



iBus[®]

Softwired

Softwired Integration Paper

**SAP WAS 6.4 SP11
Integration with
iBus//Mobile 6.0**

www.softwired-inc.com

© Softwired 2006

Abstract

This paper contains a step-by-step guide to integrating Softwired's wireless messaging Gateway with the JMS server of SAP's Netweaver product. Such integration extends Netweaver's JMS connectivity to mobile client devices over wireless networks.

This document has been prepared specifically for POST Denmark to reflect their Windows 2003 operating systems environment and SAP Web application server integration with Softwired's iBus//Mobile for deployment to this environment.

Outline of the integration

The Gateway process of iBus/Mobile is a proxy JMS client for client applications running on programmable mobile devices. The Gateway connects wireless clients to Netweaver by repeating the client applications WJMS operations on the proxy JMS connection, and by forwarding JMS messages to and from the JMS (SAP) backend and the mobile device. Essentially integration with SAP JMS means that the Gateway instantiates a JMS client runtime specific to Netweaver, to do this the Gateway configuration must be changed.

Instantiating a Netweaver client runtime.

The stepwise mechanics of instantiating a Netweaver specific JMS client are as follows:

1. Gateway instantiates (via java reflection) a JNDI implementation that is specific to Netweaver. The Gateway uses 2 JNDI contexts, for Topics and for Queues.
2. Gateway retrieves a Netweaver JMS ConnectionFactory from the JNDI. Such factory objects will reside at a well-known location in the JNDI, and must fulfill an interface defined by JMS.
3. The ConnectionFactory is used to obtain a JMS connection to Netweaver. JMS prescribes separate ConnectionFactory's for Topics and for Queues.

To instantiate the correct JNDI object the following is needed:

1. The Gateway must know the class name of the Netweaver JNDI object.
2. The Gateway process must have access to the java class files that contain the JNDI object.
3. The Netweaver JNDI object must be told where Netweaver is running.

To retrieve ConnectionFactory objects the following is needed:

1. The Gateway must know the correct credentials to be allowed to retrieve objects from JNDI, the principle (user name) and password.
2. The Gateway must know the location in JNDI, and the names under which the JMS ConnectionFactories are found.
3. The Gateway process must have access to the java class files that contain the ConnectionFactory objects.

To obtain a JMS connection from the connection factories the following is needed:

1. The Gateway must know the correct username and password to create a connection.
2. The Gateway process must have access to the java class files that contain the JMS Connection objects.

All of the above is setup by changing:

1. The configuration file that the Gateway process reads at start time.
2. The CLASSPATH of the Gateway process JVM.

Gateway internal JMS destinations

Once the correct JMS client runtime is created the Gateway will subscribe and publish to a set of JMS destinations (all of them are Topics), these destinations must have been created, and be accessible for reading and writing by the Gateway process. JMS prescribes that JMS destinations are retrieved from JNDI, the names under which the Gateway retrieves the internal destinations are included in the on-line documentation of iBus//Mobile. These destinations and notes are repeated below.

Topics and Queues used internally by iBusMobile//Gateway

The following table lists the JMS destinations used by the iBus//Mobile Gateway internally. Some Gateway subsystems (such as Administration and clustered LoadManager) need access to these destinations to function correctly. Note that JMS destination that are used by client programs are not listed here. The mechanism for creating JMS destinations, and their location in JNDI can vary depending on the JMS back-end system with which the Gateway is integrated. Steps needed to create the destinations, and allow access to them, should be found in the administration guide of the specific JMS.

Destination Type	Name	JNDI-Name
Topic	Admin-Client	iBusMobile_Admin_transport_client
Topic	Admin-System	iBusMobile_Admin_transport_system
Topic	Admin-Standard-Gateway (gateway-1)	iBusMobile_Admin_gateway_gateway-1
Topic	System-Info	iBusMobile_Admin_SystemInfo
Topic	Load reports (clustered Gateways)	iBusMobile_Clustering_LoadReports
Topic	Used to publish events	iBusMobile_Events_GatewayEvents
	Used for Web Services	iBusMobile_webservice
Default topic and queue for testing		
Topic	Test-Topic	topic1
Queue	Test-Queue	queue1

Table 1: JMS Topics and Queues Required by iBus//Mobile

Additional notes:

- iBus//MessageServer the JMS implementation packaged with iBus//Mobile does not require administrative steps to create destinations.
- The JNDI look-ups that the Gateway process does (which include those for JMS destinations) can be seen by enabling 'NameServer' level '1' logging in the Gateway config file.
- The JNDI lookups for JMS destinations can be redirected by setting 'QueueNamePrefix' or

'TopicNamePrefix' in the Gateway config file.

- Gateway configuration contains a set of characters that will not be allowed in JMS destination names, see 'destinationInvalidCharacters'.

Passwords

Two sets of passwords must be known and entered in the Gateway configuration:

1. JNDI lookup credentials. For looking up both ConnectionFactories and JMS Destinations.
2. Connection creation user name and password for JMS connections used to administer the Gateway.

Configuring the Gateway

At start time the Gateway process reads an XML syntax file from which various configurable settings are obtained. The Gateway configuration file usually resides in the directory 'ibusmobile/gateway/etc'. The usual name of the file is 'ibusmobile_config.xml'. A Netweaver specific configuration file is available with the packaged release of iBus//Mobile this file is called 'ibusmobile_config_SAP.xml' it reflects the configuration as described here. The name of the file that the Gateway will read for configuration is set by the '-config' command line parameter.

As indicated above the following are the configurable settings that the Gateway needs.

