JBoss Application Server

Chris Bonham

Third Eye Consulting, Inc.

bonhamcm@thirdeyeconsulting.com

IndyJUG

August 27, 2003

Java 2 Enterprise Edition (J2EE)

"The J2EE platform manages the infrastructure and supports the Web services to enable development of secure, robust and interoperable business applications."

J2EE 1.3 Technologies

- Enterprise JavaBeans (EJB) 2.0
- Java Message Service (JMS) 1.0.2b
- JavaMail 1.2
- JavaBeans Activation Framework (JAF) 1.0.1
- Java Database Connectivity (JDBC) 3.0, 2.1, Optional 2.0
- J2EE Connector Architecture (JCA) 1.0
- Java Naming & Directory Interface (JNDI) 1.2.1
- Java Servlet 2.3 & JavaServer Pages (JSP) 1.2
- Java Transaction API (JTA) 1.0.1b
- Java API for XML Processing (JAXP) 1.1
- Java Authentication & Authorization Service (JAAS) 1.0
- CORBA Integration (RMI/IIOP, IDL, EJB-CORBA Mapping)

JBoss Application Server

http://www.jboss.org/

- Open Source
- Hosted on Sourceforge http://sourceforge.net/projects/jboss/
- Founded by Marc Fleury
- Formerly EJBoss
- ~30 CVS commiters
- Mailing Lists (archived at http://www.mail-archive.com/)
  - jboss-user
  - Jboss-development
- JBoss Group Consultancy http://www.jbossgroup.com/

JBoss Versions

- 2.4 Latest is 2.4.11 `Branch_2_4` in CVS
  - First version using Dynamic Proxy
  - J2EE 1.2, EJB 1.1
- 3.0 Latest is 3.0.8 `Branch_3_0` in CVS
  - J2EE 1.3, EJB 2.0
  - Datasource files: `xxx-service.xml`
- 3.2 Latest is 3.2.2RC3 `Branch_3_2` in CVS
  - CMP 2.0 Improvements: PK generation, audit fields
  - Datasource files: `xxx-ds.xml` (greatly simplified!)
- 4.0 Latest is 4.0.0DR2 `HEAD` in CVS
  - Aspect Oriented Programming (AOP)
  - JMS Rewrite (using JavaGroups)
  - Distributed Optimistic Cache
  - JBossDO Java Data Objects (JDO) implementation

JBoss Installation

1. Download version from http://sourceforge.net/projects/jboss/
2. Extract from file
3. Run `run.sh/run.bat`
   - Configurations
     - Default (no arguments)
     - All (`run.sh -c all`)
       - JBoss.net (Web Services w/Apache Axis)
       - CORBA ORB (using open-source JacORB)
       - Clustering (using open-source JavaGroups)
     - Minimal (`run.sh -c minimal`)
       - Netboot
     - Custom (`run.sh -c jboss0`)

JBoss Setup

- Uses Java Management Extensions (JMX) for server management and deployment
- JMX Services deployed in `xxx-service.xml` files
- Relevant Directories
  - `bin` — `run.sh/run.bat`, `shutdown.sh/shutdown.bat`
  - `client` — client-side JARs, e.g., `jbossall-client.jar`
  - `docs/dtd` — J2EE and JBoss deployment descriptors
  - `docs/examples/jca` — datasource configuration examples
  - `lib` — core JBoss libraries
  - `server/default/conf` — configuration files for default configuration
  - `server/default/deploy` — deployment directory for default configuration
  - `server/default/lib` — libraries specific to default configuration
  - `server/default/log` — server and web access logs for default configuration

Datasources

- Uses J2EE Connector Architecture (JCA) to manage datasource connection pools
- Default database is Hypersonic (Java open source)
  - 3.2 Deployment file located in `server/default/deploy/hsqldb-ds.xml`
  - 3.0 Deployment file located in `server/default/deploy/hsqldb-service.xml`
  - Data is stored in `server/default/data/hypersonic`
- Other database configuration examples in `docs/examples/jca`
- Just copy `xxx-ds.xml` file to `server/default/deploy`
- Make sure `<jndi-name>` is unique in each `xxx-ds.xml` file

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- ===================================================================== -->
<!--                                                                       -->
<!--  JBoss Server Configuration                                           -->
<!--                                                                       -->
<!-- ===================================================================== -->

<!-- $Id: hsqldb-ds.xml,v 1.1.2.6 2003/08/19 00:18:19 ejort Exp $ -->


<datasources>
  <local-tx-datasource>
    <!-- remove this depends tag if you are not using the tcp connection url -->
    <depends>jboss:service=Hypersonic</depends>
    <jndi-name>DefaultDS</jndi-name>
    <!-- for tcp connection, allowing other processes to use the hsqldb database -->
    <connection-url>jdbc:hsqldb:hsql://localhost:1701</connection-url>
    <!-- for totally in-memory db, not saved when jboss stops.  hsql mbean is unnecessary-->
    <!--connection-url>jdbc:hsqldb:.</connection-url-->
    <!-- for in-process db,  saved when jboss stops.  hsql mbean is unnecessary-->
    <!--connection-url>jdbc:hsqldb:default-db-name</connection-url-->
    <driver-class>org.hsqldb.jdbcDriver</driver-class>
    <user-name>sa</user-name>
    <password></password>
    <!--example of how to specify class that determines if exception means connection should be destroyed-->
    <!--exception-sorter-class-name>org.jboss.resource.adapter.jdbc.vendor.DummyExceptionSorter</exception-sorter-class-name-->

