JBoss Enterprise SOA Platform 5.0

Getting Started Guide

Your guide to starting out with the JBoss Enterprise SOA Platform

JBoss Enterprise SOA Platform 5.0 Getting Started Guide
Your guide to starting out with the JBoss Enterprise SOA Platform
Edition 3

1801 Varsity Drive
Raleigh, NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588 Research Triangle Park, NC 27709 USA

A guide to the inital installation & configuration of the JBoss Enterprise SOA Platform.

Preface

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts*[1] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight key caps and key-combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a key cap, all presented in Mono-spaced Bold and all distinguishable thanks to context.

Key-combinations can be distinguished from key caps by the hyphen connecting each part of a key-combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F1`** to switch to the first virtual terminal. Press **`Ctrl`**+**`Alt`**+**`F7`** to return to your X-Windows session.

The first sentence highlights the particular key cap to press. The second highlights two sets of three key caps, each set pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`Mono-spaced Bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialogue box text; labelled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

[1] https://fedorahosted.org/liberation-fonts/

> Choose **System > Preferences > Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications > Accessories > Character Map** from the main menu bar. Next, choose **Search > Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit > Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in Proportional Bold and all distinguishable by context.

Note the **>** shorthand used to indicate traversal through a menu and its sub-menus. This is to avoid the difficult-to-follow 'Select **Mouse** from the **Preferences** sub-menu in the **System** menu of the main menu bar' approach.

`Mono-spaced Bold Italic` or ***`Proportional Bold Italic`***

Whether Mono-spaced Bold or Proportional Bold, the addition of Italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **`ssh`** ***`username@domain.name`*** at a shell prompt. If the remote machine is **`example.com`** and your username on that machine is john, type **`ssh john@example.com`**.
>
> The **`mount -o remount`** ***`file-system`*** command remounts the named file system. For example, to remount the **`/home`** file system, the command is **`mount -o remount /home`**.
>
> To see the version of a currently installed package, use the **`rpm -q`** ***`package`*** command. It will return a result as follows: ***`package-version-release`***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules* (*MPMs*). Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server.

1.2. Pull-quote Conventions

Two, commonly multi-line, data types are set off visually from the surrounding text.

Output sent to a terminal is set in `Mono-spaced Roman` and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in `Mono-spaced Roman` but are presented and highlighted as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }

}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

A note is a tip or shortcut or alternative approach to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring Important boxes won't cause data loss but may cause irritation and frustration.

Warning

A Warning should not be ignored. Ignoring warnings will most likely cause data loss.

2. We Need Feedback!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla: *http://bugzilla.redhat.com/bugzilla/* against the product **JBoss Enterprise SOA Platform.**

When submitting a bug report, be sure to mention the manual's identifier: *SOA_Getting_Started_Guide*

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Installing the JBoss Enterprise Service-Oriented Architecture Platform

This chapter will teach you how to install the JBoss Enterprise *Service-Oriented Architecture Platform* (SOA-P.)

1.1. Prerequisites

1.1.1. Hardware Requirement Recommendations

These recommendations do not represent an absolute minimum requirement for running the SOA-P but, rather, present a baseline for reasonable performance. The SOA-P will run on hardware that is below this standard but your performance will suffer accordingly.

Also, note that performance is affected by your choice of hardware platform. When using 'non-server class' hardware (such as a laptop for development & testing), performance may be worse than when the system is deployed in your production environment on dedicated servers.

These guidelines also do not take into account the requirements of the operating system or other applications on the server. It is the responsibility of the developers to provide performance requirements for their own services and those of the applications to be deployed.

CPU

Minimum: Pentium 4 class.

Recommended: Dual-core or greater.

MEMORY

One gigabyte of memory is the recommended minimum. Increasing memory is often the most effective means of improving performance.

STORAGE SPACE

The full installation of the SOA-P server requires 800 megabytes of hard disk space. The stand-alone *Enterprise Service Bus* server only requires 300 megabytes of space, including documentation. (One will require additional space for log files.) Ten gigabytes is considered adequate for a production environment, provided that log rotation has been configured.

1.1.2. Configuring a Java Environment

One requires a working installation of **JDK 1.5** before one can install and run the SOA-P. This booklet will guide you through installing the 32-bit Sun *Java Development Kit* (JDK) 5.0 on both *Red Hat Enterprise Linux* and *Microsoft Windows 2003* machines. machine.

The "Certified Compatible Configurations" section of the *Release Notes* document contains details of those Java JDKs which are supported.

1.1.3. Installing the Sun Java Development Kit on Red Hat Enterprise Linux

Procedure 1.1. Install Sun JDK on Red Hat Enterprise Linux

1. The Sun Java Software Development Kit (SDK) can be obtained via the Red Hat Enterprise Linux 4 "*Extras*" channel for your Linux variant and architecture. The channel names are as follows:

   - **`rhel-i386-as-4-extras`**
   - **`rhel-i386-es-4-extras`**
   - **`rhel-x86_64-as-4-extras`**
   - **`rhel-x86_64-es-4-extras`**

2. The Sun Java Development Kit for Red Hat Enterprise Linux 5 is provided via the *Supplementary* channel. The exact channel name will vary according to the architecture (**`i386`** or **`x86_64`**) of your Red Hat Enterprise Linux 5 installation.

   The possible channels are:

   - **`rhel-i386-server-supplementary-5`**
   - **`rhel-x86_64-server-supplementary-5`**

3. Use **`up2date`** (Red Hat Enterprise Linux 4) or **`yum`** (Red Hat Enterprise Linux 5) to install the packages named **`java-1.6.0-openjdk`** and **`java-1.6.0-openjdk-devel`**.

4. You may optionally wish to use **`alternatives`** to set the system-wide default **`java`**, **`javac`** and **`java_sdk_1.5.0`** to use the Java Virtual Machine you have chosen.

   This is only needed if you want to use the *Sys V* service script and/or want this installed Software Development Kit to be the default Java and **`javac`** libraries used by the system. This choice can often be overridden by setting the **`JAVA_HOME`** environment variable.

   The **`alternatives`** system allows one to have different versions of Java, from different sources, co-existing on one's system. One should make sure that the desired version is actually selected, so that the service script uses it.

   As the root user, issue the following command:

   ```
   /usr/sbin/alternatives --config java
   ```

   and make sure the desired one is selected (marked with a '+'). Alternatively, you can select it by entering its number as prompted.

   Make sure you do the same for both **`javac`** and **`java_sdk_1.5.0.`** Ensure that both of these also point to the same manufacturer and version.

1.1.4. Installing and Configuring the 32-bit Sun JDK 5.0 on Microsoft Windows

Procedure 1.2. Installing and Configuring the 32-bit Sun JDK 6.0 on Microsoft Windows

1. Download the Sun JDK (Java 2 Development Kit) from their website: *http://java.sun.com/javase/downloads/index.jsp*.

   You should select the most recent Java Development Kit update, (which is **`JDK 6 Update 16`** at time of writing), and select the "**`Windows Platform`**," in order to download the appropriate installation executable.

