Administration Guide

For JBoss ESB administrators

by JBoss ESB Development Team with Community Contributions

Configuration

1.1. Stand-Alone Server

If you wish to run the JBossESB server on the same machine as JBossAS, then you should look at the *Configuring Multiple JBoss Instances On One Machine* [http://www.jboss.org/community/wiki/ConfiguringMultipleJBossInstancesOnOnemachine] Wiki page.

1.2. JBossESB JMS Providers

The JBossESB supports a number of JMS providers. Currently we have successfully tested JBoss Messaging, JBossMQ, ActiveMQ and Websphere MQ Series (version 5.3 and 6.0). We recommend JBoss Messaging. At this time we know of no reasons why other JMS implementations should not also work, but have not been able to verify this.

Warning

This section is not intended as a replacement for the configuration documentation that comes with the supported JMS implementations. For advanced capabilities, such as clustering and management, you should consult that documentation as well.

How can I configure them? . JMSListeners and JMSGateways can be configured to listen to a Queue or Topic. For this you can use set the following parameters in the service configuration file: jndi-URL, jndi-context-factory, jndi-pkg-prefix, connection-factory, destination-type and destination-name. Furthermore you will need to add the client jms jars of the JMS-provider you want to use to the classpath.

Important

In the following sections we will assume that your JMS provider runs on 'localhost', that the connection-factory is 'ConnectionFactory', that we are listenening to a destination-type 'queue' and that it's name is 'queue/A'.

Note

Each JMSListener and JMSGateway can be configured to use it's own JMS provider, so you can use more then one provider in your deployment.

When using JMS, JBossESB utilizes a connection pool to improve performance. By default the size of this pool is set to 20, but can be over-ridden by setting the org.jboss.soa.esb.jms.connectionPool property in the transports section of the JBossESB configuration file. Likewise, if a session cannot be obtained initially, JBossESB will keep retrying for up to 30 seconds before giving up. This time can be configured using the org.jboss.soa.esb.jms.sessionSleep property.

1.2.1. Max Sessions Per Connection

The JBoss ESB's JmsConnectionPool pools JMS Sessions and is used by all JMS based components – JMS Listeners, JMS Couriers, JMS Router etc.

Some JMS providers limit the number of JMS Sessions per connection. This means that the JMS Components in JBoss ESB need to support a control mechanism for the maximum number of sessions created from each JMS Connection managed by a single JmsConnectionPool instance. This is done simply be specifying one or both of the following properties in the JNDI configuration of the JMS Component (JMS Provider/Bus, JMSRouter etc):

- max-sessions-per-connection: This is the maximum total number of Sessions allowed per connection i.e. XA + non-XA Session instances. This value defaults to the maximum number of JMS Sessions allowed for a JmsConnectionPool as a whole, which defaults to 20 (as configured in the jbossesb-properties.xml file).
- max-xa-sessions-per-connection: This is the maximum number of XA Sessions allowed per connection i.e. XA only. This value defaults to the value of max-sessions-per-connection.

So if neither of the above parameters are configured, the JmsConnectionPool will create a single JMS Connection and create all JMS Sessions off that Connection instance.

These configurations should be made as generic property configurations on the JMS Provider configuration e.g.

```
<jms-provider ...>

    <property name="max-sessions-per-connection" value="5" />
    <property name="max-xa-sessions-per-connection" value="1" />

    <!-- And add providers.... -->

</jms-provider>
```

1.2.2. JBossMQ or JBossMessaging

The settings for JBossMQ and JBossMessaging are identical and you should set the parameters to:

```
jndi-URL="localhost”
```

```
jndi-context-factory="org.jnp.interfaces.NamingContextFactory"
connection-factory="ConnectionFactory"
destination-type="queue"
destination-name="queue/myqueue"
```

> **Note**
>
> For JBossMQ you should have jbossmq-client.jar in your classpath. Not that this jar is included in jbossall-client.jar, which can be found in lib/ext. For JBossMessaging it should be jboss-messaging-client.jar While -for now- the JBossMQ is the default JMS provider in JBossAS, you can also use JBoss Messaging. Instructions for installing JBoss Messaging can be found on the project website (*http://docs.jboss.org/jbossmessaging/docs/userguide-1.4.0.GA/html/installation.html*).

1.2.2.1. JBoss Messaging Clustering configuration

Configuring JBoss Messaging in a clustered setup gives you loadbalancing and failover for JMS. Since this capability has changed between different versions of JBoss Messaging and may continue to do so, you should consult the relevant JBoss Messaging documentation (*http://docs.jboss.org/jbossmessaging/docs/userguide-1.4.0.SP1/html_single/index.html#conf.connectionfactory.attributes.loadbalancingfactory*).

> **Note**
>
> This functionality is continually changing as **JBoss Messaging** evolves. To learn how to configure it, consult the relevant **JBoss Messaging** documentation.)

1.2.3. ActiveMQ

For ActiveMQ you should set the parameters to:

```
jndi-URL="tcp://localhost:61616”
jndi-context-factory="org.apache.activemq.jndi.ActiveMQInitialContextFactory"
connection-factory="ConnectionFactory"
destination-type="queue"
destination-name="queue/A"
```

In your classpath you should have:

```
activemq-core-4.x
```

```
backport-util-concurrent-2.1.jar
```

Both jars can be found in lib/ext/jms/activemq. We tested with version 4.1.0-incubator.

1.2.4. Websphere MQ Series

1. For Websphere MQ Series, you should set the main JNDI parameters to:

   ```
   jndi-URL="localhost:1414/SYSTEM.DEF.SVRCONN”
   jndi-context-factory="com.ibm.mq.jms.context.WMQInitialContextFactory"
   connection-factory="ConnectionFactory"
   destination-type="queue"
   destination-name="QUEUEA"
   ```

   > **Note**
   >
   > Websphere likes all CAPS queue names and no slashes (QUEUEA), and the name of the Queue Manager in MQ should match what the value of 'connection-factory' is (or bind this name to JNDI). In our case we created a Queue Manager named “ConnectionFactory”.

2. You will also need to configure the max-xa-sessions-per-connection property to a value of “1” if using XA Connections. For more on how to configure this property, see the Max Sessions Per Connection section.

3. On your classpath you should have

   - `com.ibm.mq.pcf.jar`
   - `mqcontext.jar`
   - `com.ibm.mq.jar` (client JAR)
   - `com.ibm.mqjms.jar` (client JAR)

   > **Note**
   >
   > The client jars differ between MQ 5.3 and MQ 6.0. However the 6.0 jars should be backward compatible. The jars are not open source, and are therefor not provided by us. You will have to obtain them from your WAS and MQ installs.

Warning

You may get the following exception when running MQ 6.0, which can be fixed by adding the user that runs the jbossesb to the mqm group:

```
Message: Unable to get a MQ series Queue Manager or
 Queue Connection. Reason: failed to create connection --
javax.jms. JMSSecurityException: MQJMS2013: invalid security
 authentication supplied for MQQueueManager
```

Explanation: There is a problem with user permissions or access.

