CS 550:

Advanced Operating Systems

Message- and Stream-Oriented Communication

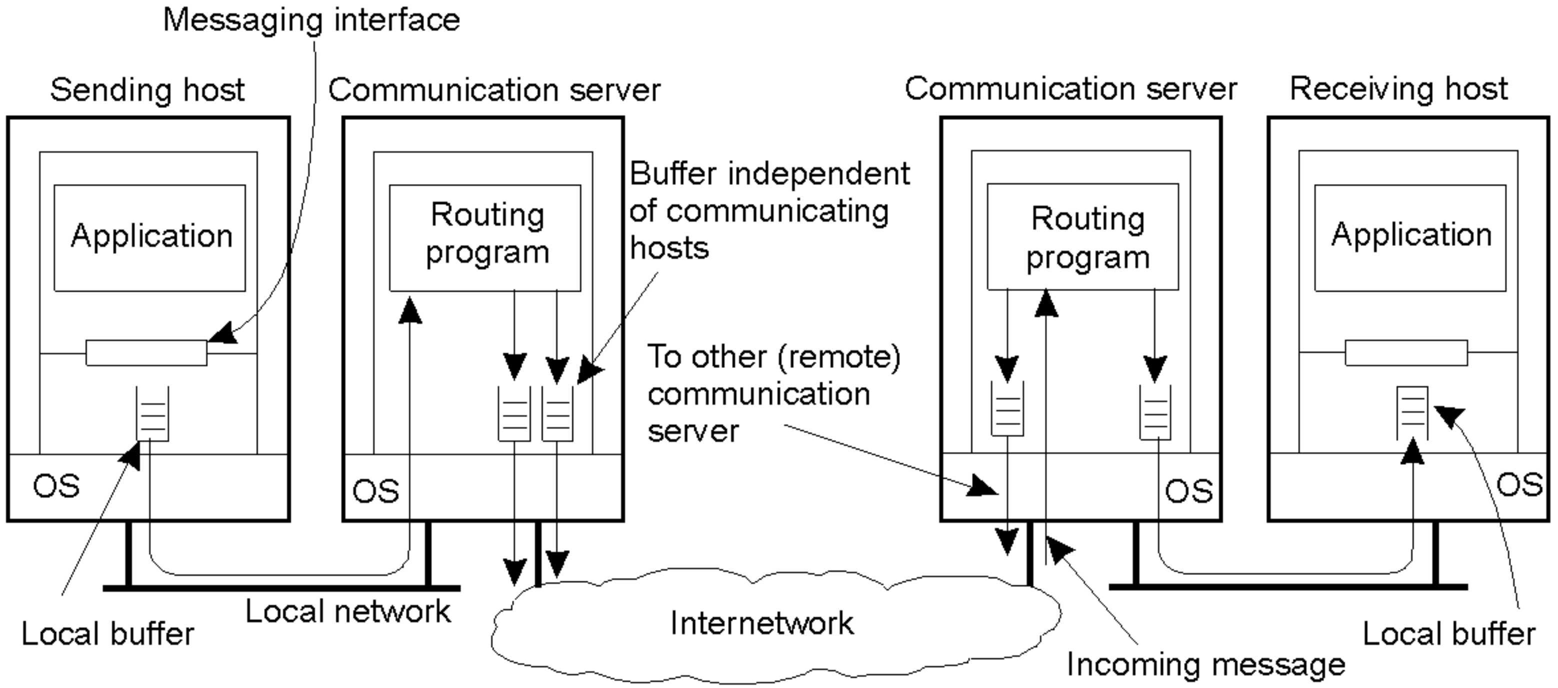
Ioan Raicu
Computer Science Department
Illinois Institute of Technology

CS 550
Advanced Operating Systems
February 8th, 2011

Outline

- Message-oriented communication
  - Persistence and synchronicity
  - Message-oriented transient communication
    - Berkeley socket
    - MPI
  - Message-oriented persistent communication
- Stream-oriented communication
  - Data stream
  - Quality of services
  - Stream synchronization