2. Create an environment variable called **`JAVA_HOME`** that points to the JDK installation directory, for example: **`C:\Program Files\Java\jdk1.6.0_16\`**. To do this, click on the **Start Menu**, open the **Control Panel**, switch to **Classic View** if necessary, open the **System Control Panel** applet, select the **Advanced Tab**, and click on the **Environment Variables** button.

3. For convenience when running the Java Development Kit tools from the command line, one can add the **`bin`** directory of the JDK installation to one's PATH.

   To do this, open the **Control Panel** from the **Start Menu**, switch to classic view if necessary, then edit the PATH environment variable found in **System** -> **Advanced** -> **Environment Variables** -> **System Variables**. Add a semicolon and **`%JAVA_HOME%\bin`** to the end of the PATH value.

4. In order to run Java from the command line add the **`jre\bin`** directory to your path. This might look, for example, like this: **`C:\Program Files\Java\jdk1.5.0_`*`11`*`\jre\bin`**.

1.1.5. Installing Apache Ant

The Java build tool *Apache Ant* is not required for one to be able to install and operate of the JBoss Enterprise SOA-P. However, it is required for some configuration tasks and also for building and deploying the Quick Starts. If you are using a development workstation, it is possible that Apache Ant is already installed.

More information about Apache Ant can be found at the project's website: *http://ant.apache.org*.

Procedure 1.3. Installing Apache Ant on Red Hat Enterprise Linux 5

- Apache Ant can be installed by using the command **`yum install ant`**

Procedure 1.4. Installing Apache Ant on Other Operating Systems

1. **Download and Extract**

   Download the Apache Ant binary release from *http://ant.apache.org/bindownload.cgi*. (At time of writing the current version is 1.7.1)

   Next, one must choose the location at which one wishes it to be installed, such as the **`c:\Program Files\Apache\Ant`** directory. Extract it there.

2. **Configure the Environmental Variables**

   Firstly, one must create an environment variable called `ANT_HOME`. Ensure that this variable contains the path on which one extracted the files.

On Unix/Linux systems, one does do this by simply adding the following line to one's **`~/.bashrc`** file, substituting with the path to where you extracted the files:

```
export ANT_HOME=/opt/apache-ant-1.7.1
```

On Microsoft Windows one undertakes this task by opening the **Control Panel** from the **Start Menu**, then selecting **System** -> **Advanced** -> **Environment Variables**. Create a new variable, call it **`ANT_HOME`** and set it to be the **`ant`** directory.

3.

```
export ANT_HOME=/opt/apache-ant-1.7.1
```

4. **Include in the Path**

   On Unix/Linux systems, one does this by simply adding the following line to the **`~/.bashrc`** file after the line setting the `ANT_HOME` variable.

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

On Microsoft Windows one undertakes this task by opening the **Control Panel** from the **Start Menu**, then editing the `PATH` environment variable found in **System** -> **Advanced** -> **Environment Variables** -> **System Variables** -> **Path**. Add a semicolon and **`%ANT_HOME%\bin`** to the end of the path value.

To test one's Apache Ant installation, run the program **`ant -version`** on the command line. The output should look something like:

```
Apache Ant version 1.7.1
            compiled on June 27 2008
```

1.2. Upgrading or Migrating Applications

This release of the JBoss Enterprise SOA-P is backwards-compatible with previous releases with one exception. The inclusion of the *JBoss Business Process Manager* (jBPM) 3.2.3 in this release has introduced one additional configuration requirement. One now required to set the jta.UserTransaction property in the **`hibernate.cfg.xml`** file to `UserTransaction`. This requirement has been added due to changes to the transaction look-ups in this new version of jBPM.

Important

Although every effort is made to ensure a smooth transition to the latest version of the software, each deployment environment is different. As a matter of best practice, Red Hat very strongly recommended that you test all of your existing applications on this new version of the SOA-P before deploying it in your production environment.

1.3. Installing the Service-Oriented Architecture Platform

Once one has their Java environment configured, installation of the SOA-P is quite straightforward.

Procedure 1.5. Installation

1. **Obtain the Appropriate SOA-P `ZIP` File**

   There are two different **`ZIP`** files available. One contains the stand-alone version of Service-Oriented Architecture Platform, whilst the other contains the version of the SOA-P which features the embedded JBoss *EAP* Server.

   That which you select will depend upon your requirements. The stand-alone version is a lightweight option with core *Enterprise Service Bus* functionality. The full version, has the embedded JBoss EAP, a fully-fledged *Java Application Server* and an *Application Framework*.

2. **Verify the Integrity of the `ZIP` File**

   Along with the **`ZIP`** archive, there is an **`.MD5`** file. This contains a check-sum which one can use to verify the integrity of the **`ZIP`** archive. It will alert one if the file has been changed since it was uploaded or, alternatively, if it has been corrupted on download.

   On a Linux system, one can use the command **`md5sum <filename>`** to run this check. On a **`Windows`** system, one will need to download an **`md5sum`** generating program such as *MD5summer*[1] from a third party.

   The program will generate an **`md5sum`** that can then be compared against the sum in the **`.MD5`** file. If the two **`md5sums`** are not identical, it means that the **`ZIP`** file you have obtained has either been corrupted upon download or has been modified since you uploaded it to the server.

3. **Expand the `ZIP` File**

   Finally, once you have checked the archive's integrity to your satisfaction, you can install the SOA-P by simply decompressing the **`ZIP`** file into a sub-directory of your choice.

4. **Upgrading the ESB Plug-In for JON**

   If one is using the *JBoss Operations Network* (JON) product to monitor or manage one's JBoss Enterprise SOA-P server, then one needs to ensure that the the correct version of the Enterprise Service Bus plug-in for JON has been installed on the JON Server prior to starting the SOA-P server for the first time. Refer to *Section 1.5, “ UPDATED The JBoss Operations Network Enterprise Service Bus Plug-In ”* for directions.

Having successfully completed these steps, you will find that the JBoss Enterprise Service-Oriented Architecture Platform is now installed on your system. In the following chapter, you will learn how to test the installation in order to make sure that everything is working satisfactorily.

1.4. Upgrading an existing SOA-P Installation

Upgrading an existing Service-Oriented Architecture Platform involves installing the new version as a separate server and then migrating one's configuration changes and applications onto it.

Before beginning such an upgrade, one should:

1. Firstly, read through *all* of the instructions.

2. Secondly, review the *Release Notes* for the new version, paying particular heed to the section entitled "Upgrading from an Earlier Version." This will contain any additional steps that one may be required to undertake.

3. Next, one must ensure that one has current backups of all data on one's existing JBoss Enterprise SOA-P server, including its databases, in case of unanticipated problems.

