Communication in the Cloud

Data at Scale

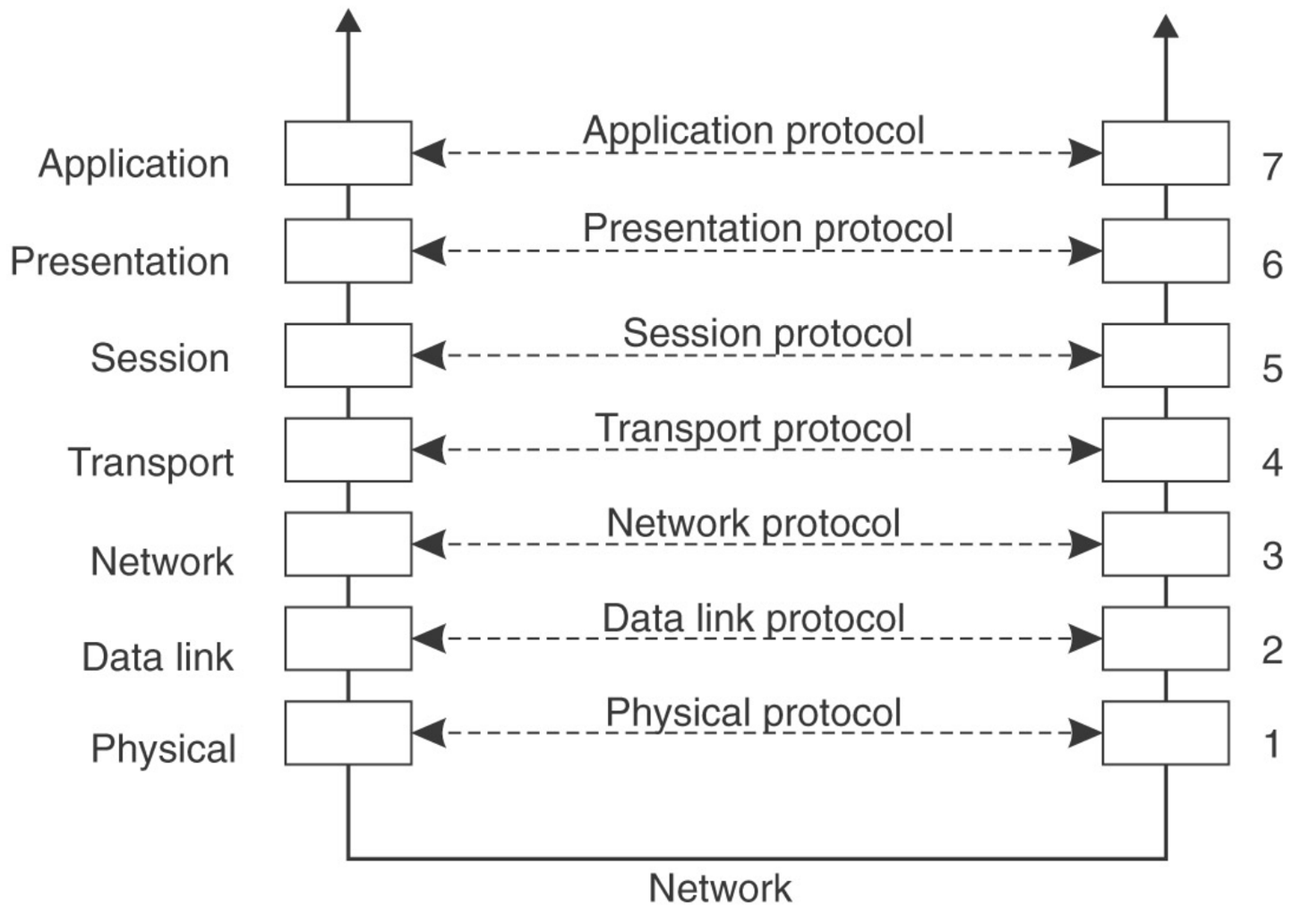
Protocols

- Networking is divided into different layers
- Each layer has its own protocol
- Allows to change one layer without changing other layers
- Use well defined interfaces between layers

- Example:
  - New communication techniques change Layer 1
  - TCP will function independently

