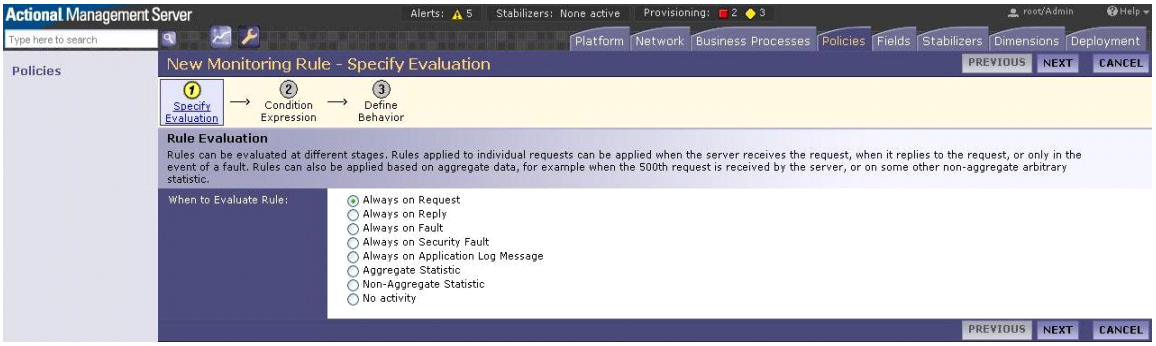

Figure 26: *Creating a Rule, selecting a type of Rule*

11. On the **New Monitoring Rule - Condition Expression** page, select **The action will always be performed** option, and click **Next.**

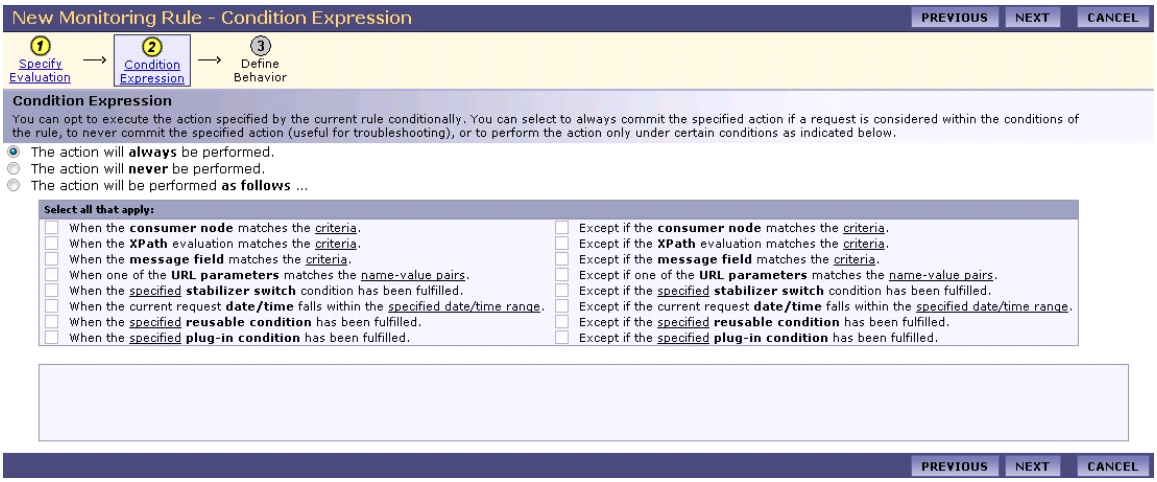

Figure 27: *Creating a Rule, selecting condition*

12. Any alerts or warnings need not to be setup, hence on the **New Monitoring Rule - Define Alerting Behavior** page, select **None** option, and click **Next**.