Warning

DO NOT ATTEMPT TO UPGRADE BY UNZIPPING THE NEW RELEASE OVER THE TOP OF YOUR EXISTING INSTALLATION. Doing so will overwrite all of your configuration changes and may result in duplicate files.

Procedure 1.6. Upgrading an Existing Installation

1. **Download and Validate**

   Download and validate the files as through performing a fresh installation. Refer to steps one and two of the above procedure, *Section 1.3, " Installing the Service-Oriented Architecture Platform "*.

2. **Shut Down the Current Server**

   Ensure that the existing 4.3.GA server that is to be upgraded is completely shutdown.

3. **Install in a New Directory**

   Rename the existing JBoss Enterprise SOA-P directory. Decompress the newly downloaded package alongside the old directory and give the subsequently-created new directory the same name as the old one had before.

   The name of this directory does not effect the operation of the SOA-P but if one using the JBoss Operations Network product to monitor or manage the SOA-P, it will affect how the JON server will treat the SOA-P server after the upgrade. Refer to *Section 1.5, " UPDATED The JBoss Operations Network Enterprise Service Bus Plug-In "* for additional information.

4. **Migrate Applications**

   Migrate applications to the new installation by copying them from the old version's **`deploy`** directory into the new one. This also includes any **`.jar`** files that one might have added to the server's **`lib`** directory upon which one's applications depend.

5. **Migrate the Configuration**

   Update the configuration on the new server to match that on the old one by copying any custom configuration changes (such as user accounts) from one to the other.

   Warning

   You will need to refer to the section of the *Release Notes* entitled "Upgrading from an Earlier Version" in order to learn which configuration files which have changed in this release. If you have also customised those files yourself then you will need to manually re-apply your changes to the new versions.

6. **Upgrading the JON Enterprise Service Bus Plug-In**

   If one is using the JBoss Operations Network software to monitor or manage one's SOA-P server, then please ensure that the correct version of the the Enterprise Service Bus plug-in has been installed on the JON Server before restarting the SOA Platform server. Refer to *Section 1.5, " UPDATED The JBoss Operations Network Enterprise Service Bus Plug-In "* for directions.

7. **Restart the SOA-P Server**
   Now, start the upgraded JBoss Enterprise SOA-P server.

   If the JBoss Operations Network program is being used, one may have to import the SOA-P server resource again. Refer to *Section 1.5.1, " Importing SOA-P Server Resources "* for more details.

1.5. *UPDATED* The JBoss Operations Network Enterprise Service Bus Plug-In

If the JBoss Operations Network software is being used to manage or monitor one's JBoss Enterprise SOA-P server, then one needs to take steps to ensure that the correct version of the ESB JON plug-in is installed on the JON server.

The updated JON ESB plug-ins are available for download, along with the JBoss Enterprise SOA-P products, at the Red Hat JBoss Customer Support Portal at this web address: *https://support.redhat.com/jbossnetwork/*.

Procedure 1.7. Updating the ESB Plug-In for the JBoss Operations Network

1. **Download Files**
   The updated plug-in packages are available from here: *https://support.redhat.com/jbossnetwork/*. On this site, one can also find the JBoss Enterprise SOA-P version with which each one is supported.

   The particular plug-ins package one requires is dictated by the version of the JBoss Operations Network software being used.

   • For JON 2.1.2, download **`rhq-jbossesb-soa-plugin-SOA.4.3.0.GA_CP02.zip`**.

   • For JON 2.3, download **`jon-plugin-pack-soa-2.3.0.GA.zip`**.

2. **Verify the Integrity of the Downloaded File**
   Verify the downloaded file as per the directions in *Section 1.3, " Installing the Service-Oriented Architecture Platform "*.

3. **Stop the JBoss Operations Network Server**
   Ensure that the JON server is halted. If it is operating as a part of an HA cluster, then only one server needs to be halted in order to perform this task.

4. **Replace the Plug-Ins**
   Extract the plug-in **`JAR`** files from the downloaded package and copy them to the **`plug-ins`** directory on the JON server. The location of this directory will depend how one's JON server has been installed and configured.

   If only the JON server has been decompressed and one has not yet gone through the installation and configuration steps, then the directory to which the plug-ins should be copied is: **`jbossas/server/default/deploy/rhq-ear.rej/rhq-downloads/rhq-plugins/`**

If one's JON server has already been installed and configured, then the directory will be: **`jbossas/server/default/deploy/rhq-ear/rhq-downloads/rhq-plugins/`**

If there are old versions of the plug-ins in this directory, one must also delete these. The older plug-ins have the following file names: **`rhq-jbossesb-plugin-SOA.4.3.0.CR1.jar`** and **`rhq-jbossesb-soa-plugin-SOA.4.3.0.GA_CP01.jar`**.

5. **Restart**
Now that the plug-in has been updated, the JON server can be re-activated.

   A *JON Agent* will update and receive the new plug-ins when one of the following occur:

   - The JON agent is restarted.
   - The **`plugins update`** command is run from the agent's command line. This can only occur if the agent is being run in "console" mode.
   - The *RHQ Agent Resource* **`Update Plugins`** command is launched from the JON Server interface. This operation can be performed on a group of agent resources.

Important

If you are upgrading your SOA-P server from 4.3.GA you may need to import the SOA-Platform server resource again after it has been restarted. Refer to *Section 1.5.1, " Importing SOA-P Server Resources "* for more details.

1.5.1. Importing SOA-P Server Resources

If one upgrades the ESB plug-in whilst one is installing the new JBoss Enterprise SOA-P server, the latter may have to be imported into the *JON Inventory* again. This will depend on whether or not installed the upgraded server is located within a directory of the same name as the previous version.

If the upgraded SOA-P server was installed in a directory using the same directory name as the old one, there is no need to import it again. However, the resource name of the server will still be the same as that of the old version.

If, however, it is in a directory with a name that differs from that of the old one, then that new directory must be imported. The server will appear in the *Auto-Discovery Pane* of the *JON Dashboard* with the resource name equal to that of the new version. The original server resource will still appear in the JON Inventory with the original name and both of them will be updated in JON as new statistics are collected. This is because both of them will have the same *JNP* uniform resource locator specified.

If one has no use for the historical data collected by the old server resource, one can simply remove it from the JON Inventory.

Post-Installation: Configuring and Starting the Server

This chapter covers post-installation tasks, including the configuration and starting of the JBoss Enterprise Service-Oriented Architecture Platform.

2.1. Post-Installation Configuration

2.1.1. Changing the Database Used by the Platform

In the **`<install-directory>/jboss-as/tools/schema`** directory, one will find a set of scripts that can be employed to reconfigure the main components of the Platform (including the *Management* and *Monitoring* consoles), to use the (supported) database of one's choice.

There are a number of caveats to this:

- The scripts expect to find the platform utilizing one of the provided configurations. (They can be run again to change the configuration unless additional changes have been made.)
- Note that the *ESB Management Console* is provided as a *technology preview only* at this point in time.