OSI Reference

- Never popular

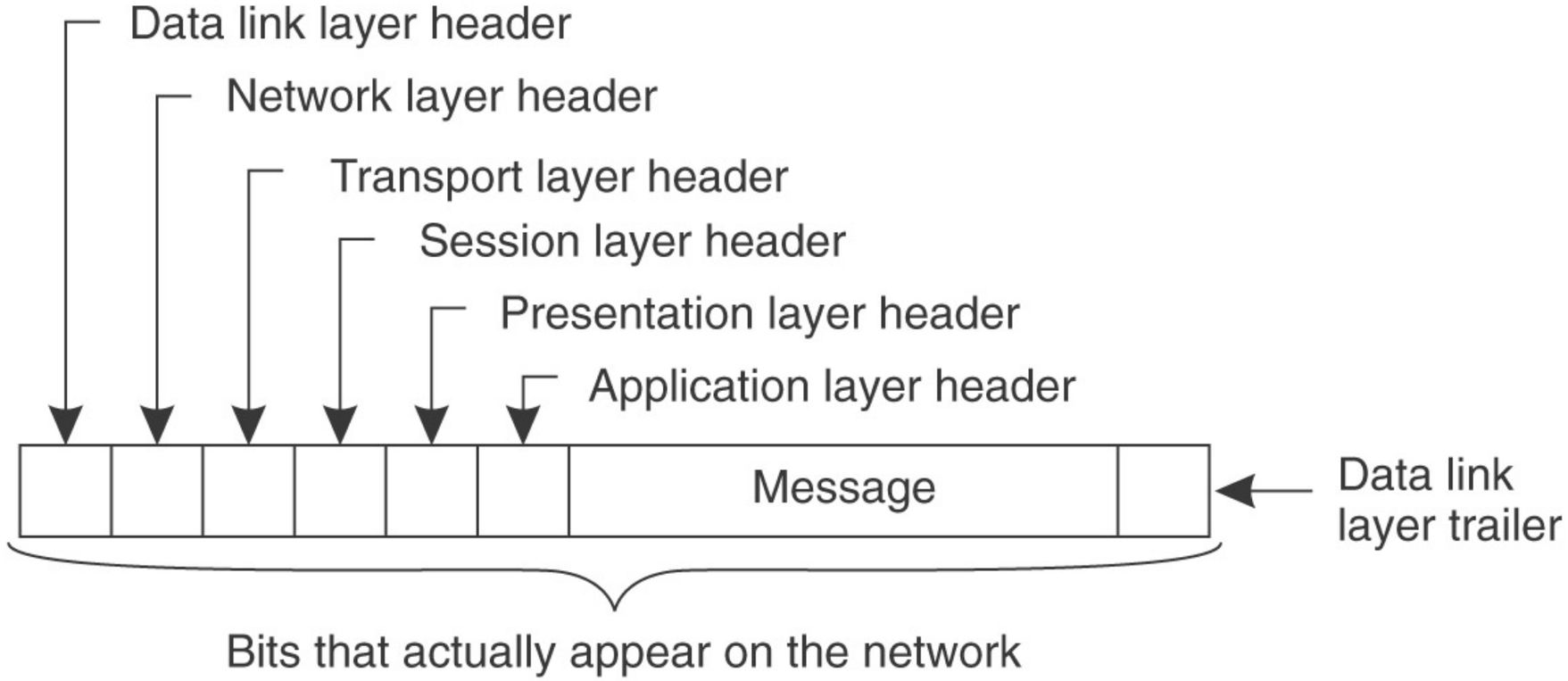


Protocols

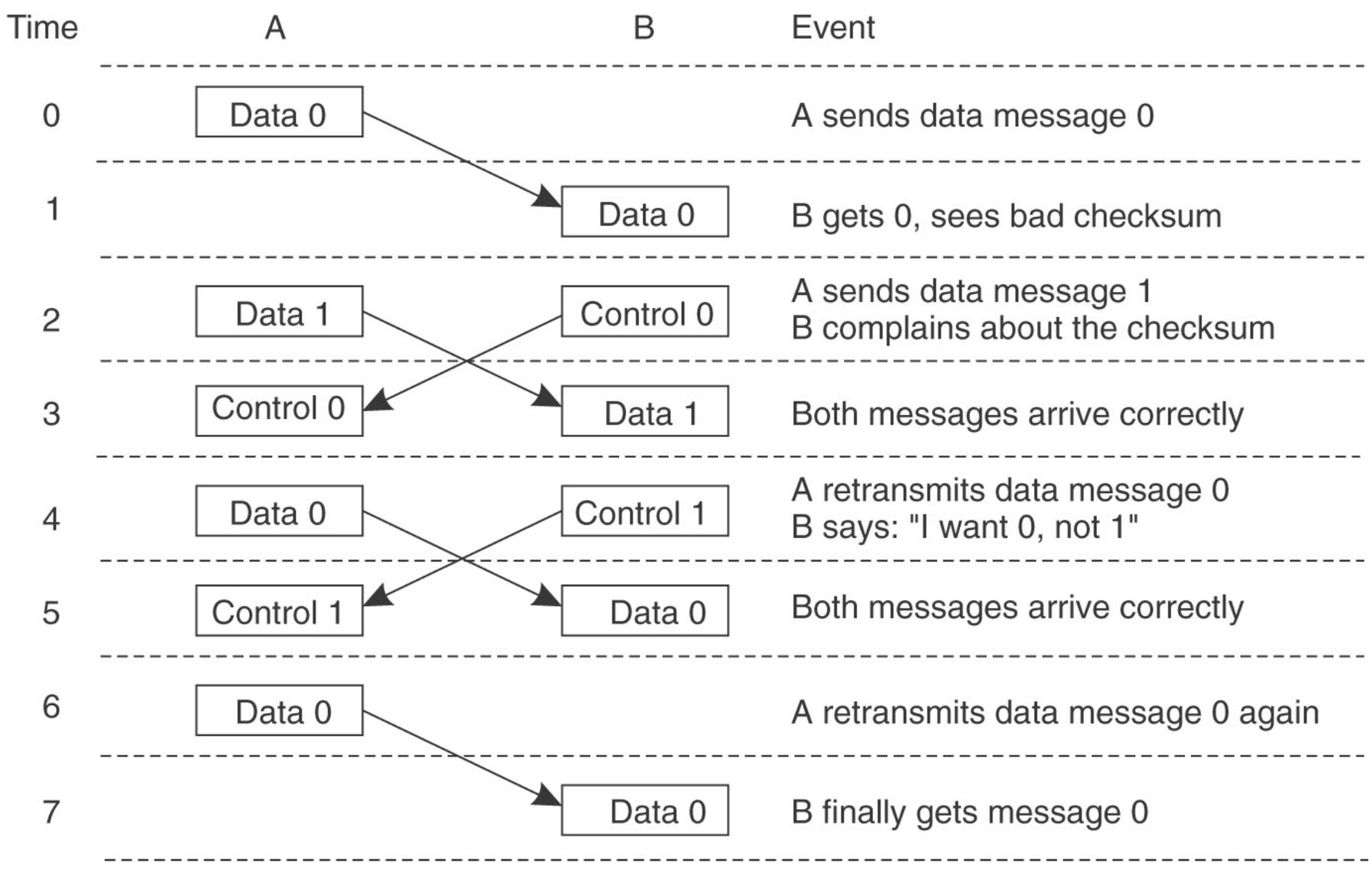
- OSI model was never popular
  - Physical layer:
    - Transmission of bits
  - Data link layer:
    - Organizes bits into frames
    - Add checksum to frames in order to detect / correct errors

Protocols

- Basic technique is framing
  - Message of upper protocol is framed by metadata from lower protocol



