Red Hat JBoss Fuse 6.3

Administration and Configuration on JBoss EAP

Administration and Configuration for JBoss Fuse on JBoss EAP

Last Updated: 2018-07-16

Red Hat JBoss Fuse 6.3 Administration and Configuration on JBoss EAP

Administration and Configuration for JBoss Fuse on JBoss EAP

JBoss A-MQ Docs Team
Content Services
fuse-docs-support@redhat.com

Legal Notice

Abstract

Use this guide to help you administer and configure JBoss Fuse on JBoss EAP

Table of Contents

CHAPTER 1. FUSE ON JBOSS EAP

1.1. WHAT IS FUSE ON JBOSS EAP?

Fuse on JBoss EAP is a platform for developing enterprise application integration (EAI) and service-oriented architecture (SOA) solutions. It consists of a service component framework, business rules/complex event processing, life-cycle governance, runtime governance, and process automation. Fuse on JBoss EAP is built on the same core as JBoss Fuse, and includes enterprise messaging, Apache Camel, and Apache CXF. Fuse on JBoss EAP enables users to design, deploy, integrate, and orchestrate business services.

Fuse on JBoss EAP can be used to integrate your major business systems into a cohesive infrastructure. Development is simplified with a transparent, lightweight service framework which uses Enterprise Integration Platform (EIP) technology. This allows developers to work with familiar technologies such as Apache Camel, BPEL, BPMN, or POJOs. To reduce the operational costs of production and maintenance, the platform utilizes an automatable, content-aware repository and service activity monitoring. These support the entire service life cycle.

1.2. CORE FUNCTIONALITY

Fuse on JBoss EAP provides the following core functionality:

Enterprise Integration Pattern (EIP) Based Development

The versatile EIP framework is implemented in routing and transformation processes for faster and more efficient integration solutions.

High Performance Messaging

A high performance messaging broker supports messaging patterns such as publish-subscribe, point-to-point, and store-forward, and multiple cross language clients.

Service Development

The web services framework exposes integration assets as services and calls external services, supporting all major web services standards. It also supports RESTful calls.

Structured Service Development

A lightweight service development framework provides full life-cycle support for developing, deploying, and managing service-based applications.

Automatable Registry with Workflow

Manage the life-cycle of services from design, development, and deployment by defining, exposing, and enforcing rules or policies.

Business Transaction Monitoring

Capture service activity information, define and collect metrics, and define alerts and SLAs.

1.3. SYSTEM INTEGRATION

Integrating your major business systems into a cohesive infrastructure can be a challenge, especially when you have legacy applications. Fuse on JBoss EAP provides a number of ways that enable you to integrate both new and legacy applications. Development is simplified with a transparent, lightweight

service framework which uses the EIP technology. This allows developers to focus on higher order concepts while still working with familiar technologies such as Apache Camel, BPEL, BPMN, or POJOs. To reduce the operational costs of production and maintenance, the platform utilizes an automatable, content-aware repository and service activity monitoring. These support the entire service life cycle and development, QA, and production teams with runtime and design-time visibility, monitoring, and alerting.

1.4. CORE AND COMPONENTS

Fuse on JBoss EAP provides an environment for easily applying SOA concepts to integrated applications. A SwitchYard application consists of components such as composite services and composite references. These provide service definitions and accessibility.

Along with SwitchYard, Fuse on JBoss EAP is made up of a number of components including a rules-based router (Apache Camel), a web service framework (Apache CXF), and a message broker (HornetQ).

1.5. COMPONENTS OF FUSE ON JBOSS EAP

Fuse on JBoss EAP ships with a number of components which enable its multi-functional capabilities.

Table 1.1. Fuse on JBoss EAP Components

Component	Function
SwitchYard	Service delivery framework
JBoss Rules	Business rules engine with complex event processing
Business Process Management	Business Process Management
Design Time Governance	A service registry/repository
Runtime Governance	Service activity monitoring
JBoss Operations Network	Operations, administration, and management tools
JBoss EAP	A full JavaEE application server
Apache Camel	Rules Based Router
Smooks	Framework for processing XML and non-XML data using Java
ModeShape	Data Store
HornetQ	Messaging and Integration Patterns Server
Apache CXF	Services Framework

These components can be used in Fuse on JBoss EAP to enable developers to build the required functionality using reliable and familiar tools. Some examples of how the components can be used are shown below:

Bean Services with CDI

SwitchYard leverages the power of Java EE6 and CDI to allow Java objects become services by adding an @Service annotation to your bean. Beans are automatically registered at runtime and references to other services can be injected as CDI beans using the @Inject annotation. Use CDI in your JSP and JSF applications to inject enterprise services into the web tier.

Declarative Transformation

With declarative transformation in SwitchYard, you can define the transformation and types to which it applies. SwitchYard automatically registers and executes the transformation. Choose from Smooks, Java, XSLT, JSON, and more.

Decision Services with JBoss Rules

Encapsulate business rules as decision services using the JBoss Rules component in SwitchYard. Each service has a well-defined contract with protocol binding details and marshaling details abstracted away by SwitchYard.

Smooks

This transformation engine can be used in conjunction with Fuse on JBoss EAP to process messages.

Business Process Execution Language (BPEL)

You can use web services to orchestrate business rules using this language. It is included with Fuse on JBoss EAP for the execution of business process instructions.

JBoss Rules (BRMS)

This is the rules engine that is packaged with Fuse on JBoss EAP. It can infer data from the messages it receives to determine which actions need to be performed.

IMPORTANT

A JBoss Fuse subscription includes an entitlement to use embedded BRMS (Drools) as a SwitchYard component only. All other uses (for example, with Apache Camel) require a separate BRMS subscription.

Business Process Management (BPM)

A pluggable container in SwitchYard that allows you to expose a business process as a service.

IMPORTANT

A JBoss Fuse subscription includes an entitlement to use embedded BPM as a SwitchYard component only. All other uses (for example, with Apache Camel) require a separate BPM subscription.

Testing

Comprehensive unit test support is provided to allow you to test services as you develop them.

1.6. FUSE ON JBOSS EAP FEATURES

SwitchYard

SwitchYard is a lightweight service delivery framework providing full life-cycle support for developing, deploying, and managing service-oriented applications.

Business Process Execution Language (BPEL)

You can use web services to orchestrate business rules using this language. It is included with Fuse on JBoss EAP for the execution of business process instructions.

Smooks

This transformation engine can be used in conjunction with Fuse on JBoss EAP to process messages. It can also be used to split messages and send them to the correct destination.

JBoss Rules

This is the rules engine that is packaged with Fuse on JBoss EAP. It can infer data from the messages it receives to determine which actions need to be performed.

1.7. INTEGRATION USE CASE

Acme Equity is a large financial service. The company possesses many databases and systems. Some are older, COBOL-based legacy systems and some are databases obtained through the acquisition of smaller companies in recent years. It is challenging and expensive to integrate these databases as business rules frequently change. The company wants to develop a new series of client-facing e-commerce websites, but these may not synchronize well with the existing systems as they currently stand.

The company wants an inexpensive solution but one that adheres to the strict regulations and security requirements of the financial sector. What the company does not want to do is to have to write and maintain “glue code” to connect their legacy databases and systems.

Fuse on JBoss EAP was selected as a middleware layer to integrate these legacy systems with the new customer websites. It provides a bridge between front-end and back-end systems. Business rules implemented with Fuse on JBoss EAP can be updated quickly and easily.

As a result, older systems can now synchronize with newer ones due to the unifying methods of Fuse on JBoss EAP. There are no bottlenecks, even with tens of thousands of transactions per month. Various integration types, such as XML, JMS and FTP, are used to move data between systems. Any one of a number of enterprise-standard messaging systems can be plugged into Fuse on JBoss EAP providing further flexibility.

An additional benefit is that the system can now be scaled upwards easily as more servers and databases are added to the existing infrastructure.

CHAPTER 2. READ ME

2.1. BACK UP YOUR DATA

WARNING

Red Hat recommends that you back up your system settings and data before undertaking any of the configuration tasks mentioned in this book.

2.2. RED HAT DOCUMENTATION SITE

Red Hat's official documentation site is at https://access.redhat.com/site/documentation/. There you will find the latest version of every book, including this one.

2.3. EAP_HOME

`EAP_HOME` refers to the root directory of the Red Hat JBoss Enterprise Application Platform installation on which JBoss Fuse is deployed.

2.4. MODE

`MODE` will either be **`standalone`** or **`domain`** depending on whether your instance of JBoss Enterprise Application Platform is running in standalone or domain mode. Substitute one of these whenever you see **`MODE`** in a file path in this documentation.

CHAPTER 3. THE ADMINISTRATION INTERFACES

The following interfaces are provided for administering the different components of JBoss Fuse.

SwitchYard

- JBoss EAP Management Console
- JBoss EAP Management CLI
- JBoss Operations Network

Business Rules Orchestration

- BPEL Console

Governance

- Runtime Governance Console
- Design Time Governance Console
- S-RAMP Console
- S-RAMP Management CLI

CHAPTER 4. MANAGING SWITCHYARD WITH JBOSS EAP MANAGEMENT CONSOLE

4.1. JBOSS EAP MANAGEMENT CONSOLE

The JBoss EAP Management Console is a web-based administration tool for JBoss EAP 6.

Use the Management Console to start and stop servers, deploy and undeploy applications, tune system settings, and make persistent modifications to the server configuration. The Management Console also has the ability to perform administrative tasks, with live notifications when any changes require the server instance to be restarted or reloaded.

4.1.1. Log in to the Management Console

Prerequisites

- JBoss EAP 6.4 must be running.

1. **Navigate to the Management Console start page**
   Launch your web browser and navigate to the Management Console in your web browser at http://localhost:9990/console/App.html

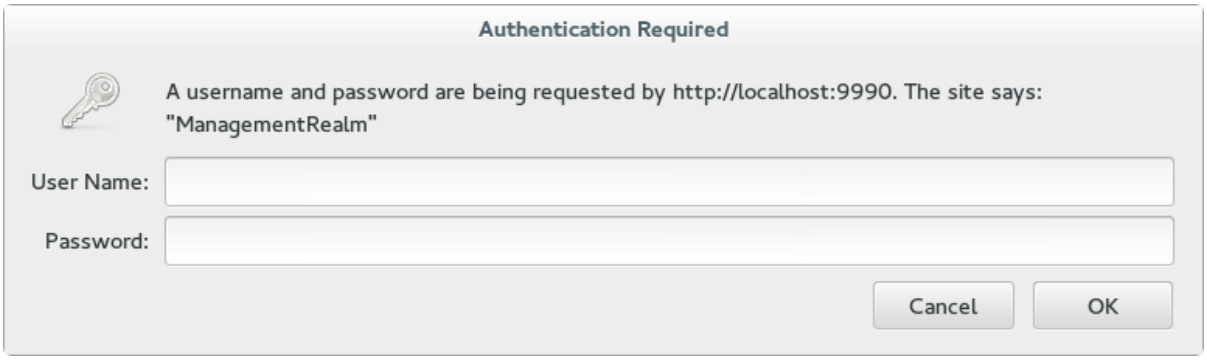
   

   NOTE

   Port 9990 is predefined as the Management Console socket binding.

2. Enter the username and password of the account that you created previously to log in to the Management Console login screen.

   Figure 4.1. Log in screen for the Management Console

Result

Once logged in, you are redirected to the following address and the the Management Console landing page appears: http://localhost:9990/console/App.html#home

4.1.2. Deploy an Application Using the Management Console

1. Select the **`Runtime`** tab at the top of the console.
2. Expand the **Server** menu and select **Manage Deployments** to go to the **`Deployments`** panel.
3. Select **`Add`**. A **`Create Deployment`** dialog box will appear.

4. In the dialog box, click **Browse**. Browse to the file you want to deploy and select it for upload. Click **Next** to proceed.

5. Verify the deployment name and runtime name that appear in the **Create Deployments** dialog box. Click **Save** to upload the file once the names are verified.

To enable the deployed application, select the application name in the **Available Deployments** table, and click **En/Disable**. Confirm the process in the pop-up dialog.

4.1.3. SwitchYard Management Console

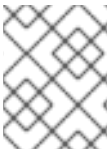
The SwitchYard management console is integrated with the standard Red Hat JBoss Enterprise Application Platform management console and provides the following:

- A view of the applications and services deployed on the server.
- A view of various execution metrics.
- A view of the SwitchYard subsystem configuration.

SwitchYard contributes views to the standard Red Hat JBoss EAP management console's Runtime and Profile pages.

4.2. METRICS VIEWS

SwitchYard Message Metrics can be accessed on the JBoss EAP Management Console via **Runtime** > **Status** > **Subsystems** > **SwitchYard** . This page provides a view of a comprehensive set of metrics aggregated at specific levels within the system.



NOTE

These instructions are for Standalone mode. For Domain mode, select the server you want to work on by clicking the **Change Server** button.