Tip: Make sure the user accessing MQ Queue Manager is part of the mqm group.

1.2.5. Oracle AQ

1. For Oracle AQ you should set the parameters to:

```
connection-factory="QueueConnectionFactory"
```

2. Use the following properties:

```
<property name="java.naming.factory.initial" value="org.jboss.soa.esb.oracle.aq.AQInitialContextFactory"/
>
<property name="java.naming.oracle.aq.user"     value="<user>"/>
<property name="java.naming.oracle.aq.password" value="<pw>"/>
<property name="java.naming.oracle.aq.server"   value="<server>"/>
<property name="java.naming.oracle.aq.instance" value="<instance>"/>
<property name="java.naming.oracle.aq.schema"   value="<schema>"/>
<property name="java.naming.oracle.aq.port"     value="1521"/>
<property name="java.naming.oracle.aq.driver"   value="thin"/>
```

3. Optionally specify a database connection url:

```
<property name="java.naming.factory.initial" value="org.jboss.soa.esb.oracle.aq.AQInitialContextFactory"/
>
<property name="java.naming.oracle.aq.user"     value="<user>"/>
<property name="java.naming.oracle.aq.password" value="<pw>"/>
<property name="java.naming.oracle.aq.url" value="jdbc:oracle:thin:@(description=(address_list=(load_balance=on)
(failover=on)(address=(protocol=tcp)(host=host1)(port=1621))(address=(protocol=tcp)
(host=host2)(port=1621)))(connect_data=(service_name=SID)(failover_mode=(type=select)
(method=basic)))) "/>
```

> **Note**
>
> The above example can be used to connect to Oracle Real Application Cluster (RAC).

> **Note**
>
> You may notice the reference to the InitialContext factory. You only need this is if you want to avoid OracelAQ to register its queues with an LDAP. The AqinitialContextFactory references code in a plugin jar that you can find in the plugins/org.jboss.soa.esb.oracle.aq directory. The jar is called org.jboss.soa.esb.oracle.aq-4.2.jar and you will have to deploy it to the jbossesb.sar/lib directory.

> **Note**
>
> When creating a Queue in Oracle AQ make sure to select a payload type of SYS AQ$_JMS_MESSAGE.

> **Note**
>
> For a sample you can check the samples/quickstarts/helloworld_action/oracle-aq directory for an example jboss-esb.xml configuration file.

1.2.6. Red Hat MRG

1. For Red Hat MRG you should set the parameters to:

```
<property name="jndi-prefixes" value="connectionfactory,.destination"/>
<property name="jndi-connection-factory" value="org.apache.qpid.jndi.PropertiesFileInitialContextFactory"/>
<property name="connectionFactory.qpidConnectionFactory" value="amqp://guest:guest@clientid/virtualHost?brokerlist='tcp://localhost:5672'"/>
```

```
<property name="destination.[queueName]Queue" value="direct://amq.direct//[queueName]?routingkey=[routingkeyname]"/>
```

2. In your classpath in your classpath you should have the Apache Qpid qpid-common and qpid-client JARs.

   - `qpid-common-0.6.jar`
   - `qpid-client-0.6.jar`

   Note

   We tested with version 0.6.

1.2.7. Tibco EMS

1. For Tibco EMS you should set the parameters to:

```
jndi-URL="tcp://localhost:7222#
jndi-context-factory=#com.tibco.tibjms.naming.TibjmsInitialContextFactory"
connection-factory="QueueConnectionFactory"
destination-type="queue"
destination-name="myqueue"
```

2. In your classpath you should have the client jars that ship with Tibco EMS, which are found in the tibco/ems/clients/java dir.

   - `jaxp.jar`
   - `jndi.jar`
   - `tibcrypt.jar`
   - `tibjmsapps.jar`
   - `tibrvjms.jar`
   - `jms.jar`
   - `jta-spec1_0_1.jar`
   - `tibjmsadmin.jar`
   - `tibjms.jar`

Note

We tested with version 4.4.1.

1.2.8. Extension properties

The JNDI configuration used to retrieve the JMS resources will, by default, inherit all properties with names prefixed by “java.naming.”. Some JMS providers may, however, specify properties that use a different naming prefix.

Note

In order to support these properties we provide a mechanism through which the property prefixes can be specified for each provider, allowing properties using these additional prefixes to be inherited.

The prefixes are configured by defining the “jndi-prefixes” property on the associated jms-provider element, containing a comma separated list of the additional prefixes. The extension properties are also configured in the same location.

```
<jms-provider name="JMS" connection-factory="ConnectionFactory">
   <property name="jndi-prefixes# value="test.prefix." />
   <property name="test.prefix.extension1" value="extension1" />
   <property name="test.prefix.extension2" value="extension2" />
</jms-provider>
```

1.3. FTP Configuration

Most configuration options are set on the FTP EPR and described in the Programmers Guide. However, the following are set at the global scope in the jbossesb-properties file:

- `org.jboss.soa.esb.ftp.renameretry`

  When transmitting files via FTP, JBossESB sends them over with one file name which prevents them being processed, before renaming them in order that they can be processed. Unfortunately some FTP servers retain locks on the file during the time it is written and then renamed, preventing the rename from happening. If this happens, JBossESB will attempt to rename the file the defined number of times (default 10), sleeping in between each attempt, before finally generating an error message if the file cannot be renamed.

1.4. Database Configuration

The ESB uses a database for persisting Registry services, and the Message-Store.

Database scripts for each of these can be found under:

Message-Store: `ESB_ROOT/services/jbossesb/src/main/resources/message-store-sql`

Service Registry: The service registry is now jUDDI v 3.0.1, which does not use SQL scripts but uses the persistence layer to initialize the database schema.

A few database types and their scripts are provided, and you should be able to easily create one for your particular database (if you do, please contribute it back to us).

For the Message-Store you will need to also update the data-source setting properties in the main ESB config file `jbossesb-properties.xml`. The following are settings you will need to change, based on the connection information appropriate to your environment – these settings are found in the DBSTORE section of the file.