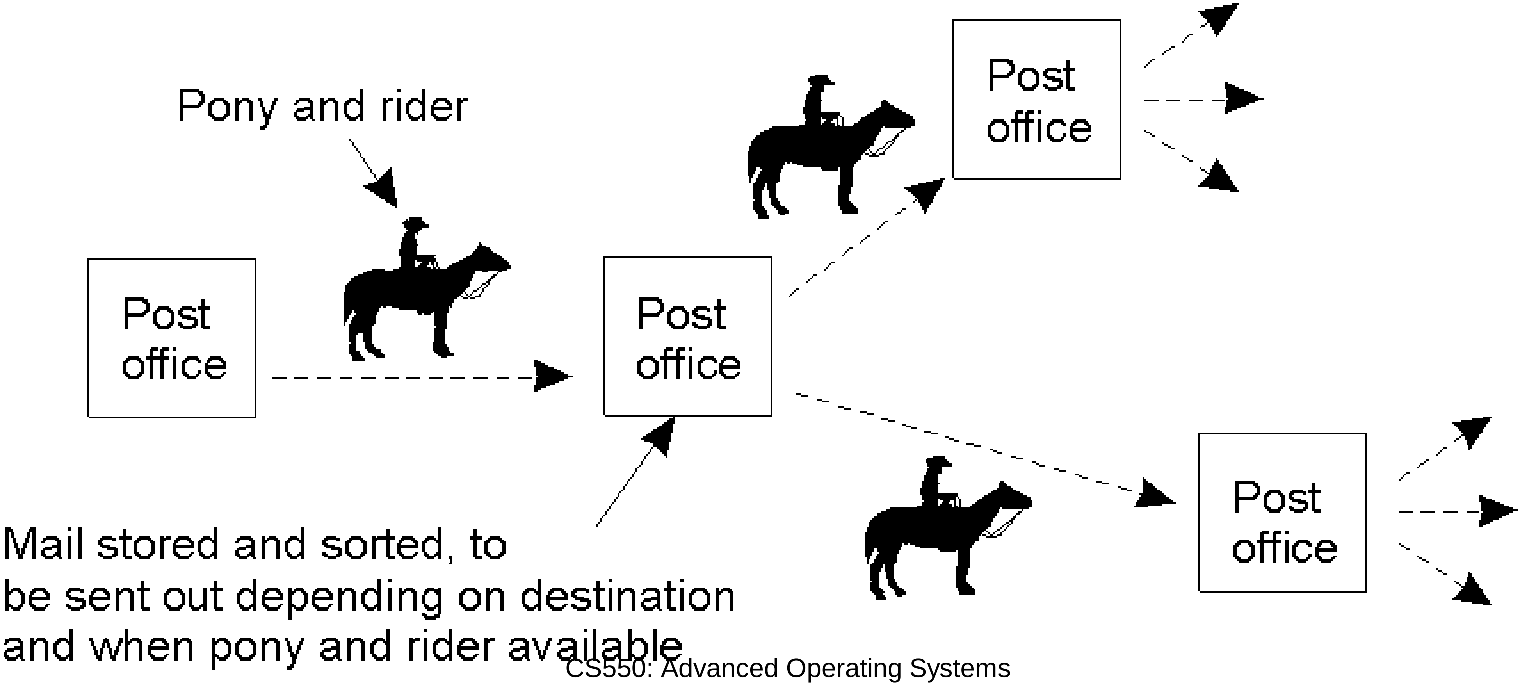
Example: Communication System



Example: e-mail system

Persistence

- Persistent communication
  - Definition:
  - Examples: email, pony express


Pony and rider
Post office
Post office
Post office
Post office
Mail stored and sorted, to be sent out depending on destination and when pony and rider available

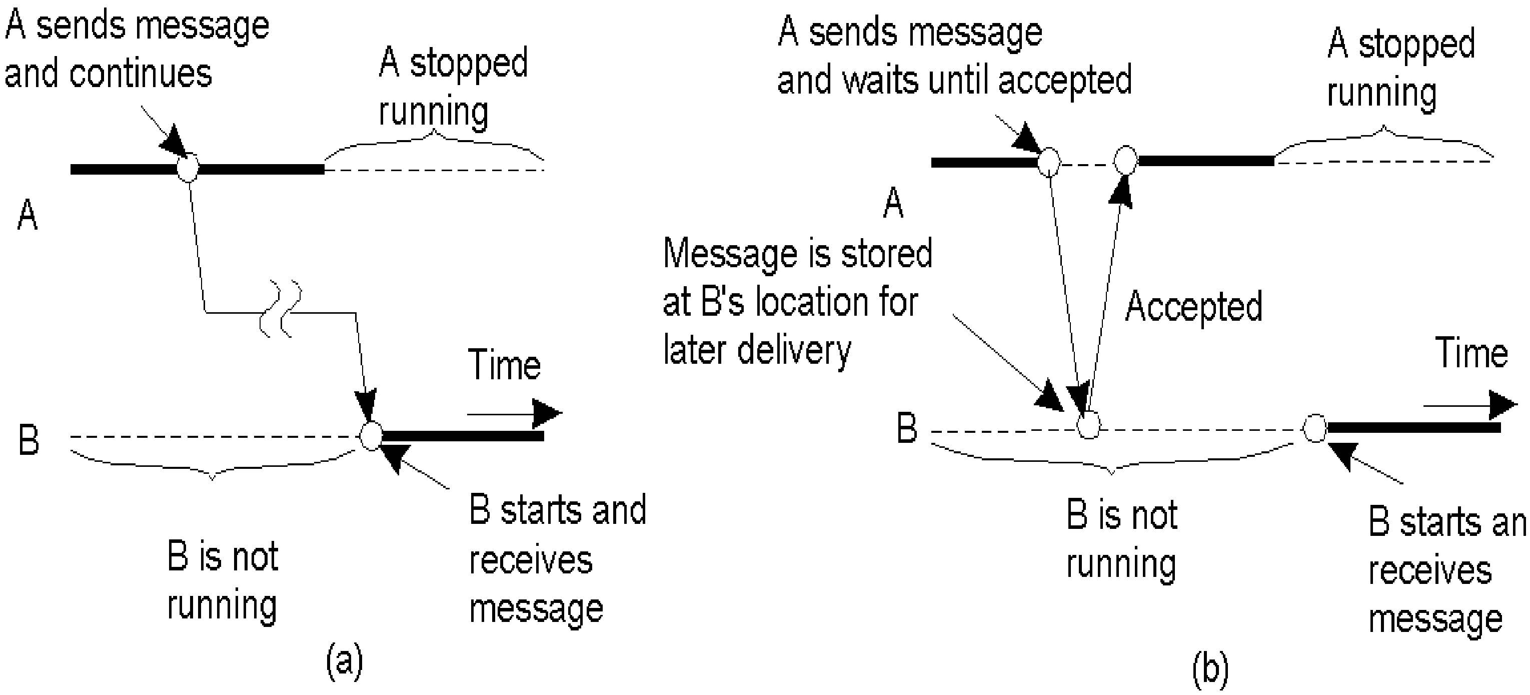
Persistence

- Transient communication
  - Example: transport-level communication services offer transient communication
  - Example: Typical network router

Synchronicity

- Asynchronous communication
- Synchronous communication
- Six combinations of persistence and synchronicity