One can reconfigure the system manually by typing **`ant`** whilst in the **`schema`** directory, or, automatically, by editing the **`build.properties`** file. Instructions to configure **`build.properties`** are included as comments within the file itself.

After the database has been re-configured, one must manually re-deploy the *ESB Management Console*. To do so:

1. Edit the **`jboss-as/tools/console/management-esb/build.xml`** file and change the value for **`org.jboss.esb.server.config`** to `production` (or `development` if one is utilizing a **`development`** profile.)
2. Run **`ant deploy`**

Important

Note that the *Hypersonic SQL* is included as the default database solely for the purposes of evaluation and development use. It neither *not* recommended nor supported in production environments.

Please refer to the *Release Notes* document for further information relating to the matter of database configuration.

2.1.2. Creating a Development Profile

Whilst testing or developing, one may wish to run the JBoss SOA-P server sans full production capabilities. For example, one may wish to have clustering disabled. The SOA-P includes a configuration profile called *default* which is suitable for such situations. One can either use this profile, or develop a customised one based upon it.

```
cd /opt/jboss-soa-p.4.3.0/jboss-as/server/
cp -rf default development
cd /opt/jboss-soa-p.4.3.0/jboss-as/bin/
./run.sh -c development
```

Example 2.1. Creating and running a new profile based upon the *default* one in Linux/Unix environment

```
cd c:\jboss-soa-p.4.3.0\jboss-as\server\
xcopy default development /E
cd c:\jboss-soa-p.4.3.0\jboss-as\bin\
run.bat -c development
```

Example 2.2. Creating and running a new profile based upon the *default* one in a Windows environment

2.1.3. Securing One's Server in a Production Environment

2.1.3.1. Securing the jBPM Console

Two distinctly different **`jbpm-console.war`** archive files are shipped with the SOA-P. The first of these files is a development version. It allows those without authenticated access to deploy processes to the server. This is for use with a graphical process design tool (such as *JBoss Developer Studio*), whilst one is developing applications. The other file is a production version, which one can use to secure the console against remote deployment. Note that one should not run one's server in a production environment with the unsecured development version of **`jbpm-console.war`** deployed. Doing so poses a threat to security.

The Stand-Alone Version of the JBoss Enterprise SOA-P

By default, Red Hat ship the unsecured up-load console in the stand-alone version of the SOA-P. The JBoss Business Processor Manager *JPDL* is able to deploy processes. Before putting it into a production environment, one should secure the console.

Procedure 2.1. Securing the Console in the Stand-Alone Version of the Platform:

- Copy the **`tools/resources/jbpm-console-production.war`** file to **`server/default/deploy/jbpm.esb/jbpm-console.war`**.

Procedure 2.2. Now, update the firewall configuration, so that these new configurations are taken into account. Enabling Remote Deployment of Processes in the Stand-Alone Version of the Platform:

- Copy the **`tools/resources/jbpm-console-development.war`** file to **`server/default/deploy/jbpm.esb/jbpm-console.war`**.

In both of these scenarios, the file must be overwritten. One cannot have two versions of the **`WAR`** archive in the deployment directory.

The Embedded JBoss EAP version of the Service-Oriented Architecture Platform

With regard to the embedded JBoss *Enterprise Application Platform* (EAP) version of the platform, the development **`WAR`** file is contained within the "**`All`**" profile, whilst the **`production`** one is within the profile of that same name. By default, one's server is configured to operate in a secure mode. To enable the development mode, one needs to run it unsecured.

Procedure 2.3. To Secure the Embedded Enterprise Application Platform Version's Console

- Start the server either without any command line parameters or with the parameter **`-c production`**

Procedure 2.4. To Enable the Remote Deployment of Processes in the Embedded Enterprise Application Platform Version

- Start the server using the parameter "**`-c all`**"

Red Hat *does not recommend* running the server on an unsecured network with the **`jbpm-console-development.war`** file deployed, nor should you use the **`all`** profile in a production environment without modifying it appropriately.

2.1.3.2. Preventing the Download of Non-RMI Classes on Port 8083 Whilst Using the Stand-Alone Version of the Server

If one uses *Remote Method Invocation* (RMI), one needs to allow the client access to port 8083 of the server. The default configuration of the Enterprise Application Platform SOA-P server permits the download of EJB classes only. The *stand-alone*SOA-P is configured to serve all of the deployable classes on this port [1].

This behaviour is controlled by the following line in the **`default/conf/jboss-service.xml`** file:

```
<!-- Should non-EJB .class files be downloadable -->
<attribute name="DownloadServerClasses">false</attribute>
```

Example 2.3. Configuration setting to allow the download of non-EJB classes

In a production environment, one must set this value to `false`.

2.2. Starting the JBoss Enterprise Service-Oriented Architecture Platform

Starting the JBoss Enterprise SOA-P is straightforward. (See *Section 2.3, " Running This Release Alongside an Earlier Version of the Enterprise Application Platform "* for specific instructions with regard to using the server with an earlier version of the JBoss Enterprise Application Platform.)

"Default" Server Profile

There are several different server profiles available for the embedded Enterprise Application Platform version of the server but there is only one for the stand-alone version. The default server profile is that

[1] This allows the Quick Starts to function correctly by default.

which will be used when the server is started without any command line parameters. This profile has the Enterprise Service Bus functionality enabled.

In the case of the stand-alone version, the default profile is named, as one would expect, "**`default`**."

In the case of the embedded Enterprise Application Platform version of the software, there is a profile named **`default`**, but this is not actually the default profile. Rather, when the embedded EAP server is started without any additional parameters, the profile named **`production`** is activated. Please refer to *Section 2.1.3, “ Securing One's Server in a Production Environment ”* for more information about this profile.

Procedure 2.5. Starting the Server

1. Navigate to the JBoss Enterprise SOA-P installation directory.

2. Change to the **`jboss-as/bin`** sub-directory.

3. If one is using Microsoft Windows, then issue the command **`run.bat`**. On Linux or Unix-derived systems, use the command **`./run.sh`**. This will start the server.

The SOA-P will now launch. Depending on if one is running the stand-alone or EAP or versions of the program one will be running either the **`default`** or the **`production`** profile, respectively. In either case, this is the profile with which the Enterprise Service Bus is deployed. The server should now by accessible at this URL: *http://localhost:8080*. Check it by attempting to load this address in a web browser.

If one desires to start the server with a different profile, simply use the start-up command shown above followed by this parameter: **`-c <profile name>`**.

2.2.1. Enabling Access to the Server Consoles

Access to the server consoles is disabled in the default configuration. To grant access, one must edit the following two files: **`soa-users.properties`** and **`soa-roles.properties`**. These files reside in the **`conf/props`** directory for the server profile for which one wishes grant access.

The **`soa-users.properties`** file contains a list of users and their passwords. These are stored in plain text. The syntax takes the form of `username=password`.