Example: Data link layer



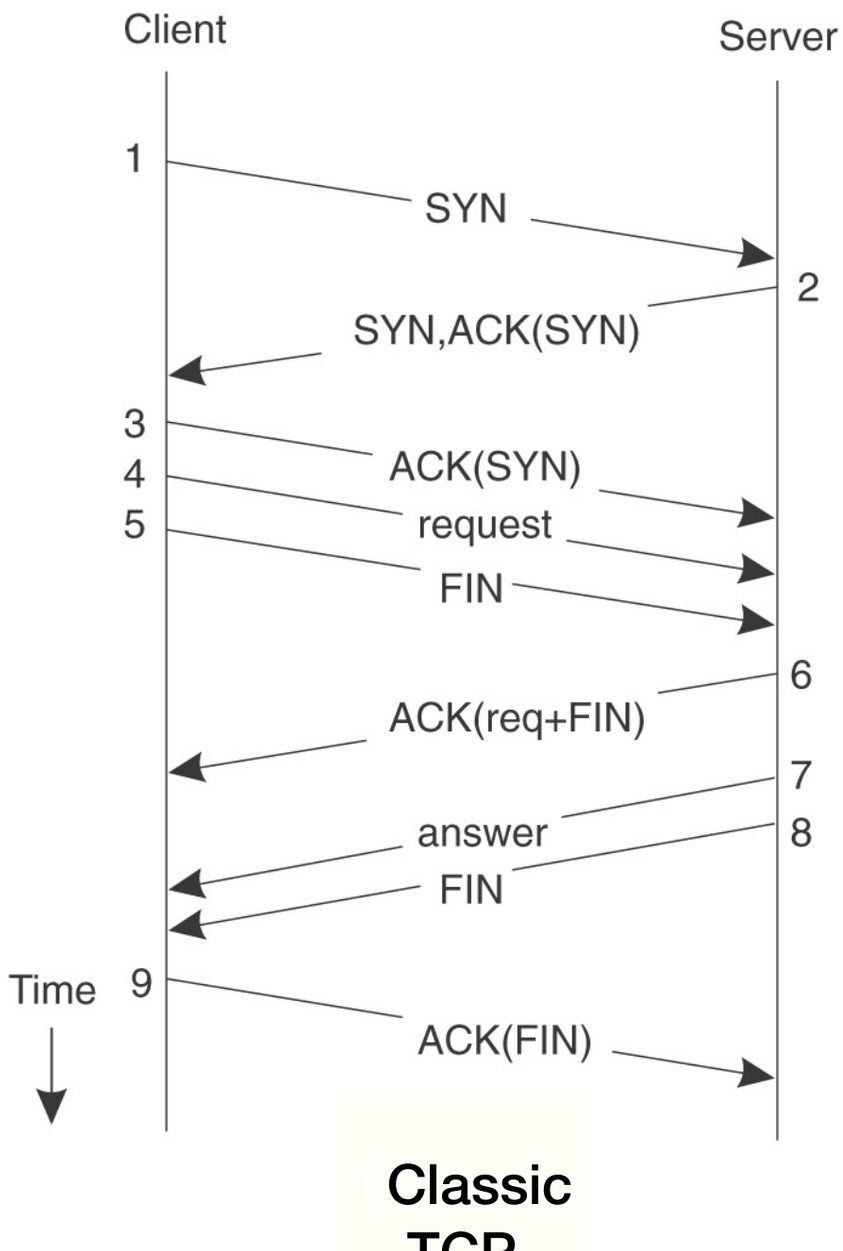
Transport Protocols

- Transport layer
  - Offers a reliable connection
    - TCP (Transmission Control Protocol): guarantees delivery in order
    - UDP (User Datagram Protocol): sends a packet and hopes for the best
    - New proposals such as Real-time transport protocol for real time data transfer

