

EJB failover

In Geronimo, you can configure [OpenEJB for failover scenarios](#), such as multipoint and multicast. Further more, Geronimo uses WADI to support failover for Stateful Session Bean (SFSB) in a unicast way.

- [Enabling multipoint/multicast failover](#)
 - [Developing the client](#)
 - [failover strategies](#)
- [Enabling unicast failover](#)
 - [Understanding unicast support in Geronimo](#)
 - [Enabling unicast support in Geronimo](#)
 - [Configuring the application](#)
 - [Configuring the client](#)

Enabling multipoint/multicast failover

By default, OpenEJB can use either **multicast** or **multipoint** strategies for failover scenarios. Basically the server cluster members maintain membership information using heartbeats and the ejb client maintains a list of servers to change target server based on availability. Each heartbeat packet contains a single URI that advertises a service, its cluster group, and its location in the form of "cluster1:ejb:ejbd://thehost:4201".

By updating different attributes in the `config-substitutions.properties` file, you can choose the appropriate discovery strategies in your cluster.

Multicast	MultiPoint
ServerHostname=<LocalIP> MulticastEnabled=true	ServerHostname=<LocalIP> MultipointEnable=true MultipointServers=<Remotehost/IP>:<port>

where

- *LocalIP* is the ip address of the current node
- *Remotehost/IP* is the host name or IP address of EJB member(s) in a cluster.
- *port* is the port number used to communicate with remote node. The port must not be the default openEJB server port 4201.

Note that for multicast scenario, you have to set `openejb.client.requestretry` property to *true* when starting up the Geronimo server. For example

```
./geronimo.sh run --long -Dopenejb.client.requestretry=true
```

Developing the client

EJB clients must know the address of EJB members to make sure that its request will be handled with high availability, regardless of whether EJB members are clustered in a **Multipoint** or **Multicast** way.

See the following code snippet for your client application development for remote host lookup.

URI for Multicast

```
Properties p = new Properties();
p.setProperty(Context.INITIAL_CONTEXT_FACTORY, "org.
apache.openejb.client.RemoteInitialContextFactory");
p.setProperty(Context.PROVIDER_URL, "multicast://239.
255.3.2:6142?group=cluster1");
...
```

URI for Multipoint/Unicast

```
Properties p = new Properties();
p.setProperty(Context.INITIAL_CONTEXT_FACTORY, "org.
apache.openejb.client.RemoteInitialContextFactory");
p.setProperty(Context.PROVIDER_URL, "failover:
ejbd://ejbd://foo:4201");
...
```

failover strategies

OpenEJB supports 3 kinds of failover strategies when the original node was shut down abnormally.

round-robin

```
Properties p = new Properties();
p.setProperty(Context.PROVIDER_URL, "failover:round-robin:ejbd://foo:4201,ejbds://bar:4201,multicast://239.
255.2.3:6142");
...
```

sticky

```
Properties p = new Properties();
p.setProperty(Context.PROVIDER_URL, "failover:
ejbd://foo:4201,ejbd://bar:4201");
...
```

random

```
Properties p = new Properties();
p.setProperty(Context.PROVIDER_URL, "failover:random:
ejbd://foo:4201,ejbd://bar:4201");
...
```

Enabling unicast failover

You can configure **unicast-based** support for EJB clustering. WADI can be configured to use unicast instead of multicast to support failover for Stateless Session Bean.

Understanding unicast support in Geronimo

The following server configurations, shipped out-of-the-box with the Java EE assemblies, must be installed and enabled for the unicast EJB failover to work:

- *org.apache.geronimo.configs/openejb-clustering-wadi//car*: it defines runtime dependencies and OpenEJB clustering contracts.
- *org.apache.geronimo.configs/openejb-clustering-builder-wadi//car*: it defines deployment time dependencies along with a *OpenEJBClusteringBuilder* GBean declaring the default clustering configuration.

This configuration must be running when a cluster is started. If it is not, then the substitution group *openejb-clustering-wadi* is not properly recognized and the deployment fails.

