JBoss Enterprise SOA Platform 5

SOA Getting Started Guide

This guide is for installation teams.

Edition 5.3.1

Last Updated: 2017-10-27

JBoss Enterprise SOA Platform 5 SOA Getting Started Guide

This guide is for installation teams.
Edition 5.3.1

David Le Sage
Red Hat Engineering Content Services

Suzanne Dorfield
Red Hat Engineering Content Services

Legal Notice

Abstract

This document will guide the user through the basic installation required to have the JBoss Enterprise SOA Platform up and running within ten minutes.

Table of Contents

PREFACE

CHAPTER 1. PREFACE

1.1. BUSINESS INTEGRATION

In order to provide a dynamic and competitive business infrastructure, it is crucial to have a service-oriented architecture in place that enables your disparate applications and data sources to communicate with each other with minimum overhead.

The JBoss Enterprise SOA Platform is a framework capable of orchestrating business services without the need to constantly reprogram them to fit changes in business processes. By using its business rules and message transformation and routing capabilities, JBoss Enterprise SOA Platform enables you to manipulate business data from multiple sources.

Report a bug

1.2. WHAT IS A SERVICE-ORIENTED ARCHITECTURE?

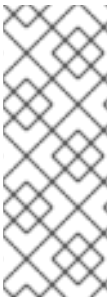
Introduction

A *Service Oriented Architecture* (SOA) is not a single program or technology. Think of it, rather, as a software design paradigm.

As you may already know, a *hardware bus* is a physical connector that ties together multiple systems and subsystems. If you use one, instead of having a large number of point-to-point connectors between pairs of systems, you can simply connect each system to the central bus. An *enterprise service bus* (ESB) does exactly the same thing in software.

The ESB sits in the architectural layer above a messaging system. This messaging system facilitates *asynchronous communications* between services through the ESB. In fact, when you are using an ESB, everything is, conceptually, either a *service* (which, in this context, is your application software) or a *message* being sent between services. The services are listed as connection addresses (known as *end-points references*.)

It is important to note that, in this context, a "service" is not necessarily always a web service. Other types of applications, using such transports as File Transfer Protocol and the Java Message Service, can also be "services."



NOTE

At this point, you may be wondering if an enterprise service bus is the same thing as a service-oriented architecture. The answer is, "Not exactly." An ESB does not form a service-oriented architecture of itself. Rather, it provides many of the tools than can be used to build one. In particular, it facilitates the *loose-coupling* and *asynchronous message passing* needed by a SOA. Always think of a SOA as being more than just software: it is a series of principles, patterns and best practices.

Report a bug

1.3. KEY POINTS OF A SERVICE-ORIENTED ARCHITECTURE

These are the key components of a service-oriented architecture:

1. the *messages* being exchanged
2. the *agents* that act as service requesters and providers
3. the *shared transport mechanisms* that allow the messages to flow back and forth.

Report a bug

1.4. WHAT IS THE JBOSS ENTERPRISE SOA PLATFORM?

The JBoss Enterprise SOA Platform is a framework for developing enterprise application integration (EAI) and service-oriented architecture (SOA) solutions. It is made up of an enterprise service bus (JBoss ESB) and some business process automation infrastructure. It allows you to build, deploy, integrate and orchestrate business services.

Report a bug

1.5. THE SERVICE-ORIENTED ARCHITECTURE PARADIGM

The service-oriented architecture (SOA) consists of three roles: requester, provider, and broker.

Service Provider

A service provider allows access to services, creates a description of a service and publishes it to the service broker.

Service Requester

A service requester is responsible for discovering a service by searching through the service descriptions given by the service broker. A requester is also responsible for binding to services provided by the service provider.

Service Broker

A service broker hosts a registry of service descriptions. It is responsible for linking a requester to a service provider.

Report a bug

1.6. CORE AND COMPONENTS

The JBoss Enterprise SOA Platform provides a comprehensive server for your data integration needs. On a basic level, it is capable of updating business rules and routing messages through an Enterprise Service Bus.

The heart of the JBoss Enterprise SOA Platform is the Enterprise Service Bus. JBoss (ESB) creates an environment for sending and receiving messages. It is able to apply “actions” to messages to transform them and route them between services.

There are a number of components that make up the JBoss Enterprise SOA Platform. Along with the ESB, there is a registry (jUDDI), transformation engine (Smooks), message queue (HornetQ) and BPEL engine (Riftsaw).

Report a bug

1.7. COMPONENTS OF THE JBOSS ENTERPRISE SOA PLATFORM

- A full Java EE-compliant application server (the JBoss Enterprise Application Platform)
- an enterprise service bus (JBoss ESB)
- a business process management system (jBPM)
- a business rules engine (JBoss Rules)
- support for the optional JBoss Enterprise Data Services (EDS) product.

Report a bug

1.8. JBOSS ENTERPRISE SOA PLATFORM FEATURES

The JBoss Enterprise Service Bus (ESB)

The ESB sends messages between services and transforms them so that they can be processed by different types of systems.

Business Process Execution Language (BPEL)

You can use web services to orchestrate business rules using this language. It is included with SOA for the simple execution of business process instructions.

Java Universal Description, Discovery and Integration (jUDDI)

This is the default service registry in SOA. It is where all the information pertaining to services on the ESB are stored.

Smooks

This transformation engine can be used in conjunction with SOA 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 SOA. It can infer data from the messages it receives to determine which actions need to be performed.

Report a bug

1.9. FEATURES OF THE JBOSS ENTERPRISE SOA PLATFORM'S JBOSSESB COMPONENT

The JBoss Enterprise SOA Platform's JBossESB component supports:

- Multiple transports and protocols
- A listener-action model (so that you can loosely-couple services together)

- Content-based routing (through the JBoss Rules engine, XPath, Regex and Smooks)
- Integration with the JBoss Business Process Manager (jBPM) in order to provide service orchestration functionality
- Integration with JBoss Rules in order to provide business rules development functionality.
- Integration with a BPEL engine.