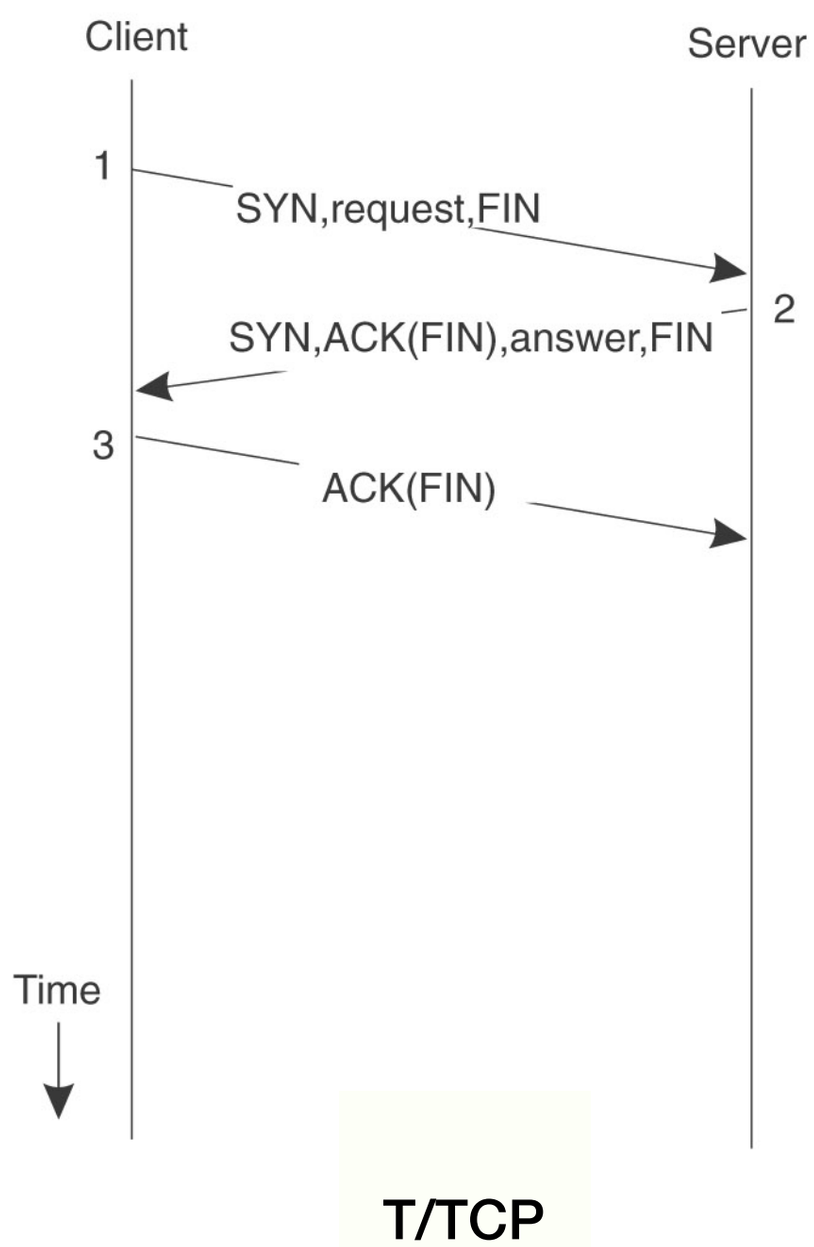
Transport Protocols

- Client-Server and TCP
  - TCP not tailored for synchronous request-reply behavior of client-server interaction
  - Can use UDP and add reliability features
  - Transactional TCP: A new proposal that combines connection setup with sending the request for client server architectures

T/TCP



Classic TCP



T/TCP

Higher Level Protocols

- Session Protocol
  - Enhancement of transport layer:
    - Dialog control
    - Synchronization