As long as there is script for your database the ESB will auto-create the schema's on startup. By default JBossESB is configured to use a JEE DataSource.

```
<properties name="dbstore">
   <property name="org.jboss.soa.esb.persistence.db.conn.manager"
      value="org.jboss.soa.esb.persistence.manager.J2eeConnectionManager"/>

   <!-- this property is only used if using the j2ee connection manager -->
   <property name="org.jboss.soa.esb.persistence.db.datasource.name"
      value="java:/JBossESBDS"/>
</properties>
```

When running from the standalone bootstrapper use:

```
<properties name="dbstore">
   <!--  connection manager type -->
   <property name="org.jboss.soa.esb.persistence.db.conn.manager"
      value="org.jboss.soa.esb.persistence.manager.StandaloneConnectionManager"/>

   <!-- FIXME: is this a typo ? -->
   <property name="org.jboss.soa.esb.persistence.db.conn.manager"



   <property name="org.jboss.soa.esb.persistence.db.connection.url"
      value="jdbc:hsqldb:hsql://localhost:9001/jbossesb"/>
```

```
    <property name="org.jboss.soa.esb.persistence.db.jdbc.driver"
        value="org.hsqldb.jdbcDriver"/>

    <property name="org.jboss.soa.esb.persistence.db.user" value="sa"/>

    <property name="org.jboss.soa.esb.persistence.db.pwd" value=""/>

    <property name="org.jboss.soa.esb.persistence.db.pool.initial.size" value="2"/>

    <property name="org.jboss.soa.esb.persistence.db.pool.min.size" value="2"/>

    <property name="org.jboss.soa.esb.persistence.db.pool.max.size" value="5"/>

    <property name="org.jboss.soa.esb.persistence.db.pool.test.table" value="pooltest"/>

    <property name="org.jboss.soa.esb.persistence.db.pool.timeout.millis" value="5000"/>
</properties>
```

Properties

org.jboss.soa.esb.persistence.db.conn.manager
    the db connection manager.

org.jboss.soa.esb.persistence.db.datasource.name
    The datasource name (used for JNDI lookup)

org.jboss.soa.esb.persistence.db.connection.url
    this is the db connection url for your database.

org.jboss.soa.esb.persistence.db.jdbc.driver
    JDBC Driver

org.jboss.soa.esb.persistence.db.user
    db user

org.jboss.soa.esb.persistence.db.pwd
    db password

org.jboss.soa.esb.persistence.db.pool.initial.size
    initial size of db connection pool

org.jboss.soa.esb.persistence.db.pool.min.size
    minimum size of db connection pool

org.jboss.soa.esb.persistence.db.pool.max.size
    maximum size of db connection pool

org.jboss.soa.esb.persistence.db.pool.test.table
: A table name (created dynamically by pool manager) to test for valid connections in the pool

org.jboss.soa.esb.persistence.db.pool.timeout.millis
: timeout period to wait for connection requests from pool

The Service Registry database information is contained in the `esb.juddi.xml` file. You should consult the Service Registry section of this document for more detailed information on what settings and their values and how they effect the behavior of the ESB.

JBoss server comes with a pre-installed hypersonic database (HSQLDB). The database can only be accessed in the same JVM. The data-source definition can be found in the `jbossesb.sar/message-store-ds.xml`.

> **Warning**
>
> Use of HSQLDB for production is not recommended.

1.4.1. Switching Databases

This section describes the steps to move from using the default hypersonic database to postgres. These steps should be the same for any other database. Just replace postgres with the database you want to switch to.

Procedure 1.1. Changing the database to PostgreSQL

1. Remove `deploy/hsqldb-ds.xml` and add the following in a file named `deploy/postgres-ds.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
   <local-tx-datasource>
      <jndi-name>DefaultDS</jndi-name>
      <connection-url>jdbc:postgresql://host:port/database</connection-url>
      <driver-class>org.postgresql.Driver</driver-class>
      <user-name>username</user-name>
      <password>password</password>
      <metadata>
         <type-mapping>PostgreSQL 7.2</type-mapping>
      </metadata>
      <check-valid-connection-sql>
         select count(*) from jbm_user
      </check-valid-connection-sql>
   </local-tx-datasource>
```

```
</datasources>
```

Modify the above to suite your needs, connection parameters and such. Make sure the name of the DS is the same though (DefaultDS)

2. Replace `deploy/jbossesb-registry.sar/juddi-ds.xml` with the same configuration in the previous step (change the database name if needed).
   Again make sure the keep the jndi-name(juddiDB).

3. Replace `deploy/jbossesb.esb/message-store-ds.xml` with the same configuration in step one (change the database name if needed).
   Again make sure the keep the jndi-name(JBossESBDS).

4. Replace the database name in the 'message-store-sql' element in `deploy/jbossesb.esb/jbossesb-service.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<server>
    <mbean code="org.jboss.internal.soa.esb.dependencies.DatabaseInitializer"
        name="jboss.esb:service=MessageStoreDatabaseInitializer">
        <attribute name="Datasource">java:/JBossESBDS</attribute>
        <attribute name="ExistsSql">select * from message</attribute>
        <attribute name="SqlFiles">
            message-store-sql/postgresql/create_database.sql
        </attribute>
        <depends>jboss.jca:service=DataSourceBinding,name=JBossESBDS</depends>
    </mbean>
</server>
```

5. Edit `jbossesb-registry.sar/META-INF/persistence.xml`. The hibernate.dialect property must be set to the type of database that you are using in your datasource (org.hibernate.dialect.PostgreSQLDialect for example).

6. Replace `deploy/jboss-messaging/hsqldb-persistence-service.xml` with the `postgres-persistence-service.xml` from the version of JBM that you are running.

   This needs to match the same version and might not work it the versions mismatch. These files can be found in `src/etc/server/default/deploy` of a JBM distribution.

7. Copy the database driver to the servers lib directory and fire up the server.

1.5. Using a JSR-170 Message Store

JBossESB allows for multiple message store implementations via a plugin-based architecture. As an alternative to the default database message store, a JSR-170 (Java content repository) message store may be used. The JCR implementation included with JBossESB is Apache Jackrabbit. To enable the JCR message store, add the following property to the "core" section of jbossesb-properties.xml in the root of the jboss-esb.sar (or the root of deployers/esb.deployer on AS5):

```
<property name="org.jboss.soa.esb.persistence.base.plugin.jcr"
value="org.jboss.internal.soa.esb.persistence.format.jcr.JCRMessageStorePlugin"/>
```

This adds the JCR plugin to the list of available message stores. The JCR message store can use an existing repository via JNDI or can create a standalone instance locally on the application server. The following list of properties should be added in the "dbstore" section of jbossesb-properties.xml to configure repository access:

```
<property name="org.jboss.soa.esb.persistence.jcr.jndi.path" value="jcr"/>
<property name="org.jboss.soa.esb.persistence.jcr.username" value="username"/>
<property name="org.jboss.soa.esb.persistence.jcr.password" value="password"/>
<property name="org.jboss.soa.esb.persistence.jcr.root.node.path"
   value="JBossESB/MessageStore"/>
```

Properties

jcr.jndi.path

Optional path in JNDI where the repository is found. If not specified, a new repository will be created based on the repository.xml located in the root of jbossesb.sar. In this case, repository data is stored in the JBossAS/server/{servername}/data/repository directory.

jcr.username

Username for getting a repository session

jcr.password

Password for gettging a repository session

jcr.root.node.path

The path relative to the root of the repository where messages will be stored.

An easy test for whether the JCR message store is configured properly is to add the org.jboss.soa.esb.actions.persistence.StoreJCRMessage action onto an existing service. The action will attempt to store the current message to the JCR store.

1.6. Message Tracing

It is possible to trace any and all Messages sent through JBossESB. This may be important for a number of reasons, including audit trail and debugging. In order to trace Messages you should ensure that they are uniquely identified using the MessageID field of the Message header: as mentioned in the Programmers Guide, this is the only way in which Messages can be uniquely identified within the ESB.

