Chapter 12

by

David G. Messerschmitt

Communication services

by

David G. Messerschmitt
Example communication services

- Messages

- Queuing and multiplexing
  - Message with reply
    - Remote method invocation

- Conversation
- Broadcast

Some objectives

- Allow modules to interact in the same way across different hosts as on a single host
- Make it easier to develop applications by capturing common generic needs
- Support a range of applications with distinct needs
### Message

- The smallest unit of information meaningful to the sender and recipient
  - Recipient must be ready and listening
- Directs or informs the recipient
  - Pure push model
- Employs a send-receive protocol

### Supporting deferred applications

- Queuing
  - Queuing allows a message to be sent even though the recipient is not waiting
  - The message sits in a queue somewhere in the infrastructure until it is actively retrieved by the recipient
  - Combines send-receive and request-reply protocols
Participants in message with queuing

Sender → Queue → Recipient

Sender can send message whenever it wants
Recipient can access message whenever it wants

Multiplexing

- Multiplexing
  - Allows recipient to receive messages from two or more senders
  - How are the senders identified?
  - Presumes queuing: what if two messages are sent simultaneously?

Supported by Message Oriented Middleware (MOM)
Multiplexing and queuing

Sender1
Mux & queue
Sender2
Infrastructure
Recipient

Recipient deals with one message at a time even if coming from many senders

Example application: workflow

Workers
Administrative assistants
Purchasing
Receiving
**What message is not**

- Delivery may not be guaranteed
- No shared context of messages from same sender to recipient
  - Application is free to create that context on its own
  - Messages may not be delivered in the same order as sent

**Message with reply**

- Service couples two messages:
  - Request and coupled response message
  - Sender freed of burden of associating response
- Requests information or a service
  - Pure pull model
- Immediate
  - Recipient presumes sender is waiting for reply and responds as quickly as possible
Remote method invocation (RMI)

Method invocation

Method: parameters return_values

An action object-oriented programming is called a method.

Computes method()

Sender is blocked waiting for response

Client object Server object

Blocked: can’t do anything else

= Blocked waiting for response = Working on request
### Comparison

<table>
<thead>
<tr>
<th>Sender</th>
<th>Recipient</th>
<th>Queue</th>
<th>Recipient</th>
</tr>
</thead>
</table>

- **Send/receive**
- **Request/response**

- Work on message or request
- Blocked waiting for response

### Conversation

- Two-way exchange of messages
- Messages in one direction are influenced by messages in the other

#### To maintain conversation
- Messages should be delivered reliably
- Messages should be delivered in the same order as sent
- No coupling of replies with messages
A session typically provides guaranteed ordered delivery to maintain proper context.

Some advantages of sessions:

- Aids application by managing a shared context for a conversation
  - Identification, ordering, reliable delivery
- Reduces overhead
  - Example: only one name server query
- Security (Chapter 8)
- Quality of service (Chapter 11)
Examples

• What are examples of applications that would naturally leverage:
  – Messages?
  – Queuing and multiplexing?
  – Message with reply?
  – Conversation?

Multimedia record-store-playback

Audio coder  Audio decoder

Record  One (big) message  Playback

Storage  Storage

Video coder  Video decoder
Streaming multimedia

A session supports streaming multimedia

Understanding Networked Applications 21 A First Course

Understanding Networked Applications 22 A First Course
**Delay and latency**

Message latency = time from sending of message until it is received in its entirety

Delay = time elapsing between an event (or series of events) and its (their) representation

**Relationship of delay and latency**

Delay = latency (of network) + message accumulation time

If delay is important, keep the messages short!