    - Not present in TCP/IP suite
- Presentation layer: meaning of bits
- Application layer:
  - OSI intended a collection of network applications
    - email, file transfer, terminal emulation

Middleware Protocols

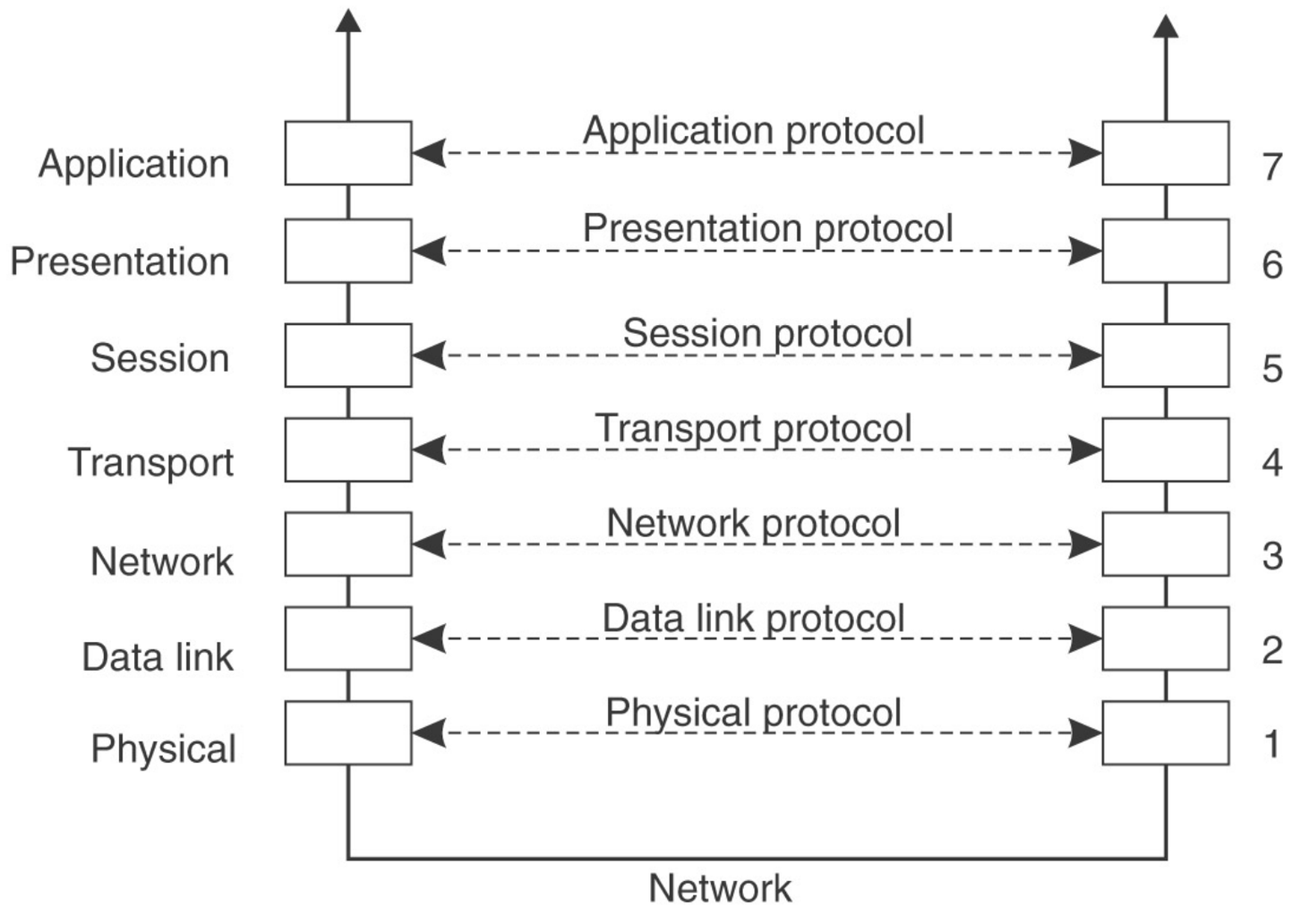
- Middleware lives at the application layer
  - Can contain general purpose protocols that warrant their own layer
  - Examples:
    - Authentication protocols
    - Authorization protocols
    - Distributed commit protocols