    <!-- this will be run before a managed connection is removed from the pool for use by a client-->
    <!--<check-valid-connection-sql>select * from something</check-valid-connection-sql> -->

    <!-- this will be run each time a connection handle is created.  Useful for setting up user-specific packages in Oracle -->
    <!-- <new-connection-sql>blah</new-connection-sql>-->
    <min-pool-size>5</min-pool-size>
    <!-- TEMPORARY FIX! - Disable idle connection removal, HSQLDB has a problem with not reaping threads on closed connections -->
    <idle-timeout-minutes>0</idle-timeout-minutes>
    <security-domain>HsqlDbRealm</security-domain>
  </local-tx-datasource>

  <!-- this mbean should be used only when using tcp connections -->
  <mbean code="org.jboss.jdbc.HypersonicDatabase"
       name="jboss:service=Hypersonic">
    <attribute name="Port">1701</attribute>
    <attribute name="Silent">true</attribute>
    <attribute name="Database">default</attribute>
    <attribute name="Trace">false</attribute>
    <attribute name="No_system_exit">true</attribute>
  </mbean>
</datasources>
```

WebServer

- Uses Apache Tomcat 4.1.x by default
- Also integrates with open-source Jetty http://jetty.mortbay.org/ (used to be default)
- Deployment file located in
  `server/default/deploy/jbossweb-tomcat41.sar/META-INF/jboss-service.xml`
- Default port is 8080
- Commonly configured attributes of the `<Connector>` element
  - `port` HTTP Connector port
  - `minProcessors` Minimum number of request processor threads
  - `maxProcessors` Maximum number of request processor threads
  - `acceptCount` Number of requests enqueued while waiting for connection before a request is dropped

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- The service configuration for the embedded Tomcat4.1.x web container
-->
<server>
   <mbean code="org.jboss.web.tomcat.tc4.EmbeddedTomcatService"
      name="jboss.web:service=WebServer">

      <!-- Get the flag indicating if the normal Java2 parent first class
      loading model should be used over the servlet 2.3 web container first
      model.
      -->
      <attribute name="Java2ClassLoadingCompliance">true</attribute>

      <attribute name="LenientEjbLink">true</attribute>

      <!-- A flag indicating if the JBoss Loader should be used. This loader
      uses a unified class loader as the class loader rather than the tomcat
      specific class loader.
      -->
      <attribute name="UseJBossWebLoader">true</attribute>


      <attribute name="Config">
         <Server>
            <Service name="JBoss-Tomcat">
               <Engine name="MainEngine" defaultHost="localhost">
                  <Logger className="org.jboss.web.tomcat.Log4jLogger"
                     verbosityLevel="debug" category="org.jboss.web.localhost.Engine"/>
                  <Host name="localhost">

                     <!-- Access logger -->
                     <Valve className="org.apache.catalina.valves.AccessLogValve"
                        prefix="localhost_access" suffix=".log"
                        pattern="common" directory="${jboss.server.home.dir}/log"/>

                     <Valve className="org.jboss.web.tomcat.security.SecurityAssociationValve" />
                     <!-- Default context parameters -->
                     <DefaultContext cookies="true" crossContext="true" override="true"/>

                  </Host>
               </Engine>

               <!-- A HTTP/1.1 Connector on port 8080 -->
               <Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
                  port="8080" minProcessors="5" maxProcessors="100"
                  enableLookups="true" acceptCount="10" debug="0"
                  connectionTimeout="20000" useURIValidationHack="false"/>
            </Service>
         </Server>
      </attribute>
      <depends>jboss:service=TransactionManager</depends>
   </mbean>
</server>
```

JMS Setup: Invocation Layers

- Uses an open source project originally called SpyderMQ, now integrated into JBoss
- No default binding to ConnectionFactory in JNDI
- Six different InvocationLayer services each providing a ConnectionFactory
- Located in `server/default/deploy/jms`
  - `rmi-il-service.xml` Uses RMI communication
  - `uil-service.xml` Uses one-port communication, e.g., applets (deprecated)
  - `uil2-service.xml` Uses one-port communication, e.g., applets
  - `oil-service.xml` Optimized multi-port communication
  - `oil2-service.xml` Alternate optimized multi-port communication
  - `jvm-il-service.xml` In-memory communication, e.g., MDBs

JMS Setup: PersistenceManager

- PersistenceManager Service
  - Stores unconsumed messages that need to be processed
  - Default PM stores data in the Hypersonic database (`hsqldb-jdbc2-service.xml`)
    - Depends on DefaultDS service
    - Add database-specific JDBC library to `server/default/lib` not `server/default/deploy`
    - Message is stored as BLOB in database; four different ways to manipulate it
      - `OBJECT_BLOB ResultStream.getObject()/setObject()`
      - `BYTES_BLOB ResultStream.getBytes()/setBytes()`
      - `BINARYSTREAM_BLOB ResultSet.getBinaryStream()/setBinaryStream()`
      - `BLOB_BLOB ResultSet.getBlob().getBinaryStream()`/not implemented
  - Other database and file PM examples in `docs/examples/jms`
  - File PM is `file-pm-service.xml`