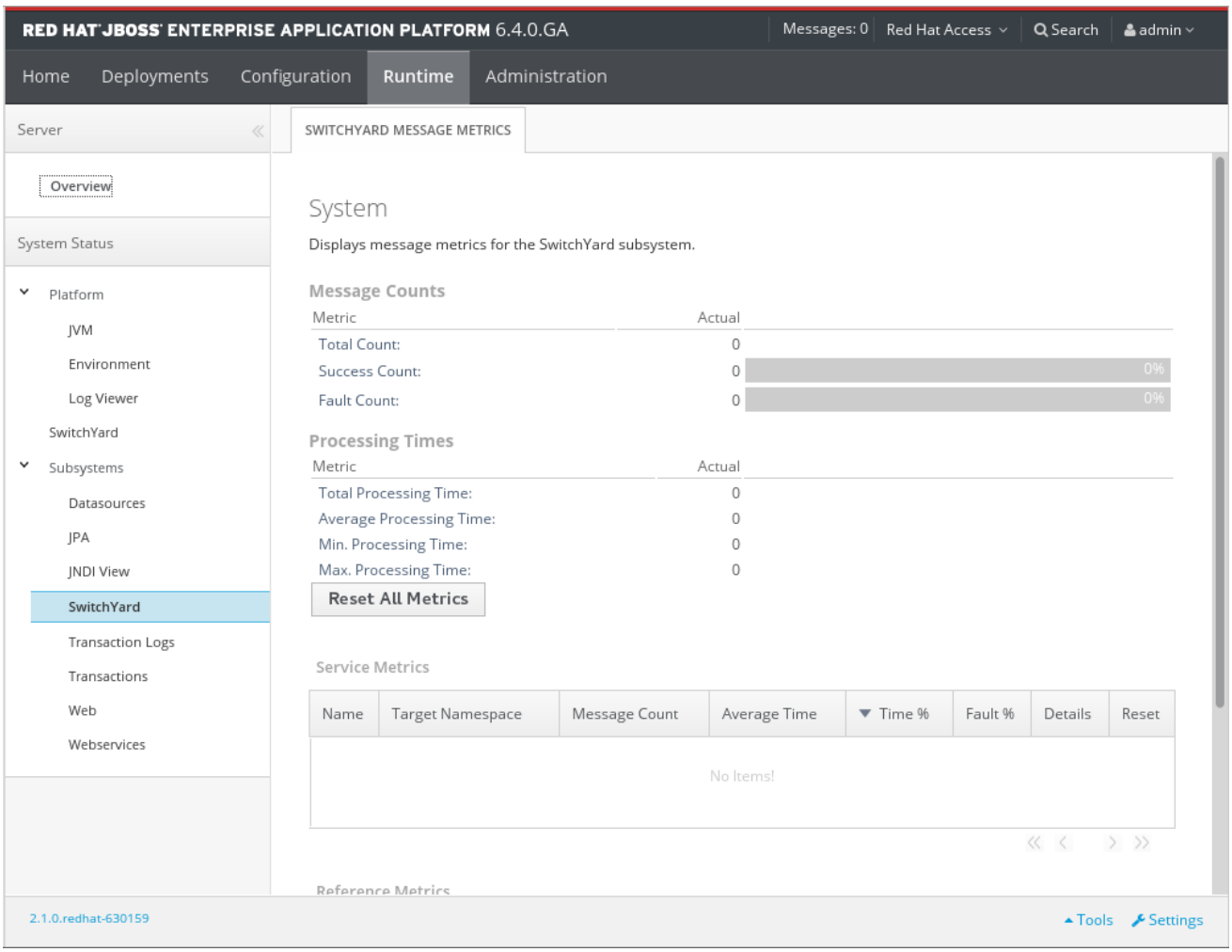
You can view collected metrics at the following levels:

- **System** : metrics for the entire SwitchYard runtime (all deployed applications)
  - **Message Counts**: This shows the total number of messages, number of successful messages and number of failed messages.
  - **Processing time**: This shows the total, minimum, average and maximum processing times.
- **Service** : metrics for a composite service in an application. Additional metric details are provided for the following:
  - **Gateway**: metrics for each binding on the service (For example, FTP metrics for service "ABC")
  - **Operation**: metrics for each operation on the service
  - **Service Reference** : metrics for references invoked by the service

- **`Reference`** : metrics for a composite reference in an application. Additional metric details are provided for the following:
  - **`Gateway`** : metrics for each binding on the reference (For example, FTP metrics for reference "ABC")
  - **`Operation`** : metrics for each operation on the reference

This page also provides the user with the ability to reset metrics.

Figure 4.2. JBoss EAP Management Console SwitchYard Metrics Page



4.3. APPLICATION VIEWS

SwitchYard contributes a page to the Runtime Operations section of the JBoss EAP Management Console, which provides views detailing various aspects of SwitchYard applications running on the system. These views may be accessed by clicking **`Runtime`** > **`Runtime Operations`** > **`SwitchYard`**.

NOTE

For Domain mode, select the server by clicking the **`Change Server`** button.

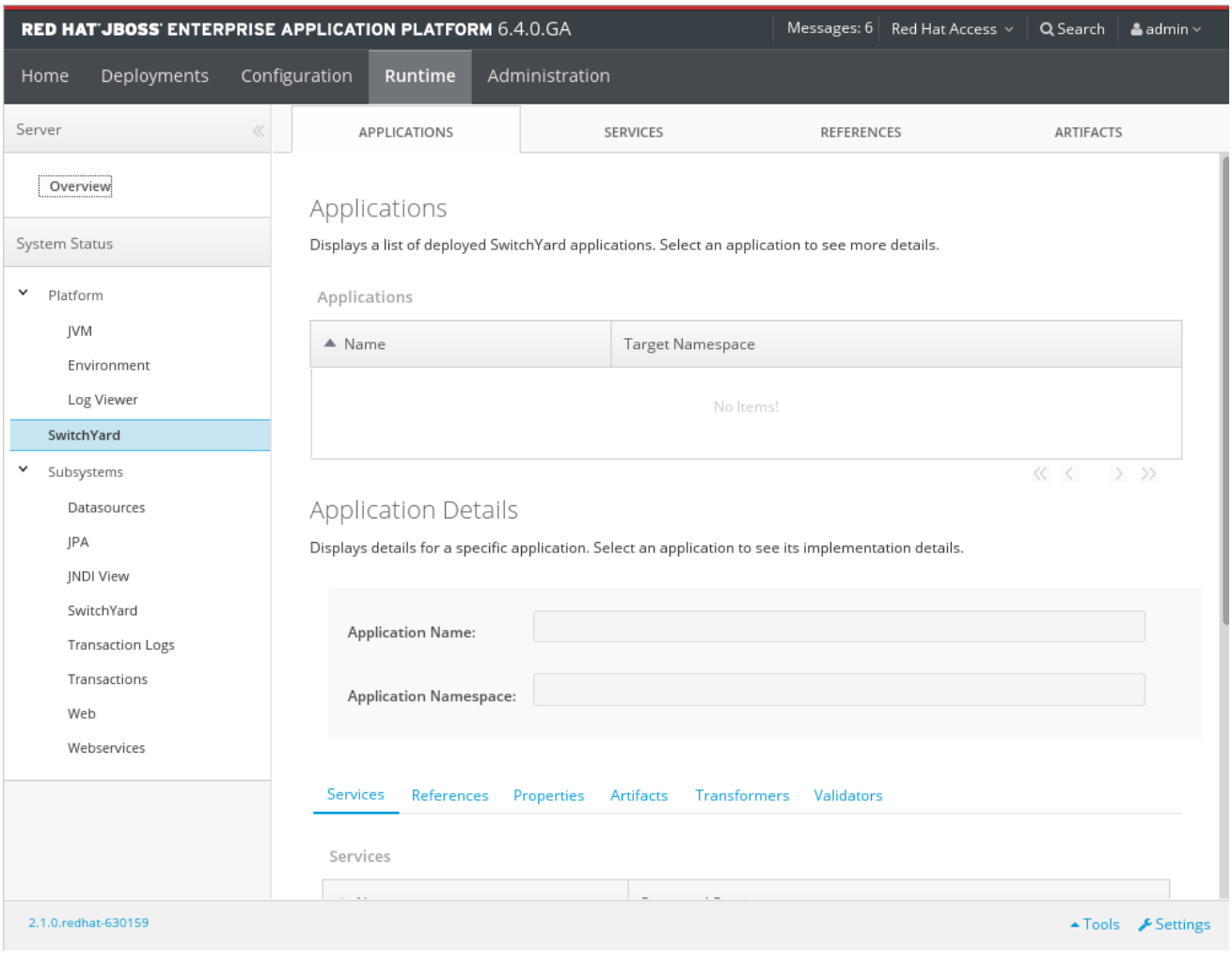
The following views are provided:

- Applications: lists all SwitchYard applications deployed on the server

For each application there is an **`Application Details`** panel. This panel contains information for each deployed SwitchYard application, and will show the details when the application is selected in the **`Applications`** list.

- Services: lists all Services provided by the applications deployed on the server
- References: lists all service References used by applications deployed on the server
- Artifacts: lists all artifacts referenced by applications deployed on the server

Figure 4.3. JBoss EAP Management Console Runtime Operations page



4.3.1. Applications

The main **Applications** tab displays all the applications deployed on the server. Select an application from the list to populate the **`Application Details`** section below. The following details are provided:

Table 4.1. JBoss EAP Management Console operations available for SwitchYard Applications page

Operation Name	Function
Services	Services provided by the application. Select a service to open the **Services** tab which will display details for the service.

Operation Name	Function
References	References used by the application. Select a reference to open the **References** tab which will display details for the reference.
Properties	Properties defined in the application. The properties may also be edited within this view.
Artifacts	Artifacts referenced by the application. Select an artifact to open the main **Artifacts** tab, which displays details for the artifact
Transformers	Transformers configured in the application.
Validators	Validators configured in the application.

The **Services** tab displays information about the services provided by the application. This information includes the services provided by the application and the component services used to implement the services.

Table 4.2. SwitchYard Applications page, Services Tab, Services Table.

Operation Name	Function
Name	The name of the component service
Promoted Service	The name of the component service providing the implementation for the service.

Select an item in the **`Name`** column of the **`Services`** table to open the main **Services** tab, which displays details for that service.

When you select an item in the **`Promoted Service`** column, the corresponding item in the **`Component Services`** table is highlighted. The **`Component Services`** table displays the component services defined in the application. This table provides the following details:

Table 4.3. SwitchYard Applications page, Services Tab, Component Services Table.

Operation Name	Function
Name	The name of the component service
Interface	The name of the component service providing the implementation for the service.
Implementation	Provides a link for viewing the implementation details of the component

Click on the **`View Details...`** button in the **`Implementation`** column to open a pop-up detailing the component's implementation. The **`Implementation Details`** dialog provides the following information:

- The technology used to implement the component (For example, Camel).

- A list of references required by the component.
- The raw configuration for the implementation.

The **References** tab lists all the composite references used by the application.

The **Properties** tab provides a list of properties defined in the application. In addition to viewing the properties, use this page to update the values for individual properties.

The **Artifacts** tab provides information about the artifacts referenced by the application and is comprised of a table providing the following details:

Table 4.4. SwitchYard Applications page, Artifacts Tab, Artifact References Table.

Operation Name	Function
Name	The name of the referenced artifact
URL	The location of the artifact.

The **Transformers** tab provides details about the transformers deployed by the application, providing the following details:

Table 4.5. SwitchYard Applications page, Transformers Tab

Operation Name	Function
From	The from type supported by the transformer.
To	The to type supported by the transformer.
Type	The implementation technology used by the transformer (For example Java and XSLT).

The **Validators** tab provides details about the validators deployed by the application, providing the following details:

Table 4.6. SwitchYard Applications page, Validators Tab

Operation Name	Function
Name	The name of the validator.
Type	The type of the validator.

4.3.2. Services

The main **Services** tab contains the **Services** table, which displays all services provided by the deployed applications. Select a service to populate the **Service Details** section below the list. The **Services** table shows:

Table 4.7. JBoss EAP Management Console operations available for SwitchYard Services page

Operation Name	Function
Name	The service name
Target Namespace	The namespace that the service is defined in.

The **`Service Details`** section shows the following details for the service selected in the **`Services table:`**

Table 4.8. SwitchYard Services page, Service Details section

Operation Name	Function
Name	The service name
Target Namespace	The namespace that the service is defined in.
Application	The application providing the service (this links to the main **`Applications`** tab)
Interface	The interface provided by the service.
Promoted Service	The component service implementing the service
Gateways	Lists the gateways providing access to the service
Throttling	Throttling configuration for the service

The **Gateways** tab in the details section provides the following information for each of the gateways provided for the service:

Table 4.9. SwitchYard Services page, Service Details section, Gateways Tab

Operation Name	Function
Name	The name pf the gateway.
Type	The type of the gateway (For example, SOAP and HornetQ)
Status	The status of the gateway (For example, started, stopped)
Start/Stop	Click the button to start or stop the gateway.
Configuration	The component service implementing the service
Gateways	Click the **`View Configuration...`** button to open a dialog that displays the raw configuration for the gateway

The **Throttling** tab in the details section allows the user to view throttling details for the service.

Table 4.10. SwitchYard Services page, Service Details section, Throttling Tab

Operation Name	Function
Edit	Switch to edit mode, allowing the user to change the throttling configuration.
Enable	Enable/disable throttling for the service
Status	The status of the gateway (For example, started, stopped)
Start/Stop	Click the button to start or stop the gateway.
Maximum Requests	The maximum number of requests per period before throttling occurs
Time Period	The time period over which requests are counted (cannot be edited)

4.3.3. References

The main **References** tab displays all service references used by the deployed applications. Select a reference from the **`References`** table to populate the **`Reference Details`** section. The **`References`** table contains:

Table 4.11. JBoss EAP Management Console operations available for SwitchYard References page

Operation Name	Function
Name	The name of the reference
Namespace	The namespace that the reference is defined in.

The **`Reference Details`** contain:

Table 4.12. SwitchYard References page, Reference Details section

Operation Name	Function
Name	The name of the reference
Namespace	The namespace that the reference is defined in.
Application	The application containing the reference (this links to the main **Applications** tab)
Interface	The interface provided by the reference.

The **Gateways** section provides the following information for each of the gateways configured for the reference:

Table 4.13. SwitchYard References page, Gateways table

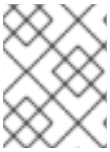
Operation Name	Function
Name	The name of the gateway
Type	The type of the gateway (For example, SOAP, HornetQ)
Status	The status of the gateway (For example, started, stopped)
Start/Stop	Click the button to start or stop the gateway
Configuration	Click the **`View Configuration...`** button to open a dialog displaying the raw configuration for the gateway

4.3.4. Artifacts

The main **Artifacts** tab displays all artifacts referenced by applications deployed to the system. Selecting a specific artifact reference will populate the **`Applications Using Artifact`** table. Selecting an application in the applications table will navigate to the main **Applications** tab.

4.4. SWITCHYARD RUNTIME DETAILS

The JBoss EAP management console contains a SwitchYard Runtime Details page, which can be accessed by selecting the **Profiles** > **Subsystems** > **SwitchYard** > **Runtime Details** node.



