Diplomarbeitspräsentationen der Fakultät für Informatik

Design and Implementation of an extended, serverless JMS Java Messaging Service using the XVSM Shared Data Space

Master Studium:
Software Engineering & Internet Computing

Robert Pitschadell

Technische Universität Wien
Institut für Computersprachen
Arbeitsbereich: Space Based Computing
Betreuerin: Ao.Univ.-Prof. Dr. Dipl.-Ing. eva Kühn

Introduction

Enterprise Application Integration (EAI) connects different independent systems, distributed in different divisions and departments of a company, into a large, integrated solution. Message oriented Middlewares (MOM) are applicable for such distributed systems with high demand on data transfer, reliability and scalability. The great advantage compared to remote procedure calls (RPC) is the asynchronous communication. This offers a loosely coupled communication in time, space and synchronisation. It is possible to provide a reliable business system even if some parts have been dropped. Besides many proprietary systems, JMS (Java Message Service), a standard developed by Sun Microsystems, established in the Java segment over the years. Most JMS implementations run in a client server architecture.
Objective of this master thesis is to analyse existing JMS providers and to develop a serverless JMS Java API implementation. The release is constructed on the top of an extended virtual shared memory XVSM system. XVSM provides a shared data space for communication and collaboration of several autonomous software components. It offers possibilities to reach requirements like reliability and high scalability with a serverless approach.

JMS Java Messaging Service

JMS offers an interface simplifying message exchange and supports different messaging strategies. On the one hand, companies implement the JMS interfaces. They become a JMS provider and offer message exchange with the defined JMS behaviour. On the other hand client applications use the JMS interfaces to develop their business logic without taking care of the message transfer realisation of the selected JMS provider. This enterprise integration standard offers two channel types. A point-to-point channel realises a consuming message queue. The second channel offers publish/subscribe functionality to deliver all published messages to all subscribers.

PTP Point to Point

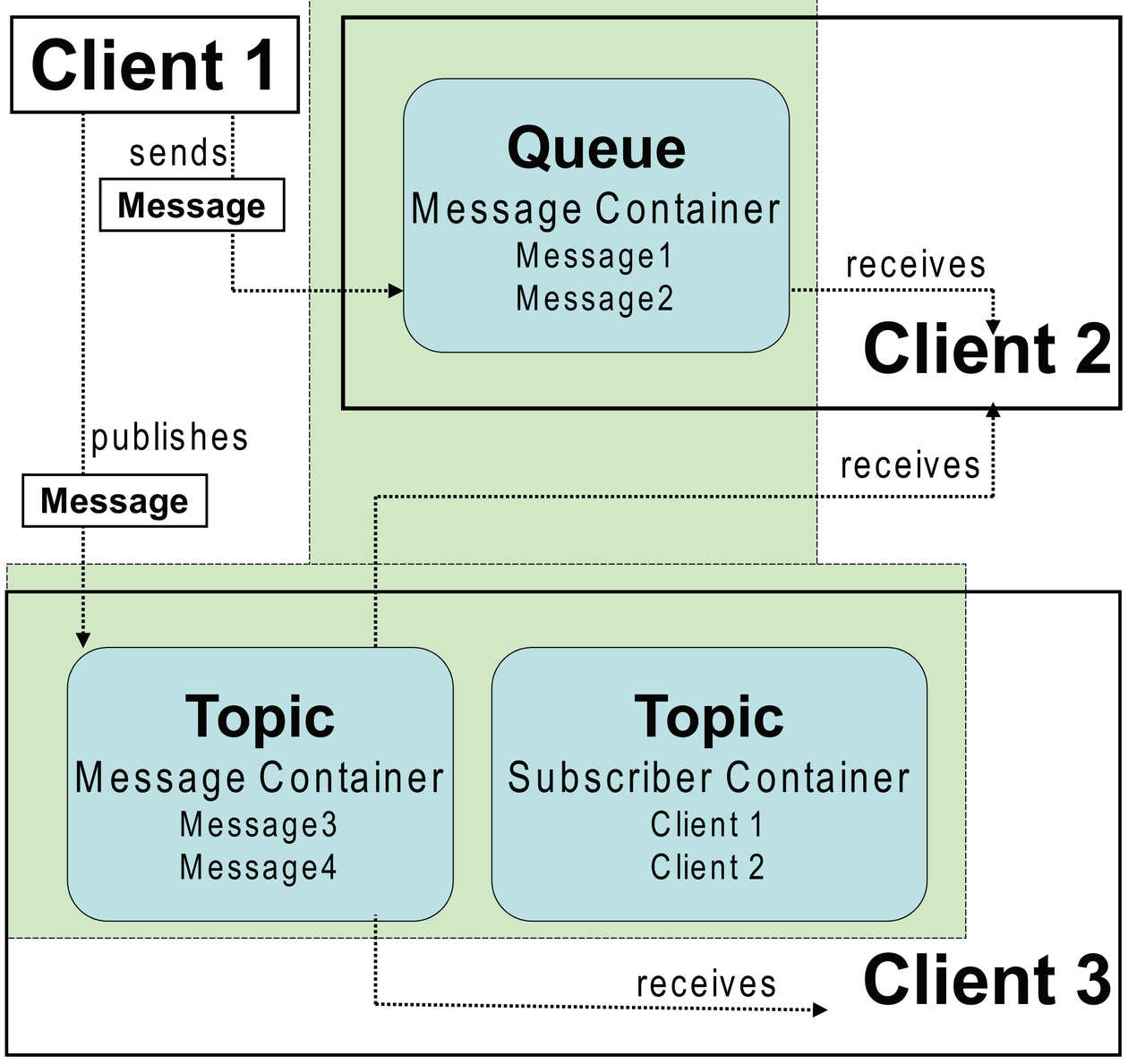
Point to point communication works with a JMS Queue, which is similar to a mailbox. Different clients address a message with a destination and send it to the specified JMS Queue. If the receiver client is online, it fetches its messages from its JMS Queue. The point to point model can be a one to one client communication or more specific a many to one peer communication. Most times there is only one consumer for a JMS Queue, if it works e.g. like a message box for one client. The amount of consumers is not restricted, but only one consumer gets the same message of the JMS Queue. The queue model is used, when every message has to be processed by one consumer. A JMS queue is realized in the shared data space with one message container and a FifoPriority coordinator, which stores the messages in a queue behavior enriched with priority levels.