Furthermore, the ESB allows you to integrate legacy systems in new deployments and have them communicate either synchronously or asynchronously.

In addition, the enterprise service bus provides an infrastructure and set of tools that can:

- Be configured to work with a wide variety of transport mechanisms (such as e-mail and JMS),
- Be used as a general-purpose object repository,
- Allow you to implement pluggable data transformation mechanisms,
- Support logging of interactions.

IMPORTANT

There are two trees within the source code: **`org.jboss.internal.soa.esb`** and **`org.jboss.soa.esb`**. Use the contents of the **`org.jboss.internal.soa.esb`** package sparingly because they are subject to change without notice. By contrast, everything within the **`org.jboss.soa.esb`** package is covered by Red Hat's deprecation policy.

Report a bug

1.10. TASK MANAGEMENT

JBoss SOA simplifies tasks by designating tasks to be performed universally across all systems it affects. This means that the user does not have to configure the task to run separately on each terminal. Users can connect systems easily by using web services.

Businesses can save time and money by using JBoss SOA to delegate their transactions once across their networks instead of multiple times for each machine. This also decreases the chance of errors ocurring.

Report a bug

1.11. 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 synchronise well with the existing systems as they currently stand.

The company wants an inexpensive solution but one that will adhere 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.

The JBoss Enterprise SOA Platform 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 the JBoss Enterprise SOA Platform can be updated quickly and easily.

As a result, older systems can now synchronise with newer ones due to the unifying methods of SOA. 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 JBoss Enterprise SOA Platform 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.

Report a bug

1.12. UTILISING THE JBOSS ENTERPRISE SOA PLATFORM IN A BUSINESS ENVIRONMENT

Cost reduction can be achieved due to the implementation of services that can quickly communicate with each other with less chance of error messages occurring. Through enhanced productivity and sourcing options, ongoing costs can be reduced.

Information and business processes can be shared faster because of the increased connectivity. This is enhanced by web services, which can be used to connect clients easily.

Legacy systems can be used in conjunction with the web services to allow different systems to "speak" the same language. This reduces the amount of upgrades and custom code required to make systems synchronise.

Report a bug

CHAPTER 2. INTRODUCING THE JBOSS ENTERPRISE SOA PLATFORM

2.1. INTENDED AUDIENCE

This book has been written for project teams wishing to quickly and easily install the JBoss Enterprise SOA Platform.

Report a bug

2.2. AIM OF THIS BOOK

Read this book in order to learn how to get the JBoss Enterprise SOA Platform up and running in approximately ten minutes on a Red Hat Enterprise Linux or Microsoft Windows computer. It guides you through the default installation path. Note that this Guide does not show you how to configure and secure the product for use on a production system. For that information, refer to the full *Installation and Configuration Guide*.

The defaults shown in this *Getting Started Guide* are not secure and should be used on development and testing systems only.

Report a bug

2.3. 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.

Report a bug

CHAPTER 3. PREREQUISITES

3.1. PREREQUISITES FOR INSTALLING THE JBOSS ENTERPRISE SOA PLATFORM

In order to install and run the JBoss Enterprise SOA Platform, you must already have the following items on your computer:

- A supported Java Virtual Machine
- A supported Java Development Kit (for running the quickstarts)
- A supported database server (needed to run the JBoss Server)
- Apache Ant 1.7 or later (needed to run the Database Schema Configuration Tool and deploy the JBoss ESB quick start examples)
- An archiving tool (such as FileRoller, ark or tar). (You need this to extract the contents of compressed files)
- JBoss Developer Studio 5.0. (Obtain it from the Red Hat Customer Portal at https://access.redhat.com/home)

Red Hat tests and certifies the JBoss Enterprise SOA Platform against several different hardware platforms, Java Virtual Machines, operating systems and databases. This is an ongoing process and the list of supported environments is always growing. Find the list of currently-supported environments at http://www.jboss.com/products/platforms/soa/testedconfigurations/.

Report a bug

3.2. JAVA VIRTUAL MACHINE

A Java Virtual Machine (JVM) is a piece of software that is capable of running Java bytecode. The JVM creates a standard environment in which the intermediate bytecode is run. By creating the standard environment irrespective of the underlying hardware and operating system combination, it allows programmers to write their Java code once and have confidence that it can be run on any system. Red Hat recommends customers use OpenJDK as it is an open source, supported Java Virtual Machine that runs well on Red Hat Enterprise Linux systems. Windows users should install Oracle JDK 1.6.

Report a bug

3.3. INSTALL OPEN JDK ON RED HAT ENTERPRISE LINUX

Procedure 3.1. Install Open JDK on Red Hat Enterprise Linux

1. **Subscribe to the Base Channel**
   Obtain the OpenJDK from the RHN base channel. (Your installation of Red Hat Enterprise Linux is subscribed to this channel by default.)

2. **Install the Package**
   Use the yum utility to install OpenJDK: **`yum install java-1.7.0-openjdk-devel`**

3. **Verify that OpenJDK is Now Your System Default**
   To ensure that the correct JDK is set as the system default, login as root and run the alternatives command: **`/usr/sbin/alternatives --config java`**

   Select /usr/lib/jvm/jre-1.6.0-openjdk/bin/java

   Set javac **`/usr/sbin/alternatives --config javac`**

   Select /usr/lib/jvm/java-1.6.0-openjdk/bin/java

Report a bug

3.4. APACHE ANT

Apache Ant ("Another Neat Tool") is a Java-based build tool. It is designed to automate the process of software compilation. It requires the developer to provide it with an XML-based build file from which it receives custom build instructions. Refer to http://ant.apache.org/ for more information.

Report a bug

3.5. INSTALL APACHE ANT

Procedure 3.2. Installing Apache Ant on Red Hat Enterprise Linux

1. **Download Apache Ant**
   Open a terminal and input this command: **`sudo yum install ant-trax`**

2. **Install Apache Ant**
   Enter **Y** when prompted to do so by the installer.