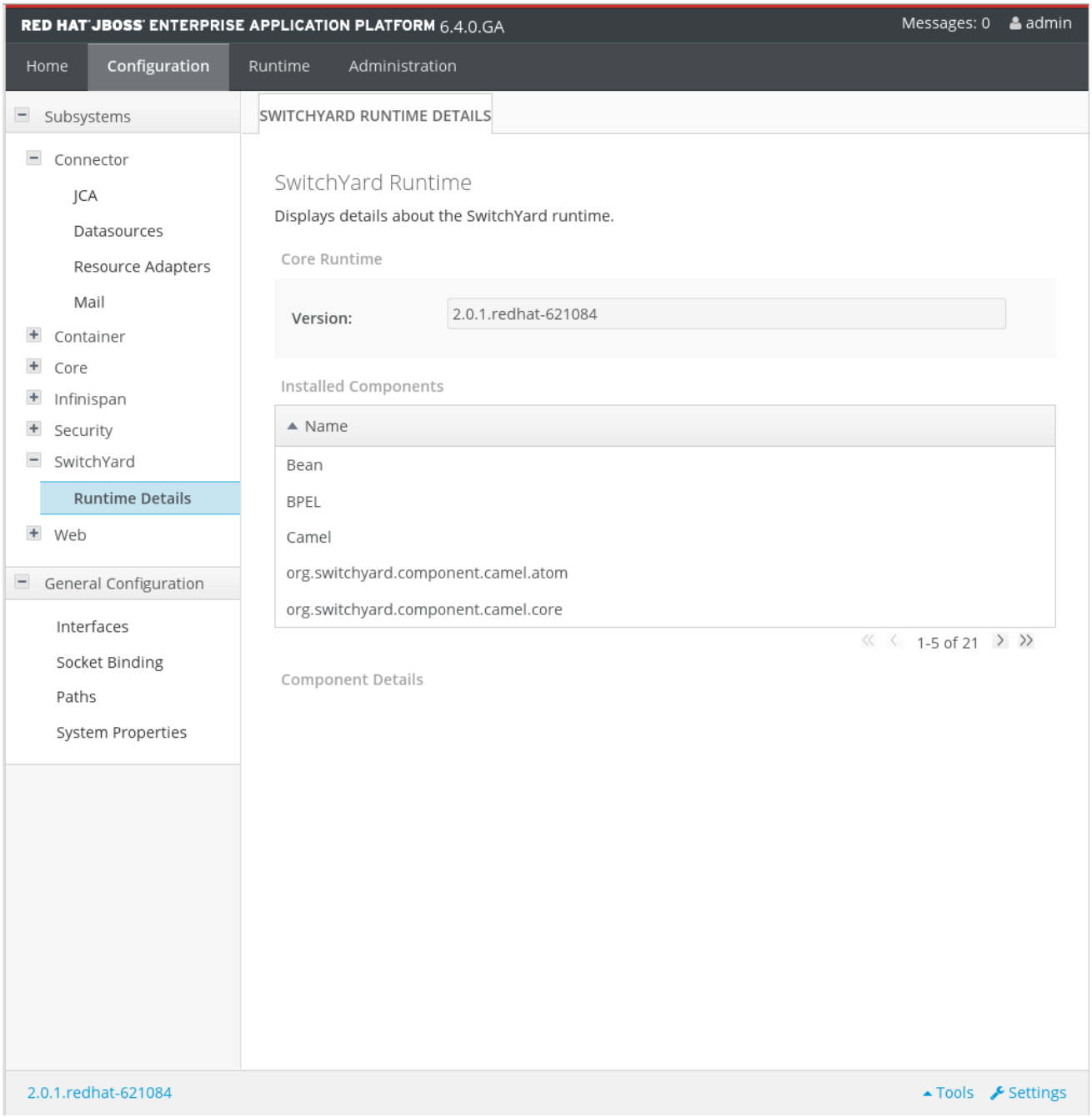
NOTE

For Domain mode, select the **`Profile`** before navigating to the SwitchYard Runtime Details page

This page displays details about components in the SwitchYard subsystem configured in JBoss EAP. When a component is selected from the list, the **`Component Details`** section is populated. The following details are displayed:

- **`Name`**: the component name, for example, SOAP, Camel.
- The **`Configured Properties`** section providing component specific details. For most components, this section lists any configurable properties and their current settings.

Figure 4.4. JBoss EAP Management Console SwitchYard Runtime Details Page



CHAPTER 5. MANAGING SWITCHYARD WITH JBOSS MANAGEMENT CLI

5.1. ABOUT THE MANAGEMENT COMMAND LINE INTERFACE (CLI)

The Management Command Line Interface (CLI) is a command line administration tool for JBoss EAP 6.

Use the Management CLI to start and stop servers, deploy and undeploy applications, configure system settings, and perform other administrative tasks. Operations can be performed in batch mode, allowing multiple tasks to be run as a group.

5.2. LAUNCH THE MANAGEMENT CLI

Procedure 5.1. Launch CLI in Linux or Microsoft Windows Server

- **Launch the CLI in Linux**
  Run the ***EAP_HOME*/bin/jboss-cli.sh** file by entering the following at a command line:

  ```
  $ EAP_HOME/bin/jboss-cli.sh
  ```

- **Launch the CLI in Microsoft Windows Server**
  Run the ***EAP_HOME*\bin\jboss-cli.bat** file by double-clicking it, or by entering the following at a command line:

  ```
  C:\>EAP_HOME\bin\jboss-cli.bat
  ```

5.3. QUIT THE MANAGEMENT CLI

From the Management CLI, enter the **quit** command:

```
[domain@localhost:9999 /] quit
```

5.4. CONNECT TO A MANAGED SERVER INSTANCE USING THE MANAGEMENT CLI

Prerequisites

- Section 5.2, “Launch the Management CLI”

Procedure 5.2. Connect to a Managed Server Instance

- **Run the `connect` command**
  From the Management CLI, enter the **connect** command:

  ```
  [disconnected /] connect
  Connected to domain controller at localhost:9999
  ```

  - Alternatively, to connect to a managed server when starting the Management CLI on a Linux system, use the ***--connect*** parameter:

```
$ EAP_HOME/bin/jboss-cli.sh --connect
```

- The **`--connect`** parameter can be used to specify the host and port of the server. To connect to the address **`192.168.0.1`** with the port value **`9999`** the following would apply:

```
$ EAP_HOME/bin/jboss-cli.sh --connect --
controller=192.168.0.1:9999
```

5.5. DEPLOY AN APPLICATION IN A STANDALONE SERVER USING THE MANAGEMENT CLI

Prerequisites

- Section 5.2, “Launch the Management CLI”
- Section 5.4, “Connect to a Managed Server Instance Using the Management CLI”

Procedure 5.3. Deploy an Application in a Standalone Server

- **Run the `deploy` command**
  From the Management CLI, enter the **`deploy`** command with the path to the application deployment.

  ```
  [standalone@localhost:9999 /] deploy /path/to/test-application.war
  ```

  Note that a successful deploy does not produce any output to the CLI.

Result

The specified application is now deployed in the standalone server.

5.6. UNDEPLOY AN APPLICATION IN A STANDALONE SERVER USING THE MANAGEMENT CLI

Prerequisites

- Section 5.2, “Launch the Management CLI”
- Section 5.4, “Connect to a Managed Server Instance Using the Management CLI”
- Section 5.5, “Deploy an Application in a Standalone Server Using the Management CLI”

Procedure 5.4. Undeploy an Application in a Standalone Server

By default the **`undeploy`** command will undeploy *and* delete the deployment content from a standalone instance of JBoss EAP. To retain the deployment content, add the parameter **`--keep-content`**.

- **Run the `undeploy` command**
  To undeploy the application and delete the deployment content, enter the Management CLI **`undeploy`** command with the filename of the application deployment.

```
[standalone@localhost:9999 /] undeploy test-application.war
```

To undeploy the application, but retain the deployment content, enter the Management CLI **undeploy** command with the filename of the application deployment and the parameter ***--keep-content***.

```
[standalone@localhost:9999 /] undeploy test-application.war --keep-content
```

Result

The specified application is now undeployed. Note that the **undeploy** command does not produce any output to the Management CLI if it is successful.

5.7. MANAGEMENT CLI COMMANDS FOR SWITCHYARD

Prerequisites

- Section 5.2, “Launch the Management CLI”
- Section 5.4, “Connect to a Managed Server Instance Using the Management CLI”

There are commands that are available for SwitchYard. To access the commands you first have to operate in the SwitchYard subsystem.

Ensure that you have started the Management CLI and connected to it.

The Management CLI employs tab completion in two ways:

- By completing partially entered commands if there is enough entered for it to be unique.
- By presenting a list of options if a unique result cannot be found, or if the command is missing.

If you enter the following command and press tab, you will get a list of operations available for the SwitchYard subsystem.

```
[standalone@localhost:9999 /] /subsystem=switchyard:
```

The available operations for the SwitchYard subsystem are:

Table 5.1. JBoss EAP Management CLI operations available for SwitchYard subsystem

Operation Name	Function
get-version	This will show the version of SwitchYard currently running on the JBoss EAP instance.
list-applications	This shows a list of the SwitchYard applications currently running on the JBoss EAP instance.
list-references	Lists the references used for SwitchYard applications currently running on the JBoss EAP instance.

Operation Name	Function
read-application	This lists the details for SwitchYard applications running on the JBoss EAP instance.
read-references	Lists the references used by SwitchYard applications running on the JBoss EAP instance.
read-service	Lists the services and details used by SwitchYard applications running on the JBoss EAP instance.
reset-metrics	Reset the message metrics for the SwitchYard subsystem.
show-metrics	List the message metrics for the SwitchYard subsystem.
start-gateway	Start the gateway for a SwitchYard service.
stop-gateway	Stop the gateway for a SwitchYard service.
update-throttling	Update message throttling for a SwitchYard service.
stop-gateway	Stop the gateway for a SwitchYard service.
uses-artifacts	Lists the artifacts used throughout the SwitchYard subsystem.

Enter the operations in the following format, replacing *operation-name* with one of the operation names from the list above:

```
[standalone@localhost:9999 /] /subsystem=switchyard:operation-name
```

The response will vary depending on the operation entered.

NOTE

For more information about the JBoss EAP Management CLI, see the Red Hat JBoss Enterprise Application Platform 6.1 Administration and Configuration Guide

CHAPTER 6. JBOSS OPERATIONS NETWORK

6.1. JBOSS OPERATIONS NETWORK

JBoss Operations Network gives administrators a single point of access to view their systems. JBoss Operations Network provides a means to develop and monitor a system's inventory. Every managed resource – from platforms to applications to services – is contained and organized in the inventory, no matter how complex the IT environment is.

JBoss Operations Network centralizes all of its operations in an installed server. The JBoss Operations Network server communicates with locally installed JBoss Operations Network agents. The agents interact directly with the platform and services to carry out local tasks such as monitoring. The types of resources that can be managed by JBoss Operations Network and the operations that can be carried out are determined by the server and agent plug-ins that have been loaded into JBoss Operations Network.

6.2. INSTALLING JBOSS OPERATIONS NETWORK FOR JBOSS FUSE SERVICE WORKS

Install JBoss Operations Network

To use JBoss Operations Network with Fuse Service Works, JBoss Operations Network must be installed as instructed in the JON Server Installation chapter of the Installing the JBoss ON Server. Note that both the Server and the Agent must be installed. See Installing and Upgrading an Agent on a Managed Platform from the JAR File for more information about installing the Agent.

In addition to the JBoss Operations Network installation, you will also have to download and install the JBoss Fuse Service Works agent Plug-in pack.

Use the installation instructions at JBoss Operations Network Plug-in Packs to install the JBoss Fuse Service Works agent plug-in pack.

Configure JBoss Operations Network for JBoss Fuse Service Works

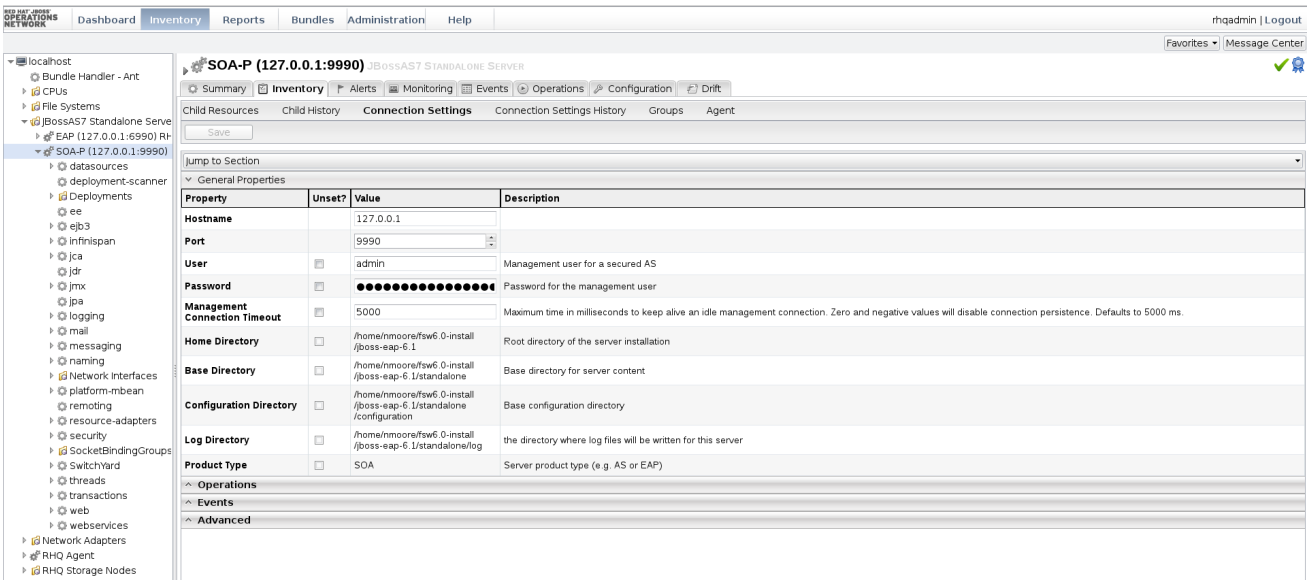
When JBoss Operations Network has been correctly installed, it will automatically detect the JBoss Fuse Service Works services.

To complete configuration for JBoss Fuse Service Works, update the **`Connection Settings`** for the SOAP-P server in the **JBossAS7 Standalone Server**:

Procedure 6.1.