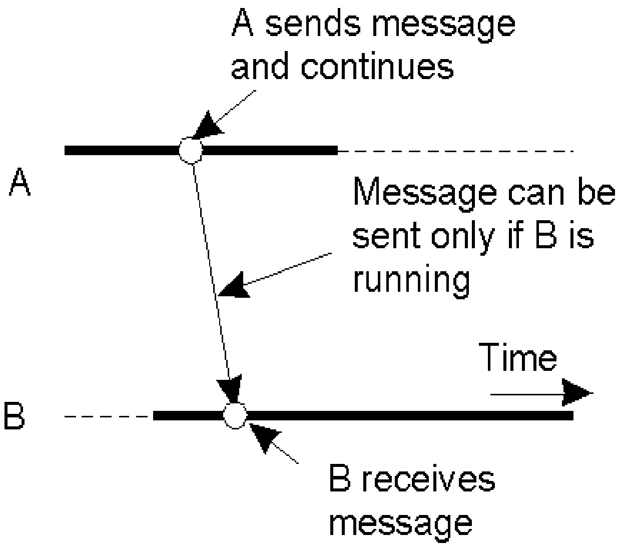
Persistence and Synchronicity Combinations


A sends message and continues
A stopped running
A
B
Time
B is not running
B starts and receives message
(a)
A sends message and waits until accepted
A stopped running
A
Message is stored at B's location for later delivery
Accepted
Time
B
B is not running
B starts and receives message
(b)

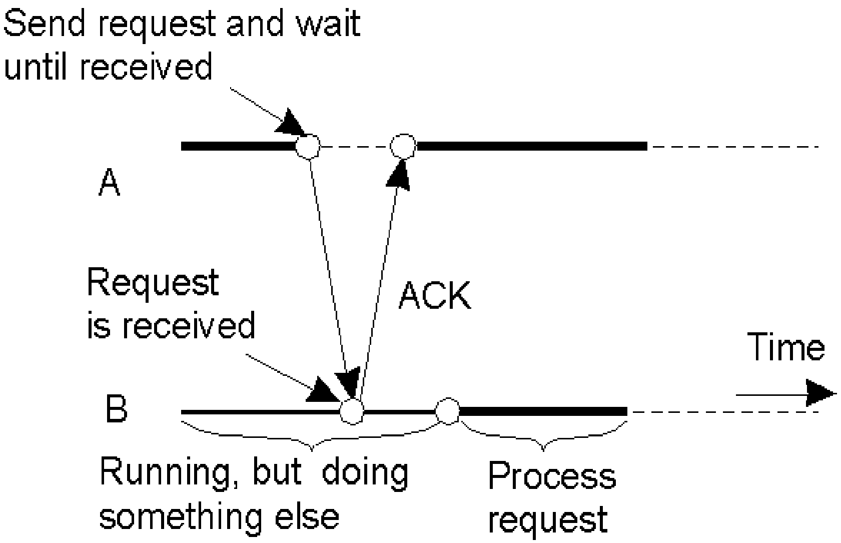
a) Persistent asynchronous communication (e.g., email)

b) Persistent synchronous communication

Persistence and Synchronicity Combinations



(c)

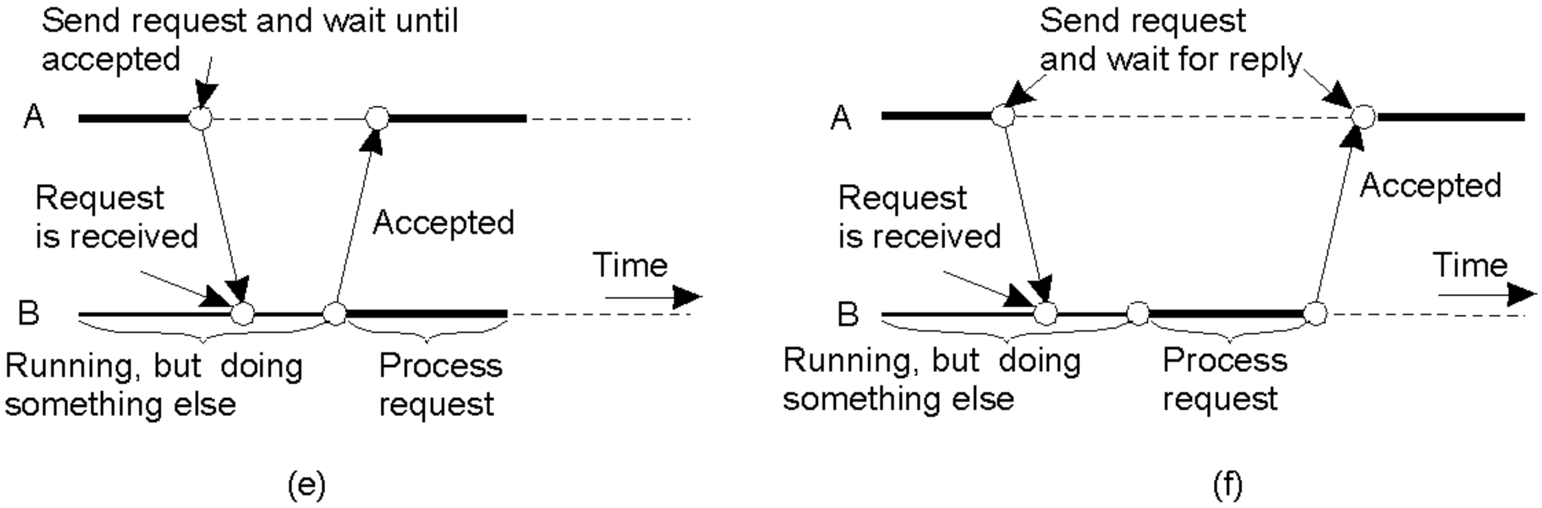


(d)

c) Transient asynchronous communication (e.g., UDP)

d) Receipt-based transient synchronous communication

Persistence and Synchronicity Combinations


Send request and wait until accepted
A
Request is received
Accepted
Time
B
Running, but doing something else
Process request
(e)
Send request and wait for reply
A
Accepted
Request is received
Time
B
Running, but doing something else
Process request
(f)

e) Delivery-based transient synchronous communication at message delivery (e.g., asynchronous RCP)

f) Response-based transient synchronous communication (RPC)