Middleware Protocols

- Other examples:
  - Remote procedure calls
  - Remote object invocation
  - Message queuing systems
  - Communication of continuous media through streams

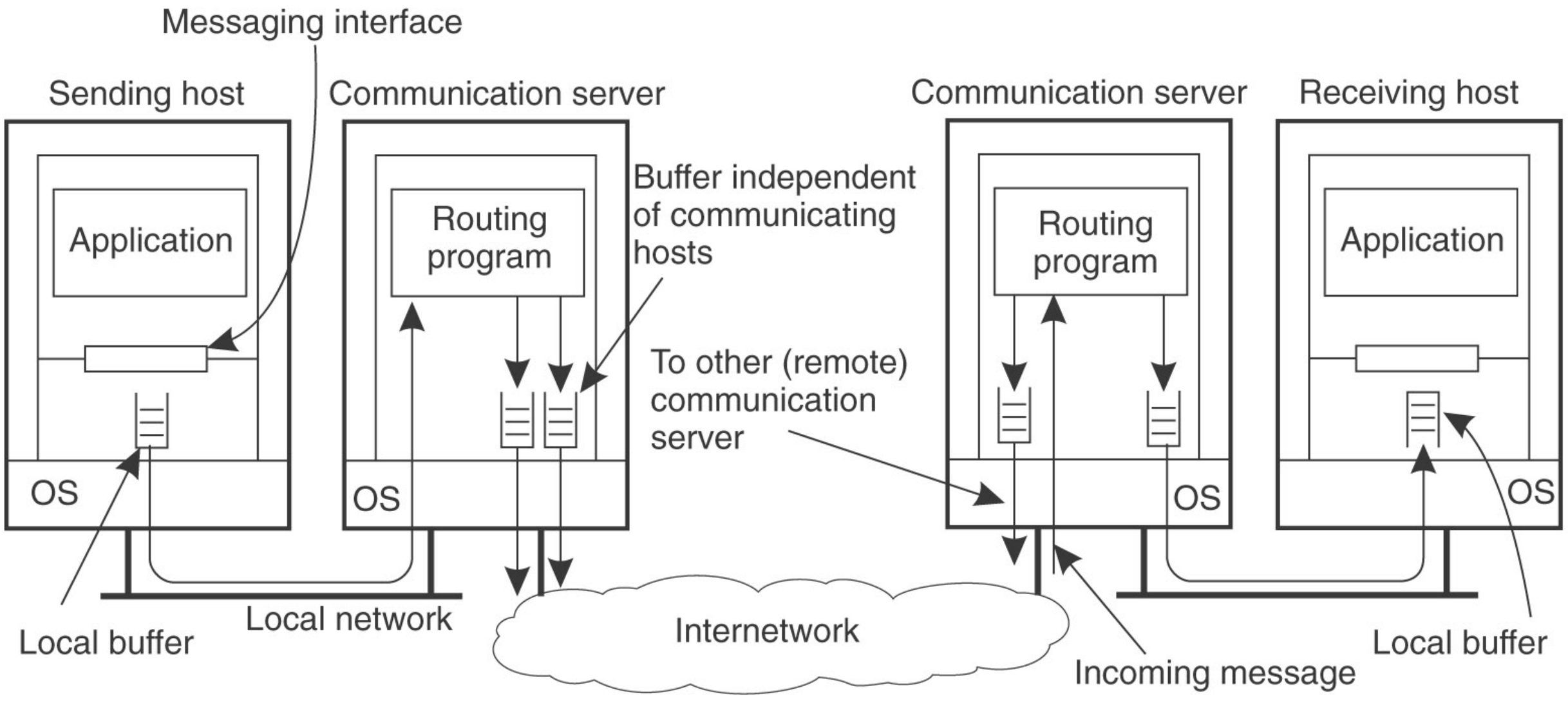
OSI Reference

- Never popular