1. Ensure that JBoss Operations Network and the JBoss Fuse Service Works instance of JBoss EAP are running. Log in to the JBoss Operations Network server GUI.
2. Click on the **`Inventory`** button at the top of the page.
3. Select **Servers** from the menu on the left hand side. Select **SOA-P** server from the list presented on the Servers page.
4. Select the **Inventory** tab on the **SOA-P** page, then click **Connection Settings**.
5. Update the **`User`** and **`Passwords`** fields on the **General Properties** section with an administrator User ID and Password for the JBoss Fuse Service Works JBoss EAP instance.

Figure 6.1. JBoss Operations Network Server Connection Settings Page



For more information regarding the configuration of JBoss EAP 6.1 on JBoss Operations Network, see the JBoss Operations Network documentation at Setting up JBoss EAP 6 Instances.

6.3. JBOSS OPERATIONS NETWORK FUNCTIONS

JBoss Operations Network provides a number of metrics and administration options for the networks it can be used to control. These functions can be accessed at different levels in the organizational hierarchy. The comprehensive documentation suite covers all the options and available processes. The JBoss Operations Network documentation suite can be found at Red Hat JBoss Operations Network 3.2 Documentation.

There are specific functions of JBoss Operations Network that are especially useful for JBoss Fuse Service Works. The most relevant functions are listed below, with links to the specific place in the JBoss Operations Network documentation.

Table 6.1. JBoss Operations Network functions most relevant for Red Hat JBoss Fuse Service Works

Function	Link
Configure the Agent to Discover JBoss EAP Instances	Configuring the Agent to Discover EAP Instances
Deploy Web Applications to a Standalone Server	Deploying Web Applications to a Standalone Server
Troubleshooting Deployments	Troubleshooting Deployments
Monitoring EAP Resources including Run Time information and metrics for SwitchYard	Setting up Monitoring for EAP 6 Resources
Set up alerts on JBoss EAP Resources	Alerting on JBoss EAP 6 Resources

6.4. AUTOMATIC SERVICE DISCOVERY

The JBoss Operations Network agent can automatically detect SwitchYard archives deployed or deleted independently of JBoss Operations Network. This is known as the Automatic Service Discovery feature. The default frequency for the Automatic Service Discovery is every 24 hours (86400 seconds). The scan frequency of this service and other scan services provided by JBoss Operations Network, can be configured by following the instruction at Setting Discovery Scan Intervals.

CHAPTER 7. BPEL CONSOLE

7.1. ABOUT THE BPEL CONSOLE

The BPEL Console is a web based interface to manage, administer and debug processes deployed on a BPEL Server. It enables you to run, test and manage BPEL processes.

It allows you to view:

- any process definitions you have deployed to the BPEL engine
- the process instances executing in the BPEL engine
- a process execution history
- the query pertaining to the execution history

7.2. LOG IN TO THE BPEL CONSOLE

Navigate to the BPEL console using http://localhost:8080/bpel-console.

NOTE

It is recommended that you only open one **`bpel-console`** window within your browser.

When the login screen is displayed, enter **`admin`** as the default userid and **`admin`** as the default password.

7.3. PROCESS DEFINITION

A BPEL Process Definition is an XML file that acts as a template for a process. When deployed as a part of the SwitchYard application, you can access the BPEL process through various bindings supported by SwitchYard. For instance, as a web service via the SOAP binding or via a JMS queue.

7.4. DEPLOYING PROCESS DEFINITION

The BPEL process definitions are deployed as a part of the packaged SwitchYard application. When deploying the SwitchYard application, ensure that it contains one or more BPEL process.

All the BPEL processes are displayed under the list of deployed process definitions in the console.

7.5. MANUALLY UNDEPLOY A PROCESS

Procedure 7.1. task

Execute the following to safely undeploy a process:

1. Launch a web browser and navigate to http://localhost:8080/bpel-console.
2. Enter your user name and password.
3. Click on the **`Runtime`** tab.

4. Select the **`Deployments`** option. You will now be able to see the version information and current status (active or retired) of each process definition.
5. Check if there is any active process consisting of any unfinished instances, retire the process.
6. Undeploy the process only when there is no active process instance.

WARNING

Undeploy the BPEL process only when the containing SwitchYard application is undeployed.

7.6. ACTIVE PROCESS INSTANCE

An active process instance is one execution of a process definition.

When you select a process definition, BPEL console displays the list of active process instances for that particular process version and definition.

7.7. VIEW THE PROCESS VERSION

Procedure 7.2. task

To view the process version, execute the following:

1. Launch a web browser and navigate to http://localhost:8080/bpel-console.
2. Enter your user name and password.
3. Click on the **Managed Instances** tab to see which BPEL processes are currently deployed. The process version information for each of these processes is also visible.
4. Only one version of a process can be active at a time. When you open a process definition, the active version is automatically selected.

NOTE

BPEL console enables you to select a different version of a process.

7.8. ABOUT BUSINESS PROCESS ANALYTICS FORMAT

The Business Process Analytics Format (BPAF) is designed to provide you with information about the efficiency and effectiveness of your organizational processes. It is an XML-based interchange format for all the process audit events.

7.9. VIEW THE BPAF DATA

Procedure 7.3. task

You can view the Business Process Analytics Format (BPAF) data with the BPEL console. Execute the following:

1. Launch a web browser and navigate to http://localhost:8080/bpel-console.
2. Enter your user name and password.
3. Click on the **`Managed Instances`** tab to see which BPEL processes are currently deployed. The process version information for each of these processes is also visible.
4. Select a process definition to open it.
5. Use the **`Execution History`** to produce a chart. Here you can specify a particular period of time to review and choose whether or not to include failed and terminated instances in the chart.

7.10. NAVIGATING THE EXECUTION HISTORY CHART

Table 7.1. List of Shortcut Keys to Use When Navigating the Execution History Chart

Keyboard and Mouse Command	Result
Up Arrow	Zoom In
Down Arrow	Zoom Out
Left Arrow	Half-Page Left
Right Arrow	Half-Page Right
Page-Up	Page Left
Page-Down	Page Right
TAB	Next Focus
Shift-TAB	Previous Focus
HOME	Max Zoom Out
ENTER	Max Zoom In to Focus
Mouse Drag	Scroll Chart
Shift Mouse Drag	Drag Select/Zoom
Mouse Wheel Up/Z	Zoom In
Mouse Wheel Down/X	Zoom Out
Backspace/Back Button	Back

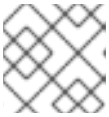
Keyboard and Mouse Command	Result
Right Mouse-Click	Context Menu
Left-Click	Set Focus
Double-Click	Maximise Zoom-In-to-Focus

7.11. CONFIGURING LOGGING FUNCTIONALITY

Procedure 7.4. task

Execute the following to activate the BPEL Web Console's Logging Functionality:

1. Open the **`deploy.xml`** file in a text editor.



NOTE

`deploy.xml` file exists with every BPEL deployment unit.

2. Configure the **`deploy.xml`** file as follows:

```
<deploy xmlns="schemas"
    xmlns:bpl="examples"
      xmlns:intf="examples/wsdl">
        <process name="bpl:HelloGoodbye">
           <active>true</active>
           <process-events generate="all"/>
           <provide partnerLink="helloGoodbyePartnerLink">
           <service name="intf:HelloGoodbyeService"
port="HelloGoodbyePort"/>
           </provide>
        </process>
</deploy>
```

NOTE

Add the ***`process-events`*** element to generate all the events. For more information, navigate to the Apache ODE events http://ode.apache.org/ode-execution-events link.

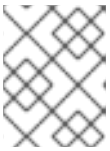
3. Save the file and exit.

7.12. VIEW THE INSTANCE DATA

Procedure 7.5. task

Execute the following to view instance data with the BPEL Console:

1. Launch a web browser and navigate to http://localhost:8080/bpel-console.
2. Enter your user name and password.
3. Under the **Managed Instances** tab, select the **Open Button** list to view the deployed process definitions. When you select any one of the process, it displays the list of process definitions and version of that particular process.
4. Select a process instance to open it. You can see its details in the **Execution Details** panel.
5. Click the **Instance Data** button. It displays all the details about the process.
6. Click the **Execution Path** button. It opens the related instance execution graph.



NOTE

The instance execution graph is a visual representation of a running instance of a process. It tells the user about the process instance's performance over time.

7. The **View** tab shows the instance execution graph, while the **Source** tab below it shows all of the "activity" events.

7.13. VIEW A HISTORY INSTANCE QUERY

Procedure 7.6. Task

1. Log into the BPEL Web Console.
2. Choose a process definition and a process status from the list box.

   You can also optionally choose to input the correlation key, the start time and the end time as search criteria.
3. Go to the History Instances List and double-click on a row. A window will pop up showing you all of the execution events that happened when that process ran.

7.14. ACTIVE PROCESS DEFINITION

When you deploy the first version of a BPEL process definition, it automatically becomes the active definition. If this definition is subsequently changed and redeployed, then that version is "retired", and a new version becomes active automatically.

7.15. RETIRED PROCESS DEFINITION

If the active process definition is changed and redeployed, the old version is "retired". The new version automatically becomes active. The only difference between an active version and a retired one is that a retired one can no longer create new process instances. However, if there are active process instances associated with the retired process version, then these will continue to run.

7.16. MANUALLY RETIRE AN ACTIVE PROCESS DEFINITION

Procedure 7.7. Task

1. Launch a web browser and go to http://localhost:8080/bpel-console.
2. Input your user name and password.
3. Click on the **Runtime** tab.
4. Select the **Deployments** option.

   You will now be able to see the version information and current status (active or retired) of each process definition.
5. Select the particular version of the process definition you want to retire and then press the **Retire** button.

NOTE

If you undeploy a process, its end-points will only deactivate if no previous versions of that process have ever existed.

7.17. ENABLE EXECUTION EVENTS

The BPEL Engine generates events to let you track all the activities happening within the engine and produces detailed information about all the process executions. These events exists in the database and can be examined.

Procedure 7.8. task

When you run a process, Apache ODE generates a set of events. To achieve the satisfactory performance, you can deactivate few events that are not in use. All the events may cause a non-negligeable overhead.

To enable these events, execute the following:

1. Open the **deploy.xml** file in a text editor.
2. Add ***<process-events generate="all"/>*** parameter in the **deploy.xml** file.

Result

Events are successfully enabled.

CHAPTER 8. RUNTIME GOVERNANCE CONSOLE

8.1. OVERLORD RUNTIME GOVERNANCE

Overlord RTGov uses a gadget server to display runtime governance information via a set of configurable gadgets. These can be accessed via the gadget server.

8.2. ACCESSING OVERLORD RUNTIME GOVERNANCE

Accessing the Overlord Runtime Governance console allows you to view governance related information. You can monitor events like response time, call tracing, situations that have been raised, and other details. Once you have configured a login for the server and signed in, you are free to peruse this information.

Procedure 8.1. Task

1. Start the server.
2. Access the following URL:

   ```
   http://localhost:8080/gadget-web
   ```

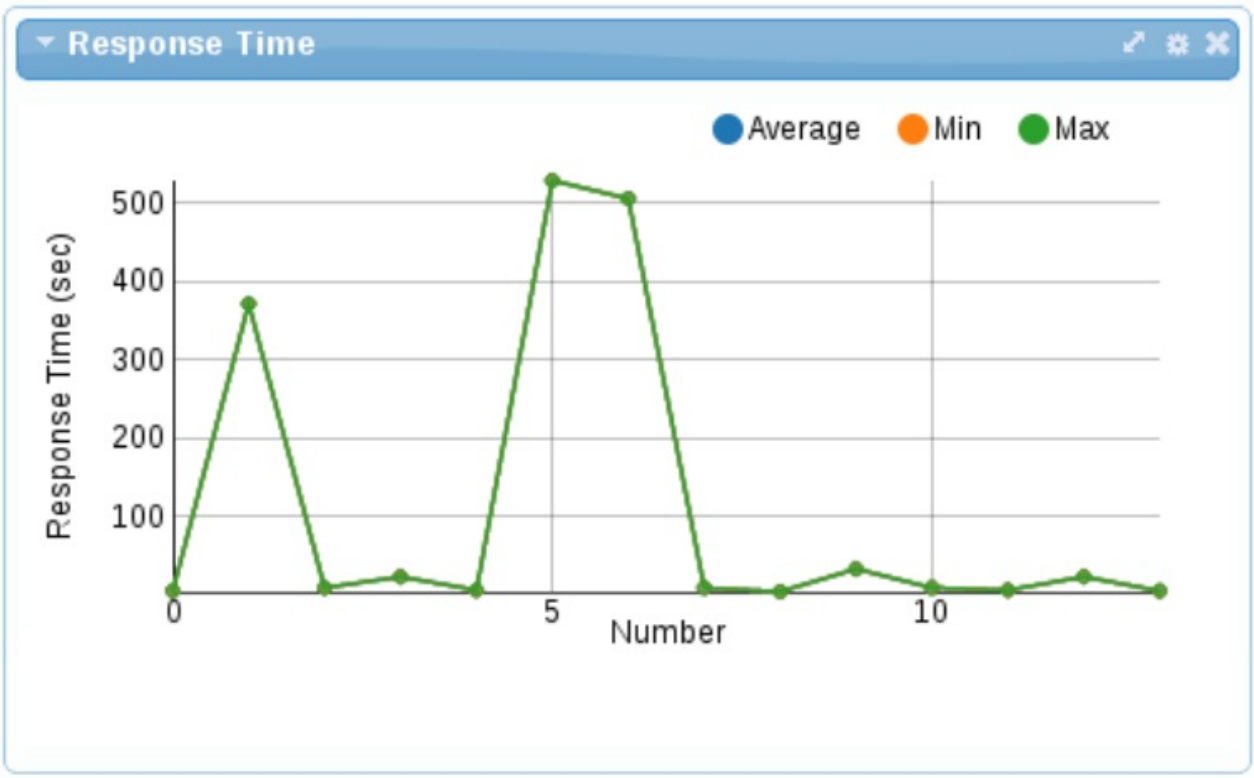
3. Enter a username and specify a password. A default user is provided called *fswAdmin*.
4. When the Overlord Runtime Governance UI is displayed the first time, you must enter a new page by pressing the *+* button.
5. To browse gadgets, click on the *Gadget Store* button.

8.3. GADGETS

Response Time

The Response Time gadget shows average, minimum and maximum summary metrics from the service operations invoked over a period of time. The gadget configuration can be used to select a particular service to display. It is also possible to customize the gadget further to display only the metrics from a particular operation on that service.