JNDI settings:

The JNDI setup is configured in the 'JndiProperties' property group it defines values that the Gateway will pass to the Netweaver JNDI object:

```
<!-- ===== -->
<!-- JNDI Setup -->
<!-- ===== -->
<PROPERTYGROUP
  name="JndiProperties"
  description="Various provider-specific properties that are required in JNDI.">
  <PROPERTY
    name="java.naming.provider.url"
    description="JNDI Provider URL for topics/queues (optional)"
    value="localhost:50004">
  </PROPERTY>
  <PROPERTY
    name="java.naming.security.principal"
    description="Context.SECURITY_PRINCIPAL (optional)"
    value="Administrator">
```

```

</PROPERTY>
<PROPERTY
    name="java.naming.security.credentials"
    description="Context.SECURITY_CREDENTIALS (optional)"
    value="admin">
</PROPERTY>
</PROPERTYGROUP>

```

The JNDI property group contains:

1. The location (DNS hostname and port) at which Netweaver is running: "localhost: 50004".
2. The principle and credentials that allow JNDI lookups.

Class name of the JNDI object

The Gateway instantiates 2 JNDI objects, for topics and for queues. The names of the JNDI objects are provided in the Gateway configuration.

```

<PROPERTY
    name="JndiContextFactoryForTopics"
    description="JNDI Factory class used to look up Topics and
TopicConnectionFactories."
    value="com.sap.engine.services.jndi.InitialContextFactoryImpl">
</PROPERTY>
<PROPERTY
    name="JndiContextFactoryForQueues"
    description="JNDI Factory class used to look up Queues and
QueueConnectionFactories."
    value="com.sap.engine.services.jndi.InitialContextFactoryImpl">
</PROPERTY>

```

Note that the same class can return JMS ConnectionFactories for both Topics and for Queues.

JNDI location of Topic and Queue ConnectionFactories:

```

<PROPERTY
    name="TopicConnectionFactory"
    description="JMS TopicConnectionFactory default-name to look up in JNDI."
    value="jmsfactory/default/TopicConnectionFactory">
</PROPERTY>
<PROPERTY
    name="QueueConnectionFactory"
    description="JMS QueueConnectionFactory default-name to look up in JNDI."

```

```
        value="jmsfactory/default/QueueConnectionFactory">
    </PROPERTY>
```

The Gateway will ask the JNDI object to return the objects at these locations. The Gateway expects these to fulfill JMS ConnectionFactory contracts.

Prefixes to JNDI locations for topics and queues:

```
<PROPERTY
    name="TopicNamePrefix"
    description="Topic's location in the JNDI."
    value="jmsttopics/default/">
</PROPERTY>
<PROPERTY
    name="QueueNamePrefix"
    description="Queue's location in the JNDI."
    value="jmsqueus/default/">
</PROPERTY>
```

Every time the Gateway looks up a JMS destination these prefixes will be prepended to the destination name to give the JNDI lookup location that the Gateway uses.

Administration passwords:

```
<!-- ===== ->
<!-- Administration GUI settings -->
<!-- ===== ->
<PROPERTYGROUP
    name="Admin">
    <PROPERTY
        name="adminConnectionLogin"
        description="Login for admin JMS connection of the Gateway."
        value="Administrator">
    </PROPERTY>
    <PROPERTY
        name="adminConnectionPassword"
        description="Password for admin JMS connection of the Gateway."
        value="admin">
    </PROPERTY>
    ...
```

The username and password used to create JMS connections for the Gateway administration system. These connections are to Gateway internal destinations.

Tomcat username and password:

Note that the Tomcat (catalina) server that is embedded in the Gateway process must share these Administration passwords. The tomcat username and password must be entered in the tomcat users file at:

```
.../gateway/tomcat/conf/tomcat-users.xml
```

with an entry of the form:

```
<user name="Administrator" password="admin" roles="mobile-admin" />
```

CLASSPATH of the Gateway process

Usually the Gateway process is started by a script file called 'startgateway.[bat|sh]'. As with the Gateway configuration file a Netweaver specific start script is available with the packaged release of iBus//Mobile this file is called 'startgateway_SAP.[bat|sh]'. The start script sets two values that are important for integration:

1. Set the CLASSPATH to include the jar files needed to run the Netweaver client runtime.
2. Point the Gateway process to the correct configuration (the -config parameter). So startgateway_SAP points to 'ibusmobile_config_SAP.xml'.

For WebAS 6.4 the necessary jar files are:

```
.../usr/sap/J2E/JC00/j2ee/j2eeclient/sapj2eeclient.jar  
.../usr/sap/J2E/JC00/j2ee/j2eeclient/jms.jar  
.../usr/sap/J2E/JC00/j2ee/j2eeclient/exception.jar  
.../usr/sap/J2E/JC00/j2ee/j2eeclient/logging.jar
```

Usually the startgateway script sets the CLASSPATH via a JMS_CLASSPATH variable that is included in the final CLASSPATH setting.

Integration tips and tricks

Some suggestions for determining the status of integration:

1. The Gateway process writes a log file, which contains useful logging statements about what the process is doing. The log file location is ibusmobile/gateway/gatewaylog.txt.
2. When first running the integration it is advisable to increase logging in some Gateway classes. Logging can be adjusted in the configuration file by setting the 'logmodules' property and the 'loglevel'. The set of classes that will provide information useful to integration:

```
<PROPERTY  
    name="logmodules"
```

```
description="Module names to log."
value="MobileGateway NameServer AdminDispatcher AdminRequestor
AdminConnectionBean">
</PROPERTY>
```

3. The NameServer object will print the stack trace of any caught Exceptions if the following java system property is set on the JVM: -
Dibus.PrintStackTraceOnException.ch.softwired.gateway.admin.gateway.NameServer=true.