The **`soa-roles.properties`** file contains a list of users and the server roles to which they have been assigned. The syntax takes the form of **`username=role1,role2,role3`**. (Note that any number of roles can be assigned.)

These user and role details do not correspond to any other accounts, (such as operating system user accounts.) One can create new user accounts here on an arbitrary basis.

Procedure 2.6. Enabling Access to the Server Consoles

1. One either needs to add the required user name and password to the **`soa-users.properties`** file, or enable the user "`admin`" by un-commenting the line containing that name within the file. (If one chooses to enable the `admin` user account, one should also change the password with which it is associated.

```
#admin=admin
harold=@dm1nU53r
```

Example 2.4. A new user added in **`soa-users.properties`**

2. Having done that, one must also add the user to the **`soa-roles.properties`** file. The roles to which the user must be assigned to gain server console access are `JBossAdmin`, `HttpInvoker`, `user` and `admin`.

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

   Example 2.5. Assigning user roles in **`soa-roles.properties`**file

2.2.2. Troubleshooting the Start-Up Process

Here are a few common problems, accompanied by suggested solutions.

`JAVA_HOME` Set Incorrectly

An incorrectly-set environmental variable named `JAVA_HOME` will result in the server failing to start. A related error message will be displayed onscreen. To resolve this problem, one must set the `JAVA_HOME` variable to point to one's Java Development Kit installation.

VM Cannot Allocate Sufficient Memory

This error occurs when there is not enough free memory available to the system to satisfy the resource requirements of the SOA-P. One will need to increase the amount available in one of three ways: by exiting applications, allocating more virtual memory, or physically increasing the amount of RAM installed on the system.

2.3. Running This Release Alongside an Earlier Version of the Enterprise Application Platform

This release of the JBoss Enterprise SOA-P can run alongside an earlier version of Enterprise Application Platform. One may want to run them together if one has applications that depend upon a different version of the EAP than that which is included in the embedded release.

If one find's oneself in in that situation, two options are available:

1. Firstly, one can *multi-home* one's network card. This term means to configure it so that it can handle multiple Internet Protocol (IP) addresses. Once this is done, one can launch each instance of the EAP and use the "**`-b`**" command line option to bind them to different IP addresses:

```
./run.sh -b {ip-address or host}
```

Example 2.6. Using the -b option to bind to an IP or hostname in Linux/Unix

```
run.bat -b {ip-address or host}
```

Example 2.7. Using the -b option to bind to an IP or hostname in Windows

2. Alternatively, one may use the *Service Bindings Manager* to configure a different port for each server instance.

   This does not require changes to the configuration of the host operating environment or hardware. However, it will require one to change one's firewall rules.

   Procedure 2.7. Enabling the Service Binding Manager

   1. Open the **`jboss-service.xml`** file in a text editor of choice.

      This file is located in either the **`jboss-as/server/production/conf/`** directory if one is using the embedded JBoss Enterprise Application Platform or in the **`default/conf/`** directory if one is using the stand-alone version of the platform.

   2. Remove the comments the `Service Binding` section of the file and select a `ServerName` value from **`docs/examples/binding-manager/sample-bindings.xml`** (This could, for instance, be `ports-01` or `ports-02`). Alternatively, one could make one's own named-port configuration.

   3. Now, update the firewall configuration, so that these new configurations are taken into account.

Starting Out

3.1. *Updated.* Starting Out With the JBoss Enterprise Service Bus

The quickest way with which to become accustomed to the JBoss Enterprise Service Bus is by running one of the "Quick Starts" in the **`samples/quickstarts`** directory. This has an added advantage, as it will allow one to perform a basic validation of one's' system, the minimum requirements for which are as follows:

- **`JDK 5`** (v1.5.0_06 recommended)
- **`Ant`** (v1.6.5 recommended)
- **`JBoss`** Application Server 4.2.x.GA or **`JBoss`** ESB Server 4.5.GA

Note

Many of the Quick Starts are dependent upon services that may, possibly, not be deployed on one's SOA-P system. This is a concern primarily if one is using the "minimal" configuration. The "Known Issues" section of the *Release Notes* contains specific information related to this problem.

There are three ways in which to run the JBoss Enterprise Service Bus. One can deploy it to either the *JBoss Application Server* or the JBoss ESB Server, or one can run it in stand-alone mode. This guide concentrates on the JBoss Application Server and JBoss ESB Server scenarios because they are those which are most common and provide the largest range of functionality to users. Red Hat recommends that one uses the JBoss ESB Server as one follows this *Guide*.

Note that the ESB components can also be deployed directly to the JBoss Application Server. If one requires an **`EAR`** or EJB3 deployment, then the JBoss Application Server will be the preferable choice. Below, one can read the steps to follow in order to install the JBoss ESB on the Application Server. (If one plans to use the JBoss ESB Server, no additional installation is required.)

The JBoss Enterprise Service Bus Server functions as a convenient and lightweight container to which one can deploy one's WAR and SAR files. However, as it is a "stripped-down" version of the JBoss Application Server, it does not contain EJB3 libraries or "deployers." The main advantage of using the ESB Server over the JBoss ESB 4.4 GA Server is that it has a much quicker boot time, which one may well find advantageous during the development process.

3.2. Downloading

This document assumes one already has **`Ant`** (1.6.5 or higher) and Java 5 installed on one's machine, along with a fresh copy of the JBoss Application Server. Now download the JBoss ESB 4.4 GA distribution from *http://labs.jboss.com/portal/jbossesb/downloads*. There are three corresponding distribution versions: **`jbossesb-server-{version}`**, **`jbossesb-{version}`** and **`jbossesb-{version}-src`**. The execution of the stand-alone **`JBossESB`** Server execution requires the **`jbossesb-server-{version}`** distribution whilst the execution of the deployed version requires the **`jbossesb-{version}`** distribution.

Hence, if one wishes to use the JBoss Application Server, download the ESB 4.4 GA release from the above URL. Then, download the JBoss Application Server 4.2.2.GA from *http://labs.jboss.com/portal/jbossas/downloads*. (Starting from version 5.1.0.GA, the JBoss AS 5 is supported. It can be downloaded from the same location.)

3.3. Installing the JBoss Application Server

Note that this procedure is not required for the ESB Server.

1. Use **`jbossesb-{version}/install/deployment.properties.example`** as a template to create the **`install.deployment.properties`** file.

2. Open the **`install/deployment.properties`** file in a text editor and amend the following lines if need be, in order to represent both the directory in which one has installed one's **`JBoss`** Application Server and the configuration profile selected. (Most users choose "default" to be their configuration setting.) Here is an example:

```
# application server root directory
                org.jboss.esb.server.home=/jboss-4.2.2.GA
                # the instance of jboss you are running(default)
                org.jboss.esb.server.config=default
```

3. Now, from the **`install`** directory, run the 'ant' command (default target). This will deploy the JBoss Enterprise Service Bus one's JBoss AS instance. The program copies several **`.sar`** and **`.esb`** archive files (**`jbossesb.esb`**, **`jbpm.esb`**, **`jbrules.esb`**, **`smooks.esb`**, **`spring.esb`**, **`soap.esb`**) into the application server's deploy directory.