Figure 8.1. Response Time

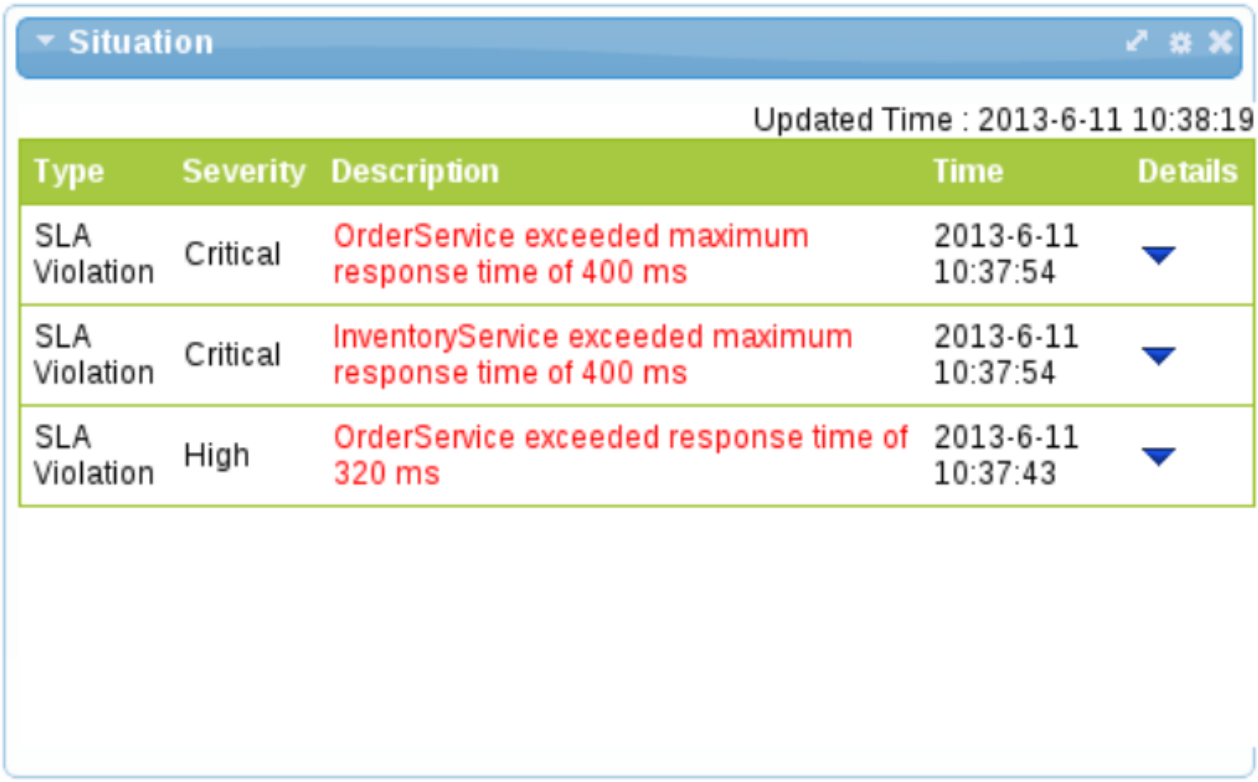


The X-axis represents the number of response time metrics captured for the particular services or operation. Note that it is not related to time. The collection of metrics in the underlying active collection displayed has a maximum size, hence it only discards the metrics when it reaches that maximum threshold.

Situations

Situations are created by event processors when they detect a "situation of interest". For example, an error or service level agreement violation. These situations are captured in a predefined active collection that is configured to only show the latest situations. They are automatically removed from the collection after a specified time period. This ensures that the gadget displaying the situations only shows the latest problems that are occurring. However, situations can also be reported via JMX for more long term management.

Figure 8.2. Situations

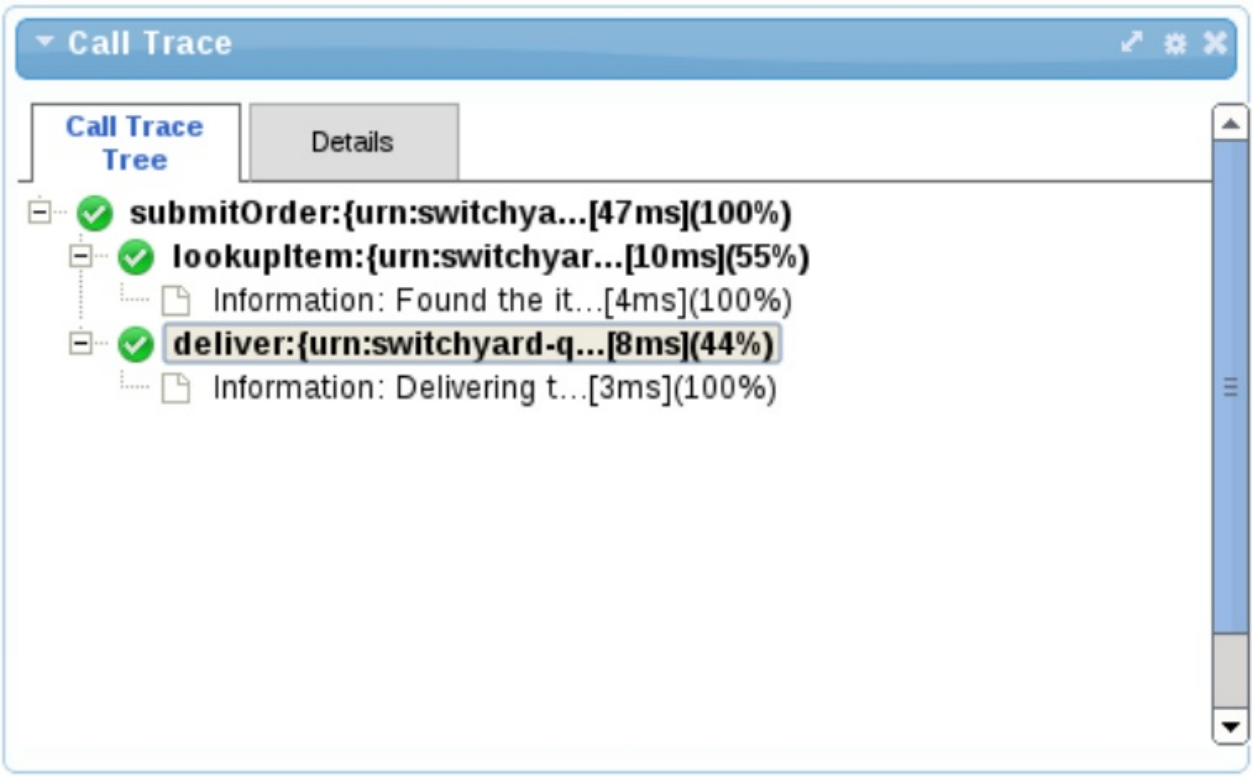


Call Trace

The Call Trace gadget provides the means to display the invocation history of a particular business transaction instance. The business transaction is identified by entering its globally unique identifier in the gadget's configuration.

The business transaction id is specific to the messages being monitored. For example, a purchasing business transaction may have a purchase order number that is carried with the various business messages. This information can be extracted from the message contents using an "information processor", and the extract value specified in the call trace gadget's configuration.

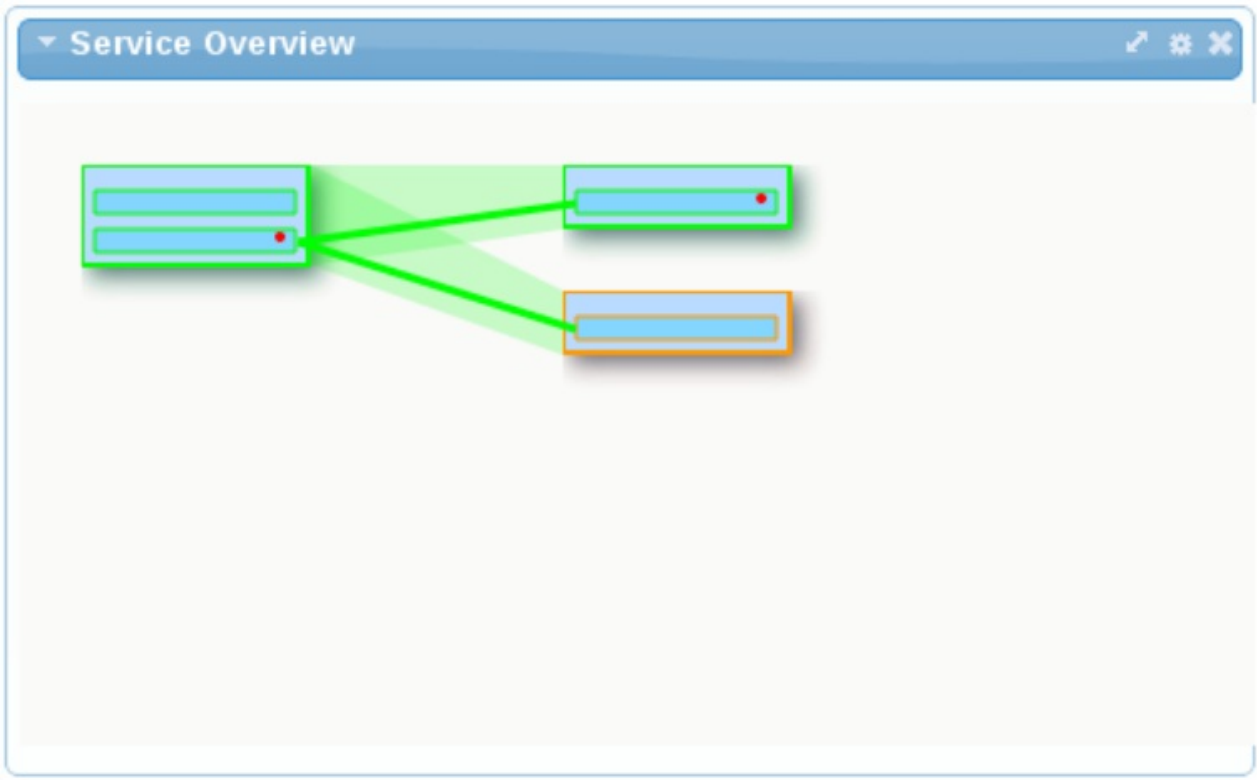
Figure 8.3. Call Trace



Service Overview

The Service Overview gadget provides a graphical representation of the dependencies (invocation/usage) between services. When displayed as part of a group of gadgets, the representation shows the status of each service and link as a color (green being normal). If the gadget is enlarged, further details are included, such as the service and operation names. Metrics are visible by hovering over an object. When a situation is reported, a red dot will be displayed on that component.

Figure 8.4. Service Overview



CHAPTER 9. DESIGN TIME GOVERNANCE CONSOLE

9.1. ACCESSING THE DTGOV DASHBOARD

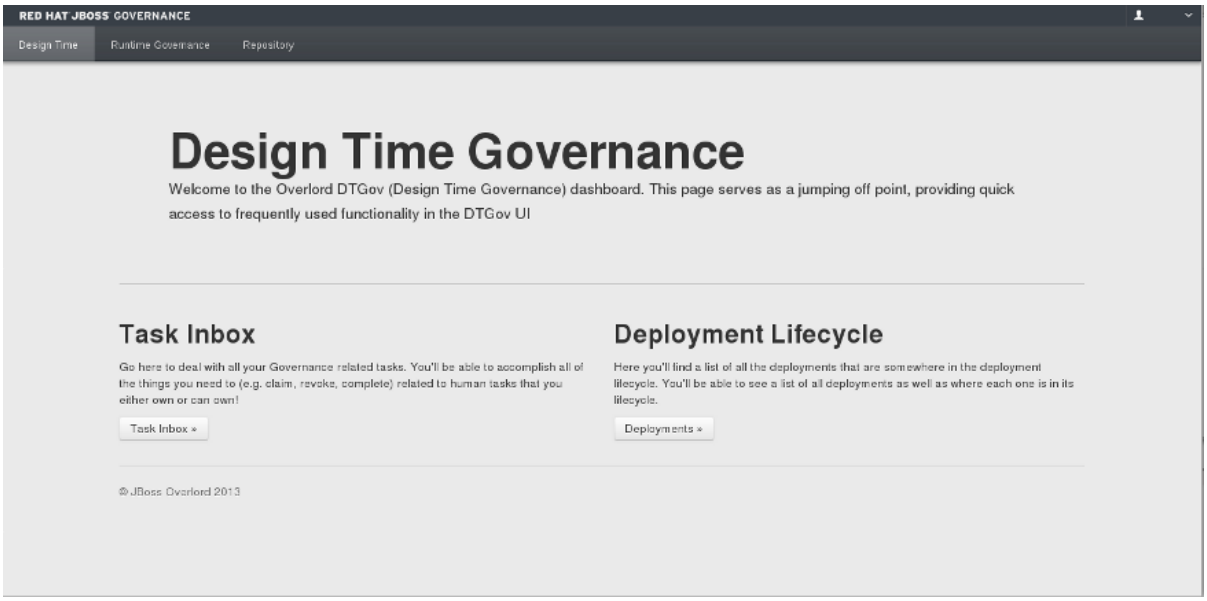
Accessing the DTGov Dashboard allows you to view the options available for governance. It includes information such as properties, deployments, and repositories.

Procedure 9.1. Task

1. Start the server.
2. Access the following URL:

```
http://localhost:8080/dtgov-ui
```

3. Enter your credentials and log in. The Dashboard will be displayed.



9.2. DTGOV DASHBOARD HOME SCREEN OPTIONS

Task Inbox

Clicking this option will take you to a list of all the governance-related tasks within JBoss Fuse. Here you can view task filters, task owners, priorities and the dates for which the task is set to be performed.

Deployment Lifecycle

Clicking this option will show you a list of all the deployments that exist within the current lifecycle. You can view the type, environment and dates initiated for any deployment. You can also perform a search to find a deployment.

Repository

By clicking this tab at the top of the screen, you are able to view artifacts in the S-RAMP Browser. Clicking the **`Artifacts`** button will show you the core porpertioes, classifiers, and custom properties for the artifacts in the current lifecycle.