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- $Id: hsqldb-jdbc2-service.xml,v 1.1.2.1 2003/08/23 22:21:28 ejort Exp $ -->

<server>
  <!-- ===================================================================== -->
  <!-- Persistence and caching using HSQLDB                                  -->
  <!-- See docs/examples/jms for other configurations                        -->
  <!-- ===================================================================== -->
  <!--
     | The jdbc2 PersistenceManager is the new improved JDBC implementation.
     | This implementation allows you to control how messages are stored in
     | the database.
     |
     | Use this PM if you want the reliablity a relational database can offer
     | you.  The default configuration is known to work with hsqldb, other
     | databases will requrie teaking of the SqlProperties e.g. changing table
     | colum definitions to database supported types.
     |
     | Take care that the selected blob column type in jms_messages can store
     | all message data. Some databases (e.g. mySQL) offer blob types with
     | different maximum capacity (e.g. mySQL-type BLOB=64K, LONGBLOB=2G).
     |
     | If you encouter problems with the configured BLOB_TYPE try a different
     | setting. Valid settings are OBJECT_BLOB, BINARYSTREAM_BLOB and BYTES_BLOB.
   -->
  <mbean code="org.jboss.mq.pm.jdbc2.PersistenceManager"
       name="jboss.mq:service=PersistenceManager">
    <depends optional-attribute-name="ConnectionManager">jboss.jca:service=LocalTxCM,name=DefaultDS</depends>
    <attribute name="SqlProperties">
      BLOB_TYPE=OBJECT_BLOB
      INSERT_TX = INSERT INTO JMS_TRANSACTIONS (TXID) values(?)
      INSERT_MESSAGE = INSERT INTO JMS_MESSAGES (MESSAGEID, DESTINATION, MESSAGEBLOB, TXID, TXOP) VALUES(?,?,?,?,?)
      SELECT_ALL_UNCOMMITED_TXS = SELECT TXID FROM JMS_TRANSACTIONS
      SELECT_MAX_TX = SELECT MAX(TXID) FROM JMS_MESSAGES
      SELECT_MESSAGES_IN_DEST = SELECT MESSAGEID, MESSAGEBLOB FROM JMS_MESSAGES WHERE DESTINATION=?
      SELECT_MESSAGE = SELECT MESSAGEID, MESSAGEBLOB FROM JMS_MESSAGES WHERE MESSAGEID=? AND DESTINATION=?
      MARK_MESSAGE = UPDATE JMS_MESSAGES SET TXID=?, TXOP=? WHERE MESSAGEID=? AND DESTINATION=?
      UPDATE_MESSAGE = UPDATE JMS_MESSAGES SET MESSAGEBLOB=? WHERE MESSAGEID=? AND DESTINATION=?
      UPDATE_MARKED_MESSAGES = UPDATE JMS_MESSAGES SET TXID=?, TXOP=? WHERE TXOP=?
      UPDATE_MARKED_MESSAGES_WITH_TX = UPDATE JMS_MESSAGES SET TXID=?, TXOP=? WHERE TXOP=? AND TXID=?
      DELETE_MARKED_MESSAGES_WITH_TX = DELETE FROM JMS_MESSAGES WHERE TXID IN (SELECT TXID FROM JMS_TRANSACTIONS) AND TXOP=?
      DELETE_TX = DELETE FROM JMS_TRANSACTIONS WHERE TXID = ?
      DELETE_MARKED_MESSAGES = DELETE FROM JMS_MESSAGES WHERE TXID=? AND TXOP=?
      DELETE_MESSAGE = DELETE FROM JMS_MESSAGES WHERE MESSAGEID=? AND DESTINATION=?
      CREATE_MESSAGE_TABLE = CREATE TABLE JMS_MESSAGES ( MESSAGEID INTEGER NOT NULL, \
         DESTINATION VARCHAR(255) NOT NULL, TXID INTEGER, TXOP CHAR(1), \
         MESSAGEBLOB OBJECT, PRIMARY KEY (MESSAGEID, DESTINATION) )
      CREATE_TX_TABLE = CREATE TABLE JMS_TRANSACTIONS ( TXID INTEGER )
      CREATE_TABLES_ON_STARTUP = TRUE
    </attribute>
  </mbean>
</server>
```

JMS Setup: Topics/Queues

- Default Topics/Queues in `server/default/deploy/jms/jbossmq-destinations-service.xml`
  - `topic/testTopic`
  - `topic/securedTopic`
  - `topic/testDurableTopic`
  - `queue/testQueue`
  - `queue/A` through `queue/D`
  - `queue/ex`
- Deploy application specific Topics/Queues in `server/default/deploy/jms` using different file

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- $Id: jbossmq-destinations-service.xml,v 1.2 2002/05/04 03:07:49 chirino Exp $ -->

<!--
   | This file defines the default Queues and Topics that JBossMQ
   | ships with.  The default Queues and Topics are used by the
   | JBoss test suite and by the sample jms programs.
   |
   | You can add other destinations to this file, or you can create other
   | *-service.xml files to contain your application's destinations.
 -->

<server>
  <!-- Destination without a configured SecurityManager or without a
       a SecurityConf will default to role guest with read=true, write=true,
       create=false.
  -->

  <mbean code="org.jboss.mq.server.jmx.Topic"
       name="jboss.mq.destination:service=Topic,name=testDurableTopic">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
    <depends optional-attribute-name="SecurityManager">jboss.mq:service=SecurityManager</depends>
    <attribute name="SecurityConf">
       <security>
         <role name="guest" read="true" write="true"/>
         <role name="publisher" read="true" write="true" create="false"/>
         <role name="durpublisher" read="true" write="true" create="true"/>
      </security>
    </attribute>
  </mbean>

  <mbean code="org.jboss.mq.server.jmx.Queue"
       name="jboss.mq.destination:service=Queue,name=testQueue">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
    <depends optional-attribute-name="SecurityManager">jboss.mq:service=SecurityManager</depends>
    <attribute name="SecurityConf">
      <security>
        <role name="guest" read="true" write="true"/>
        <role name="publisher" read="true" write="true" create="false"/>
        <role name="noacc" read="false" write="false" create="false"/>
      </security>
    </attribute>
  </mbean>

</server>
```