4. Next, one must start the server. There is a **`bin`** directory within the JBoss Application Server (or the JBoss ESB Server, depending upon which one has chosen to use.) Execute the "run" script (**`run.sh`** on Linux/MacOS X/Unix or **`run.bat`** on Windows).

5. Verify that the chosen server is, indeed, running by visiting *http://localhost:8080* with one's web browser.

3.3.1. *Updated.* Installation to the JBoss Application Server 5.1.0.GA

Installation steps:

1. Download the JBoss AS 5.1.0.GA and decompress it.

2. Follow the same steps as listed above for “Installation to the JBoss Application Server (not required for ESB Server).”

Depending on whether one has access certain application (like the *JOPR* console) one may need to increase the amount of memory when starting the server. This is altered from the **`run.conf`** file:

```
-Xms128m -Xmx512m -XX:PermSize=200m -XX:MaxPermSize=500m
```

Example 3.1. Configuring the **`run.conf`** File

3.3.2. *Updated.* Update to the Windows Script for JBoss AS 5

The **`run.bat`** script contains an error which prevents "**`wise`**" from compiling the web service proxies. In order to fix this issue, one must edit the **`run.bat`** script. Firstly, open the file in one's text editor of choice and then locate the following lines:

```
if "x%JAVAC_JAR%" == "x" (
    set "RUNJAR=%JAVAC_JAR%;%JBOSS_HOME%\bin\run.jar"
  ) else (
    set "RUNJAR=%JBOSS_HOME%\bin\run.jar"
  )
```

Example 3.2. Lines to Be Modified in the **`run.bat`** File

Now, switch the two "set" statements so that they read as follows:

```
if "x%JAVAC_JAR%" == "x" (
    set "RUNJAR=%JBOSS_HOME%\bin\run.jar"
  ) else (
    set "RUNJAR=%JAVAC_JAR%;%JBOSS_HOME%\bin\run.jar"
  )
```

Example 3.3. Rectified **`run.bat`** File

The batch file should now run correctly.

3.3.3. *Updated.* Scoped Deployments in the JBoss AS 5

In Enterprise Service Bus 4.x, one could specify that a deployment be scoped by configuring this option in the **`deployment.xml`** file:

```
<jbossesb-deployment>
 <depends>jboss.esb:test=server</depends>
 <depends>jboss.esb.qa.junit.destination:service=Queue,name=esb_gateway_channel_versioned<
depends>
 <depends>jboss.esb.qa.junit.destination:service=Queue,name=esb_channel_versioned</
depends>
 <loader-repository>
  org.jboss.soa.esb:loader=simple-scoped
  <loader-repository-config>
   java2ParentDelegaton=false
 </loader-repository-config>
 </loader-repository>
</jbossesb-deployment>
```

Example 3.4. Configuring the **`deployment.xml`** File

However, the AS5 "deployers" ignore this section of the **`deployment.xml`** file. Instead, users themselves are required to create a **`META-INF/jboss-classloading.xml`** file for an ESB 4.x scoped deployment. Here is an example of this:

```
<classloading xmlns="urn:jboss:classloading:1.0" domain="simple-scoped"
 parent-first="false" />
```

Example 3.5. Configuring the **`classloading.xml`** File

Finally, when one uses Java 6, ensure that the following **`JAR`** archive files are available in the **`(server_home)/lib/endorsed`** directory:

1. **`jaxb-api.jar`**
2. **`jbossws-native-jaxrpc.jar`**
3. **`jbossws-native-jaxws-.jar`**
4. **`jbossws-native-jaxws-ext.jar`**
5. **`bossws-native-saaj.jar`**

3.3.4. ESB Archive Deployment Strategies

The JBoss Enterprise Service Bus is packaged and shipped with a number of base services. A service should be deployed as an *ESB archive*, which will consist of both an action code and configuration data. Conceptually, the idea of an ESB archive is that it is a *deployable service unit*. An ESB archive is simply a **`ZIP`** file with an **`.esb`** extension. One may deploy as many ESB archives as one wishes. One can even influence the order in which the archives are deployed by configuring this functionality via settings found in the **`deployment.xml`** file. There is a section in this file in which one can specify start-order dependencies.

Typically, one would deploy an ESB archive to the **`deploy`** directory. As the data is bundled in ZIP files, one can move services between servers very simply, just by shifting the corresponding ESB archive.

- The **`jboss-esb.xml`** file contains both the "service" (listener and action) and "provider" configurations.
- The **`deployment.xml`** file is optional. One should consider using it for two reasons:
  - Firstly, to make this **`.esb`** archive depend on other archives, by specifying the class-loading order.
  - Secondly, to make the deployment of this **`.esb`** archive "scoped."
- <**`java`** classes> These are the customised action classes in a standard package hierarchy.
- <jars> These are the additional **`jar`** archives upon which one's actions depend.
- The **`queue-service.xml`** file: if the providers section of the **`jboss-esb.xml`** contains references to either queues or topics, one can deploy their configurations in the ESB archive. Note that this is strictly a convenience and the other ways to deploy these queues equally acceptable. However, Red Hat recommend this approach, in order that one can keep one's deployments as self-contained as possible, thereby keeping dependency management simple.

The JBoss Enterprise Service Bus ships with a number of standard service archives:

- **`jbossesb.esb`** - This is used for internal services, an example of which is the `DeadLetterService`.
- **`jbrules.esb`** - This file is needed by services like the *Content-Based Router* for rules evaluation. It contains the `JBossRules` functionality.
- **`jbpm.esb`** - This is needed as it is the default provider of Business Process Management (jBPM.)
- **`smooks.esb`** - This file is the default "message transformation engine," *Smooks*.

These services are deployed by default but one is able to remove those that are not needed, if one so desires.

Quick Starts

4.1. The "`Hello World`" Quick Start

This Quick Start allows one to start running the JBoss SOA-P "out of the box." It is located in the distribution in this directory: **`samples/quickstarts/helloworld`**.

In order to run this Quick Start, one must:

1. **Set Quick Start Properties:**
   Ensure that the ***`${SOA_ROOT}`*****`/samples/quickstart/conf/quickstarts.properties`** file has the correct configuration and home directory settings for one's server. (This configuration file is used by all of the Quick Starts.) Here is sample data:

   ```
   # Location of your JBoss Application Server installation.
   org.jboss.esb.server.home=/opt/jboss-soa-p.4.3.0/jboss-as
   # JBossAS server name.
   org.jboss.esb.server.config=default
   ```

   Example 4.1. Quick Start Deployment Properties

2. **Start the Server:**
   Manually start the server using the start-up scripts as described in *Section 2.2, " Starting the JBoss Enterprise Service-Oriented Architecture Platform "*.