CHAPTER 10. S-RAMP

10.1. S-RAMP

SOA Repository Artifact Model and Protocol (S-RAMP) is supported by a Technical Committee at OASIS. It is a specification of SOA repository that provides a common data model and protocol for interacting with a repository of SOA artifacts. For more information, see https://www.oasis-open.org/committees/s-ramp/charter.php.

S-RAMP supports interoperability between repository implementations by standardizing on a data model and API. The S-RAMP specification includes the following:

- A foundation document that describes the core concepts.
- An Atom based protocol binding document that describes an Atom based API.

An S-RAMP repository primarily stores artifacts. An artifact comprises of the following metadata:

- Core Properties
- Custom Properties
- Classifiers
- Relationships

10.2. S-RAMP CONSOLE

10.2.1. S-RAMP Dashboard

The S-RAMP Dashboard is where you can import artifacts and modify them. A list of DTGov data is presented in the Repository view. Additionally, the following core properties are presendted for configuration:

- Type
- Date Created
- Date Last Modified
- Created By
- Last Modified
- Origin (Any, Primary or Derived)

10.2.2. Accessing the S-RAMP Dashboard

You can view S-RAMP's graphical interface in your browser. The S-RAMP dashboard gives you a full visual representation of all the options available to you. Browse the menu options and buttons to see what's on offer.

Procedure 10.1. Task

1. In a browser, navigate to http://localhost:8080/s-ramp-ui .
2. Enter your administrator username and password when prompted.

Result

The S-RAMP dashboard will be displayed.

10.3. S-RAMP MANAGEMENT CLI

10.3.1. Connect to the S-RAMP Server

Connecting to the S-RAMP server gives you access to the S-RAMP dashboard and repository. This allows you to access artifacts and their metadata in one place. Once you have signed in, you can browse the repository and inspect your data.

1. Open a command terminal and navigate to ***EAP_HOME*/bin** directory.
2. Start the JBoss EAP server by entering following command:

   ```
   $ ./standalone.sh
   ```

3. Open another terminal and type **./s-ramp.sh**. Then type **connect** and press Tab key. The command will auto-complete to say **s-ramp:connect http://localhost:8080/s-ramp-server.**

   Enter username and password to connect via the S-RAMP shell. The username and password are defined in sramp.properties.

Result

When you press Tab key, the command will auto-complete to say **s-ramp:connect http://localhost:8080/s-ramp-server** and when you press the return key the cursor will go from red to green indicating successful connection to S-ramp server.

10.3.2. Browse the S-RAMP Repository

Running the commands in this task will allow you to access the S-RAMP repository and peruse its contents. You can view server output and metadata for artifacts. This lets you view additional details of indexed items.

1. Start the S-RAMP Management CLI.

   ```
   ./s-ramp.sh
   ```

2. Connect to S-RAMP server.

   ```
   s-ramp:connect http://localhost:8080/s-ramp-server
   ```

3. Upload an artifact to the repository.

   ```
   s-ramp:uploadArtifact /path/to/overlord.demo.SimpleReleaseProcess.bpmn
   ```

4. Browse the s-ramp repository.

   ```
   s-ramp:query /s-ramp
   ```

Here is an example of the output of this command:

```
Querying the S-RAMP repository:
    /s-ramp
Atom Feed (9 entries)
  Idx                    Type Name
  ---                    ---- ----
    1           ImageDocument user-properties.png
    2                Document overlord.demo.CheckDeployment-
taskform.flt
    3         BrmsPkgDocument SRAMPPackage.pkg
    4           ImageDocument overlord.demo.SimpleReleaseProcess-
image.png
    5           ImageDocument run-build-install.png
    6                Document overlord.demo.SimpleReleaseProcess-
taskform.flt
    7           ImageDocument audio-input-microphone-3.png
    8            BpmnDocument
overlord.demo.SimpleReleaseProcess.bpmn
    9            TextDocument HttpClientWorkDefinitions.wid
```

5. Get metadata. To obtain the metaData of overlord.demo.SimpleReleaseProcess.bpmn (which is number 8 in the list in the output example), run the following command:

   `getMetaData --feed 8`

   The output of the command is:

```
Meta Data for: 31b3acbc-cda8-4856-9e34-d3e645283035
--------------
  -- Core S-RAMP Info --
  Type: BpmnDocument
  Model: ext
  UUID: 31b3acbc-cda8-4856-9e34-d3e645283035
  Name: overlord.demo.SimpleReleaseProcess.bpmn
  Derived: false
  Created By: anonymous
  Created On: 2013-03-08T14:00:37.036-05:00
  Modified By: anonymous
  Modified On: 2013-03-18T14:58:46.328-04:00
s-ramp>
```

   You can also use the UUID of the artifact to obtain the metadata.

   `getMetaData --uuid 31b3acbc-cda8-4856-9e34-d3e645283035`

6. Set property. To set the core property, use command **`property set name _overlord.demo.SimpleReleaseProcess.bpmn_`**. To set custom property, use **`property set cliProp "this is an important document"`**

7. To add a comment to the artifact file, run command:

   `addComment "Comment added to the file"`

8. Once the changes are completed use **`updateMetaData`** to push the changes.

9. Run the s-ramp query with Property.

   query /s-ramp[@cliProp]

   ```
   1, 31b3acbc-cda8-4856-9e34-d3e645283035, BpmnDocument,
   _overlord.demo.SimpleReleaseProcess.bpmn_
   ```

10. Download the artifact.

    getContent --uuid 31b3acbc-cda8-4856-9e34-d3e645283035 --outputFile downloaded-overlord.demo.SimpleReleaseProcess.bpmn

CHAPTER 11. RUNTIME GOVERNANCE CONFIGURATION

11.1. ENABLING RUNTIME GOVERNANCE

You can enable or disable Runtime Governance through the collectionEnabled property in the **`$JBOSS_HOME/standalone/configuration/overlord-rtgov.properties`** file. By default, the value of the collectionEnabled is set to false.

This property will determine whether activity information is collected when the server is initially started.

11.2. CONFIGURING RUN-TIME GOVERNANCE

Introduction

SwitchYard uses Run-Time Governance to facilitate complete control over the shared services by following a policy based approach. SwitchYard classifies Run-Time Governance into two aspects:

- Policy definition and enforcement
- Collection and exposure of run-time metrics for services and service references

Run-Time Governance configurations allow you to add or modify classes to manage Run-Time Governance and also define usernames and passwords.

11.3. OVERLORD RUN-TIME GOVERNANCE CONFIGURATION

Procedure 11.1. Task

1. To view the Overlord Run-Time Governance properties, access the properties file located in **`$JBOSS_HOME/standalone/configuration/overlord-rtgov.properties`**.
2. Add and modify classes to manage governance.

11.4. COMMON PROPERTIES

Table 11.1. Common Properties

Name	Description
collectionEnabled	This property will determine whether activity information is collected when the server is initially started. This value can be changed at runtime using the ActivityCollector MBean.
ActivityServerLogger.maxThreads	This property will determine the maximum number of threads used by the activity collector to report all the activity units to the server.
ActivityServerLogger.durationBetweenFailureReports	This property controls the duration (in milliseconds) between a failure being reported, if the client is having trouble contacting the activity server.

Name	Description
ActivityServerLogger.activityListQueueSize	This property defines the queue size for pending activity lists, that are awaiting being reported to the activity server.
ActivityServerLogger.freeActivityListQueueSize	This property defines the queue size to manage free activity lists that can be reused.
BatchedActivityUnitLogger.maxUnitCount, BatchedActivityUnitLogger.maxTimeInterval	Activity unit batch logging properties.
infinispan.container	JNDI name for default cluster to use when not specified explicitly.

11.5. SERVER PROPERTIES

Table 11.2. Server Properties

Name	Description
MVELSeverityAnalyzer.scriptLocation	Optional location of a MVEL script used to determine severity levels for nodes and links within the service overview diagram.

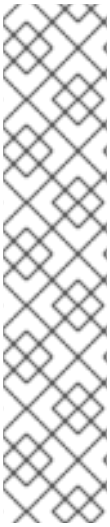
11.6. CLIENT PROPERTIES

Table 11.3. Client Properties

Name	Description
RESTActivityServer.serverURL	This is the URL of the activity server collecting the activity events.
RESTActivityServer.serverUsername	The username used to access the REST service.
RESTActivityServer.serverPassword	The password used to access the REST service.

11.7. DATABASE

The installer sets up the datasources for Runtime Governance, jBPM, SRAMP, and BPEL (Riftsaw). The datasources can be configured in **`$JBOSS_HOME/standalone/configuration/standalone.xml`**.



NOTE

By default, the datasource validation function is disabled. To enable this function, add the validate-on-match property as follows:

```
<validation>
<validate-on-match>true</validate-on-match>
<valid-connection-checker
class-name="org.jboss.jca.adapters.jdbc.extensions.postgres.Pos
tgreSQLValidConnectionChecker"/>
<exception-sorter
class-name="org.jboss.jca.adapters.jdbc.extensions.postgres.Pos
tgreSQLExceptionSorter"/>
</validation>
```

11.8. CACHING

The EPN and Active Collection mechanisms both have the ability to make use of caching provided by Infinispan. When running the Red Hat JBoss Fuse in clustered mode (that is, with **`standalone-full-ha.xml`**), it provides a default clustered cache container, which is referenced in the **`infinispan.container`** property in the **`overlord-rtgov.properties`** file.

NOTE

To make sure the individual named caches are clustered correctly, it is necessary to add an entry for each cache into the **`standalone-full-ha.xml`** file.

For more information on Infinispan configuration, refer JBoss Enterprise Application Platform 6.1 Development Guide.

11.9. CACHING EXAMPLE

The following cache entry for the "Principals" cache has been defined in the *standalone-full-ha.xml* file:

```
<cache-container name="cluster" aliases="ha-partition" default-
cache="default">
    <transport lock-timeout="60000"/>
    <replicated-cache name="default" mode="SYNC" batching="true">
        <locking isolation="REPEATABLE_READ"/>
    </replicated-cache>

    <!-- Configuration for Runtime Governance caches -->

    <replicated-cache name="Principals" mode="SYNC">
        <locking isolation="REPEATABLE_READ"/>
        <transaction mode="FULL_XA" locking="PESSIMISTIC"/>
    </replicated-cache>
</cache-container>
```

CHAPTER 12. DESIGN-TIME GOVERNANCE CONFIGURATION AND WORKFLOWS

12.1. OVERVIEW

Design-Time Governance consists of the following configurations that can be modified to suit a particular deployment and business.

- Design-Time Governance Back-End Configuration - includes **`dtgov.properties`** file
- Design-Time Governance User Interface Configuration - includes **`dtgov-ui.properties`** file

This section describes the two configuration files so that the user can configure Design-Time Governance for their particular deployment environment and organization's unique business processes.

12.2. DESIGN-TIME GOVERNANCE BACK-END CONFIGURATION

You can modify the configuration of the back-end system, after you make appropriate changes to an external configuration file located under the configuration directory of the JBoss EAP. However, by default, it is located under:

```
jboss-eap/standalone/configuration/dtgov.properties
```

If the configuration file does not exists, you can create the file and modify. The Design-Time Governance will automatically identify this file during the server startup. You can set the location of the configuration file by accessing the following Java system property:

```
governance.file.name
```

12.3. DESIGN-TIME GOVERNANCE USER INTERFACE (UI) CONFIGURATION

You can configure the Design-Time Governance User Interface for a specific deployment and business environment. To modify the User Interface configuration, make the appropriate changes to an external configuration file located under the configuration directory of the **JBoss EAP**. However, by default, the JBoss EAP configuration file is located under:

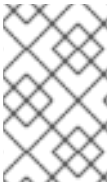
```
jboss-eap/standalone/configuration/dtgov-ui.properties
```

You can set the location of the configuration file by accessing the following Java system property:

```
 dtgov-ui.config.file.name
```

12.4. DESIGN-TIME GOVERNANCE CONFIGURATION PROPERTIES

A number of configuration properties drive the integration between Design-Time Governance and S-RAMP.



NOTE

The Design-Time Governance back-end and the Design-Time Governance User Interface each have their own separate configuration. This is because the back-end and User Interface are separate applications that can be independently deployed.

