1. Introduction

1.1. Project/Component Working Name:
Message Queue integration

1.2. Name(s) and e-mail address of Document Author(s)/Supplier:
Satish Kumar [sats@dev.java.net, satish.kumar@Sun.COM]
[with inputs from Siva Kumar T., Ramesh Parthasarathy, Sanjeeb Sahoo].

1.3. Date of this Document:
v 1.0 - Draft

2. Project Summary
2.1. Project Description:

To outline the new features and packaging changes required to modularize SUN MQ and the glue code to enable integration of the JMS module with GlassFish V3.

2.2. Risks and Assumptions:
The integration depends on timely delivery of binaries from the MQ team.

3. Problem Summary

3.1. Problem Area:

The modular architecture of V3 poses new requirements on the way modules are packaged and integrated into it. To address these requirement, the JMS module and the integration code would need changes in the way they are packaging and delivered to GF. The MQ administration services would also need to integrate into the new GF admin framework. This project aims to integrate the latest version of the Message Queue product with the Application Server.

3.2. Justification:
Plug-able integration of JMS services into GF without impacting performance.

4. Technical Description:
4.1. Details:
Packaging: The MQ binaries are currently delivered to GlassFish as a file archive (JAR). The broker lifecycle management (for embedded and local JMS integration modes) is currently handled by the JMSRA that is shipped by SUN MQ. by In conformance to the V3 theme of modularity, MQ would need to package their module as an OSGi bundle. OSGi bundles are archives which include metadata regarding their contents and information how they can be run. Further details on OSGi are availability here ñ http://
wiki.glassfish.java.net/attach/V3FunctionalSpecs/GFv3Prelude-OSGi-onepager-

v0.2.txt. Details on how to add a new module in GF V3 are here - http://wiki.glassfish.java.net/Wiki.jsp?page=AddingModule

Mode of delivery:
Currently the MQ binaries are manually staged in the GF java.net maven repository. For V3, the plan is to explore the possibility of the MQ team directly uploading the MQ bundles to the GlassFish Maven repository (in download.java.net) so they can be consumed by GF.

JMS API ñ
The JMS API is currently duplicated in two locations ñ included with javaee.jar and shipped with the MQ binaries. In V3, we intend to remove this duplication and use the API provided by the MQ team and will be no longer bundle with Javaee.jar

Separation of client and broker binaries -

IF a user is running GF V3 with MQ in the remote mode, only the client libraries are required. Hence, in this scenario a user would only need to download the client bundle(containing the client jars and the RA only). Currently MQ does not provide separate client and broker bundles. The bundles required would depend on the integration type configured for the installation. GF would need to ensure that the appropriate libraries are available for the particular installation type.

Port Unification -
Port unification is an important theme of GF V3 where by all services would be accessible though a common port. We will need to explore the option of unifying the JMS ports also and making it accessible though the unified port. Currently, the JMS module is assigned two distinct ports ñ one for the port mapper utility (7676 by default) and one RMI port. Port unification in GF V3 is provided by Grizzly through filters that are part of the request processing chain. This throws up two new requirements -
An API or algorithm provided by MQ to decipher if an incoming request, based on it bytes is a MQ request.
A filter that would plug into the Grizzly framework and query the above API to determine if the request is indeed meant for MQ.
Lockhard Integration / Admin console
GF V3 provides an extensible framework for plugging administration components. MQ should be able to plugin its admin services to this framework
MYSQL

GF V3 will support MYSQL for HA (this needs to be conformed. At this moment we are not sure which profile this will be in). We would need to explore the option of sharing drivers and resources.

RA Consolidation

There are currently three JCA 1.5 compliant Resource Adapters with a lot of functional overlap from SUN. Two of these are java.net projects. There is an effort to consolidate these three RA's into a single RA.

4.2. Bug/RFE Number(s):
Need to be identified

4.3. In Scope:

4.4. Out of Scope:

4.5. Interfaces:

4.5.1 Exported Interfaces

4.5.2 Imported interfaces
This integration depends on the interfaces exposed by the Sun Java System Message Queue Resource Adapter for broker lifecycle control, MQ "HA-cluster" integration.

4.5.3 Other interfaces (Optional)

4.6. Doc Impact:
Application Server's Developer's Guide and Administration Guide.

4.7. Admin/Config Impact:
WIP

4.8. HA Impact:
None

4.9. I18N/L10N Impact:
None

4.10. Packaging & Delivery:
See above

4.11. Security Impact:
None

4.12. Compatibility Impact
None

4.13. Dependencies:
Dependent on Sun Java System Message Queue 4.x

5. Reference Documents:

6. Schedule:
6.1. Projected Availability:
Aligns with GlassFish V3 JavaOne 2009 release