   If one is running the server on a Microsoft Windows operating system, one should not run it as a "**`Windows Service`**" whilst using the examples in this *Guide*. That is because one needs to be able to view the console output.

3. **Build and Deploy the Quick Start:**
   Launch a terminal window and, from the shell, navigate to the ***`${SOA_ROOT}`*****`/samples/quickstarts/helloworld`** directory.

   Run the **`ant deploy`** command. This builds the **`helloworld.esb`** archive and deploys it onto one's application server.

4. **Invoke the Deployed ESB:**
   Run the **`ant runtest`** command. This sends a JMS message to the Enterprise Service Bus service that was deployed in the previous step.

5. **Check the Server Console for the Result:**
   One should now see a

   ```
   Hello
                       World
   ```

   message appear on one's application server's console. It will look like this:

```
17:25:12,614 INFO [STDOUT] &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
17:25:12,615 INFO [STDOUT] Body: Hello World
17:25:12,615 INFO [STDOUT] &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
17:25:12,615 INFO [STDOUT] Message structure:
17:25:12,615 INFO [STDOUT] [ message: [ JBOSS_XML ]
```

Example 4.2. **`Hello World`** Quick Start console output

Note that, depending on one's server configuration, it is possible that it may not appear in the console. If so, check that it exists in the server log. The log is located at the following filepath address: **`${SOA_ROOT}/server/${CONFIG}/log/server.log`**

That is all one needs to do. The Quick Start has run successfully and the environment has been properly configured.

One can learn more about each Quick Start example, including detailed instructions, by running the **`ant help-quickstarts`** command in the directory of each specific Quick Start. Alternatively, one can consult the Quick Start's **`readme.txt`** file. Also, further information about how to run a particular Quick Start under different deployment scenarios can be obtained by running the **`ant help`** command in the directory of that specific Quick Start.

4.2. *Updated.* The Components of a Quick Start

The following diagram illustrates the sequence of events that take place when a Quick Start is executed. This is a useful learning aid as it depicts a number of the key concepts within the JBoss Service-Oriented Architecture Platform.

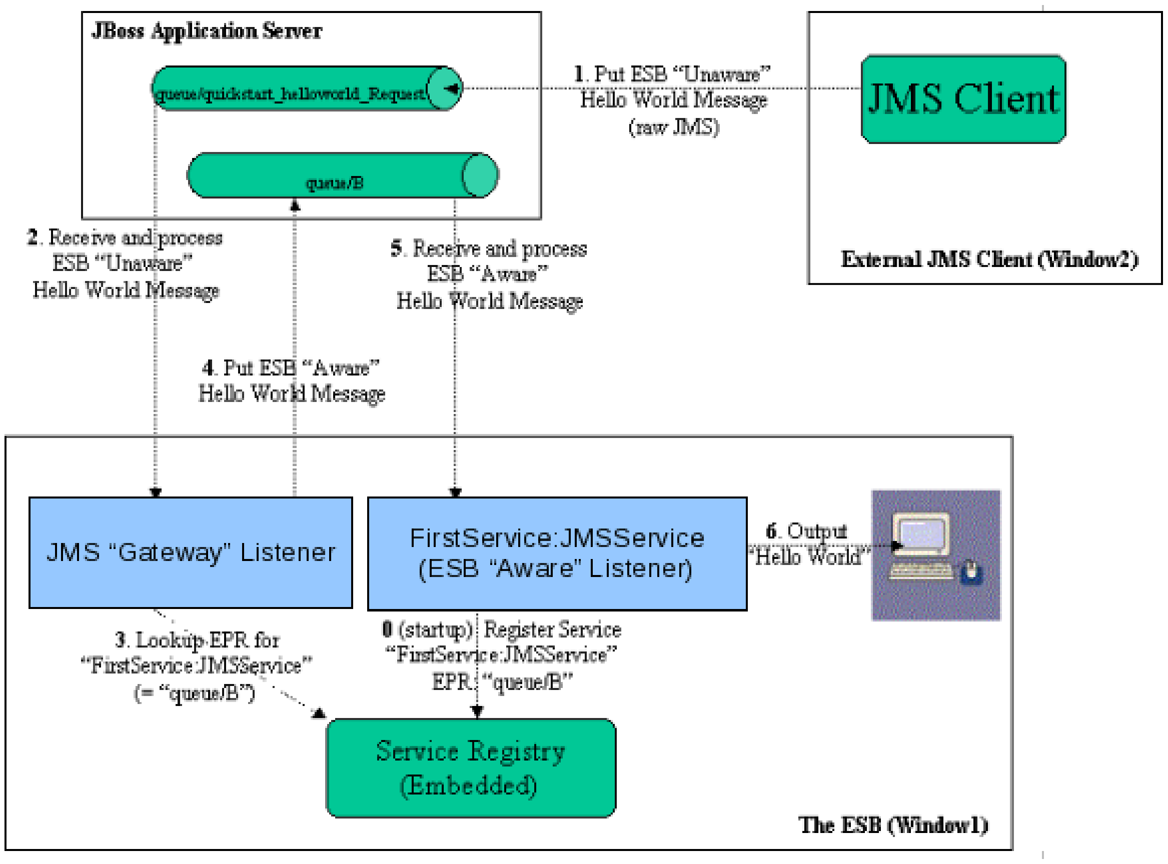
JBoss Application Server
queue/quickstart_helloworld_Request
queue/B
1. Put ESB "Unaware" Hello World Message (raw JMS)
JMS Client
External JMS Client (Window2)
2. Receive and process ESB "Unaware" Hello World Message
5. Receive and process ESB "Aware" Hello World Message
4. Put ESB "Aware" Hello World Message
JMS "Gateway" Listener
FirstService:JMSService (ESB "Aware" Listener)
6. Output "Hello World"
3. Lookup EPR for "FirstService:JMSService" (= "queue/B")
0 (startup): Register Service "FirstService:JMSService" EPR: "queue/B"
Service Registry (Embedded)
The ESB (Window1)

Figure 4.1. Quick Start Components and Sequence of Events

Service Registry

This is a *JAXR Registry* implementation. In this Quick Start, the registry uses Remote Method Invocation-based communications. (Note that the JBoss SOA-P *Services Guide* contains more details on the Registry Service.)

JMS Gateway Listener

A *Gateway Listener* is one of the key architectural components of the JBoss SOA-P. This type of listener provides an interface between the SOA-P and external service End Points. In this case, one will be using a Java Message Service (JMS) Gateway.

The ESB-Aware Service Listener

The **`FirstService:ESB:SimpleListener" ESB-Aware Service Listener`** listens for "ESB-Aware" messages on "**`queue/quickstart_helloworld_Request_esb`**." This introduces one to the concepts of ESB "*Aware*" and "*Unaware*" messages which shall be examined in more detail in the next section.