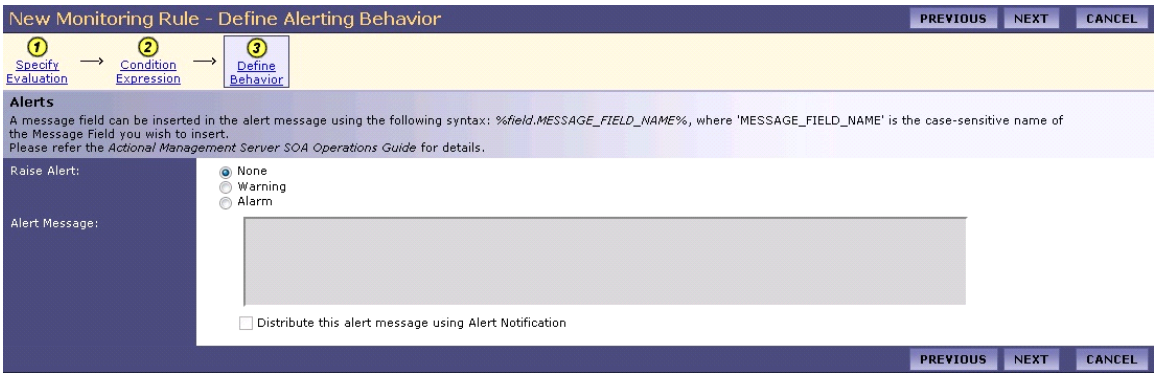

Figure 28: *Creating a rule, specifying alerts*

13. Do not specify any alerts to be shown.
14. Click **Next** Button. Select all items in the request to be audited, including the message fields (that are already defined). Ensure the last check box **Audit Only if Alarm is Raised** is not selected. If this is selected, the policy will not audit the calls unless an alarm/fault is raised.
15. Click **Next** until you get the **Finish** button on the screen. Clicking **Finish** button takes you to the **Policy Group** view.

Figure 29: *Creating a rule, and specifying message fields*

16. Create a new policy, by clicking the Policy Group, and clicking **Add** on the **Policies** section.
17. Select the Policy that needs to be applied for all sites, and click **Next**.