Summary of Synchronicity

- Persistent communication
  - Messages are stored by communication middleware for as long as needed to ensure delivery of message
  - Example: email
- Transient communication
  - Messages are stored by communication middleware only for as long as the sending and receiving application are executing
  - Example: TCP/UDP
- Asynchronous communication
  - Sender continues immediately after message sent
- Synchronous communication
  - Sender blocks until the request is known to be accepted

Persistence and Synchronicity: Comments

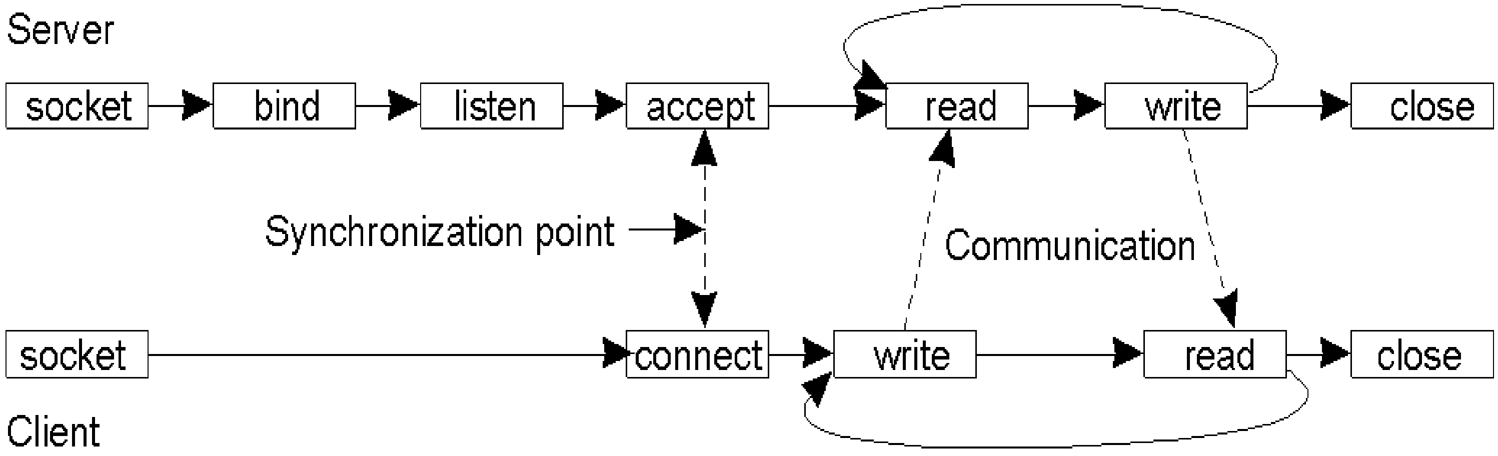
- Transient synchronous comm: response-based, delivery-based and reply-based

- Transient asynchronous comm: message-passing systems

- Persistent comm: developing of middleware for large-scale interconnected networks; failure masking and recovery

Message-oriented Transient Communication

- Many distributed systems built on top of simple message-oriented model
  - Example: Berkeley sockets
  - Socket?



Berkeley Socket Primitives

Primitive	**Meaning**
Socket	Create a new communication endpoint
Bind	Attach a local address to a socket
Listen	Announce willingness to accept connections
Accept	Block caller until a connection request arrives
Connect	Actively attempt to establish a connection
Send	Send some data over the connection
Receive	Receive some data over the connection
Close	Release the connection

Message-Passing Interface (MPI)

- Sockets designed for network communication (e.g., TCP/IP)
  - Support simple send/receive primitives
  - Use general-purpose protocol stacks such as TCP/IP
- Abstraction not suitable for other protocols in clusters of workstations or massively parallel systems
  - Need an interface with more advanced primitives
- Large number of incompatible proprietary libraries and protocols
  - Need for a standard interface

Message-Passing Interface (MPI)

- Message-passing interface (MPI)
  - Hardware independent
  - Designed for parallel applications (uses transient communication)
- Key idea: communication between groups of processes
  - Each endpoint is a *(groupID, processID)* pair
- Support most of the forms of transient communication (c)-(f)

MPI Primitives

Primitive	Meaning
MPI_bsend	Append outgoing message to a local send buffer
MPI_send	Send a message and wait until copied to local or remote buffer
MPI_ssend	Send a message and wait until receipt starts
MPI_sendrecv	Send a message and wait for reply
MPI_isend	Pass reference to outgoing message, and continue
MPI_issend	Pass reference to outgoing message, and wait until receipt starts
MPI_recv	Receive a message; block if there are none
MPI_irecv	Check if there is an incoming message, but do not block