3. **Add the ANT_HOME Environment Variable**

   a. **`vi ~/.bash_profile`**.

   b. Add the following line:

   ```
   export ANT_HOME=/FILEPATH/ant
   ```

   where filepath is the directory in which Apache Ant was installed. Save and exit vi.

   Example 3.1.

   ```
   export ANT_HOME=/opt/apache-ant-1.8.2
   ```

4. **Append the Ant installation's `bin` directory to the Path environmental variable.**

   a. **`vi ~/.bash_profile`**.

   b. Add the following line and then save and exit vi:

```
export PATH=$PATH:$ANT_HOME/bin
```

5. **Test the installation**
   Go back to your terminal and run **`ant -version`**. The output should resemble the following:

```
[localhost]$ ant -version
Apache Ant(TM) version 1.8.2 compiled on July 6 2011
```

Procedure 3.3. Installing Apache Ant on Microsoft Windows

1. **Download Apache Ant**
   Download the latest Apache Ant binary release from http://ant.apache.org/.
2. **Extract Apache Ant**
   Extract the files to a preferred installation location such as:
   - **`c:\Program Files\Apache\Ant\`**
3. **Add the ANT_HOME Environment Variable**
   a. Click on the **Start Menu**.
   b. Open the **Control Panel**.
   c. Select **System** → **Advanced** → **Environment Variables**
   d. Create a new variable called **`ANT_HOME`**.
   e. Configure the ANT_HOME variable so that it points to the Apache Ant directory.
4. **Append the bin directory of the Ant installation to the Path environmental variable.**
   a. Click on the **Start Menu**.
   b. Open the **Control Panel**.
   c. Select **System** → **Advanced** → **Environment Variables** → **System Variables**
   d. Edit the **PATH** variable and append the text:

```
;%ANT_HOME%\bin
```

5. **Test the installation**
   Run **`ant -version`** in a command line terminal. The version number should appear.

Report a bug

CHAPTER 4. DOWNLOAD THE PRODUCT

4.1. RED HAT CUSTOMER PORTAL

The Red Hat Customer Portal is a website, located at https://access.redhat.com/home. It provides a central point from where you can manage and maintain your subscription, access the Red Hat Knowledgebase and engage with Red Hat and our partners.

Report a bug

4.2. PACKAGES AVAILABLE FOR DOWNLOAD

Table 4.1. Packages Available for Download

Package	Description
JBoss Enterprise SOA Platform Package	The SOA Platform package is a complete JBoss application deployment environment. This single installation provides a complete environment for deploying SOA applications. It includes Seam, Hibernate, clustering, and transaction services.
JBoss Enterprise SOA Platform Standalone Edition Package	The SOA Standalone package provides a light-weight solution for deployments where only the core SOA functionality is needed. It does not support clustering.
JBoss Enterprise SOA Platform Source Code Package	The source code package contains the complete source code for the JBoss Enterprise SOA Platform product.
SOA Platform JavaDocs	The JavaDocs package contains the complete JavaDocs for the JBoss Enterprise SOA Platform's APIs.

Report a bug

4.3. DIFFERENCES BETWEEN VERSIONS OF THE JBOSS ENTERPRISE SOA PLATFORM

Table 4.2. Differences Between Versions of the JBoss Enterprise SOA Platform

	SOA Platform Package	SOA Standalone Package
JBoss ESB	YES	YES
JBoss Rules	YES	YES

	SOA Platform Package	SOA Standalone Package
JBoss JBPM	YES	YES
JBoss EAP	YES	YES
BPEL Engine	YES	YES
EJB3	YES	NO
JBoss RestEasy	YES	NO
JBoss Seam	YES	NO
Support for JBoss Enterprise Data Services Deployment	YES	NO

Report a bug

4.4. JAVADOCS

JavaDocs are automatically-generated documentation for Java APIs. They are created from the comments developers add to source code as they write it. JavaDocs have become the de facto standard way of documenting Java APIs.

Report a bug

4.5. DOWNLOAD FILES FROM THE RED HAT CUSTOMER PORTAL

Task Prerequisites

Before you begin this task, you need a Customer Portal account. Browse to https://access.redhat.com and click the **Register** link in the upper right corner to create an account.

Procedure 4.1. Task:

1. Browse to https://access.redhat.com and click the **Log in** link in the top right corner. Enter your credentials and click **Log In**.

   Result:

   You are logged into RHN and you are returned to the main web page at https://access.redhat.com.

2. **Navigate to the Downloads page.**
   Use one of the following options to navigate to the **Downloads** page.

   - Click the **Downloads** link in the top navigation bar.

- Navigate directly to https://access.redhat.com/downloads/.

3. **Select the product and version to download.**
   Use one of the following ways to choose the correct product and version to download.
   - Step through the navigation one level at a time.
   - Search for your product using the search area at the top right-hand side of the screen.
4. **Download the appropriate file for your operating system and installation method of choice.**
   Depending on the product you choose, you may have the choice of a Zip archive, RPM, or native installer for a specific operating system and architecture. Click either the file name or the **Download** link to the right of the file you want to download.

Result:

The file is downloaded to your computer.

Report a bug

4.6. CHECKSUM VALIDATION

Checksum validation is used to ensure a downloaded file has not been corrupted. Checksum validation employs algorithms that compute a fixed-size datum (or checksum) from an arbitrary block of digital data. If two parties compute a checksum of a particular file using the same algorithm, the results will be identical. Therefore, when computing the checksum of a downloaded file using the same algorithm as the supplier, if the checksums match, the integrity of the file is confirmed. If there is a discrepancy, the file has been corrupted in the download process.