JBoss Default Ports

Port	Description	Location
1099	JNDI	`server/default/conf/jboss-service.xml`
1100	HAJNDI for clustering	`server/all/deploy/cluster-service.xml`
8083	WebService for downloading RMI classes	`server/default/conf/jboss-service.xml`
4444	RMI/JRMP Invoker, invokes JMX services	`server/default/conf/jboss-service.xml`
4445	RMI/JRMPHA Invoker, invokes clustered JMX services	`server/all/deploy/cluster-service.xml`
3528	CORBA ORB	`server/all/conf/jacorb.properties`
8089	JMS RMI InvocationLayer	`server/default/deploy/jms/rmi-il-service.xml`
8090	JMS OIL InvocationLayer	`server/default/deploy/jms/oil-service.xml`
8091	JMS UIL InvocationLayer	`server/default/deploy/jms/uil-service.xml`
8092	JMS OIL2 InvocationLayer	`server/default/deploy/jms/oil2-service.xml`
8093	JMS UIL2 InvocationLayer	`server/default/deploy/jms/uil2-service.xml`
1701	Hypersonic Database	`server/default/deploy/hsqldb-ds.xml`
8080	WebServer (Tomcat/Jetty)	`server/default/deploy/jbossweb-tomcat41.sar/META-INF/jboss-service.xml`

Administration Consoles

- JMX Console `/jmx-console/`
  - Displays all JMX services, some attributes can be changed
  - Deprecated after 3.2.2beta
- Web Console `/web-console/`
  - Navigation applet
  - Displays all JMX services, some attributes can be changed
  - Introduced in 3.2.2beta
- Useful Services
  - JNDIView
  - LoaderRepository

JBoss Application Deployment

- Easy deployment; just drop the file into the `server/default/deploy` directory!
- Supports hot-deploy (no server restarts)
- Handles many different file extensions, including embedded files
  - `.xml` expects a `<server>` doctype
  - `*-ds.xml` Datasource deployment descriptor (JBoss 3.2+)
  - `.jar` Java Archive
  - `.war` Web Application Archive
  - `.ear` Enterprise Application Archive
  - `.rar` Resource Application Archive
  - `.sar` Service Application Archive (JBoss specific)
  - `.wsr` Web Service Archive (JBoss specific)
- Supports exploded file formats, e.g., `server/default/deploy/jbossweb-tomcat41.sar/`

JBoss Classloading

- UnifiedClassLoader
  - Classes are visible between deployments
  - Can't have the different class versions in different deployment units
- Scoped Classloading
  - If different versions are needed, classes can be scoped at the EAR or WAR level
  - EAR `META-INF/jboss-app.xml`

```
<jboss-app>
   <loader-repository>jmx.loading:loader=scoped.ear</loader-repository>
</jboss-app>
```

  - WAR `WEB-INF/jboss-web.xml`

```
<jboss-web>
   <loader-repository>jmx.loading:loader=scoped.war</loader-repository>
</jboss-web>
```

EJB Interceptors

- EJB calls are intercepted and passed along the interceptor stack
- Interceptors for a container configuration can be changed declaratively by editing `server/default/conf/standardjboss.xml`
- Many Container Configurations
  - Standard/Clustered/Instance Per Transaction CMP 2.x EntityBean
  - Standard/Clustered/Instance Per Transaction CMP EntityBean
  - Standard/Clustered Stateless SessionBean
  - Standard/Clustered Stateful SessionBean
  - Standard/Clustered/Instance Per Transaction BMP EntityBean
  - Standard Message Driven Bean

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jboss PUBLIC
   "-//JBoss//DTD JBOSS 3.2//EN"
   "http://www.jboss.org/j2ee/dtd/jboss_3_2.dtd">