Design-Time Governance Back-End Configuration Properties

```
# # S-RAMP Connection details
sramp.repo.url
sramp.repo.auth.provider
sramp.repo.user
sramp.repo.password
sramp.repo.validating
sramp.repo.auth.saml.issuer
sramp.repo.auth.saml.service

# Location of the DTGov WAR
governance.url
# Frequency with which to poll S-RAMP for query matches
governance.query.interval
# Location in JNDI of the email service
governance.jndi.email.reference
# "From" information to use when sending email (domain and address)
governance.email.domain
governance.email.from

# RHQ connection info
rhq.rest.user
rhq.rest.password
rhq.base.url

# BPM connection info
governance.bpm.user
governance.bpm.password
governance.bpm.url

# JAAS user used to invoke DTGov provided services
governance.user
governance.password

# Deployment targets configured for the DTGov deployment service
governance.targets

# Mapping of S-RAMP query to governance workflow
governance.queries

# Location of the DTGov UI
dtgov.ui.url

# S-RAMP
s-ramp-wagon
dtgov.s-ramp-wagon.snapshots
dtgov.s-ramp-wagon.releases

# DTGov Workflow maven info
```

```
dtgov.workflows.group
dtgov.workflows.name
dtgov.workflows.version
dtgov.workflows.package
```

Design-Time Governance User Interface Configuration Properties

```
# # S-RAMP API connection endpoint
dtgov-ui.s-ramp.atom-api.endpoint
# Whether to validate the S-RAMP connection
dtgov-ui.s-ramp.atom-api.validating
# What kind of authentication to use (class name)
dtgov-ui.s-ramp.atom-api.authentication.provider
# Only used when the provider is basic auth
dtgov-ui.s-ramp.atom-api.authentication.basic.username
dtgov-ui.s-ramp.atom-api.authentication.basic.password
# Only used when the provider is SAML bearer token auth
dtgov-ui.s-ramp.atom-api.authentication.saml.issuer
dtgov-ui.s-ramp.atom-api.authentication.saml.service
dtgov-ui.s-ramp.atom-api.authentication.saml.sign-assertions
dtgov-ui.s-ramp.atom-api.authentication.saml.keystore
dtgov-ui.s-ramp.atom-api.authentication.saml.keystore-password
dtgov-ui.s-ramp.atom-api.authentication.saml.key-alias
dtgov-ui.s-ramp.atom-api.authentication.saml.key-password

# Task API connection endpoint
dtgov-ui.task-api.endpoint
# Implementation of a task client
dtgov-ui.task-client.class
# Authentication to use when invoking the task API
dtgov-ui.task-api.authentication.provider
# Only used when using basic auth
dtgov-ui.task-api.authentication.basic.username
dtgov-ui.task-api.authentication.basic.password
# Only used when using saml bearer token auth
dtgov-ui.task-api.authentication.saml.issuer
dtgov-ui.task-api.authentication.saml.service
dtgov-ui.task-api.authentication.saml.sign-assertions
dtgov-ui.task-api.authentication.saml.keystore
dtgov-ui.task-api.authentication.saml.keystore-password
dtgov-ui.task-api.authentication.saml.key-alias
dtgov-ui.task-api.authentication.saml.key-password

# Deployment lifecycle base classifier
dtgov-ui.deployment-lifecycle.classifiers.base
dtgov-ui.deployment-lifecycle.classifiers.initial
# Classifier to use when querying for all deployments
dtgov-ui.deployment-lifecycle.classifiers.all
dtgov-ui.deployment-lifecycle.classifiers.in-progress
# This next one is a prefix for any property that will indicate a possible
classifier stage that
# should be displayed in the UI.  In the dtgov ui configuration file,
multiple properties would
# be specified that begin with this prefix and have a value of the format
{label}:{classifier}
dtgov-ui.deployment-lifecycle.classifiers.stage
```

```
# And another one that is a prefix for any property that will indicate a
possible deployment type
# that should be displayed in the UI.  In the dtgov ui configuration file,
multiple properties would
# be specified that begin with this prefix and have a value of the format
{label}:{type}
dtgov-ui.deployment-lifecycle.types

# S-RAMP UI integration properties
dtgov-ui.s-ramp-browser.url-base
```

12.5. DESIGN-TIME GOVERNANCE CONFIGURATION EXAMPLES

Back-End Configuration Example

Here is an example of the back-end configuration:

```
sramp.repo.url=http://localhost:8080/s-ramp-server/
sramp.repo.auth.provider=org.overlord.sramp.governance.auth.BasicAuthentic
ationProvider
sramp.repo.user=dtgov
sramp.repo.password=DTG_PASSWORD
sramp.repo.validating=true
```

The above configuration uses BASIC authentication when connecting to the S-RAMP repository. It connects to S-RAMP on localhost (port 8080).

User Interface Configuration Example

Here is an example of the User Interface configuration:

```
dtgov-ui.s-ramp.atom-api.endpoint=http://localhost:8080/s-ramp-server
dtgov-ui.s-ramp.atom-
api.authentication.provider=org.overlord.dtgov.ui.server.services.sramp.SA
MLBearerTokenAuthenticationProvider
dtgov-ui.s-ramp.atom-api.authentication.saml.issuer=/dtgov-ui
dtgov-ui.s-ramp.atom-api.authentication.saml.service=/s-ramp-server
dtgov-ui.s-ramp.atom-api.validating=true
```

The above configuration connects to S-RAMP on localhost (port 8080) and uses SAML bearer token authentication.

12.6. GOVERNANCE WORKFLOWS

One of the most important features of Design-Time Governance is the ability to trigger Governance Workflows based on changes detected in the S-RAMP repository. Once you have installed S-RAMP and Design-Time Governance, and configured the database for S-RAMP, you can:

- Create a new workflow, or adapt a sample workflow.
- Deploy the workflow to S-RAMP.
- Map S-RAMP query to BPMN2 process name. That is, configure a workflow to execute when repository content changes.

12.7. INSTALLING SAMPLE WORKFLOW

This section provides installation steps of a sample workflow called "SimpleReleaseProcess".

1. Navigate to jboss-eap-6.1/dtgov-data and create a local **`settings.xml`** containing the user credentials for the S-RAMP repository. This file must also contain the Overlord admin password that you configured during installation. Here is an example:

```
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
                      http://maven.apache.org/xsd/settings-
1.0.0.xsd">
  <servers>
    <server>
      <id>local-sramp-repo</id>
      <username>admin</username>
      <password>cust0mpwd</password>
    </server>
    <server>
      <id>local-sramp-repo-snapshots</id>
      <username>admin</username>
      <password>cust0mpwd</password>
    </server>
  </servers>
</settings>
```

2. Ensure that the server is up and running, and deploy the "SimpleReleaseProcess" sample workflow to the server by running the following command:

```
mvn -s local-settings.xml deploy
```

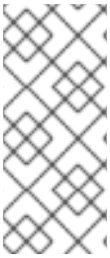
CHAPTER 13. MANAGING USER ACCOUNTS

13.1. ADDING A NEW USER FOR THE MANAGEMENT INTERFACES

Overview

The management interfaces in JBoss Fuse are secured by default as there are no user accounts initially available, unless you have installed the platform using the graphical installer. This is a security precaution to prevent security breaches from remote systems due to simple configuration errors. Local non-HTTP access is protected by a SASL mechanism, with a negotiation happening between the client and server each time the client connects for the first time from the localhost.

This task describes how to create the initial administrative user, which can use the web-based Management Console and remote instances of the Management Command Line Interface.



NOTE

HTTP communication with JBoss EAP 6 is considered to be remote access, even if the traffic originates on the localhost. Therefore, you must create at least one user in order to be able to use the management console. If you attempt to access the management console before adding a user, you will receive an error because it does not even deploy until the user is added.

Procedure 13.1. Create the Initial Administrative User for the Remote Management Interfaces

1. **Invoke the `add-user.sh` or `add-user.bat` script.**
   Change to the ***`EAP_HOME`*`/bin/`** directory. Invoke the appropriate script for your operating system.

   Red Hat Enterprise Linux

   ```
   [user@host bin]$ ./add-user.sh
   ```

2. **Choose to add a Management user.**
   Hit **`ENTER`** to select the default option **`a`** to add a Management user. This user is added to the **`ManagementRealm`** and is authorized to perform management operations using the web-based Management Console or command-line based Management CLI. The other choice, **`b`**, adds a user to the **`ApplicationRealm`**, and provides no particular permissions. That realm is provided for use with applications.

3. **Enter the desired username and password.**
   When prompted, enter the username and password. You will be prompted to confirm the password.

4. **Review the information and confirm.**
   You are prompted to confirm the information. If you are satisfied, type **`yes`**.

5. **Choose whether the user represents a remote JBoss EAP 6 server instance.**
   Besides administrators, the other type of user which occasionally needs to be added to JBoss EAP 6 in the **`ManagementRealm`** is a user representing another instance of JBoss EAP 6, which must be able to authenticate to join a cluster as a member. The next prompt allows you to

designate your added user for this purpose. If you select **yes**, you will be given a hashed **secret** value, representing the user's password, which would need to be added to a different configuration file. For the purposes of this task, answer **no** to this question.

6. **Enter additional users.**
   You can enter additional users if desired by repeating the procedure. You can also add them at any time on a running system. Instead of choosing the default security realm, you can add users to other realms to fine-tune their authorizations.

7. **Create users non-interactively.**
   You can create users non-interactively, by passing in each parameter at the command line. This approach is not recommended on shared systems, because the passwords will be visible in log and history files. The syntax for the command, using the management realm, is:

   ```
   [user@host bin]$ ./add-user.sh username password
   ```

   To use the application realm, use the **-a** parameter.

   ```
   [user@host bin]$ ./add-user.sh -a username password
   ```

8. You can suppress the normal output of the add-user script by passing the **--silent** parameter. This applies only if the minimum parameters **username** and **password** have been specified. Error messages will still be shown.

Result

Any users you add are activated within the security realms you have specified. Users active within the **ManagementRealm** realm are able to manage JBoss EAP 6 from remote systems.

13.2. ADD-USER COMMAND ARGUMENTS

The following table describes the arguments available for the **add-user.sh** or **add-user.bat** command.

Table 13.1. Add-user Command Arguments

Command Line Argument	Argument Value	Description
-a	N/A	This argument specifies to create a user in the application realm. If omitted, the default is to create a user in the management realm.
-dc	*DOMAIN_CONFIGURATION_DIRECTORY*	This argument specifies the domain configuration directory that will contain the properties files. If it is omitted, the default directory is ***EAP_HOME*/domain/configuration/**.
-sc	*SERVER_CONFIGURATION_DIRECTORY*	This argument specifies an alternate standalone server configuration directory that will contain the properties files. If it is omitted, the default directory is ***EAP_HOME*/standalone/configuration/**.

Command Line Argument	Argument Value	Description
-up --user-properties	*USER_PROPERTIES_FILE*	This argument specifies the name of the alternate user properties file. It can an absolute path or it can be a file name used in conjunction with the **-sc** or **-dc** argument that specifies the alternate configuration directory.
-g --group	*GROUP_LIST*	A comma-separated list of groups to assign to this user.
-gp --group-properties	*GROUP_PROPERTIES_FILE*	This argument specifies the name of the alternate group properties file. It can an absolute path or it can be a file name used in conjunction with the **-sc** or **-dc** argument that specifies the alternate configuration directory.
-p --password	*PASSWORD*	The password of the user. The password must satisfy the following requirements: • It can not be the same as the user name. • It must contain at least 8 characters. • It must contain at least one alphanumeric character. • It must contain at least one digit. • It must contain at least one non-alphanumeric symbol
-u --user	*USER_NAME*	The name of the user.
-r --realm	*REALM_NAME*	The name of the realm used to secure the management interfaces. If omitted, the default is "ManagementRealm".
-s --silent	N/A	Run the add-user script with no output to the console.
-h --help	N/A	Display usage information for the add-user script.

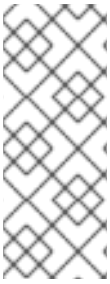
13.3. ALTERNATIVE PROPERTIES FILES FOR USER MANAGEMENT INFORMATION

Overview

By default, user and role information created using the **add-user.sh** script is stored in properties files located in the server configuration directory. The server configuration information is stored in the ***EAP_HOME*/standalone/configuration/** directory and the domain configuration information is stored in the ***EAP_HOME*/domain/configuration/** directory. This topic describes how to override the default file names and locations.

Procedure 13.2. Alternative Properties Files

- To specify an alternative directory for the server configuration, use the **-sc** argument. This argument specifies an alternative directory that will contain the server configuration properties files.
- To specify an alternative directory for the domain configuration, use the **-dc** argument. This argument specifies an alternative directory that will contain the domain configuration properties files.
- To specify an alternative user configuration properties file, use the **-up** or **--user-properties** argument. It can an absolute path or it can be a file name used in conjunction with the **-sc** or **-dc** argument that specifies the alternative configuration directory.
- To specify an alternative group configuration properties file, use the **-gp** or **--group-properties** argument. It can an absolute path or it can be a file name used in conjunction with the **-sc** or **-dc** argument that specifies the alternative configuration directory.



NOTE

The **add-user** command is intended to operate on existing properties files. Any alternative properties files specified in command line arguments must exist or you will see the following error:

```
JBAS015234: No appusers.properties files found
```

13.4. GOVERNANCE

13.4.1. S-RAMP User Management

By default S-RAMP uses the standard EAP Application Realm configuration as its authentication source. This means that adding users is a simple matter of using the existing EAP add-user script. If you are running on Windows you can use the **add-user.bat** script. Otherwise run the **add-user.sh** script. Both of these scripts can be found in EAP's bin directory.

Here is an example of how to add an S-RAMP user using the add-user.sh script:

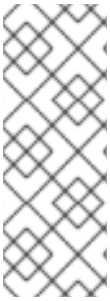
```
[user@host jboss-eap-6.1]$ pwd
/home/user/FSW6/jboss-eap-6.1
[user@host jboss-eap-6.1]$ ./bin/add-user.sh

What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): b
```

```
Enter the details of the new user to add.
Realm (ApplicationRealm) : ApplicationRealm
Username : fitzuser
Password : P4SSW0RD!
Re-enter Password : P4SSW0RD!
What roles do you want this user to belong to? (Please enter a comma
separated list, or leave blank for none)[  ]: overlorduser,admin.sramp
About to add user 'fitzuser' for realm 'ApplicationRealm'
Is this correct yes/no? yes
Added user 'fitzuser' to file '/home/user/FSW6/jboss-eap-
6.1/standalone/configuration/application-users.properties'
Added user 'fitzuser' to file '/home/user/FSW6/jboss-eap-
6.1/domain/configuration/application-users.properties'
Added user 'fitzuser' with roles overlorduser,admin.sramp to file
'/home/user/FSW6/jboss-eap-6.1/standalone/configuration/application-
roles.properties'
Added user 'fitzuser' with roles overlorduser,admin.sramp to file
'/home/user/FSW6/jboss-eap-6.1/domain/configuration/application-
roles.properties'
Is this new user going to be used for one AS process to connect to another
AS process?
e.g. for a slave host controller connecting to the master or for a
Remoting connection for server to server EJB calls.
yes/no? no
```

Required Roles

There are several roles that the user must have in order to interact with the S-RAMP repository. These roles are as follows:

- overlorduser : users must have this role in order to access the S-RAMP user interface (browser)
- admin.sramp : users must have this role in order to access the S-RAMP repository (both read and write)



NOTE

If you change the S-RAMP repository name in the **`standalone.xml`** file and set the new repository name in **`standalone/configuration/sramp.properties`** (under sramp.config.jcr.repository.jndi-path), make sure you modify the user's roles. If the role that grants users access to the ModeShape repository is *admin.sramp*, where the ModeShape role is *admin* on repository named *sramp*, ensure you change this value to *admin.<new repository name>* in the **`application-roles.properties`** file.

13.4.2. Design-Time Governance User Management

By default Design-Time Governance uses the standard EAP Application Realm configuration as its authentication source. This means that adding users is a simple matter of using the existing EAP add-user script. If you are running on Windows you can use the **`add-user.bat`** script. Otherwise run the **`add-user.sh`** script. Both of these scripts can be found in EAP's bin directory.

This example creates a user who can view and complete Development and Test environment human tasks. Any other human tasks is not visible.