Report a bug

4.7. VERIFY THE DOWNLOADED FILE

Procedure 4.2. Verify the Downloaded File

1. To verify that a file downloaded from the Red Hat Customer Portal is error-free, whilst still on the portal site, go to that package's **Software Details** page. Here you will find **MD5** and **SHA256**"checksum" values that you will use to check the integrity of the file.
2. Open a terminal window and run either either the **md5sum** or **sha256sum** command, supplying the filename of the downloaded **ZIP** as an argument. The program will output the checksum value for the file.
3. Compare the checksum value returned by the command to the corresponding value displayed on the **Software Details** page for the file.

NOTE

Microsoft Windows does not come equipped with a checksum tool. Users of that operating system will have to download a third-party product instead.

Result

If the two checksum values are identical then the file has not been altered or corrupted and is, therefore, safe to use.

If the two checksum values are not identical, then download the file again. A difference between the checksum values means that the file has either been corrupted during download or has been modified since it was uploaded to the server. If, after several downloads, the checksum will still not successfully validate, please contact Red Hat Support for assistance.

Report a bug

4.8. RED HAT DOCUMENTATION SITE

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

Report a bug

CHAPTER 5. INSTALLATION

5.1. VARIABLE NAME: SOA_ROOT DIRECTORY

SOA Root (often written as SOA_ROOT) is the term given to the directory that contains the application server files. In the standard version of the JBoss Enterprise SOA Platform package, SOA root is the **`jboss-soa-p-5`** directory. In the Standalone edition, though, it is the **`jboss-soa-p-standalone-5`** directory.

Throughout the documentation, this directory is frequently referred to as **`SOA_ROOT`**. Substitute either **`jboss-soa-p-5`** or **`jboss-soa-p-standalone-5`** as appropriate whenever you see this name.

Report a bug

5.2. VARIABLE NAME: PROFILE

PROFILE can be any one of the server profiles that come with the JBoss Enterprise SOA Platform product: default, production, all, minimal, standard or web. Substitute one of these that you are using whenever you see "PROFILE" in a file path in this documentation.

Report a bug

5.3. INSTALL THE JBOSS ENTERPRISE SOA PLATFORM ON RED HAT ENTERPRISE LINUX

Prerequisites

- a Java Development Kit (Red Hat recommends OpenJDK)
- a database server
- Apache Ant 1.7 or later
- an archiving tool that can open ZIP files

WARNING

These instructions explain how to install and configure the product on a test system only. These defaults will not result in the level of security needed for a production system. Please read the full Installation Guide for information on installing and configuring this product for production use.

Follow these steps after you have downloaded either the Standalone or full SOA package from the Red Hat Customer Portal and verified its integrity.

Procedure 5.1. Installation

Procedure 5.1. Installation

1. Extract the installation directory by running **`unzip soa-p-VERSION.zip`**.
2. Open the user account settings file in your text editor: **`vi SOA_ROOT/jboss-as/server/default/conf/props/soa-users.properties`**.

   The contents of this file using the following syntax: **`username=password`**.

   ```
   #admin=admin
   ```

   Confirm that the security roles for your admin account are enabled by removing a leading hash character, if it is present.

   WARNING

   Because this account is not secure and the password can be easily guessed, use admin=admin for test purposes only. Do not use **`admin`** as a password on production systems as this may compromise security.

3. Save the file and exit vi.
4. Open the security permissions file in your text editor: **`vi SOA_ROOT/jboss-as/server/default/conf/props/soa-roles.properties`**.

   ```
   #admin=JBossAdmin,HttpInvoker,user,admin
   ```

   Confirm that the security roles for your admin account are enabled by removing a leading hash character, if it is present.
5. Save the changes to the file and exit vi.

Result

The JBoss Enterprise SOA Platform is now installed and configured for basic use.

Report a bug

5.4. INSTALL THE JBOSS ENTERPRISE SOA PLATFORM ON MICROSOFT WINDOWS

Prerequisites

- a Java Development Kit
- a database server
- Apache Ant 1.7 or later

- an archiving tool that can open ZIP files

WARNING

These instructions explain how to install and configure the product on a test system only. These defaults will not result in the level of security needed for a production system. Please read the full Installation Guide for information on installing and configuring this product for production use.

Follow these steps after you have downloaded either the Standalone or full SOA package from the Red Hat Customer Portal and verified its integrity.

Procedure 5.2. Installation

1. Extract the **`soa-p-VERSION.zip`** using the ZIP tool of your choice.
2. Confirm that the security roles for your admin account are enabled by removing a leading hash character, if it is present.

WARNING

Because this account is not secure and the password can be easily guessed, use admin=admin for test purposes only. Do not use **`admin`** as a password on production systems as this may compromise security.

3. Save the file and exit Notepad.
4. Open **`SOA_ROOT\jboss-as\server\default\conf\props\soa-roles.properties`** in Notepad.

```
#admin=JBossAdmin,HttpInvoker,user,admin
```

Remove the hash to enable the security permissions for your admin account.

5. Save the file and exit Notepad.

Result

The JBoss Enterprise SOA Platform is now installed and configured for basic use.

Report a bug

CHAPTER 6. BASIC OPERATION TUTORIAL

6.1. RUNNING THE JBOSS ENTERPRISE SOA PLATFORM FOR THE FIRST TIME

Introduction

In the following section you will learn how to launch and run the JBoss Enterprise SOA Platform for the first time. The simplest thing you can do is run the demonstration code found in the "Hello World" quick start.

As you run this quick start, you will be taught how it works.

Report a bug

6.2. START THE JBOSS ENTERPRISE SOA PLATFORM

Prerequisites

The following software must be installed:

- JBoss Enterprise SOA Platform

Procedure 6.1. Start the JBoss Enterprise SOA Platform

- **Start the SOA server in a *server window***
  - **Red Hat Enterprise Linux**
    a. Open a terminal and navigate to the **bin** directory by entering the command **cd *SOA_ROOT*/jboss-as/bin**.
    b. Enter **./run.sh** to start the SOA server. (Because you are not specifying a server profile, "default" will be used.)
  - **Microsoft Windows**
    a. Open a terminal and navigate to the **bin** directory by entering the command **chdir *SOA_ROOT*\jboss-as\bin**.
    b. Enter **run.bat** to start the SOA server. (Because you are not specifying a server profile, "default" will be used.)