<!-- ===================================================================== -->
<!--  Standard JBoss EJB Configurations                                    -->
<!-- ===================================================================== -->
<!-- $Id: standardjboss.xml,v 1.47.2.13 2003/08/06 18:38:56 loubyansky Exp $ -->

<jboss>
   <container-configurations>
      <container-configuration>
         <container-name>Standard CMP 2.x EntityBean</container-name>
         <call-logging>false</call-logging>
         <invoker-proxy-binding-name>entity-rmi-invoker</invoker-proxy-binding-name>
         <sync-on-commit-only>false</sync-on-commit-only>
         <insert-after-ejb-post-create>false</insert-after-ejb-post-create>
         <container-interceptors>
            <interceptor>org.jboss.ejb.plugins.ProxyFactoryFinderInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.LogInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.SecurityInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.TxInterceptorCMT</interceptor>
            <interceptor metricsEnabled="true">org.jboss.ejb.plugins.MetricsInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.EntityCreationInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.EntityLockInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.EntityInstanceInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.EntityReentranceInterceptor</interceptor>
            <interceptor>org.jboss.resource.connectionmanager.CachedConnectionInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.EntitySynchronizationInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.cmp.jdbc.JDBCRelationInterceptor</interceptor>
         </container-interceptors>
         <instance-pool>org.jboss.ejb.plugins.EntityInstancePool</instance-pool>
         <instance-cache>org.jboss.ejb.plugins.InvalidableEntityInstanceCache</instance-cache>
         <persistence-manager>org.jboss.ejb.plugins.cmp.jdbc.JDBCStoreManager</persistence-manager>
         <locking-policy>org.jboss.ejb.plugins.lock.QueuedPessimisticEJBLock</locking-policy>
         <container-cache-conf>
            <cache-policy>org.jboss.ejb.plugins.LRUEnterpriseContextCachePolicy</cache-policy>
            <cache-policy-conf>
               <min-capacity>50</min-capacity>
               <max-capacity>1000000</max-capacity>
               <overager-period>300</overager-period>
               <max-bean-age>600</max-bean-age>
               <resizer-period>400</resizer-period>
               <max-cache-miss-period>60</max-cache-miss-period>
               <min-cache-miss-period>1</min-cache-miss-period>
               <cache-load-factor>0.75</cache-load-factor>
            </cache-policy-conf>
         </container-cache-conf>
         <container-pool-conf>
            <MaximumSize>100</MaximumSize>
         </container-pool-conf>
         <commit-option>B</commit-option>
      </container-configuration>
```

```
      <container-configuration>
         <container-name>Standard Stateless SessionBean</container-name>
         <call-logging>false</call-logging>
         <invoker-proxy-binding-name>stateless-rmi-invoker</invoker-proxy-binding-name>
         <container-interceptors>
            <interceptor>org.jboss.ejb.plugins.ProxyFactoryFinderInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.LogInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.SecurityInterceptor</interceptor>
            <!-- CMT -->
            <interceptor transaction="Container">org.jboss.ejb.plugins.TxInterceptorCMT</interceptor>
            <interceptor transaction="Container" metricsEnabled="true">org.jboss.ejb.plugins.MetricsInterceptor</interceptor>
            <interceptor transaction="Container">org.jboss.ejb.plugins.StatelessSessionInstanceInterceptor</interceptor>
            <!-- BMT -->
            <interceptor transaction="Bean">org.jboss.ejb.plugins.StatelessSessionInstanceInterceptor</interceptor>
            <interceptor transaction="Bean">org.jboss.ejb.plugins.TxInterceptorBMT</interceptor>
            <interceptor transaction="Bean" metricsEnabled="true">org.jboss.ejb.plugins.MetricsInterceptor</interceptor>
            <interceptor>org.jboss.resource.connectionmanager.CachedConnectionInterceptor</interceptor>
         </container-interceptors>
         <instance-pool>org.jboss.ejb.plugins.StatelessSessionInstancePool</instance-pool>
         <instance-cache></instance-cache>
         <persistence-manager></persistence-manager>
         <container-pool-conf>
            <MaximumSize>100</MaximumSize>
         </container-pool-conf>
      </container-configuration>
      <container-configuration>
         <container-name>Standard Message Driven Bean</container-name>
         <call-logging>false</call-logging>
         <invoker-proxy-binding-name>message-driven-bean</invoker-proxy-binding-name>
         <container-interceptors>
            <interceptor>org.jboss.ejb.plugins.ProxyFactoryFinderInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.LogInterceptor</interceptor>
            <interceptor>org.jboss.ejb.plugins.RunAsSecurityInterceptor</interceptor>
            <!-- CMT -->
            <interceptor transaction="Container">org.jboss.ejb.plugins.TxInterceptorCMT</interceptor>
            <interceptor transaction="Container" metricsEnabled="true">org.jboss.ejb.plugins.MetricsInterceptor</interceptor>
            <interceptor transaction="Container">org.jboss.ejb.plugins.MessageDrivenInstanceInterceptor</interceptor>
            <!-- BMT -->
            <interceptor transaction="Bean">org.jboss.ejb.plugins.MessageDrivenInstanceInterceptor</interceptor>
            <interceptor transaction="Bean">org.jboss.ejb.plugins.MessageDrivenTxInterceptorBMT</interceptor>
            <interceptor transaction="Bean" metricsEnabled="true">org.jboss.ejb.plugins.MetricsInterceptor</interceptor>
            <interceptor>org.jboss.resource.connectionmanager.CachedConnectionInterceptor</interceptor>
         </container-interceptors>
         <instance-pool>org.jboss.ejb.plugins.MessageDrivenInstancePool</instance-pool>
         <instance-cache></instance-cache>
         <persistence-manager></persistence-manager>
         <container-pool-conf>
            <MaximumSize>100</MaximumSize>
         </container-pool-conf>
      </container-configuration>