By default, JBossESB components (e.g., gateways, ServiceInvoker and load balancing) log all interactions with Messages through standard logger messages. Such log messages will contain the entire header information associated with the Message which will enable correlation across multiple JBossESB instances. You can identify these messages by looking for the following in your output:

```
header: [ To: EPR: PortReference < <wsa:Address ftp://foo.bar/> >, From: null, ReplyTo:
EPR: PortReference < <wsa:Address http://bar.foo/> >, FaultTo: null, Action: urn:dowork,
MessageID: urn:foo/bar/1234, RelatesTo: null ]
```

Furthermore, you can enable a logging MetaData Filter, whose only role is to issue log messages whenever a Message is either input to an ESB component, or output from it. This filter, org.jboss.internal.soa.esb.message.filter.TraceFilter, can be placed within the Filter section of the JBossESB configuration file, in conjunction with any other filters: it has no effect on the input or output Message. Whenever a Message passes through this filter, you will see the following log at info level:

```
TraceFilter.onOutput ( header: [ To: EPR: PortReference < <wsa:Address ftp://foo.bar/> >,
From: null, ReplyTo: EPR: PortReference < <wsa:Address http://bar.foo/> >, FaultTo: null,
Action: urn:dowork, MessageID: urn:foo/bar/1234, RelatesTo: null ] )

TraceFilter.onInput ( header: [ To: EPR: PortReference < <wsa:Address ftp://foo.bar/> >,
From: null, ReplyTo: EPR: PortReference < <wsa:Address http://bar.foo/> >, FaultTo: null,
Action: urn:dowork, MessageID: urn:foo/bar/1234, RelatesTo: null ] )
```

TraceFilter will only log if the property org.jboss.soa.esb.messagetrace is set to on/ON (the default setting is off/OFF). By default, if enabled it will log all Messages that pass through it. However, for finer grained control you may enable finer grained control over which Messages are logged and which are ignored. To do this make sure that the property org.jboss.soa.esb.permessagetrace is set to on/ON (the default is off/OFF). Once enabled, those Messages with a Property of org.jboss.soa.esb.message.unloggable set to yes/YES will be ignored by this filter.

1.7. Clustering and Fail-Over Support

Beginning with JBossESB 4.2, there is now support for fail-over of stateless services. You should consult the Programmers Guide for further details, but the pertinent issues to note are:

- Because ServiceInvoker hides much of the fail-over complexity from users, it necessarily only works with native ESB Messages. Furthermore, not all gateways have been modified to use the

ServiceInvoker, so incoming ESB-unaware messages to those gateway implementations may not always be able to take advantage of service fail-over.

- When the ServiceInvoker tries to deliver a message to our Service it may get a choice of potentially multiple EPRs now. In order to help it determine which one to select, you can configure a Policy. In the jbossesb-properties.xml you can set the 'org.jboss.soa.esb.loadbalancer.policy'. Right now three Policies are provided, or you can create your own.

  1. First Available. If a healthy ServiceBinding is found it will be used unless it dies, and it will move to the next EPR in the list. This Policy does not provide any load balancing between the two service instances.

  2. Round Robin. Typical Load Balance Policy where each EPR is hit in order of the list.

  3. Random Robin. Like the other Robin but then random.

- The EPR list the Policy works with may get smaller over time as dead EPRs will be removed from the (cached) list. When the list is exhausted or the time-to-live of the list cache is exceeded, the ServiceInvoker will obtain a fresh list of EPRs from the Registry. The 'org.jboss.soa.esb.registry.cache.life' can be set in the jbossesb-properties file, and is defaulted to 60,000 milliseconds. What if none of the EPRs work at the moment? This is where we may use Message Redelivery Service.

- If you would like to run the same service on more than one node in a cluster you have to wait for service registry cache revalidation before the service is fully working in the clustered environment. You can setup this cache revalidation timeout in deploy/jbossesb.sar/jbossesb-properties.xml (or deploy/esb.deployers/jbossesb-properties.xml in AS5) :

```
<properties name="core">
   <property name="org.jboss.soa.esb.registry.cache.life" value="60000"/>
   <!-- 60 seconds is the default -->
</properties>
```

- If you set the org.jboss.soa.esb.failure.detect.removeDeadEPR property to true, then whenever ServiceInvoker suspects an EPR has failed it will remove it from the Registry. The default setting is false, because this should be used with extreme care: for example, if the service represented by the EPR is simply overloaded and slow to respond then it may be excluded from future users. Therefore, if you allow ServiceInvoker to remove EPRs it is possible orphan services (ones that eventually receive no further interactions) may result and you may have to restart them.

Registry

At the heart of all JBossESB deployments is the registry. This is fully described elsewhere in the Services Guide, where configuration information is also discussed. However, it is worth noting the following:

- When services run they typically place the EPR through which they can be contacted within the registry. If they are correctly developed, then services should remove EPRs from the registry when they terminate. However, machine crashes, or incorrectly developed services, may leave stale entries within the registry that prevent the correct execution of subsequent deployments. In that case these entries may be removed manually. However, it is obviously important that you ensure the system is in a quiescent state before doing so.
- If you set the optional remove-old-service tag name in the EPR to true then the ESB will remove any existing service entry from the Registry prior to adding this new instance. However, this should be used with care, because the entire service will be removed, including all EPRs.

Configuring Web Service Integration

JBoss ESB 4.8 exposes Webservice Endpoints for through the SOAPProcessor action. This action integrates the JBoss Webservices v2.x container into JBossESB, allowing you to invoke JBossWS Endpoints over any channel supported by JBossESB. See the Programmers' Guide for more details.

Important

The SOAPProcessor action requires JBossWS 2.0.1.SP2 (native) or higher to to be properly installed on your JBoss Application Server (v4.2.x.GA).

Default ReplyTo EPR

JBossESB uses Endpoint References (EPRs) to address messages to/from services. As described in the Programmers Guide, messages have headers that contain recipient addresses, sequence numbers (for message correlation) and optional addresses for replies, faults etc. Because the recommended interaction pattern within JBossESB is based on one-way message exchange, responses to messages are not necessarily automatic: it is application dependent as to whether or not a sender expects a response.

As such, a reply address (EPR) is an optional part of the header routing information and applications should be setting this value if necessary. However, in the case where a response is required and the reply EPR (ReplyTo EPR) has not been set, JBossESB supports default values for each type of transport. Some of these ReplyTo defaults require system administrators to configure JBossESB correctly.

- For JMS, it is assumed to be a queue with a name based on the one used to deliver the original request: <request queue name>_reply
- For JDBC, it is assumed to be a table in the same database with a name based on the one used to deliver the original request: <request table name>_reply_table. The new table needs the same columns as the request table.
- For files (both local and remote), no administration changes are required: responses will be written into the same directory as the request but with a unique suffix to ensure that only the original sender will pick up the response.