Result

The server starts. Note that this will take approximately two minutes, depending on the speed of your hardware.

NOTE

To verify that there have been no errors, check the server log: **less *SOA_ROOT*/jboss-as/server/*PROFILE*/log/server.log**. As another check, open a web browser and go to http://localhost:8080. Make sure you can login to the admin console with the user name and password you have set.

Report a bug

6.3. TROUBLESHOOTING THE BOOT PROCESS

Errors in the server.log are indicated by the keyword "ERROR". If you see an error in the log, look through this list to find the cause:

1. "Address already in use" - There is already a server running on port 8080.
2. "Java not found" - The Java JRE may not be installed, or if it is, your PATH environment variable is not set to locate the java runtime.
3. "Class not found" - The CLASSPATH environment variable is not set properly. You really don't need to set this variable as the server startup script sets it for you.
4. If you see any of these errors, examine the server.log messages that come before and after the error message for additional information regarding the root cause of the error.

Report a bug

6.4. RUNNING THE "HELLO WORLD" QUICKSTART

6.4.1. Quickstart

The quickstarts are sample projects. Each one demonstrates how to use a specific piece of functionality in order to aid you in building services. There are several dozen quickstarts included in the **`SOA_ROOT/jboss-as/samples/quickstarts/`** directory. Build and deploy every quickstart by using **Apache Ant**.

Report a bug

6.4.2. Important Notes About Quickstarts

When intending to run a quickstart, remember the following points:

1. Each quickstart needs to be built and deployed using Apache Ant.
2. Each quickstart uses the **`samples/quickstarts/conf/quickstarts.properties`** file to store environment-specific configuration options such as the directory where the server was installed. You must create a **`quickstarts.properties`** file that matches your server installation. An example properties file (**`quickstarts.properties-example`**) is included.
3. Each quickstart has different requirements. These are documented in their individual **`readme.txt`** files.
4. Not every quickstart can run under every server profile.
5. The jBPM quickstarts require a valid jBPM Console user name and password. Supply these by adding them as properties in the **`SOA_ROOT/jboss-as/samples/quickstarts/conf/quickstarts.properties`** file:

```
# jBPM console security credentials
```

```
jbpm.console.username=admin
jbpm.console.password=adminpassword
```

The quickstarts that are affected by this requirement are **bpm_orchestration1**, **bpm_orchestration2**, **bpm_orchestration3** and **bpm_orchestration4**.

6. You can only execute some of the quickstarts (such as **groovy_gateway**) if the server is not running in *headless* mode. (The JBoss Enterprise SOA Platform is configured to launch in headless mode by default.)

IMPORTANT

Red Hat recommends that you run production servers in headless mode only.

Report a bug

6.4.3. Deploy the "Hello World" Quickstart on Your Test Server

Prerequisites

- Check that the setting in **SOA_ROOT/jboss-as/samples/quickstarts/conf/quickstarts.properties-example** matches the server configuration (**default** in a testing environment).

Procedure 6.2. Deploy the "Hello World" Quickstart

1. Check that the server has fully launched.
2. Open a second terminal window and navigate to the directory containing the quick start: **cd SOA_ROOT/jboss-as/samples/quickstarts/helloworld** (or **chdir SOA_ROOT\jboss-as\samples\quickstarts\helloworld** in Microsoft Windows).
3. Run **ant deploy** to deploy the quickstart. Look for messages such as this to confirm if the deployment was successful:

```
deploy-esb:
     [copy] Copying 1 file to
/jboss/local/53_DEV2/jboss-soa-p-5/jboss-as/server/default/deploy

deploy-exploded-esb:

quickstart-specific-deploys:
     [echo] No Quickstart specific deployments being made.

display-instructions:
     [echo]
     [echo] ******************
     [echo] Quickstart deployed to target JBoss ESB/App Server at
'/jboss/local/53_DEV2/jboss-soa-p-5/jboss-as/server/default/deploy'.
     [echo] 1.  Check your ESB Server console to make sure the
deployment was
executed without errors.
     [echo] 2.  Run 'ant runtest' to run the Quickstart.
```

```
     [echo] 3.  Check your ESB Server console again.  The Quickstart
should
have produced some output.
     [echo] ******************

deploy:

BUILD SUCCESSFUL
```

Also, check for this in the **SOA_ROOT/jboss-as/server/default/log/server.log**:

```
10:58:52,998 INFO  [NamingHelper] JNDI InitialContext properties:{}
11:00:58,154 INFO  [QueueService]
Queue[/queue/quickstart_helloworld_Request_esb] started,
fullSize=200000,
pageSize=2000, downCacheSize=2000
11:00:58,186 INFO  [QueueService]
Queue[/queue/quickstart_helloworld_Request_gw] started,
fullSize=200000,
pageSize=2000, downCacheSize=2000
11:00:58,427 INFO  [EsbDeployment] Starting ESB Deployment
'Quickstart_helloworld.esb'
```

4. Run the quickstart by issuing this command: **ant runtest**. When the quickstart is run, messages such as this are written to the **SOA_ROOT/jboss-as/server/default/log/server.log**:

```
11:03:02,190 INFO  [STDOUT]
&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
11:03:02,191 INFO  [STDOUT] Body: Hello World
11:03:02,191 INFO  [STDOUT]
&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
11:03:02,192 INFO  [STDOUT] Message structure:
11:03:02,192 INFO  [STDOUT] [Hello World].
```

Result

The words "Hello World" will appear on the server terminal. This message will also be appended to the **SOA_ROOT/jboss-as/server/default/log/server.log** file.