Persistence and Synchronization

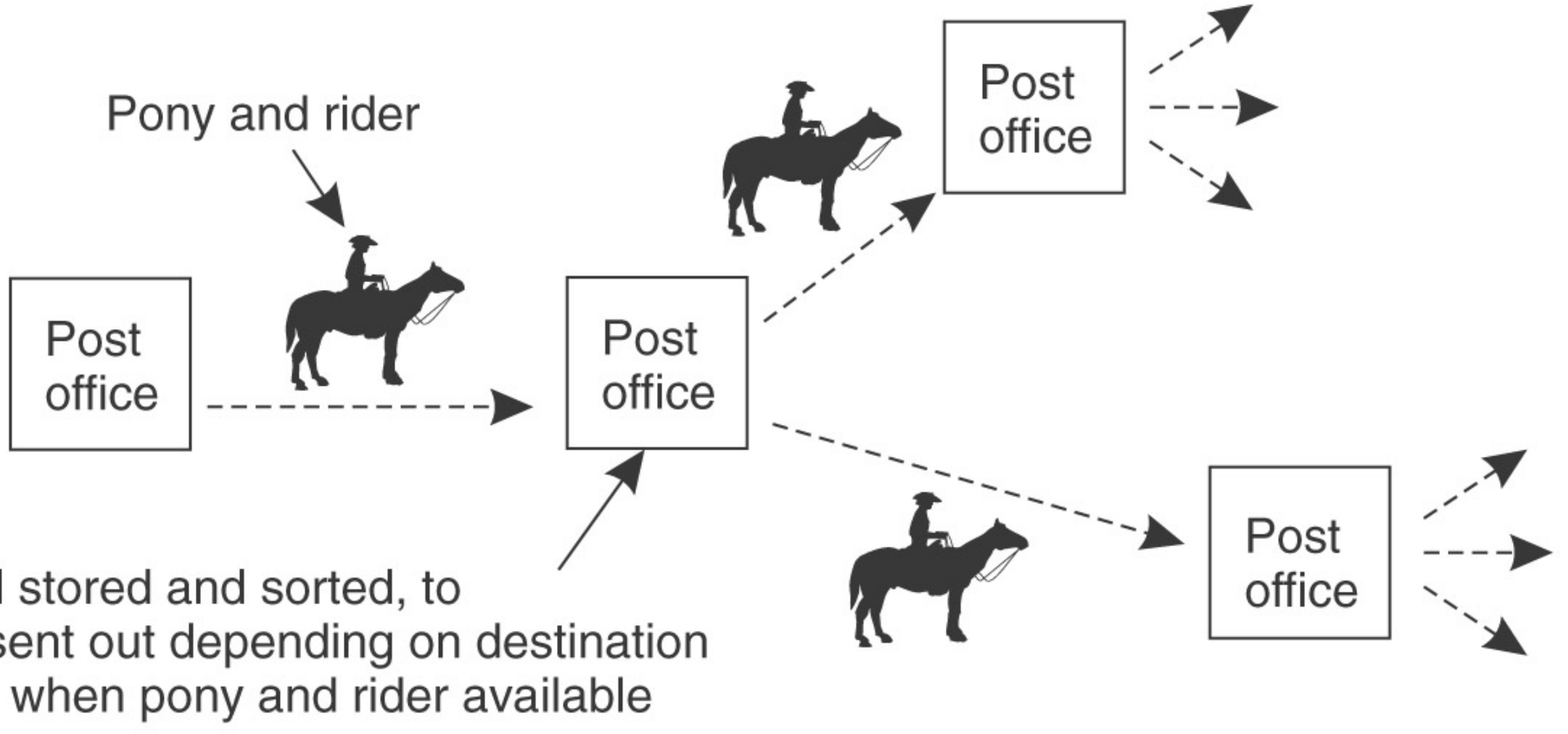
Communication System



Persistence and Synchronization

- Persistence
  - Message is stored until it can be delivered
    - Sender or receiver might be down while message travels between communication servers