ServiceBinding Manager

If you wish to run multiple ESB servers on the same machine, you may want to use JBoss ServiceBinding Manager. The binding manager allows you to centralize port configuration for all of the instances you will be running. The ESB server ships with a sample bindings file in docs/examples/binding-manager/sample-bindings.xml. Chapter Ten of the JBoss appplication server documentation contains instructions on how to set up the ServiceBinding manager. Note:

Note

`remoting-service.xml` – If you are using jboss-messaging as your JMS provider, please note that what you specify in your ServiceBinding manager xml for jboss-messaging configuration must match what is in remoting-service.xml.

Monitoring and Management

There are a number of options for monitoring and managing your ESB server. Shipping with the ESB are a number of useful JMX MBeans that help administrators monitor the performance of their server.

Under the jboss.esb domain, you should see the following MBean types:

deployment=<ESB package name>

Deployments show the state of all of the esb packages that have been deployed and give information about their XML configuration and their current state.

listener-name=<Listener name>

All deployed listeners are displayed, with information on their XML configuration, the start time, maxThreads, state, etc. The administrator has the option of initialising/starting/stopping/destroying a listener.

category=MessageCounter

Message counters break all of the services deployed for a listener down into their separate actions and give counts of how many messages were processed, as well as the processing time of each message.

service-name=<Service name>

Displays statistics per-service (message counts, state, average size of message, processing time, etc). The message counts may be reset and services may be stopped and started.

Note

Additionally, jms domain MBeans show statistics for message queues, which is useful information when debugging or determining performance.

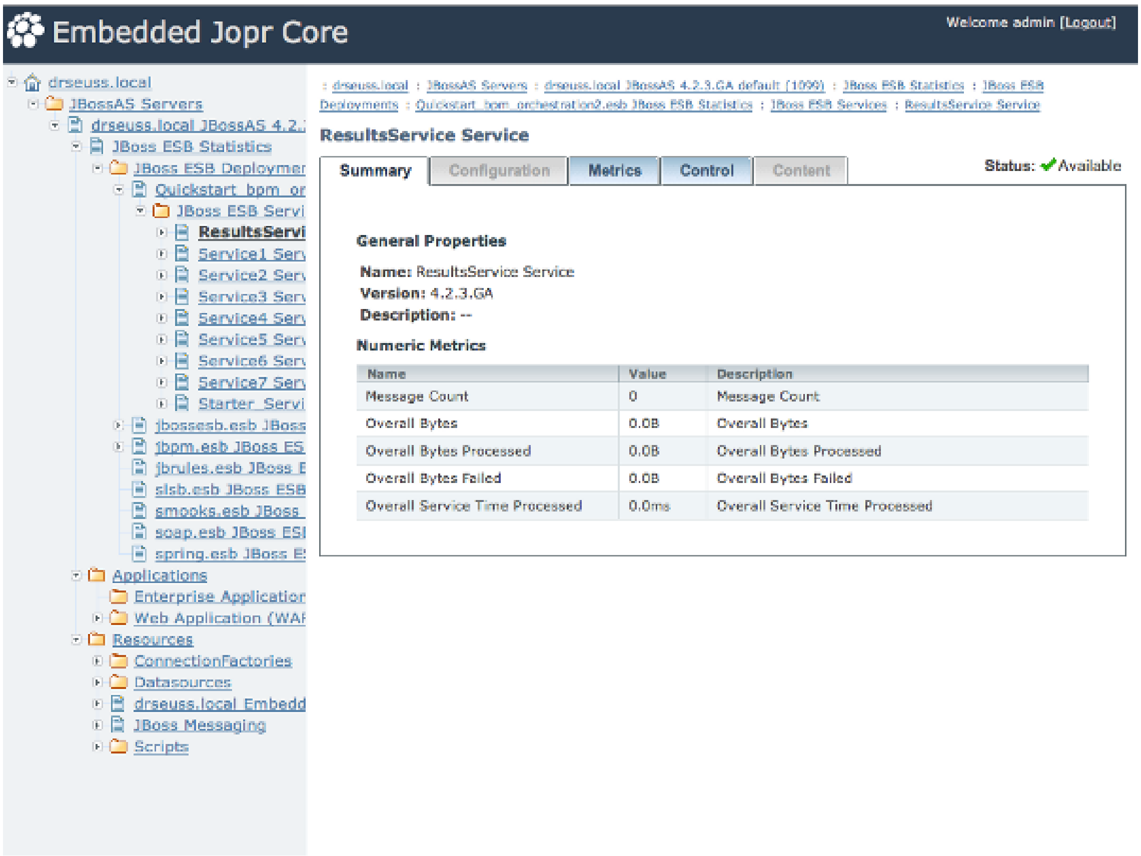
6.1. Monitoring and Management

JBossESB provides management and monitoring through Embedded JOPR: (*http://localhost:8080/admin-console*).

The JBossESB monitoring console gathers information on the performance of different ESB services that are deployed . As of JBoss ESB 4.2.0.GA, the monitoring console allows users to get message counts by service, action, and node, as well as other information like processing time, number of failed messages, bytes transferred, and last successful and failed message date time. As of JBoss ESB 4.6, the previous ESB monitoring tool has been deprecated.

The monitoring console is installed automatically in the stand-alone ESB server and JBossAS.

Below is a screenshot of the console. The console requests MBean information from each node within the ESB registry, and then displays it back.



ResultsService Service

General Properties

Name: ResultsService Service
Version: 4.2.3.GA
Description: --

Numeric Metrics

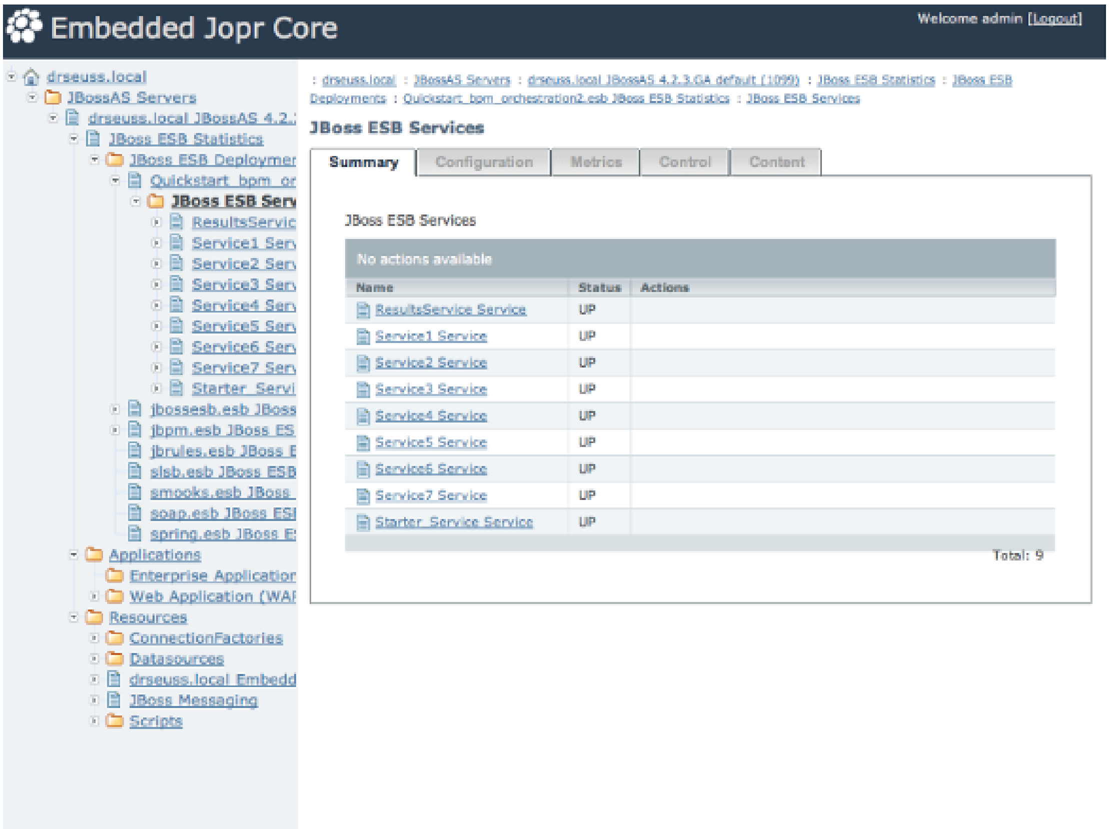
Name	Value	Description
Message Count	0	Message Count
Overall Bytes	0.0B	Overall Bytes
Overall Bytes Processed	0.0B	Overall Bytes Processed
Overall Bytes Failed	0.0B	Overall Bytes Failed
Overall Service Time Processed	0.0ms	Overall Service Time Processed