Report a bug

6.4.4. ant deploy

ant deploy compiles a quickstart's source code in the **build** directory, then generates an .ESB file (such as **Quickstart_helloworld.esb**) in the server profile's **deploy** directory. (Note that it generates .JAR files for BPEL quickstarts.) The server detects the presence of the new .esb archive and deploys it. In the .ESB archive is a deployment.xml file that **ant deploy** uses to configure a queue.

ant deploy also uses an XSL template to transform generic JMS queue names into the specific JMS queues needed by the target server's messaging provider. Ant selects the correct messaging provider by examining the server for a messaging provider deployment. Only JBoss Messaging, JBoss MQ, and

HornetQ are detected by the build script. Other messaging providers are not supported by the quick start. Ant then puts the **deployment.xml** file into the **build/META-INF** directory before including it in the same .ESB archive as the rest of the quickstart.

Report a bug

6.4.5. ant runtest

ant runtest sends an ESB-unaware "Hello World" message (which is a plain String object) to the JMS Queue (**queue/quickstart_helloworld_Request_gw**). It instructs Java to run the sender class (in the case of the "Hello World" quick start, this is called **org.jboss.soa.esb.samples.quickstart.helloworld.test.sendJMSMessage**). By doing so, it sends a message directly to the deployed process.

Report a bug

6.4.6. ant sendesb

The **ant sendesb** command sends an ESB message to the SOA Server. It sends the ESB-aware message directly to the ESB listener, meaning that it does not have to utilize a gateway.

Report a bug

6.4.7. Undeploy the "Hello World" Quickstart

Procedure 6.3. Task

1. Navigate to the quickstart's directory: **cd SOA_ROOT/jboss-as/samples/quickstarts/helloworld** (or **chdir SOA_ROOT\jboss-as\samples\quickstarts\helloworld** if you are running Microsoft Windows).
2. Run the **ant undeploy** command. You should see messages such as this displayed:

```
undeploy:
[delete] Deleting:
/jboss/local/53_DEV2/jboss-soa-p-5/jboss-
as/server/default/deploy/Quickstart_helloworld.esb

BUILD SUCCESSFUL
```

   And messages such as this written to the server.log:

```
11:10:08,205 INFO  [EsbDeployment] Stopping
'Quickstart_helloworld.esb'
11:10:08,577 INFO  [EsbDeployment] Destroying
'Quickstart_helloworld.esb'
```

Report a bug

6.5. STOP THE JBOSS ENTERPRISE SOA PLATFORM SERVER

Procedure 6.4. Stop the JBoss Enterprise SOA Platform Server

- **Stop the SOA server**
  Press **`ctrl-c`** in the *server window* (the terminal window where the SOA server was started).

Result

The server will shut down. Note that this process will take a few minutes. Look for this line in the **`server.log`** file to confirm that the server has shut down successfully:

```
        12:17:02,786 INFO  [ServerImpl] Shutdown complete
```

Report a bug

6.6. EXAMINING THE "HELLO WORLD" QUICKSTART

6.6.1. Overview of How the "Hello World" Quickstart Works

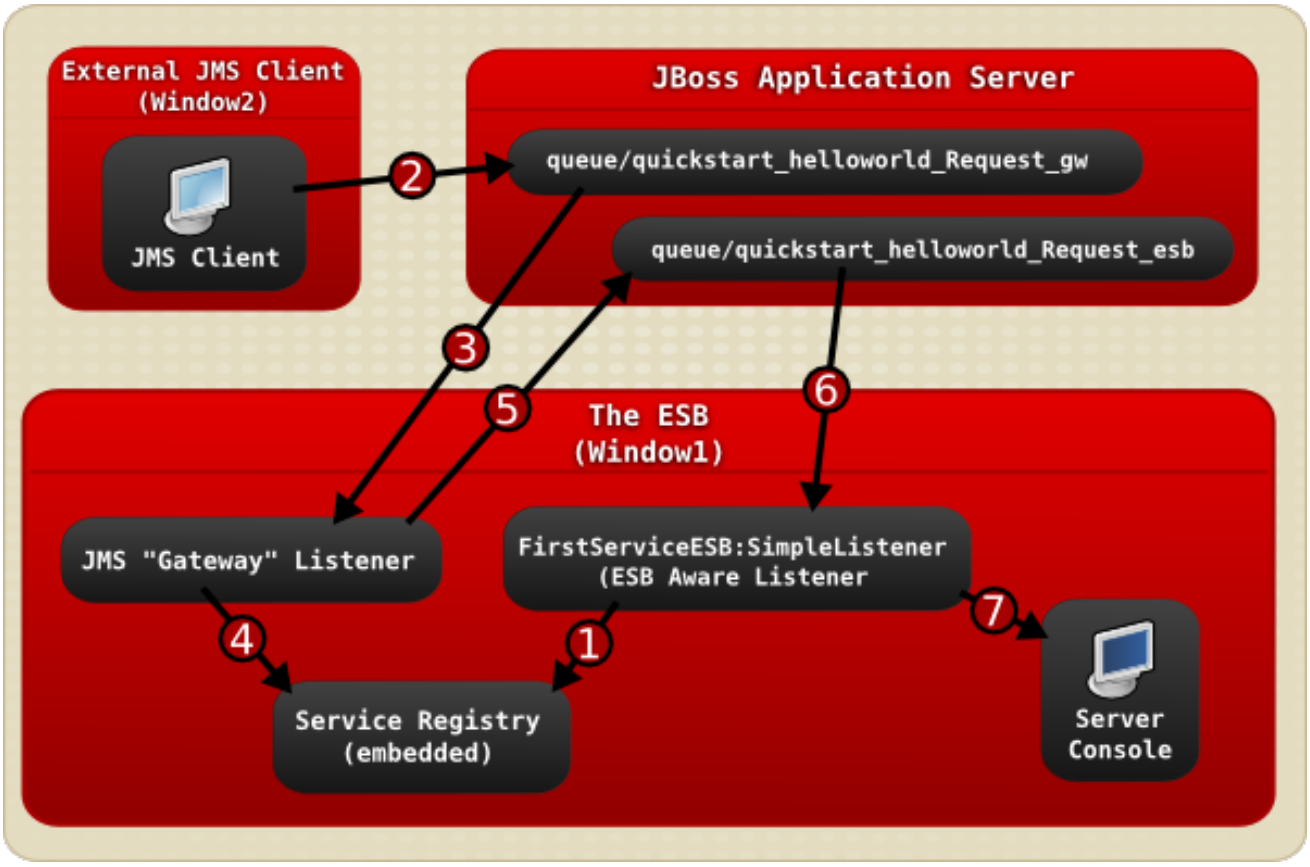


Figure 6.1. Image

1. The **JBoss Enterprise SOA Platform** server is launched in **`Window1`** and then the **`FirstServiceESB:SimpleListener`** service is added to the Service Registry service when the helloworld quickstart is deployed.