   </container-configurations>

</jboss>
```

JNDI Properties

- Client application requires `client/jbossall-client.jar`
- JNDI Properties needed for client connections

```
java.naming.factory.initial=org.jnp.interfaces.NamingContextFactory
java.naming.factory.url.pkgs=org.jboss.naming:org.jnp.interfaces
java.naming.provider.url=localhost:1099
java.naming.security.principal=nobody
java.naming.security.credentials=
```

- ***Don't put a `jndi.properties` file with these properties in your application JAR, it will cause JNDI lookup failures!***

JBoss Security

- Declarative security based on Java Authentication & Authorization Service (JAAS)
- Uses LoginModules, Subjects to define Principals and Roles
- Define security policies in `server/default/conf/login-conf.xml`
- Map application to security domains/policies (must be present for security to work)
  - JAR `META-INF/jboss.xml`

```
<jboss>
   <security-domain>java:/jaas/other</security-domain>
</jboss>
```

  - WAR `WEB-INF/jboss-web.xml`

```
<jboss-web>
   <security-domain>java:/jaas/other</security-domain>
</jboss-web>
```

Common JBoss Security LoginModules
(can be chained)

- `org.jboss.security.auth.spi.IdentityLoginModule`
  - all authenticated Subjects mapped to one Principal
- `org.jboss.security.auth.spi.UsersRolesLoginModule`
  - authenticated username-password mappings in `users.properties`
  - authenticated username-role mappings in `roles.properties`
- `org.jboss.security.auth.spi.LdapLoginModule`
  - users and roles authenticated via LDAP
- `org.jboss.security.auth.spi.DatabaseServerLoginModule`
  - users and roles authenticated via logical `PRINCIPALS` and `ROLES` tables
- `org.jboss.security.auth.spi.RunAsLoginModule`
  - pushes run as role in order to authenticate additional secured login modules
- `org.jboss.security.ClientLoginModule`
  - ***Copies security & principal for server EJB invocation layer***

```
<?xml version='1.0'?>
<!DOCTYPE policy PUBLIC
      "-//JBoss//DTD JBOSS Security Config 3.0//EN"
      "http://www.jboss.org/j2ee/dtd/security_config.dtd">

<!-- The XML based JAAS login configuration read by the
org.jboss.security.auth.login.XMLLoginConfig mbean. Add
an application-policy element for each security domain.
-->
<policy>
    <!-- Used by clients within the application server VM such as
    mbeans and servlets that access EJBs.
    -->
    <application-policy name = "client-login">
       <authentication>
          <login-module code = "org.jboss.security.ClientLoginModule"
             flag = "required">
          </login-module>
       </authentication>
    </application-policy>

    <!-- A template configuration for the jmx-console web application. This
      defaults to the UsersRolesLoginModule the same as other and should be
      changed to a stronger authentication mechanism as required.
    -->
    <application-policy name = "jmx-console">
       <authentication>
          <login-module code="org.jboss.security.auth.spi.UsersRolesLoginModule"
             flag = "required" />
       </authentication>
    </application-policy>

    <!-- A template configuration for the web-console web application. This
      defaults to the UsersRolesLoginModule the same as other and should be
      changed to a stronger authentication mechanism as required.
    -->
    <application-policy name = "web-console">
       <authentication>
          <login-module code="org.jboss.security.auth.spi.UsersRolesLoginModule"
             flag = "required" />
       </authentication>
    </application-policy>