Enabling unicast support in Geronimo

To enable unicast-based failover for Stateless Session Bean, the wadi-clustering module in <GERONIMO_HOME>/var/config/config.xml should be configured for each server node. You must specify the real IP address to *ServerHostname* and the node name to *ClusterNodeName* for each static member in the config-substitutions.properties file. Each member has its own GBean configuration in the wadi-cluster module in config.xml. Ensure that you define all the static members in it. The following example shows the configuration with two static members.

```
...
<module name="org.apache.geronimo.configs/wadi-clustering/3.0-SNAPSHOT/car">
  <gbean name="DefaultBackingStrategyFactory">
    <attribute name="nbReplica">${ReplicaCount}</attribute>
  </gbean>
  <gbean name="DefaultDispatcherHolder">
    <attribute name="disableMCastService">true</attribute>
    <attribute name="receiverPort">4002</attribute>
    <attribute name="endPointURI">${EndPointURI}</attribute>
    <attribute name="clusterName">${WADIClusterName}</attribute>
    <reference name="staticMember">
      <pattern>
        <groupId>org.apache.geronimo.configs</groupId>
        <artifactId>wadi-clustering</artifactId>
        <version>2.2.1-SNAPSHOT</version>
        <type>car</type>
        <name>firstStaticMember</name>
      </pattern>
    </reference>
  </gbean>
  <gbean name="org.apache.geronimo.configs/wadi-clustering/3.0-SNAPSHOT/car?ServiceModule=org.apache.geronimo.configs/wadi-clustering/3.0-SNAPSHOT/car,j2eeType=GBean,name=firstStaticMember"
    gbeanInfo="org.apache.geronimo.clustering.wadi.WadiStaticMember">
    <attribute name="className">org.apache.catalina.tribes.membership.StaticMember</attribute>
    <attribute name="port">4002</attribute>
    <attribute name="securePort">-1</attribute>
    <attribute name="domain">test-domain</attribute>
    <attribute name="UniqueId">0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0</attribute>
    <attribute name="host">Node1_IP</attribute>
    <reference name="nextWadiStaticMember">
      <pattern>
        <groupId>org.apache.geronimo.configs</groupId>
        <artifactId>wadi-clustering</artifactId>
        <version>2.2.1-SNAPSHOT</version>
        <type>car</type>
        <name>secondStaticMember</name>
      </pattern>
    </reference>
  </gbean>
  <gbean name="org.apache.geronimo.configs/wadi-clustering/3.0-SNAPSHOT/car?ServiceModule=org.apache.geronimo.configs/wadi-clustering/3.0-SNAPSHOT/car,j2eeType=GBean,name=secondStaticMember"
    gbeanInfo="org.apache.geronimo.clustering.wadi.WadiStaticMember">
    <attribute name="className">org.apache.catalina.tribes.membership.StaticMember</attribute>
    <attribute name="port">4003</attribute>
    <attribute name="securePort">-1</attribute>
    <attribute name="domain">test-domain</attribute>
    <attribute name="UniqueId">0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1</attribute>
    <attribute name="host">Node2_IP</attribute>
    <reference name="nextWadiStaticMember" />
  </gbean>
</module>
...
```

where

Node1_IP is the IP address of the first server node.
Node2_IP is the IP address of the second server node.

Configuring the application

The Geronimo-specific deployment plan for an EJB application, which is usually packaged as an EJB JAR file, is called "**openejb-jar.xml**". The **openejb-jar.xml** deployment plan is used in conjunction with the **ejb-jar.xml** Java EE deployment plan to deploy enterprise applications to the Geronimo application server.

To enable WADI configuration for your application, add `openejb-clustering-wadi` into the deployment plan as follows:

```
<ejb-jar xmlns="http://geronimo.apache.org/xml/ns/j2ee/ejb/openejb-2.0"
  xmlns:wadi="http://geronimo.apache.org/xml/ns/openejb-clustering-wadi-1.2">
...
  <wadi:openejb-clustering-wadi>
    <wadi:deltaReplication>false</wadi:deltaReplication>
  </wadi:openejb-clustering-wadi>
</ejb-jar>
```

Configuring the client

An application client must find an EJB node and make sure that the client could find another node when the original one was shut down abnormally in a unicast scenario. Use the following code snippet when programming your client applications.

```
Properties p = new Properties();
p.setProperty(Context.INITIAL_CONTEXT_FACTORY, "org.apache.openejb.client.RemoteInitialContextFactory");

p.setProperty(Context.PROVIDER_URL, "ejbd://foo:4201");
Context context1 = new InitialContext(p);
...
p.setProperty(Context.PROVIDER_URL, "ejbd://bar:4201");
Context context2 = new InitialContext(p);
```