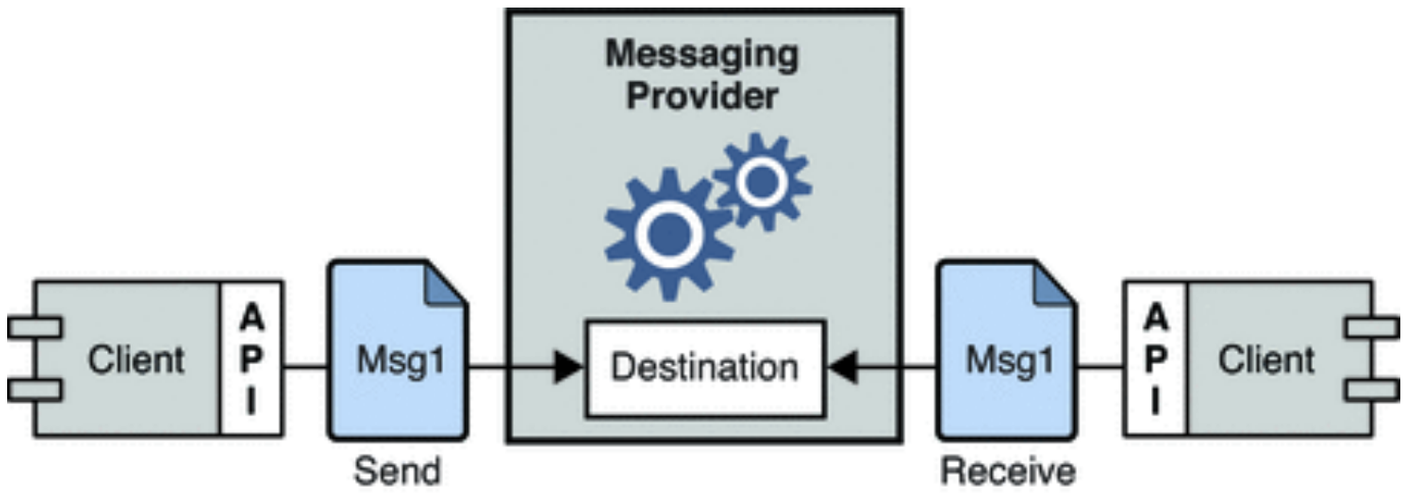
Communication

- Request-response pattern
    - Tightly coupled software components
    - Synchronous communication (ex. Knock knock jokes)
- Messaging
    - Method of communicating between s/w components
    - Loosely coupled components
    - Peer-to-peer
        - Client can send to/receive from any other client
        - Not client-server

Message Oriented Middleware (MOM)

- A MOM provider (aka broker) mediates messages between sender and receiver

- Main components: clients, messages, MOM provider which includes an API and admin tools



- Administrative interface allows tuning reliability, security, scalability, and performance

Image source: docs.oracle.com

MOM Advantages

- Asynchronous communication
    - Message queues provide temporary storage when the receiver is busy or not connected
- Routing (implementation-dependent)
    - A message may be broadcast to multiple receivers
- Transformation
    - A sender can send a message in its native format. Multiple receivers can receive the message in their native formats.

MOM Disadvantages

- The added component (MOM Provider) might reduce performance or reliability
- might increase the cost to maintain
- MOM doesn’t work for inherently synchronous system (ex real-time or near-real-time systems)

Java Message Service

A MOM Implementation

JMS

- Specification is non-proprietary
- Point-to-point messaging
- Publish-subscribe messaging
- Synchronous and asynchronous receipt of messages
- Reliable communication

JMS API Architecture

- JMS Provider
    - Implements Java Message Service
    - Has admin controls
- JMS Clients
    - Programs written in Java
    - Produce/Consume messages
- Messages
    - Objects sent between JMS clients
    - javax.jms.Message

Point-to-Point Message Domain

- Message queues, Senders, and Receivers
- Each message is addressed to a specific queue
- Queues hold all messages until the messages are consumed or expire
- Each message has only one consumer
- Sender and Receiver are loosely coupled
- Receiver acknowledges each message

Publish/Subscribe Message Domain - 1

- Message topics, Publishers, Subscribers
- A topic is like a bulletin board
- A topic can have multiple publishers and multiple subscribers
- Topics retain messages as long as it takes to deliver them to the current subscribers*

Publish/Subscribe Message Domain - 2

- Publishers and Subscribers have a timing dependency
    - A client who subscribes to a topic can only consume messages after the client has connected.
    - The client must continue to be connected for it to consume messages.
-

Publish/Subscribe Message Domain - 3

- * Durable subscriptions
  - Receive messages sent while the subscriber is not active
  - Allows flexibility and reliability of queues and sending to multiple clients