    <!-- The default login configuration used by any security domain that
    does not have a application-policy entry with a matching name
    -->
    <application-policy name = "other">
       <authentication>
          <login-module code = "org.jboss.security.auth.spi.UsersRolesLoginModule"
             flag = "required" />
       </authentication>
    </application-policy>
</policy>
```

Stock Trading Example

- Stateless Session Beans
  - **StockTrader**
    - **Collection getStocks()**
    - **Collection getStocksByExchange(String exchange)**
    - **OrderValue trade(*String exchange, String symbol, String action,***
      double price, double shares)
- CMP 2.x Entity Beans
  - Stock
    - exchange
    - symbol (primary key)
    - name
    - price
  - Order
    - id
    - stock
    - action (BUY|SELL)
    - date
    - price
    - Shares

Web Services

- "Web services...are services offered by one application to other applications via the World Wide Web. Clients of these services can aggregate them to form an end-user application, enable business transactions, or create new Web services."
- Use eXtensible Markup Language (XML) via HyperText Transfer Protocol (HTTP)
- Web Services XML format is Simple Object Access Protocol (SOAP)
  - Envelope describes message and how to process it
  - Encoding rules express instances of application-defined datatypes
  - Convention for representing remote procedure calls and responses
- SOAP Namespace (NS) Schemas

NS Prefix	Definition	Namespace URI
SOAP-ENV	SOAP Envelope	http://schemas.xmlsoap.org/soap/envelope/
SOAP-ENC	SOAP Encoding	http://schemas.xmlsoap.org/soap/encoding/
xsi	XML Schema instance	http://www.w3.org/2001/XMLSchema-instance
xsd	XML Schema definition	http://www.w3.org/2001/XMLSchema

Web Service Definition Language (WSDL)

- "WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information."
- Element Definitions
  - **Types**–a container for data type definitions using some type system (such as XSD).
  - **Message**–an abstract, typed definition of the data being communicated.
  - **Operation**–an abstract description of an action supported by the service.
  - **Port Type**–an abstract set of operations supported by one or more endpoints.
  - **Binding**–a concrete protocol and data format specification for a particular port type.
  - **Port**–a single endpoint defined as a combination of a binding and a network address.
  - **Service**–a collection of related endpoints.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://localhost/jboss-net/services/StockTrader"
    xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:apachesoap="http://xml.apache.org/xml-soap"
    xmlns:impl="http://localhost/jboss-net/services/StockTrader" xmlns:intf="http://localhost/jboss-net/services/StockTrader"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns1="http://net.jboss.org/jmx"
    xmlns:tns2="http://net.jboss.org/stocktrader" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <schema targetNamespace="http://net.jboss.org/jmx" xmlns="http://www.w3.org/2001/XMLSchema">
      <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
      <simpleType name="ObjectNameType"><simpleContent><extension base="xsd:string"/></simpleContent></simpleType>
    </schema>
    <schema targetNamespace="http://net.jboss.org/stocktrader" xmlns="http://www.w3.org/2001/XMLSchema">
      <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
      <complexType name="StockValue">
        <sequence>
          <element name="exchange" nillable="true" type="xsd:string"/>
          <element name="name" nillable="true" type="xsd:string"/>
          <element name="price" nillable="true" type="xsd:double"/>
          <element name="symbol" nillable="true" type="xsd:string"/>
        </sequence>
      </complexType>
      <complexType name="OrderValue">
        <sequence>
          <element name="action" nillable="true" type="xsd:string"/>
          <element name="date" nillable="true" type="xsd:dateTime"/>
          <element name="price" nillable="true" type="xsd:double"/>
          <element name="shares" nillable="true" type="xsd:double"/>
          <element name="stock" nillable="true" type="tns2:StockValue"/>
        </sequence>
      </complexType>
    </schema>
  </wsdl:types>
  <wsdl:message name="getStocksByExchangeResponse">
    <wsdl:part name="stocks" type="soapenc:Array"/>
  </wsdl:message>
  <wsdl:message name="tradeResponse">
    <wsdl:part name="order" type="tns2:OrderValue"/>
  </wsdl:message>
  <wsdl:message name="tradeRequest">
    <wsdl:part name="exchange" type="xsd:string"/>
    <wsdl:part name="symbol" type="xsd:string"/>
    <wsdl:part name="action" type="xsd:string"/>
    <wsdl:part name="price" type="xsd:double"/>
    <wsdl:part name="shares" type="xsd:double"/>
  </wsdl:message>
  <wsdl:message name="getStocksRequest">
  </wsdl:message>
  <wsdl:message name="getStocksResponse">
    <wsdl:part name="stocks" type="soapenc:Array"/>
  </wsdl:message>
  <wsdl:message name="getStocksByExchangeRequest">
    <wsdl:part name="exchange" type="xsd:string"/>
  </wsdl:message>
```

```
  <wsdl:portType name="StockTrader">
    <wsdl:operation name="getStocks">
      <wsdl:input message="impl:getStocksRequest" name="getStocksRequest"/>
      <wsdl:output message="impl:getStocksResponse" name="getStocksResponse"/>
    </wsdl:operation>
    <wsdl:operation name="getStocksByExchange" parameterOrder="exchange">
      <wsdl:input message="impl:getStocksByExchangeRequest" name="getStocksByExchangeRequest"/>
      <wsdl:output message="impl:getStocksByExchangeResponse" name="getStocksByExchangeResponse"/>
    </wsdl:operation>
    <wsdl:operation name="trade" parameterOrder="exchange symbol action price shares">
      <wsdl:input message="impl:tradeRequest" name="tradeRequest"/>
      <wsdl:output message="impl:tradeResponse" name="tradeResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="StockTraderSoapBinding" type="impl:StockTrader">
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="getStocks">
      <wsdlsoap:operation soapAction="StockTrader"/>
      <wsdl:input name="getStocksRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                       namespace="http://localhost/jboss-net/services/StockTrader" use="encoded"/>
      </wsdl:input>
      <wsdl:output name="getStocksResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://localhost/jboss-net/services/Sto
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getStocksByExchange">
      <wsdlsoap:operation soapAction="StockTrader"/>
      <wsdl:input name="getStocksByExchangeRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://localhost/jboss-net/services/Sto
      </wsdl:input>
      <wsdl:output name="getStocksByExchangeResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://localhost/jboss-net/services/Sto
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="trade">
      <wsdlsoap:operation soapAction="StockTrader"/>
      <wsdl:input name="tradeRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://localhost/jboss-net/services/Sto
      </wsdl:input>
      <wsdl:output name="tradeResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://localhost/jboss-net/services/Sto
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="StockTraderService">
    <wsdl:port binding="impl:StockTraderSoapBinding" name="StockTrader">
      <wsdlsoap:address location="http://localhost/jboss-net/services/StockTrader"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Apache Axis/JBoss.net

http://ws.apache.org/axis/

- Formerly ApacheSOAP (which was based on IBM SOAP4J)
- Pluggable/Embeddable SOAP Engine
- Java/WSDL Emitter (`wsdl2java`/`java2wsdl`)
- TCP Monitoring Tool (`tcpmon`)
- Axis embedded in JBoss as JBoss.net (JBoss Web Service provider)
- JBoss.net usually only available in the `all` configuration; just copy `server/all/deploy/jboss-net.sar` to `server/default/deploy`
- JBoss.net rooted at `/jboss-net` context
- AxisServlet
  - `/jboss-net/servlet/AxisServlet`
  - Displays available web services
  - WSDL available at `/jboss-net/services/<servicename>?wsdl`

Web Service Deployment

- JBoss.net implements an ***EJB Provider that exposes the methods of*** any session bean as a web service endpoint
- Exposed methods defined in a `web-service.xml` file
- Must be contained in the `META-INF` dir of a `.wsr` (JBoss specific Web Service Archive) file
- Deployment Hierarchy