Public Subscribe

The publish/subscribe model works with a JMS Topic, which is similar to a newsgroup. Clients publish messages to a JMS Topic. Clients can subscribe to a JMS Topic. All subscribed clients get the published messages of the Topic. The publish/subscribe model can be used to realize one to one or many to many communication. A message can have multiple consumers. Normally only online subscribers receive the messages missing all messages published while they are offline. This can be avoided with the additional feature for JMS Topic, the durable subscription. This publish subscribe model is used when a message has to be processed by none, one or many consumers. A JMS Topic is realized in the XVSM-JMS shared data space with a message repository container and additional containers for subscriptions.

XVSM Shared Data Space

The eXtended virtual shared memory XVSM is a technology that belongs to the space based computing paradigm. A user interacts with the space by storing and retrieving data objects. XVSM offers collaboration structures to develop distributed systems much faster and safer. It provides coordination containers to retrieve data objects in a specific manner or order. E.g. a FIFO coordinator sorts their data objects in a first in first out order, like a queue behavior. Every client acts as an independent peer instead of a client server system. A Client can be part of the data space by storing data objects or be only participant by querying and inserting data objects with the space.
The Figure below shows a part of the XVSM-JMS data space. Client 2 is a part of the data spaces and stores one replication of the message container for a specific JMS queue. Client 3 stores one replication of the message and subscriber container for a specific JMS topic. Client 1 stores no data, but sends a message to the JMS queue and another message to the JMS topic. Client 2 is the receiver for the queue and is subscribed to the JMS topic, therefore it receives both messages. Client 3 is subscribed to the JMS topic and receives only that message.



Further outlook and evaluation

The XVSM-JMS system can be extended by several useful features, but they should not disturb the main JMS specification to keep compatibility. One realized extension is to combine the two models, PTP and Pub/Sub, to an extended destination. The client decides for each message the behavior of the destination and how much receivers should receive the message.
Another advantage of the XVSM-JMS provider is the serverless architecture. The JMS parts in XVSM container can be distributed in many different ways and the strategies can be changed automatically according to requirements. The benefit is a system, which is scaling very well and avoiding the bottleneck of a centralized JMS server.
The XVSM system supports aspect programming that gives the ability to extend application functionalities at runtime. This can be easily used to extend XVSM-JMS with additional features like authentication, encryption, resource management etc.

Robert_Pitschadell@gmx.at