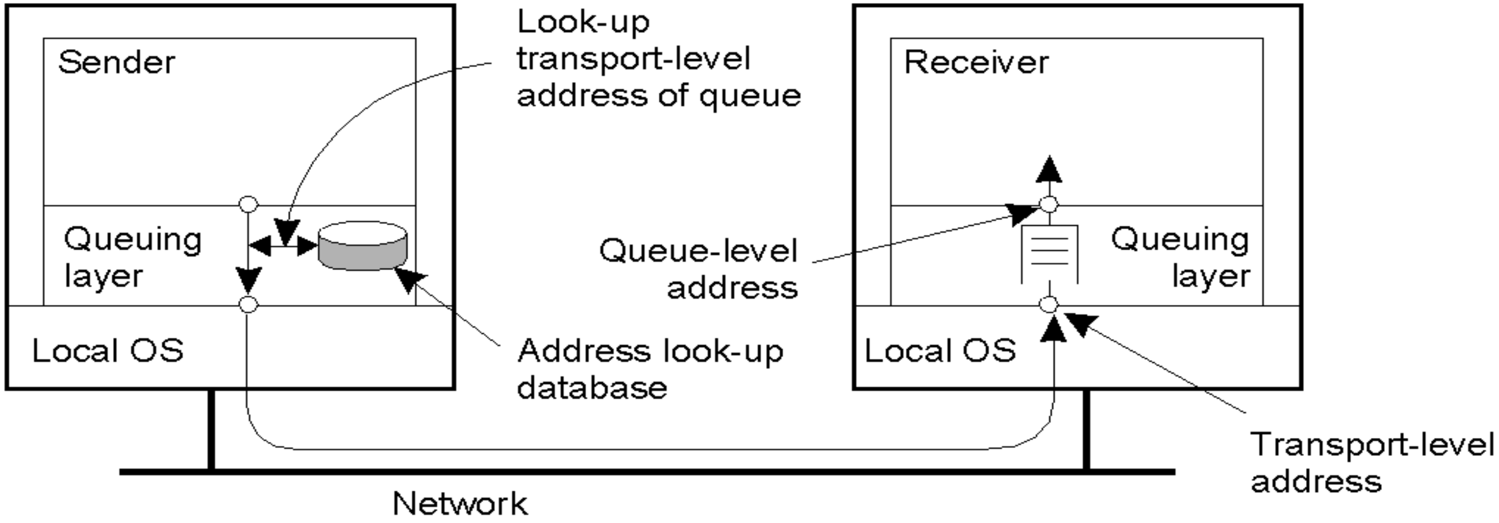
- MPI reference: http://www.mcs.anl.gov/mpi/

Message-oriented Persistent Communication

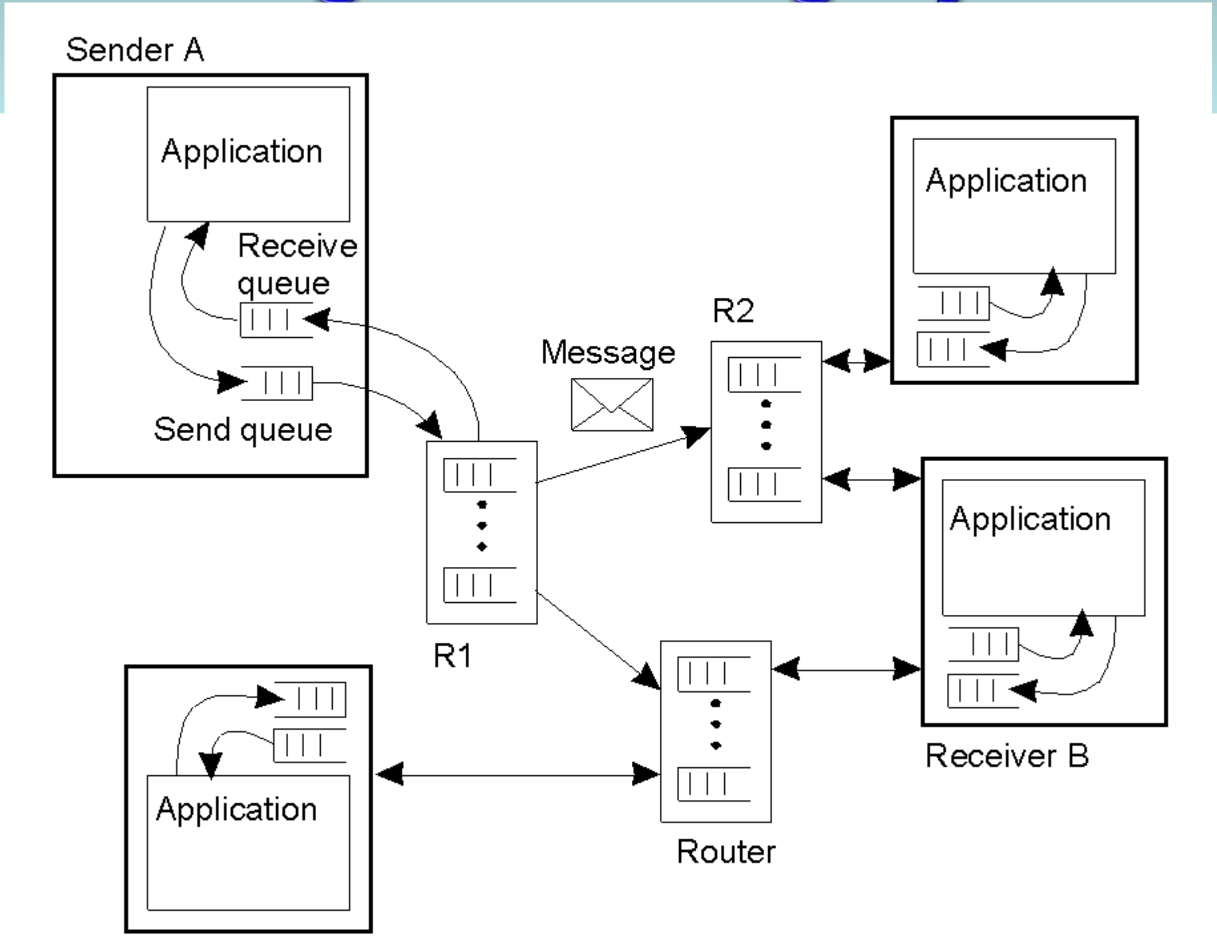
- Message queuing systems or Message-Oriented Middleware (MOM)
  - Support asynchronous persistent communication
  - Intermediate storage for message while sender/receiver are inactive
  - Example application: email
- Communicate by inserting messages in queues
- Sender is only guaranteed that message will be eventually inserted in recipient’s queue
  - When/if the message will be read?