```
stock.ear
    META-INF/application.xml
    stock.jar
      META-INF/ejb-jar.xml
      META-INF/jboss.xml
      META-INF/jbosscmp-jdbc.xml
      indyjug/jboss/ejb/*.class
    stock.wsr
      META-INF/web-service.xml
```

```
<?xml version="1.0" encoding="UTF-8"?>

<deployment
    name="StockTrader"
    xmlns="http://xml.apache.org/axis/wsdd/"
    targetNamespace="http://net.jboss.org/stocktrader"
    xmlns:stocktrader="http://net.jboss.org/stocktrader"
    xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">

<!-- The following are declarations of service endpoints targetted to
     session beans -->

  <service name="StockTrader" provider="Handler">
    <parameter name="handlerClass" value="org.jboss.net.axis.server.EJBProvider"/>
    <parameter name="beanJndiName" value="ejb/StockTrader"/>
    <parameter name="allowedMethods" value="getStocks getStocksByExchange trade"/>
    <operation name="getStocks" returnQName="stocks"/>
    <operation name="getStocksByExchange" returnQName="stocks">
      <parameter name="exchange"/>
    </operation>
    <operation name="trade" returnQName="order">
      <parameter name="exchange"/>
      <parameter name="symbol"/>
      <parameter name="action"/>
      <parameter name="price"/>
      <parameter name="shares"/>
    </operation>
    <requestFlow name="StockTraderRequest">
    </requestFlow>
    <responseFlow name="StockTraderResponse">
    </responseFlow>
  </service>

<!-- The following are typemappings for entity beans for implementing
     the implicit web-service value-object pattern -->

<!-- The following are typemappings for bean-type value-objects -->

  <typeMapping
      qname="stocktrader:StockValue"
      type="java:indyjug.jboss.ejb.StockValue"
      serializer="org.apache.axis.encoding.ser.BeanSerializerFactory"
      deserializer="org.apache.axis.encoding.ser.BeanDeserializerFactory"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />

  <typeMapping
      qname="stocktrader:OrderValue"
      type="java:indyjug.jboss.ejb.OrderValue"
      serializer="org.apache.axis.encoding.ser.BeanSerializerFactory"
      deserializer="org.apache.axis.encoding.ser.BeanDeserializerFactory"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</deployment>
```

Build Example

- Use Jakarta Ant to build
- Tasks
  - main → deploy → ear → jar,wsr → compile-resources → compile → init
  - generate-client generates Web Service client classes
    - **`java org.apache.axis.wsdl.WSDL2Java -o <output-dir> -a http://localhost/jboss-net/services/StockTrader?wsdl -p <package-name> [-t]`**
    - **`-t` generates JUnit testcase used by test task**
  - test → compile-client → generate-client → deploy
  - dbmgr starts Hypersonic DatabaseManager
  - java2html converts Java source to HTML

Summary

- Advantages
  - JBoss is very easy to use for J2EE beginners (NB: J2EE has a high learning curve!)
  - JBoss is also very customizable (subclassing and declarative definitions)
  - Hot deployment makes development very convenient!
  - Use 3.2 branch for production implementations
  - JBoss.net makes providing Web Services easy
- Disadvantages
  - Very long stacktraces due to interceptor stack
  - Some exceptions cryptic or based on earlier exceptions