Figure 6.1. JBoss Monitoring and Management Console

6.1.1. Services

Each ESB service is displayed along with the processing time per action, processed count per action, failed count per action, and overall message count (per service).

See below:



: drseuss.local : JBossAS Servers : drseuss.local JBossAS 4.2.3.GA default (1099) : JBoss ESB Statistics : JBoss ESB Deployments : Quickstart_bpm_orchestration2.esb JBoss ESB Statistics : JBoss ESB Services

JBoss ESB Services

JBoss ESB Services

No actions available

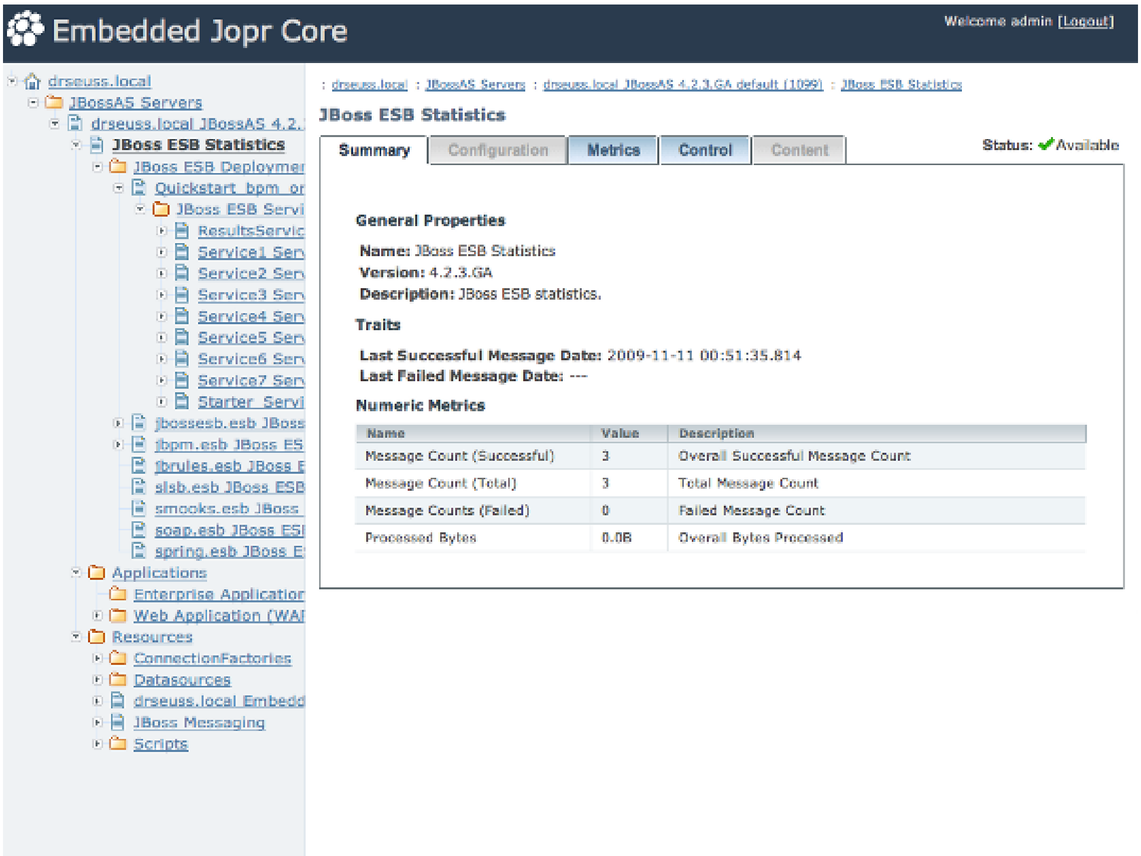
Name	Status	Actions
ResultsService Service	UP	
Service1 Service	UP	
Service2 Service	UP	
Service3 Service	UP	
Service4 Service	UP	
Service5 Service	UP	
Service6 Service	UP	
Service7 Service	UP	
Starter_Service Service	UP	

Total: 9

Figure 6.2. Services

6.1.2. The Message Counter

Seen above, the monitoring console also provides an overall counter which counts all messages that pass through the ESB. The MessageCounter keeps track of the successful and failed message counts, as well as time and date.



JBoss ESB Statistics

General Properties

Name: JBoss ESB Statistics
Version: 4.2.3.GA
Description: JBoss ESB statistics.

Traits

Last Successful Message Date: 2009-11-11 00:51:35.814
Last Failed Message Date: ---

Numeric Metrics

Name	Value	Description
Message Count (Successful)	3	Overall Successful Message Count
Message Count (Total)	3	Total Message Count
Message Counts (Failed)	0	Failed Message Count
Processed Bytes	0.0B	Overall Bytes Processed

Figure 6.3. Monitoring and Management Console Message Counter

6.1.3. Transformations

For each Smooks Transformation that is registered, a Mbean keeps track of the processed count for each transformation, processing time for each transformation, and the overall count for the transformation chain. You can see this information in the jmx-console.

6.1.4. DeadLetterService

As has been mentioned in the Programmers Guide, the DeadLetterService (DLQ) can be used to store messages that cannot be delivered. This is a JBossESB service and can be monitored and inspected. Note, however, that the DLQ is not used if the underlying transport has native support, e.g., JMS. In which case you should inspect the JBossESB DLQ as well as any transport-specific equivalent.