Message-Queuing Model

- General architecture of MOM



Message-Queuing System



The general organization of a message-queuing system with routers

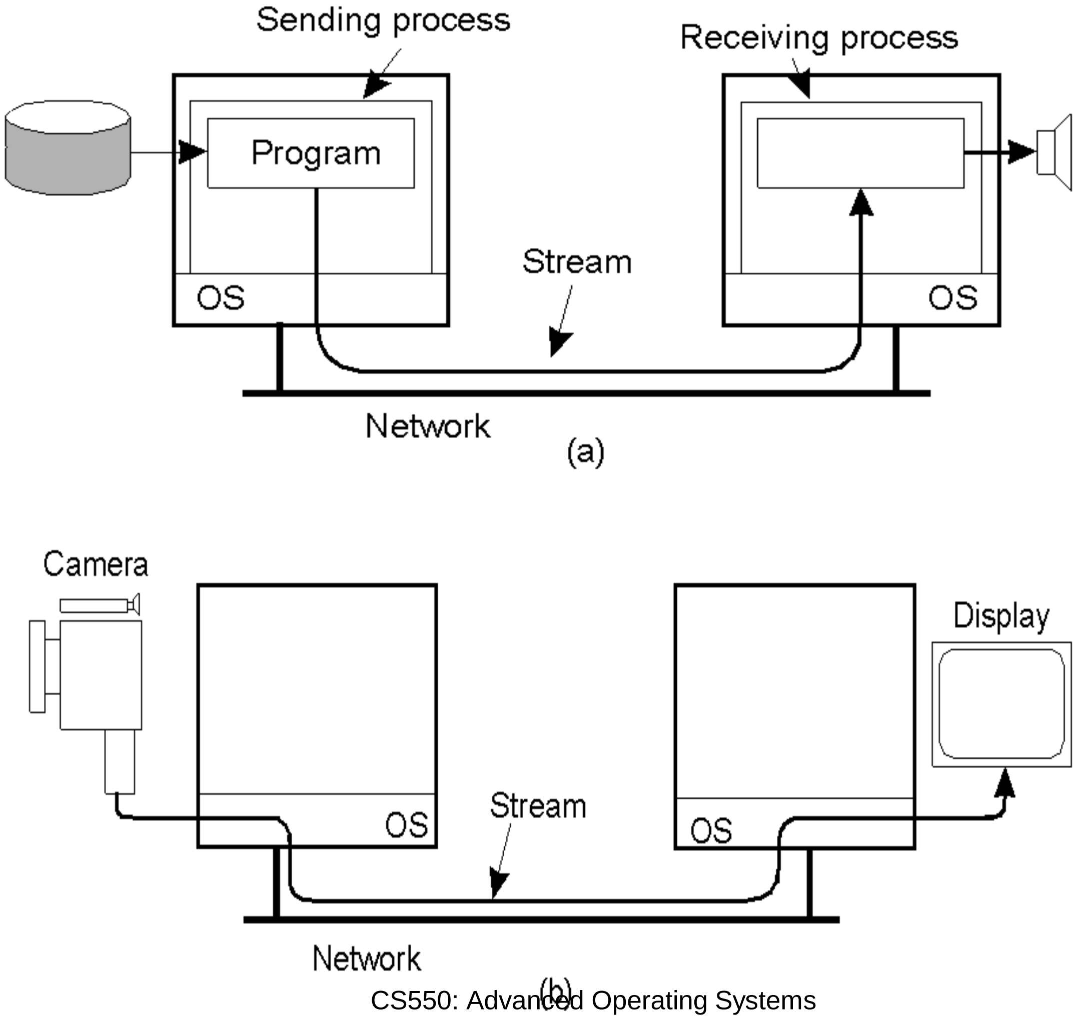
Stream Oriented Communication

- Message-oriented communication: request-response
  - When communication occurs and speed do not affect correctness
- Timing is crucial in certain forms of communication
  - Examples: audio and video (“continuous media”)
  - 30 frames/s video => receive and display a frame every 33ms
- Stream oriented comm is required!

Data Stream

- A data stream is a sequence of data units
- Discrete or continuous:
  - Discrete stream
  - Continuous stream
- For continuous stream, three transmission modes:
  - Asynchronous transmission mode
    - No timing requirements
  - Synchronous transmission mode
    - Maximum end-to-end delay
  - Isochronous transmission mode
    - Both minimum and maximum end-to-end delay