18. On the **New Policy - Configuration** page, select **Every message** option, and click **Next**.

Figure 30: *Creating a new Policy*

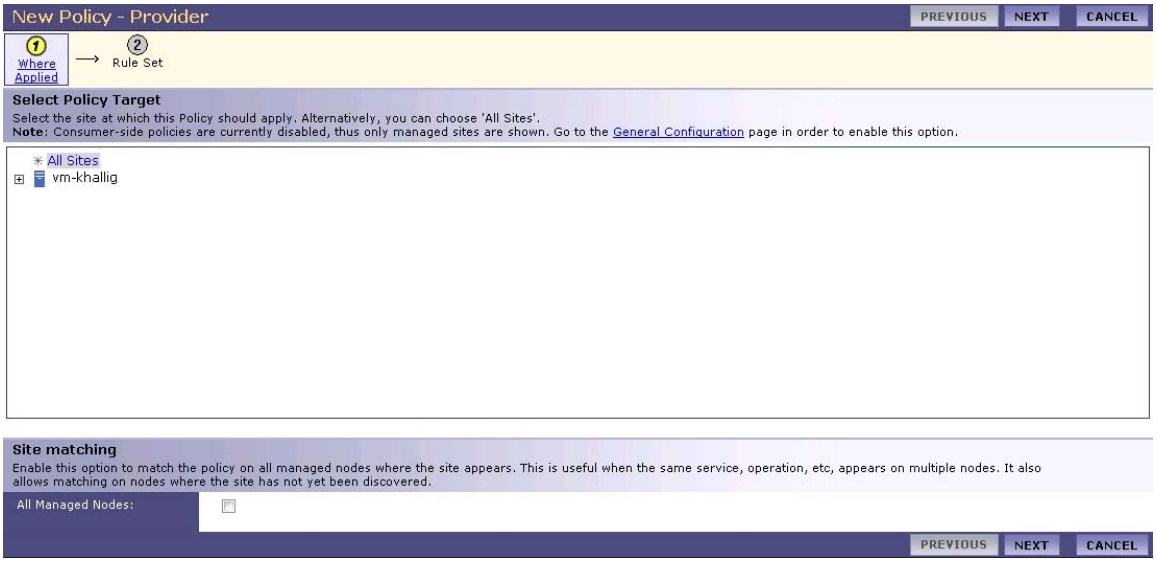

Figure 31: *Creating a new policy, select sites for policy*

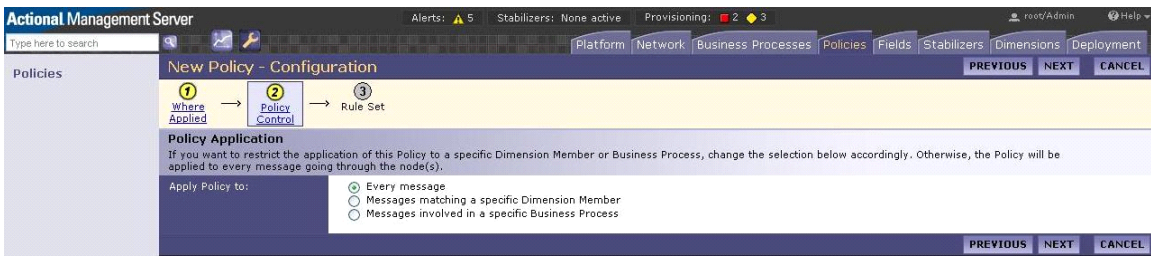

Figure 32: *Create a new Policy, selecting message type*

19. Select the **RuleSet** that needs to be associated with this policy, and click **OK.**

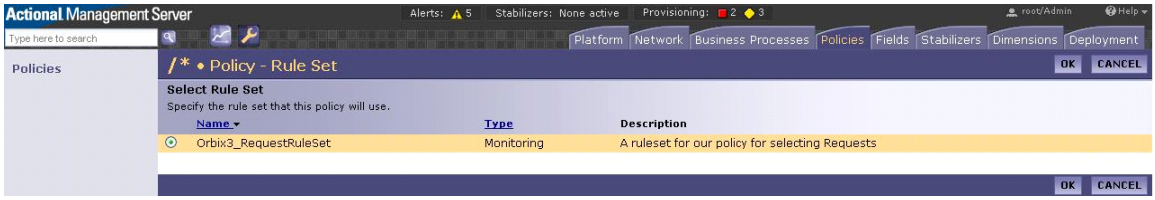

Figure 33: *Create a new Policy, Selecting a ruleset for the policy*

20. Once you are completed creating your policy, click **ACTIVATE THIS REVISION** which locks the policy and applies it to all sites. All the nodes in the network need to be re-provisioned.