6.1.5. Alerts

The JBoss Web Console (*http://community.jboss.org/wiki/WebConsole*) is a utility within both the JBoss AS and the JBoss ESB Server that is capable of monitoring and sending alerts based off of JMX MBean properties. You can use this functionality to receive alerts for ESB-related events – such as the DeadLetterService counter reaching a certain threshold.

1. Configure ./deploy/mail-service.xml with your SMTP settings.

2. Change ./deploy/monitoring-service.xml – uncomment the EmailAlertListener section and add appropriate header related information.

3. Create a file ./deploy to serve as your monitor MBean.

```
<?xml version="1.0" encoding="UTF-8"?>
<server>
<mbean code="org.jboss.monitor.ThresholdMonitor"
       name="jboss.monitor:service=ESBDLQMonitor">
  <attribute name="MonitorName">ESB DeadLetterQueue Monitor</attribute>
  <attribute name="ObservedObject">
                                  jboss.esb:category=MessageCounter,deployment=jbossesb.esb,service-name=DeadLetterService
  </attribute>
  <attribute name="ObservedAttribute">overall service message count</attribute>
  <attribute name="Threshold">4</attribute>
  <attribute name="CompareTo">-1</attribute>
  <attribute name="Period">1000</attribute>
  <attribute name="Enabled">true</attribute>
  <depends-list optional-attribute-name="AlertListeners">
<depends-list-element>jboss.alerts:service=ConsoleAlertListener</depends-list-element>
<depends-list-element>jboss.alerts:service=EmailAlertListener</depends-list-element>
  </depends-list>
  <depends>jboss.esb:deployment=jbossesb.esb</depends>
</mbean>
</server>
```

This MBean will serve as a monitor, and once the DeadLetterService counter reaches 5, it will send an e-mail to the address(es) specified in the monitoring-service.xml. Note that the alert is only sent once – once the threshold has been reached. If you want to be alerted again once resetting the counter, you can reset the alerted flag on your monitoring service MBean (in this case jboss.monitor:service=ESBDLQMonitor).

Note

For more details on how to use the JBoss Web Console monitoring, please see *http://community.jboss.org/wiki/JBossMonitoring*.

Hot Deployment

JBossAS as well as the JBossESB-Server are always checking the 'deploy' directory for new files to deploy. So we're really talking about hot redeployment. So here is what you have to do to make it redeploy an existing deployment for the different components.

1. SAR files

   The jbossesb.sar is hot deployable. It will redeploy when

   - the timestamp of the archive changes, if the sar is compressed archive.
   - the timestamp of the META-INF/jboss-service.xml changes, if the sar is in exploded from.

2. ESB files

   Any *.esb archive will redeploy when

   - the timestamp of the archive changes, if the esb is compressed archive.
   - the timestamp of the META-INF/jboss-esb.xml changes, if the esb is in exploded from.

   Our actions have lifecycle support, so upon hot deployment it goes down gracefully, finishes active requests, and does not accept any more incoming messages until it is back up. All of this can be done by simply redeploying the .esb archive. If you want to update just one action, you can use groovy scripting to modify an action at runtime (see the groovy QuickStart: *http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossESBQuickStart*.)

3. rule files

   There are two options to refresh rule files (drl or dsl)

   a. redeploy the jbrules.esb (see 2)

   b. turn on the 'ruleReload' in the action config (see *http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossESBContentBasedRouting*). Now if a rule file *changes* it will be reloaded. .) After doing so, if a rule file is changed, it will be re-loaded.

4. transformation files

   The only way to refresh transformation files is to redeploy the esb archive in which the transformation file resides.

5. Business Process Definitions

   When using jBPM new Business Process Definitions can be deployed. From within the jBPM eclipse plugin you can deploy a new definition to the jbpm database. New process instances will get the new version, in flight processes will finish their life cycle on the previous definitions. For details please see the documentation on jBPM.

7.1. Standalone (bootstrap) mode

The bootstrapper does not deploy esb archives. You can only have one jboss-esb.xml configuration file per node. It will monitor the timestamp on this file and it will reread the configuration if a change occurs. To updates rules you will have to use the 'ruleReload'. And finally to update BPDs you can follow the same process mentioned above.

Contract Publishing

8.1. Overview

Integrating to certain ESB endpoints may require information about that endpoint and the operations it supports. This is particularly the case for Webservice endpoints exposed via the SOAPProcessor action (see Message Action Guide).

8.2. "Contract" Application

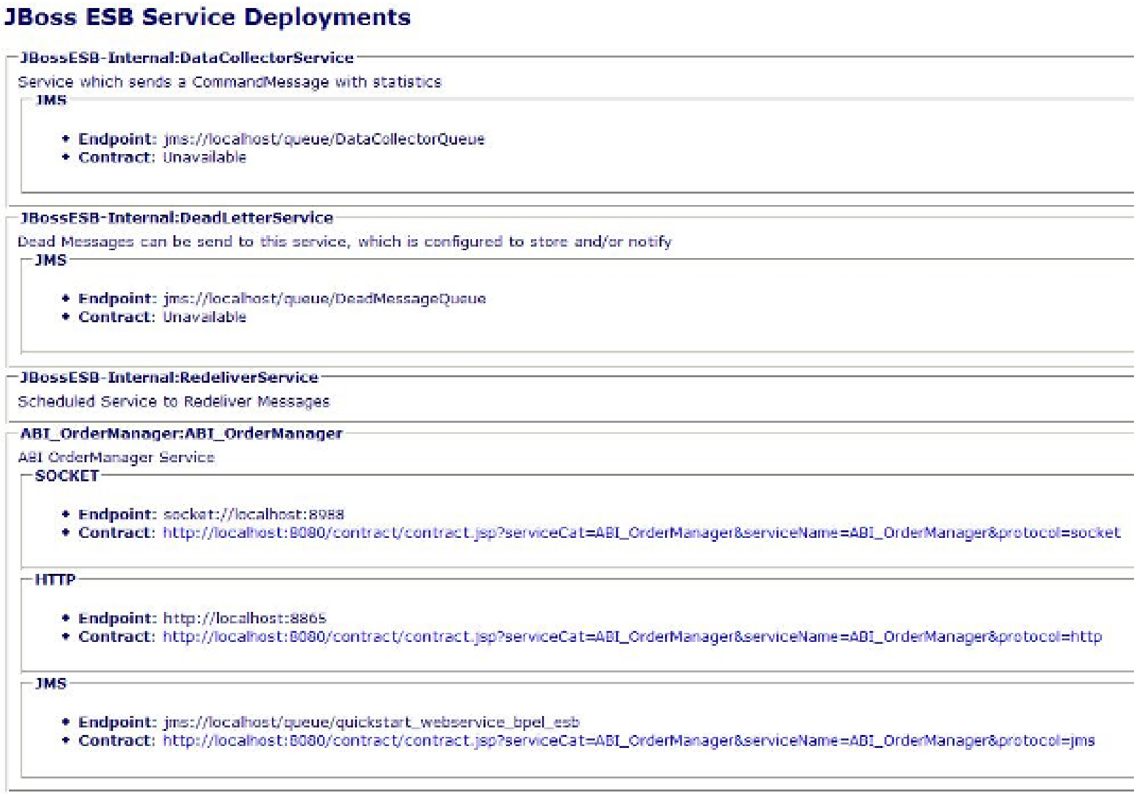
Important

This application is only being offered as a Technical Preview. It will be superseded in a later release.