4.2.1. ESB-Aware and Unaware Messages

Messages passed between components within the JBoss SOA-P *Enterprise Message Bus* are expected to comply with the definition of a message as detailed in the **`xml/message.xsd`** file. This well-defined concept is often referred to as an *ESB Message*[1] .

By complying with this definition, messages facilitate easy and effective communication between services within the SOA-P domain.

However, service end-points that reside outside of the Enterprise Service Bus are not expected to be aware of these requirements, nor comply with them. They are referred to as *ESB unaware* or *non-ESB Endpoints* or Messages. A gateway listener provides a translation layer for messages passing between components that are ESB-aware and unaware.

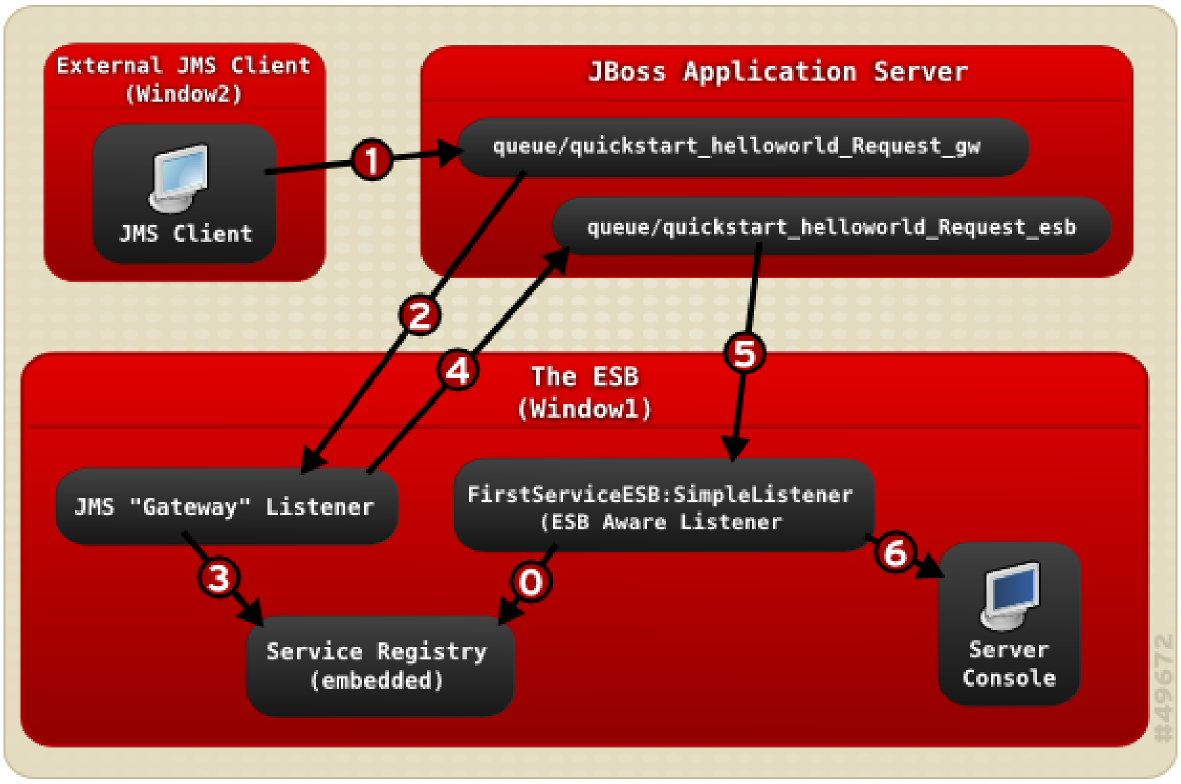

Figure 4.2. Sequence of Events for Quick Start

4.3. *Updated.* `Helloworld` Quick Start Sequence of Events

After starting the **`JBoss`**SOA-P Server in Window1 and prior to any "Hello World" messages being deployed to the Enterprise Service Bus, the “**`FirstServiceESB:SimpleListener`**“ Service is added to the Registry Service.

The sequence of events in the **`HelloWorld`** Quick Start are as follows:

- An ESB-Unaware Java Message Service Client end-point puts an ESB-Unaware "Hello World" Message (a plain String Object) into the JMS Queue at "queue/quickstart_helloworld_Request_gw".

[1] This topic is more fully discussed in the SOA-P*Programmers' Guide*.

- Next, the JMS Gateway Listener receives the ESB-Unaware message. The Gateway's role is to adapt this message by making it ESB-Aware. Once done, this will mean that it can be processed by an ESB-Aware end-point.
- The JMS Gateway Listener uses the registry to find the *Endpoint Reference* (EPR) for “`FirstServiceESB:SimpleListener`” service. This is derived to be JMS Queue “**`queue/quickstart_helloworld_Request_esb`**.”
- The JMS Gateway Listener "adapts" the message into one which is ESB-Aware and places it into the JMS Queue, which resides in the “**`queue/quickstart_helloworld_Request_esb`**” file.
- Next, the “`FirstServiceESB:SimpleListener`” service receives the message.
- Finally, the “`FirstServiceESB:SimpleListener`” service extracts the payload from the message and outputs it on the console.

4.4. *Updated.* Running other Quick Starts

Once one has successfully run the "Hello World" Quick Start and grasped an understanding of the concepts involved in its execution, one can try some of the other Quick Starts.

Important

Please note that each of the Quick Starts have different requirements. These are documented in their respective **`readme.txt`** files. It is also important to realise that not all of the Quick Starts will run on every server configuration.

There are several dozen Quick Starts included. Below is a list which suggests which ones should be studied first.

- helloworld
- helloworld_action
- custom_action
- helloworld_file_action
- helloworld_ftp_action
- simple_cbr
- fun_cbr
- business_service
- business_rules_service
- scripting_groovy
- transform_CSV2XML
- transform_XML2POJO
- transform_XML2XML_simple

- transform_XML2XML_date_manipulation
- aggregator
- bpm_orchestration1
- bpm_orchestration2
- webservice_consumer1
- webservice_producer

Warning

The "**`groovy_gateway`**" Quick Start does not work when the server is running in "*headless*" mode. The Service-Oriented Architecture Platform runs in this mode by default due to the requirements of other components.

One can learn more information about this issue by reading the "Known Issues" section of the SOA-P*Release Notes* or the online bug report at *http://jira.jboss.org/jira/browse/SOA-906*.

Appendix A. Revision History

Revision 3.0	Mon 12 Oct 2009	David Le Sage *dlesage@redhat.com*

Updated for Release 5.0

Revision 1.2	Fri 24 Jul 2009	Darrin Mison *dmison@redhat.com*

Updated for 4.3 CP02
SOA-1151 - Added details of certified and compatible configurations. Section 1.1.2
Some configuration information has been moved to this guide from the Release Notes. Section 2.1
SOA-1489 - JON ESB plugin directions have been updated. Section 1.6

Revision 1.0	Wed 10 Sep 2008	Darrin Mison *dmison@redhat.com*

Created

Index