ExoLab OpenJMS

Java™ Message Service Messaging Engine

Jim Alateras
alateras@exoffice.com

Jim Mourikis
mourikis@exoffice.com

Features Overview

- JMS 1.0.2 compliant
- JDK 1.2 compliant
- Point-to-point messaging
- Publish-subscribe messaging
- Transacted sessions
- JAAS support
- Persistence adapter
- ORB adapter
- More at www.openjms.org

Overview

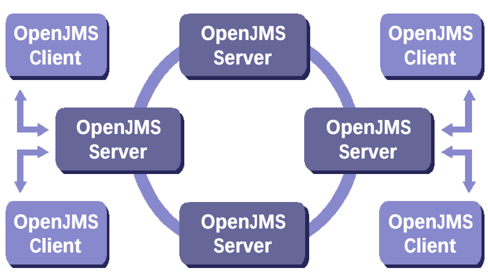
The Java™ Message Service (JMS), a messaging infrastructure defined by Sun Microsystems, Inc. provides a common way for Java™ programs to create, send, receive and read enterprise messaging system's messages. OpenJMS is an Open Source implementation of version 1.0.2 of the JMS specification, which supports both publish-subscribe and point-to-point messaging semantics in addition to reliable and guaranteed Quality of Service (QoS) delivery models.

Quality of Service

A higher Quality of Service inevitably requires more processing resources than a lower one. The JMS specification defines two levels of reliable and guaranteed service, with the latter offering the highest Quality of Service. The OpenJMS messaging engine provides an environment where clients with differing Quality of Service requirements can coexist without adversely affecting the overall performance of the others.

Scalability

Scalability and high-availability concerns are addressed through a peer-to-peer communication protocol between servers. This feature allows multiple servers to be connected together to build a federation, facilitating inter-server client communication.



Federated Configuration of OpenJMS Servers

Security

Security considerations, such as authentication and authorization, are addressed through the Java™ 2 Security Model. OpenJMS relies on security policies to provide access control to resources such as queues and topics, securely restricting the capabilities of publishers and subscribers accessing those destinations. In addition, OpenJMS delegates the responsibility of securing all end-to-end communications to the underlying Object Request Broker (over IIOP, RMI or any other protocol).

Transactions

OpenJMS supports transacted sessions with the Java™ Transaction Service (JTS) and distributed transactions via the JTA XAResource API. The Tyrex Transaction Processing Monitor implements both JTS and JTA.

Deployment

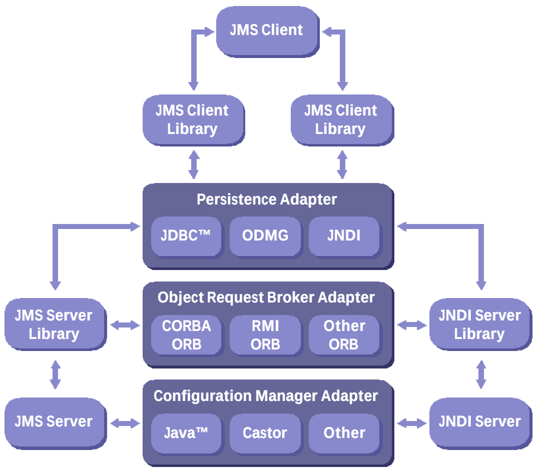
OpenJMS has been designed and packaged so that it can be easily deployed as an embeddable component or run as a standalone service. In embedded mode it is integrated and controlled by a parent component, such as an application server, which is then responsible for managing all aspects of configuration and lifecycle. Alternatively, if run in standalone mode OpenJMS takes full responsibility for these tasks.

Administration & Monitoring

OpenJMS supports both console-based and web-based administration and monitoring. The Administration Facility is capable of configuring all aspects of the server including persistence, debugging, resource restrictions, etc. Similarly, the Monitoring Facility is capable of starting, stopping and supervising the OpenJMS messaging engine. The engine can be configured to run with its own embedded web server or can run with any other web server supporting the Servlet API.

Architecture

OpenJMS has been designed with an adapter-based architecture relying on open and extensible interfaces to the persistence mechanisms, Object Request Broker, and configuration manager.



Adapter-based OpenJMS Architecture

Persistence Mechanism

OpenJMS can be configured to use an RDBMS, an OODBMS, or the plain file system as persistence mechanism. Such a persistence mechanism is used to support durable subscribers and destinations through a JNDI service provider, as well as a storage area for messages requiring guaranteed delivery.

Object Request Broker

OpenJMS provides a generic interface allowing any Object Request Broker to be used for supporting protocols like IIOP or RMI. The OpenORB CORBA Object Request Broker is used to support RMI-IIOP.

Configuration Manager

The configuration manager offers the ability to use several types of configuration file formats. Java™ property files, as well as Castor configuration files are supported by the current OpenJMS implementation.

Features

- JMS 1.0.2 compliant
- JDK 1.2 compliant
- Point-to-point messaging
- Publish-subscribe messaging
- Reliable and guaranteed QoS delivery models
- Synchronous and asynchronous message delivery
- Hierarchical topic namespace
- JAAS authentication and authorization
- Transactional and non-transactional sessions
- Web-based and console-based administration
- Adapter-based architecture
- CORBA, RMI and proprietary ORB support
- JDBC™, ODMG and JNDI-based persistence
- Multi-threaded messaging engine
- Embeddable or standalone service
- Open Source licensing (ExoLab Public License)

www.exolab.org
info@exolab.org
1-650-259-9796
1-603-719-9409