Required Roles

There are several roles that the user must have in order to interact with Design-Time Governance. These roles are as follows:

- overlorduser : users must have this role in order to access the DTGov user interface
- admin.sramp : users must have this role in order to access the S-RAMP repository (both read and write)
- dev : users with this role will be able to view and complete Dev environment and developer human tasks
- test : users with this role will be able to view and complete Test environment human tasks
- stage : users with this role will be able to view and complete Staging environment human tasks
- prod : users with this role will be able to view and complete Production environment human tasks
- ba : users with this role will be able to view and complete business analyst human tasks
- arch : users with this role will be able to view and complete architect human tasks

CHAPTER 14. CONFIGURING RED HAT JBOSS FUSE TO RUN AS A BACKGROUND SERVICE

14.1. INTRODUCTION

Headless mode is a mode that allows you to run **JBoss Fuse** on a server without a monitor attached.

The main advantage of running JBoss Fuse in this particular mode is that the product will launch automatically when the server reboots, saving your valuable time. There is no need to intervene and launch JBoss Fuse manually.

To achieve this, **JBoss Fuse** runs as a background daemon (Linux) or service (Microsoft Windows).

14.2. RUNNING JBOSS FUSE AS A SERVICE ON A HEADLESS SERVER

Prerequisites

- Install JBoss EAP 6 or later
- Administrator privileges on the server are required.

Summary

Use the following procedure to install JBoss Fuse as a service on Red Hat Enterprise Linux when the installation has been done using the graphical installer.

Procedure 14.1. Setup the Service

1. **Locate the start-up script and configuration file**
   The start-up script and an associated configuration file are located in the ***EAP_HOME*/bin/init.d/** directory. Open the configuration file **jboss-as.conf** to edit it.

2. **Customize the start-up options in the `jboss-as.conf` file**
   There are several options within the **jboss-as.conf** file. At the minimum, specify the correct values for **JBOSS_HOME** and the **JBOSS_USER** variables. If these variables are absent, add them.

3. **Copy files into system directories**

   a. Copy the modified configuration file to the **/etc/jboss-as** directory.

   ```
   [user@host init.d]$ sudo mkdir /etc/jboss-as
   ```

   ```
   [user@host init.d]$ sudo cp jboss-as.conf /etc/jboss-as/
   ```

   b. Copy the start-up script to the **/etc/init.d** directory.

   ```
   [user@host init.d]$ sudo cp jboss-as-standalone.sh /etc/init.d
   ```

4. **Add the start-up script as a service.**

Add the new **jboss-as-standalone.sh** service to list of automatically started services, using the **chkconfig** service management command.

```
[user@host init.d]$ sudo chkconfig --add jboss-as-standalone.sh
```

5. **Start the service.**
   Test that the service has been installed correctly by using the standard syntax for starting Red Hat Enterprise Linux services.

   ```
   [user@host bin]$ sudo service jboss-as-standalone.sh start
   ```

   If everything has gone correctly, you should get a green [OK]. If you get an error, check the error logs and make sure your paths are correct in the configuration file.

6. **Make the service start automatically when you restart your server.**
   To add the service to the list of services which start automatically when your server restarts, issue the following command.

   ```
   [user@host init.d]$ sudo chkconfig jboss-as-standalone.sh on
   ```

Result

JBoss Fuse starts automatically when Red Hat Enterprise Linux reaches its default run-level, and stops automatically when the operating system goes through its shutdown routine.

CHAPTER 15. CONFIGURING RED HAT JBOSS FUSE TO RUN IN A CLUSTERED ENVIRONMENT

15.1. ABOUT CLUSTER SERVICE REGISTRY IN SWITCHYARD

The purpose of the clustered service registry is to store information about the services that are used in a cluster of SwitchYard instances.

Following are the details that are recorded in the registry for each published service:

- *Service Name*: It stores the name of a service.
- *Domain Name*: It stores the name of a domain.
- *Endpoint Address*: It stores the physical access point for the service.
- *Node Name*: It stores the cluster node on which the service is deployed.
- *Service Contract*: It stores the abstract invocation contract for the service which consists of the message exchange pattern and message types.

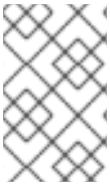
15.2. SETTING UP SWITCHYARD IN A CLUSTERED ENVIRONMENT

The two fundamental building blocks that are used for setting up SwitchYard in a clustered environment are:

- *Shared Runtime Registry*: A shared, distributed runtime registry that enable individual instances to publish and query service endpoint details.
- *Remote Communication Channels*: An internal communication protocol that enables a service client to invoke a service, hosted in a remote instance.

The runtime registry supports a replicated Infinispan cache. Each instance in a cluster points to the same replicated cache. When a node joins a cluster, it immediately has access to all remote service endpoints published in the registry. If a node leaves the cluster due to failure or shutdown, all service endpoint registrations are immediately removed for that node. The registry is not persisted, so manual clean-up and maintenance is not required.

The communication channel is a private intra-cluster protocol used by instances to invoke a remote service. At present, the channel is based on HTTP.



NOTE

The SwitchYard registry is a runtime registry and not a publication registry. The state of services within the registry is tied directly to the current state of services deployed within a cluster.

15.3. CREATING A CLUSTER OF SWITCHYARD INSTANCES

To create a cluster of SwitchYard instances, start two or more instances with a shared Infinispan cache.

```
# start instance 1
node1> bin/standalone.sh -c standalone-ha.xml -Djboss.node.name=node1
# start instance 2
```

```
node> bin/standalone.sh -c standalone-ha.xml -Djboss.node.name=node2 -
Djboss.socket.binding.port-offset=1000
```

When the instances are up, you can deploy applications independently to each instance. A homogeneous cluster (a cluster of identical machines) consists of identical applications deployed on each node. However, a heterogeneous cluster consists of different applications and services deployed on each instance. To verify the cluster setup, deploy a consumer application to one instance and a provider application to another.

15.4. ENABLING CLUSTERING IN YOUR SWITCHYARD APPLICATION

To configure a cluster:

- Start two or more instances on different nodes.
- Indicate the services that are clustered in the **`switchyard.xml`** file.

NOTE

By default, SwitchYard uses the default cache in the cluster cache container which comes pre-defined in the **`standalone-ha.xml`** file.

You can control the services in the application that can be published in the cluster's runtime registry and the references that can be resolved by clustered services.

To enable a service to be published in the cluster's runtime registry, promote the service in the application and add a **`<binding.sca>`** with clustering enabled to it.

```
<sca:service name="Goodbye" promote="GoodbyeBean/Goodbye">
  <sca:interface.java interface="com.example.Goodbye"/>
  <sca:binding.sca sy:clustered="true"/>
</sca:service>
```

To invoke a service in a cluster, promote the reference and add an SCA binding with clustering enabled.

```
<sca:reference name="Goodbye" multiplicity="0..1"
promote="GreetingBean/Goodbye">
  <sca:interface.java interface="com.example.Goodbye"/>
  <sca:binding.sca sy:clustered="true"/>
</sca:reference>
```

15.5. GOVERNANCE AND S-RAMP IN A CLUSTERED ENVIRONMENT

In a clustered environment, it is recommended to have a dedicated runtime governance server separate from the execution servers. In this case, all execution servers connect to the governance server via the runtime governance client.

Depending upon the load of the runtime governance server, it may be necessary to set up a cluster of governance servers as well. If you have a cluster of runtime governance servers, then they only need to connect to the same database instance. However, it is also possible to configure each runtime governance server to have its own S-RAMP database. The JBoss Fuse installer configures all components to use the same database by default so that runtime governance and S-RAMP share the same database instance. However, this is not mandatory.

Procedure 15.1. Configuring Governance and S-RAMP in a Clustered Environment

Perform the following steps on all servers in the cluster. Please note that this is just one of several possible configurations of governance components in a clustered environment.

1. Add the **<distributable/>** entry inside the **<web-app>** element of the following **web.xml** files:
   - **standalone/deployments/s-ramp-server.war/WEB-INF/web.xml**
   - **standalone/deployments/s-ramp-ui.war/WEB-INF/web.xml**
   - **standalone/deployments/dtgov.war/WEB-INF/web.xml**
   - **standalone/deployments/dtgov-ui.war/WEB-INF/web.xml**
   - **standalone/deployments/overlord-commons-idp.war/WEB-INF/web.xml**
   - **standalone/deployments/overlord-rtgov/gadget-web.war/WEB-INF/web.xml**
   - **standalone/deployments/overlord-rtgov/gadgets.war/WEB-INF/web.xml**
   - **standalone/deployments/overlord-rtgov/overlord-rtgov.war/WEB-INF/web.xml**
2. Add the **<replication-config>** entry:

   ```
   <replication-config>
     <cache-name>web/sso</cache-name>
   </replication-config>
   ```

   inside the **<jboss-web>** element of the following **jboss-web.xml** files:
   - **standalone/deployments/s-ramp-server.war/WEB-INF/jboss-web.xml**
   - **standalone/deployments/s-ramp-ui.war/WEB-INF/jboss-web.xml**
   - **standalone/deployments/dtgov.war/WEB-INF/jboss-web.xml**
   - **standalone/deployments/dtgov-ui.war/WEB-INF/jboss-web.xml**
   - **standalone/deployments/overlord-commons-idp.war/WEB-INF/jboss-web.xml**
   - **standalone/deployments/overlord-rtgov/gadget-web.war/WEB-INF/jboss-web.xml**
   - **standalone/deployments/overlord-rtgov/gadgets.war/WEB-INF/jboss-web.xml**
   - **standalone/deployments/overlord-rtgov/overlord-rtgov.war/WEB-INF/jboss-web.xml**
3. Configure the ModeShape and Infinispan subsystems, and the jGroups TCP Stack in the **standalone/configuration/standalone-ha.xml** file as in the following code sample:

```
<!-- Infinispan subsystem configuration -->

<cache-container name="modeshape" module="org.modeshape"
  start="EAGER">
  <transport lock-timeout="60000" />
  <replicated-cache name="sramp" mode="SYNC" batching="true">
    <locking isolation="NONE" />
    <transaction mode="NON_XA" />
    <string-keyed-jdbc-store
datasource="java:jboss/datasources/srampDS"
      passivation="false" purge="false">
      <string-keyed-table prefix="ispn_bucket">
        <id-column name="id" type="VARCHAR(500)" />
        <data-column name="datum" type="VARBINARY(60000)" />
        <timestamp-column name="version" type="BIGINT" />
      </string-keyed-table>
    </string-keyed-jdbc-store>
  </replicated-cache>
</cache-container>
<cache-container name="modeshape-binary-cache-container"
  aliases="modeshape-binary-cache" module="org.modeshape">
  <transport lock-timeout="60000" />
  <replicated-cache name="sramp-binary-fs" mode="SYNC"
    batching="true">
    <transaction mode="NON_XA" />
    <file-store relative-to="jboss.server.data.dir"
      path="modeshape/binary-store/sramp-binary-data-
${jboss.node.name}"
      passivation="false" purge="false" />
  </replicated-cache>
</cache-container>

<!-- ModeShape subsystem configuration -->

<subsystem xmlns="urn:jboss:domain:modeshape:1.0">
  <repository name="sramp" cache-name="sramp" cache-
container="modeshape"
    security-domain="overlord-idp" anonymous-roles="readonly"
    cluster-name="sramp-cluster" cluster-stack="tcp">
    <indexing rebuild-upon-startup="if_missing" />
    <local-file-index-storage
      path="modeshape/clustered-repo/${jboss.node.name}_indexes" />
    <cache-binary-storage data-cache-name="binary-fs"
      metadata-cache-name="binary-fs-meta" cache-
container="modeshape-binary-cache-container" />
  </repository>
</subsystem>

<!-- jgroups tcp stack configuration -->

<subsystem xmlns="urn:jboss:domain:jgroups:1.1" default-stack="udp">
  ...
  <stack name="tcp">
  ...
    <protocol type="TCPPING">
      <property
```

```
name="initial_hosts">0.0.0.0[7600],0.0.0.0[7600]</property>
      <property name="num_initial_members">2</property>
      <property name="port_range">0</property>
      <property name="timeout">2000</property>
    </protocol>
  </stack>
</subsystem>
```

CHAPTER 16. STARTING RED HAT JBOSS FUSE IN AN ENTERPRISE ENVIRONMENT

16.1. INTRODUCTION

Isolating services so that they are accessible only to the clients who need them increases the security of your network. JBoss EAP 6 includes two interfaces in its default configuration, both of which bind to the IP address **`127.0.0.1`**, or **`localhost`**, by default. One of the interfaces is called **`management`**, and is used by the Management Console, CLI, and API. The other is called **`public`**, and is used to deploy applications. These interfaces are not special or significant, but are provided as a starting point.

The **`management`** interface uses ports 9990 and 9999 by default, and the**`public`** interface uses port 8080, or port 8443 if you use HTTPS.

WARNING

If you expose the management interfaces to other network interfaces which are accessible from remote hosts, be aware of the security implications. Most of the time, it is not advisable to provide remote access to the management interfaces.

16.2. SPECIFY THE NETWORK INTERFACE

You can specify the IP address of the management interface, public interface, or both.

1. **Stop JBoss EAP 6.**
   Stop JBoss EAP 6 by sending an interrupt in the appropriate way for your operating system. If you are running JBoss EAP 6 as a foreground application, the typical way to do this is to press **`Ctrl+C`**.

2. **Restart JBoss EAP 6, specifying the bind address.**
   Use the **`-b`** command-line switch to start JBoss EAP 6 on a specific interface.

Example 16.1. Specify the public interface.

```
EAP_HOME/bin/domain.sh -b 10.1.1.1
```

Example 16.2. Specify the management interface.

```
EAP_HOME/bin/domain.sh -bmanagement=10.1.1.1
```

Example 16.3. Specify different addresses for each interface.

```
EAP_HOME/bin/domain.sh -bmanagement=127.0.0.1 -b 10.1.1.1
```

Example 16.4. Bind the public interface to all network interfaces.

```
EAP_HOME/bin/domain.sh -b 0.0.0.0
```

It is possible to edit your XML configuration file directly, to change the default bind addresses. However, if you do this, you will no longer be able to use the **-b**command-line switch to specify an IP address at run-time, so this is not recommended. If you do decide to do this, be sure to stop JBoss EAP 6 completely before editing the XML file.