2. A JMS client sends an ESB-unaware "Hello World" message, (it is a plain **`String`** object), to the JMS Queue (**`queue/quickstart_helloworld_Request_gw`**).

3. The JMS Gateway Listener receives the ESB-unaware message and creates from it an ESB-aware message for use by ESB-aware end-points.

4. The **`JMS Gateway Listener`** uses the **`service registry`** to find the **`FirstServiceESB:SimpleListener`** service's *end-point reference* (EPR). In this case, the EPR is the **`queue/quickstart_helloworld_Request_esb`** JMS queue.

5. The **`JMS Gateway Listener`** takes the new ESB-aware message and sends it to the **`queue/quickstart_helloworld_Request_esb`** JMS queue.

6. The **`FirstServiceESB:SimpleListener`** service receives the message.

7. The **`FirstServiceESB:SimpleListener`** service extracts the payload from the message and outputs it to the console.

Report a bug

6.6.2. ESB Message

An ESB message is a message that takes the form defined by the **`org.jboss.soa.esb.message`** interface. This standardized format consists of a header, body (payload) and attachments. All ESB-aware clients and services communicate with one another using messages.

Report a bug

6.6.3. Components of an ESB Message

An ESB message is made up of the following components:

Header

The header contains such information as the destination end-point reference, the sender end-point reference, and where the reply goes. This is all general message-level functional information.

Context

This is additional information that further explains the message; for example, transaction or security data, the identity of the ultimate receiver or HTTP-cookie information.

Body

The actual contents of the message.

Fault

Any error information associated with the message.

Attachment

Any attachments (additional files) associated with the message.

Properties

Any message-specific properties.(For example, the jbossesb.message.id property specifies a unique value for each message).

Here is a code representation:

```
    <xs:complexType name="Envelope">
 <xs:attribute ref="Header" use="required"/>
 <xs:attribute ref="Context" use="required"/>
 <xs:attribute ref="Body" use="required"/>
 <xs:attribute ref="Attachment" use="optional"/>
 <xs:attribute ref="Properties" use="optional"/>
 <xs:attribute ref="Fault" use="optional"/>
</xs:complexType>
```

Report a bug

6.6.4. How Message Objects are Sent to the Queue

Overview

The JBoss Enterprise SOA Platform product uses a properties object that is populated with parameters to identify the presence of JNDI on the local server. It is then used as the parameter for a call to create a new Naming Context which is used to obtain the ConnectionFactory. The Connection Factory, in turn, creates the QueueConnection, which creates the QueueSession. This QueueSession creates a Sender object for the Queue. The Sender object is used to create an ObjectMessage for the sender and to then send it to the Queue.

Report a bug

6.6.5. Properties Object

The Properties object is a container for a generic list of keys and values. You should populate the Properties object with the parameters needed to identify JBoss JNDI on the local server and then use it as the parameter for a call to create a new Naming Context.

Report a bug

6.6.6. Naming Context

A Naming Context is an object that connects to a directory and allows the user to obtain objects by name reference.

Report a bug

6.6.7. ConnectionFactory

The ConnectionFactory is an object (**`org.jboss.jms.client.JBossConnectionFactory`**) that creates a QueueConnection. The Naming Contect obtains the ConnectionFactory from JNDI.

Report a bug

6.6.8. QueueConnection

The QueueConnection is an object created by the ConnectionFactory. It is used to create and start a QueueSession.

Report a bug

6.6.9. QueueSession

The QueueSession is created by the QueueConnection. It creates a Sender object for the queue and an ObjectMessage containing a string.

Report a bug

6.6.10. SOA_ROOT/jboss-as/samples/quickstarts/helloworld/build.xml

The **`build.xml`** file contains the instructions used by **`ant deploy`** to compile the quickstart's source code in the **`build`** directory. You can edit this file to add your own custom instructions.

Report a bug

6.6.11. SOA_ROOT/jboss-as/samples/quickstarts/helloworld/deployment.xml

The **`deployment.xml`** file is used by **`ant runtest`** to create and configure a messaging queue.

`ant deploy` generates the **`deployment.xml`** file in the **`build/META-INF`** directory during the compilation process. It then populates it when it determines which of the hard-coded JMS queues should be used. Once populated, the file is packaged as part of the .ESB archive. (Ant uses an XSL template to transform generic JMS queue names into the specific JMS queues required by the target server's messaging provider. It is from this template that the **`deployment.xml`** file is created.)

Report a bug

6.6.12. Messaging Queues

A message queue is a queue that is generated when an application is deployed. Messages are sent to these queues where they await the message listener.

Report a bug

6.6.13. Message Listeners

Message listeners encapsulate the communications end-points needed to receive SB-aware messages. Listeners are defined by services and their role is to monitor queues. They receive any messages as they land in those queues. When a listener receives a message, the ESB server calls the appropriate action class defined in the action definition. The methods in this class process the message. In other words, listeners act as inbound routers, directing messages to the action pipeline. When the message has been modified by the actions on the pipeline, the listener sends the result to the replyTo end-point.

You can configure various parameters for listeners. For instance, you can set the number of active worker threads.