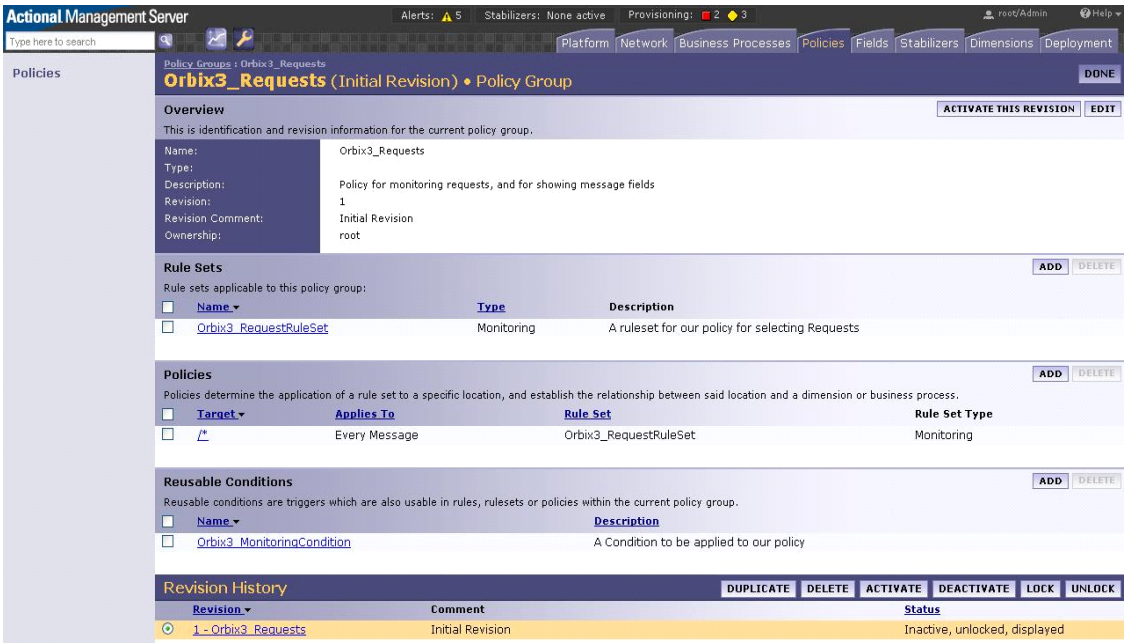

Figure 34: *Overview of the Policy Group*

21. Once you complete re-provisioning, any new monitored calls are available in the Audit Logs. If you click on any of the new log entries, details of the message fields are displayed. To view the **Audit Logs,** click Stabilizers tab | Stabilizer Management page link | Audit Logs.

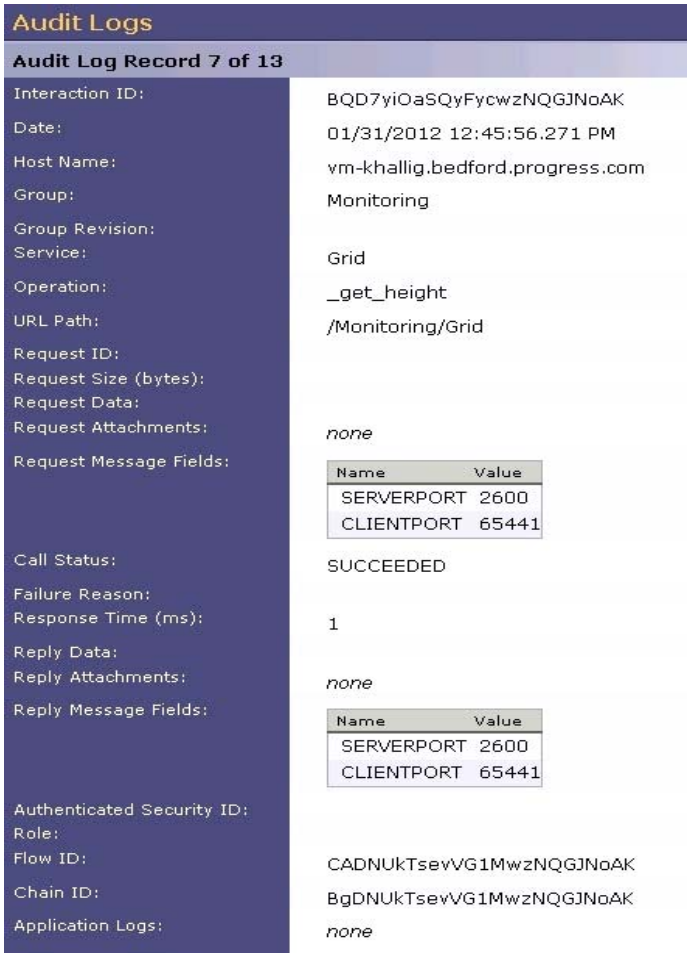

Figure 35: *Audit log entry showing message fields*

Index

T

U

W