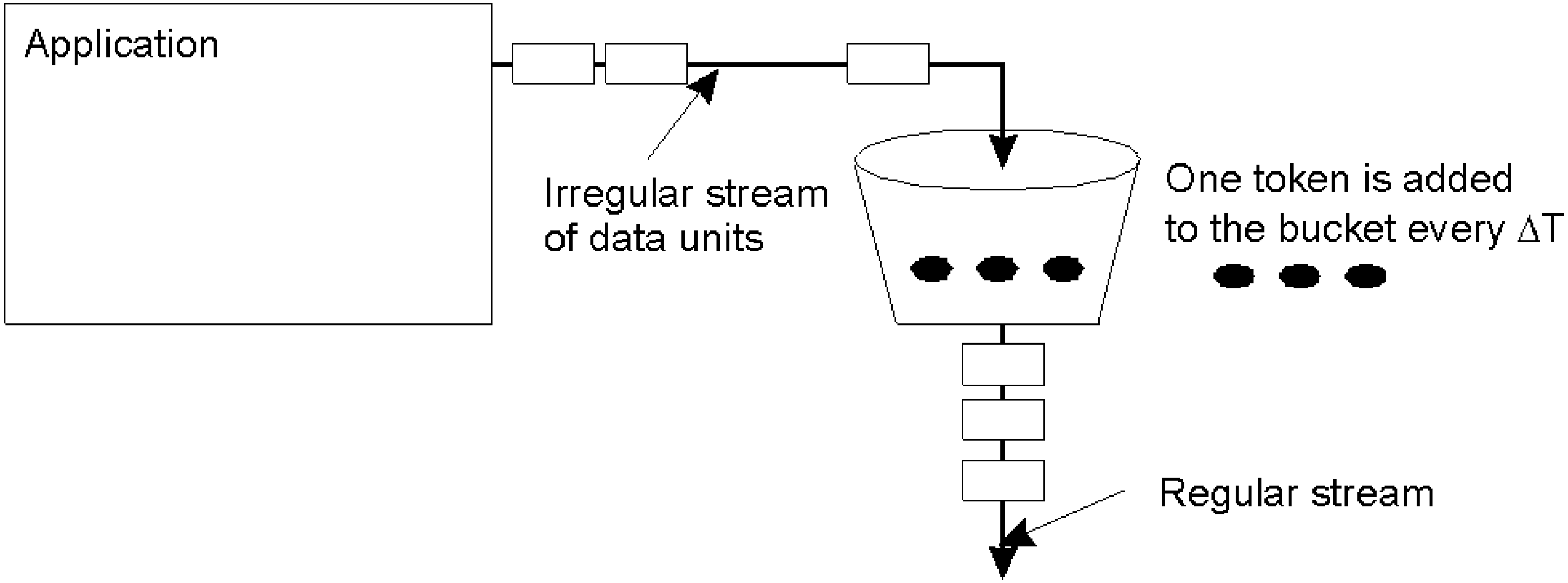
Set Up Stream



Quality of Service (QoS)

- Time-dependent and other requirements are specified as *quality of service (QoS)*
  - Requirements/desired guarantees from the underlying systems
  - Application specifies workload and requests a certain service quality
  - Contract between the application and the system

Specify QoS: Token bucket


Application
Irregular stream of data units
One token is added to the bucket every ΔT
Regular stream

- The principle of a token bucket algorithm
  - Parameters (rate r, burst b)
  - Rate is the average rate, burst is the maximum number of packets that can arrive simultaneously

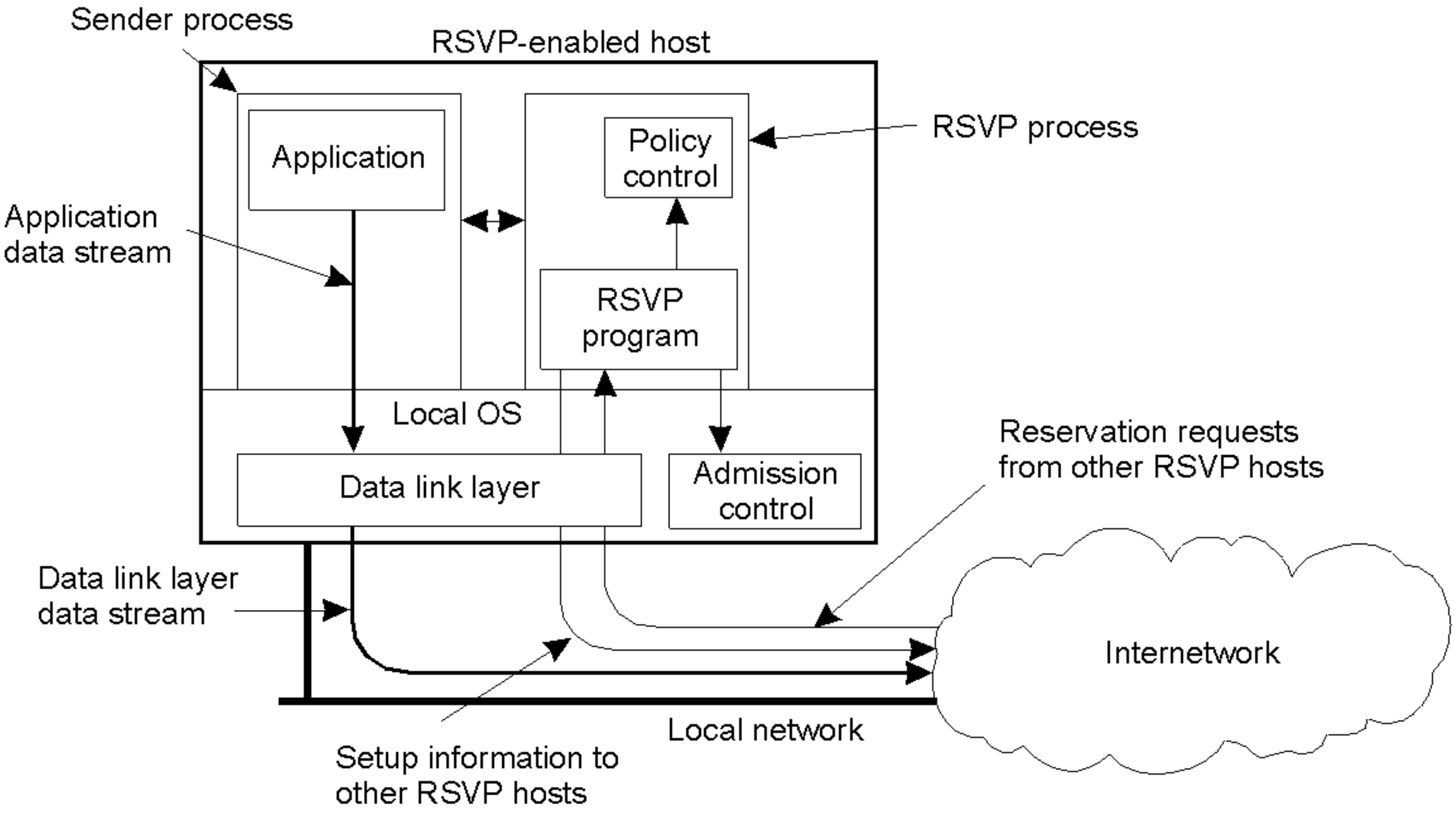
Specify QoS: Flow Specification

Characteristics of the Input	Service Required
•maximum data unit size (bytes) •Token bucket rate (bytes/sec) •Toke bucket size (bytes) •Maximum transmission rate (bytes/sec)	•Loss sensitivity (bytes) •Loss interval (μsec) •Burst loss sensitivity (data units) •Minimum delay noticed (μsec) •Maximum delay variation (μsec) •Quality of guarantee