For this purpose, we bundle the "Contract" application with the ESB. This application is installed by default with the ESB (after running "ant deploy" from the install directory of the distro). Note that the Contract application is also bundled inside the JBossESB Console. If you are deploying the console, you will first need to undeploy the default Contract application. Just remove contract.war from the default/deploy folder of your ESB/App Server.

It can be accessed through the URL *http://localhost:8080/contract/*.

Figure 8.1, “The Contract Application” is a screenshot of this application.



JBoss ESB Service Deployments

JBossESB-Internal:DataCollectorService

Service which sends a CommandMessage with statistics

JMS

- **Endpoint:** jms://localhost/queue/DataCollectorQueue
- **Contract:** Unavailable

JBossESB-Internal:DeadLetterService

Dead Messages can be send to this service, which is configured to store and/or notify

JMS

- **Endpoint:** jms://localhost/queue/DeadMessageQueue
- **Contract:** Unavailable

JBossESB-Internal:RedeliverService

Scheduled Service to Redeliver Messages

ABI_OrderManager:ABI_OrderManager

ABI OrderManager Service

SOCKET

- **Endpoint:** socket://localhost:8988
- **Contract:** http://localhost:8080/contract/contract.jsp?serviceCat=ABI_OrderManager&serviceName=ABI_OrderManager&protocol=socket

HTTP

- **Endpoint:** http://localhost:8865
- **Contract:** http://localhost:8080/contract/contract.jsp?serviceCat=ABI_OrderManager&serviceName=ABI_OrderManager&protocol=http

JMS

- **Endpoint:** jms://localhost/queue/quickstart_webservice_bpel_esb
- **Contract:** http://localhost:8080/contract/contract.jsp?serviceCat=ABI_OrderManager&serviceName=ABI_OrderManager&protocol=jms

Figure 8.1. The Contract Application

As you can see, it groups the endpoint according to Service with which they are associated (servicing). Another thing you'll notice is how some of them have an active "Contract" hyperlink. The ones visible here are for Webservice endpoints exposed via the SOAPProcessor. This hyperlink links off to the WSDL.

8.3. Publishing a Contract from an Action

JBossESB discovers endpoint contracts based on the action pipeline that's configured on a Service. It looks for the first action in the pipeline that publishes contract information. If none of the actions publish contract information, then the Contract application just displays "Unavailable" on Contract for that endpoint.

An Action publishes contract information by being annotated with the org.jboss.internal.soa.esb.publish.Publish annotation as follows (using the SOAPProcessor as an example):

```
@Publish(WebserviceContractPublisher.class)
public class SOAPProcessor extends AbstractActionPipelineProcessor {
    ...
}
```

See the SOAPProcessor code as an example at *http://anonsvn.labs.jboss.com/labs/jbossesb/trunk/product/services/soap/src/main/java/org/jboss/soa/esb/actions/soap/SOAPProcessor.java*.

You then need to implement a "ContractPublisher" (org.jboss.soa.esb.actions.soap.ContractPublisher), which just requires implementation of a single method:

```
public ContractInfo getContractInfo(EPR epr);
```

See the WebserviceContractPublisher code as an example at *http://anonsvn.labs.jboss.com/labs/jbossesb/trunk/product/services/soap/src/main/java/org/jboss/soa/esb/actions/soap/WebserviceContractPublisher.java*.

jBPM

9.1. jBPM Console

The jBPM Web Console is deployed by default as part of jbpm.esb and can be found at: *http://localhost:8080/jbpm-console/*. Please refer to the jBPM documentation for information regarding the console.

9.2. jBPM Message and Scheduler service

The jBPM integration within ESB now support additional Message and Scheduler services, distinct from those offered natively by jBPM. In addition to the standard jBPM configurations we now also support a JMS based Message Service, driven using JCA inflow, and three additional Scheduling Services, based on JBoss Messaging, hornetq and quartz.

The configuration, as shipped by default within ESB, uses the jBPM JobExecutor and the database implementations of the Message and Scheduler service.

```
<service name="message" factory="org.jbpm.msg.db.DbMessageServiceFactory" />
<service name="scheduler" factory="org.jbpm.scheduler.db.DbSchedulerServiceFactory" />

<bean name="jbpm.job.executor" class="org.jbpm.job.executor.JobExecutor">
   ...
</bean>
```

In order to utilize the alternative services it is necessary to replace the active configurations with the versions specific to your requirements.

The configuration for the alternative services can be found within the jbpm.esb/config directory

1. jbpm.esb/config/hornetqscheduler for configuring the hornetq based message and scheduler services
2. jbpm.esb/config/jmsscheduler for configuring the JBoss Messaging based message and scheduler services
3. jbpm.esb/config/quartzscheduler for configuring the JMS message service and a quartz based scheduler service.

The configuration files within the appropriate directory should be used to replace the active configurations within the jbpm.esb directory, remembering to remove the .config suffix from each. It should also be noted that only one of the quartz message queue service definitions should be used, either hornetq-jms.xml, jbm-queue-service.xml or jbmq-queue-service.xml, depending on which JMS implementation is currently in use.

Performance Tuning

Please refer to the JBossESB wiki for information on Performance Tuning: *http://community.jboss.org/wiki/JBossESBPerformanceTuning*.

Appendix A. GNU General Public License

Version 2, June 1991

Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor,
Boston, MA 02110-1301
USA

Version 2, June 1991

A.1. Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software - to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. copyright the software, and
2. offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

A.2. TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

A.2.1. Section 0

This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

A.2.2. Section 1

You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

A.2.3. Section 2

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of *Section 1* above, provided that you also meet all of these conditions:

a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: If the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

A.2.4. Section 3

You may copy and distribute the Program (or a work based on it, under *Section 2* in object code or executable form under the terms of *Sections 1* and *2* above provided that you also do one of the following:

a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

A.2.5. Section 4

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

A.2.6. Section 5

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

A.2.7. Section 6

Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

A.2.8. Section 7

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

A.2.9. Section 8

If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

A.2.10. Section 9

The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

A.2.11. Section 10

If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

A.2.12. NO WARRANTY Section 11

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

A.2.13. Section 12

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

A.3. How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type “show w”. This is free software, and you are welcome to redistribute it under certain conditions; type “show c” for details.

The hypothetical commands “show w” and “show c” should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than “show w” and “show c”; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program “Gnomovision” (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Appendix B. Revision History

Revision History

Revision 1	Fri Jul 16 2010	DavidLe Sage`<dlesage@redhat.com>`, DarrinMison`<dmison@redhat.com>`

Initial conversion from OpenOffice ODT files.