There are two types of listeners: ESB-aware listeners and gateway listeners. Gateway listeners are different from ESB-aware listeners in that they accept data in different formats (such as objects in files, SQL tables and JMS messages). They then convert them from these formats to the ESB messaging format. By contrast, ESB-aware listeners can only accept messages that are in the **`org.jboss.soa.esb.message.Message`** format. Each gateway listener must have a corresponding ESB listener defined.

With ESB-aware listeners, RuntimeExceptions can trigger rollbacks. By contrast, with a gateway listener, the transaction simply sends the message to the JBoss ESB. The message is then processed asynchronously. In this way, message failures are separated from message receipts.

Report a bug

6.6.14. ESB-Awareness

If application clients and services are referred to as being ESB-aware, this means that they can understand the message format and transport protocols used by the SOA Platform's enterprise service bus.

Report a bug

6.6.15. Gateway Listener

A gateway listener is used to bridge the ESB-aware and ESB-unaware worlds. It is a specialized listener process that is designed to listen to a queue for ESB-unaware messages that have arrived through an external (ESB-unaware) end-point. The gateway listener receives the messages as they land in the queue. When a gateway listener "hears" incoming data arriving, it converts that data (the non-ESB messages) into the **`org.jboss.soa.esb.message.Message`** format. This conversion happens in a variety of different ways, depending on the gateway type. Once the conversion has occurred, the gateway listener routes the data to its correct destination.

Report a bug

6.6.16. Senders

Senders are created by QueueSessions. There is a sender for each queue. The Sender's **`send`** method is called by its QueueSession's ObjectMessage when **`ant runtest`** is executed. When this happens, the client sends a message to the queue.

Report a bug

6.6.17. Learn More About a Quickstart

To learn more about a particular quickstart:

Procedure 6.5. Task

1. Study the quickstart's **readme.txt** file.
2. Run the **ant help** command in the quickstart's directory.

Report a bug

6.7. THE "HELLO WORLD" QUICKSTART'S SOURCE CODE

6.7.1. SOA_ROOT/jboss-as/samples/quickstarts/helloworld/src

The "Hello World" quickstart's **src** directory contains the uncompiled programming instructions. The classes are in files nested in subdirectories. ant deploy compiles this source code.

Report a bug

6.7.2. SOA_ROOT/jboss-as/samples/quickstarts/helloworld/lib

The **lib** directory contains the classes needed by **ant deploy**, (in addition to the source code), to compile the quick start.

Report a bug

6.7.3. SOA_ROOT/jboss-as/server/SERVER_PROFILE/deploy

ant deploy moves the compiled version of a quick start (in the form of an .ESB archive file) from the **build** directory to the **/jboss-as/server/default/deploy/** directory. The JBoss Enterprise SOA Server is polling this directory and, when it detects the presence of a new .ESB file, it deploys it.

Report a bug

CHAPTER 7. REMOVAL

7.1. REMOVE THE JBOSS ENTERPRISE SOA PLATFORM FROM YOUR SYSTEM

Procedure 7.1. Remove the JBoss Enterprise SOA Platform from Your System

1. **Remove the JBoss Enterprise SOA Platform from a Red Hat Enterprise Linux System**
   Having made sure the server is shut down, navigate to the level above the SOA_ROOT directory and issue this command: `rm -Rf SOA_ROOT`.

2. **Remove the JBoss Enterprise SOA Platform from a Microsoft Windows System**
   Having made sure the server is shut down, open Windows Explorer, go to the directory in which the SOA_ROOT is located, select the SOA_ROOT and delete it.

3. Delete the database.

Report a bug

APPENDIX A. SOME DEFINITIONS

A.1. ENTERPRISE SERVICE BUS

An enterprise service bus is a concrete implementation of an abstract SOA design philosophy. An enterprise service bus (ESB) has two roles: it provides message routing functionality and allows you to register services. The enterprise service bus that lies at the center of the JBoss Enterprise SOA Platform is called JBoss ESB.

An enterprise service bus deals with infrastructure logic, not business logic (which is left to higher levels). Data is passed through the enterprise service bus on its way between two or more systems. Message queuing may or may not be involved. The ESB can also pass the data to a transformation engine before passing it to its destination.

Report a bug

A.2. JBOSS RULES

JBoss Rules is the name of the business rule engine provided as part of the JBoss Enterprise SOA Platform product.

Report a bug

A.3. SOA-USERS.PROPERTIES

The **`soa-users.properties`** file is where the user accounts and passwords for accessing the SOA Web consoles are stored. Administrators control access to the system by editing this file. Note that the passwords are saved in clear text so for production systems, password encryption should be used instead.

Report a bug

A.4. SOA-ROLES.PROPERTIES

The **`soa-roles.properties`** file is where user access privileges are defined. This file uses the following syntax: **`username=role1,role2,role3`**. You can assign any number of roles. Note that a user must be assigned the **`JBossAdmin`**, **`HttpInvoker`**, **`user`**, and **`admin`** roles in order to be able to log into the server consoles.

Report a bug

A.5. RUN.SH

`run.sh` is the shell script the user runs to launch the JBoss Enterprise SOA Platform. The Microsoft Windows equivalent is **`run.bat`**. The script contains the commands needed to start the server with the profile and port binding which the user has specified in the shell. The script is found in the **`SOA_ROOT/jboss-as/bin`** directory.

Report a bug

A.6. SERVER PROFILES

A server profile is a set of pre-determined settings for running the JBoss Enterprise SOA Platform in different ways. The following profiles come with the product: all, default, minimal, production, standard and web. They are found in the **SOA_ROOT/jboss-as/server/** directory. The user specifies which profile to run when launching the software by using the **-c** switch. If none is specified, the "Default" profile is used.

Report a bug

APPENDIX B. REVISION HISTORY

Revision 5.3.1-31.400 **2013-10-31** **Rüdiger Landmann**

Rebuild with publican 4.0.0

Revision 5.3.1-31 **Tue Feb 05 2013** **David Le Sage**

Built from Content Specification: 6405, Revision: 371691 by dlesage