QoS: Set Up Stream

- Lack of a model
  - Specify QoS parameter
  - Generically describe resources in any communication system
  - Translate QoS parameters to resource usage
- Expressing and establishing QoS is often difficult
- Incompatible approaches exist

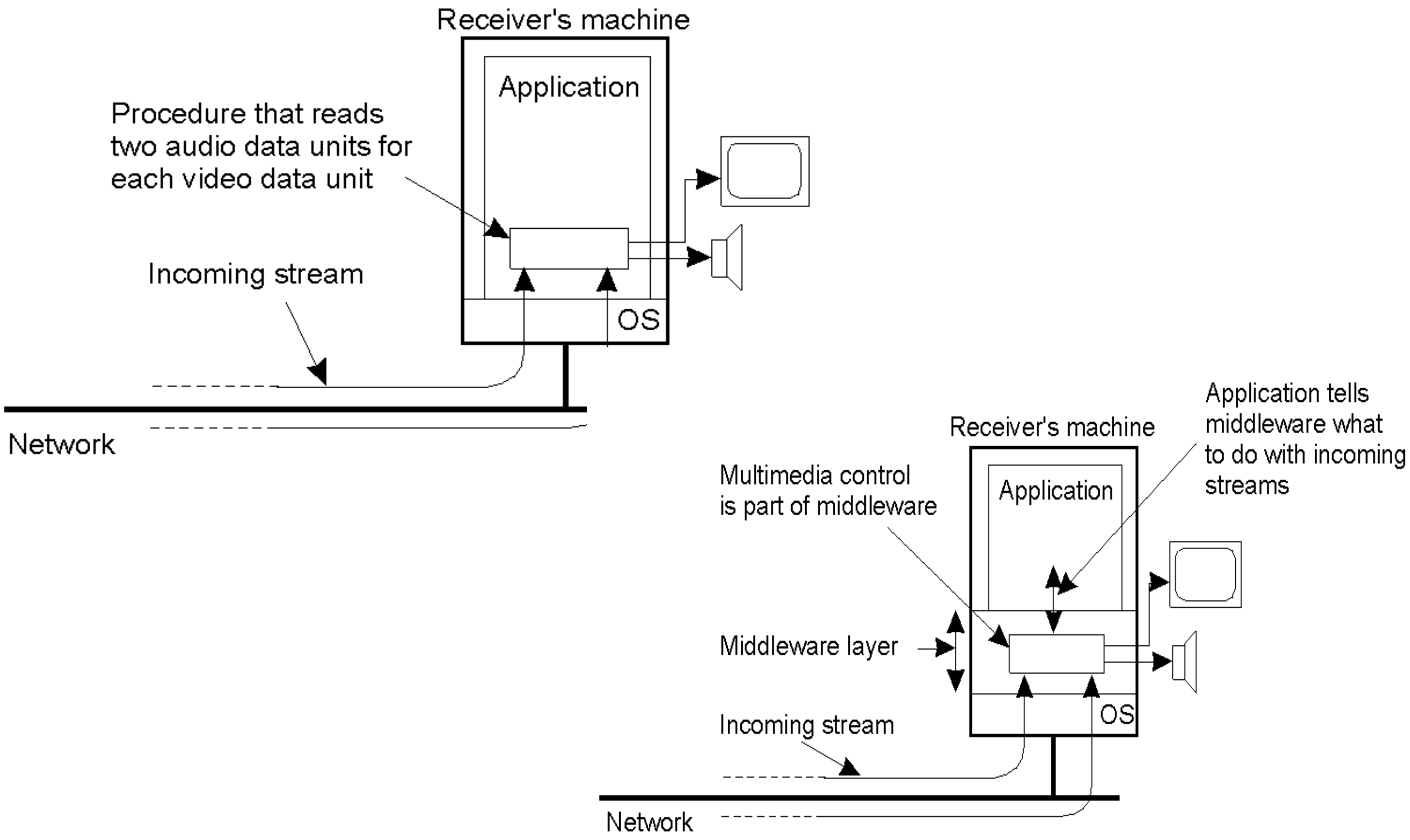
QoS: RSVP


Sender process
RSVP-enabled host
Application
Policy control
RSVP process
Application data stream
RSVP program
Local OS
Data link layer
Admission control
Reservation requests from other RSVP hosts
Data link layer data stream
Internetwork
Local network
Setup information to other RSVP hosts

Stream Synchronization

- Deal with maintaining temporal relations between streams
- Example:
  - A slide show on the Web that has been enhanced with audio
  - A movie play
- Two issues:
  - Synchronization mechanism
  - The distribution of synchronization mechanisms

Synchronization Mechanisms


Receiver's machine
Application
Procedure that reads two audio data units for each video data unit
Incoming stream
OS
Network
Receiver's machine
Application tells middleware what to do with incoming streams
Multimedia control is part of middleware
Application
Middleware layer
Incoming stream
OS
Network

Distribution of Synchronization Mechanisms

- Whether synchronization should take place at the sending or receiving side?

- What is the local synchronization specification?

Summary

- Message-oriented communication
  - Persistence and synchronicity
  - Message-oriented transient communication
    - Berkeley socket
    - MPI
  - Message-oriented persistent communication
- Stream-oriented communication
  - Data stream
  - Quality of services
  - Stream synchronization
- Readings:
  - Chpt 4 of AST

Questions

?