Persistence and Synchronization



Persistence and Synchronization

- Transient communication
  - Message is stored by the communication system only as long as sender and receiver execute

Persistence and Synchronization

- Asynchronous communication
  - Sender continues immediately after submitting message for transmission

Persistence and Synchronization

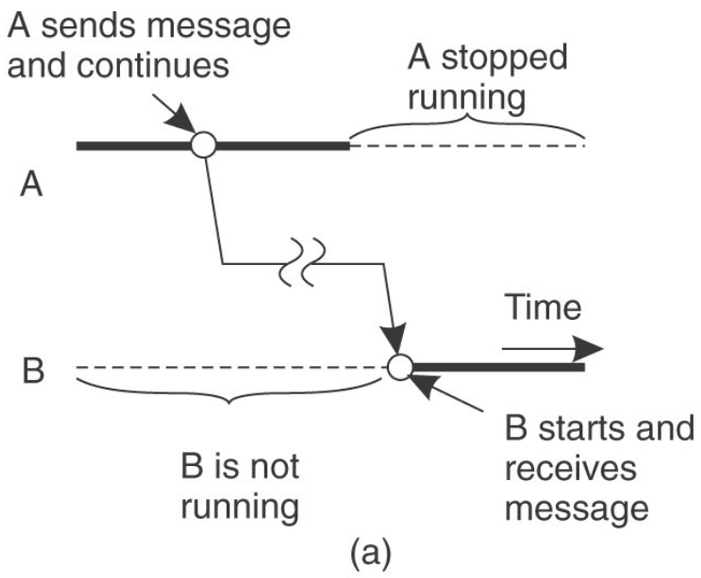
- Synchronous communication
  - Sender blocks until its message is
    - stored in the local buffer at the receiver
    - or actually delivered to the receiver

Persistence and Synchronization

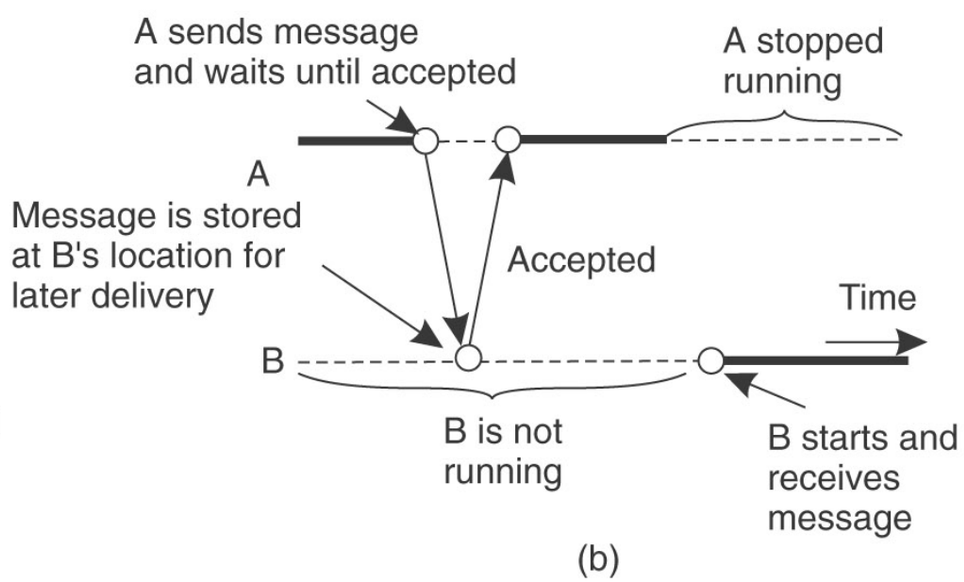
- Persistent asynchronous communication
  - Electronic mail
- Persistent synchronous communication
  - Messages are stored only at the receiving host
  - Sender is blocked until message reaches receiver's buffer
  - (But receiver does not have to run)

Persistence and Synchronization

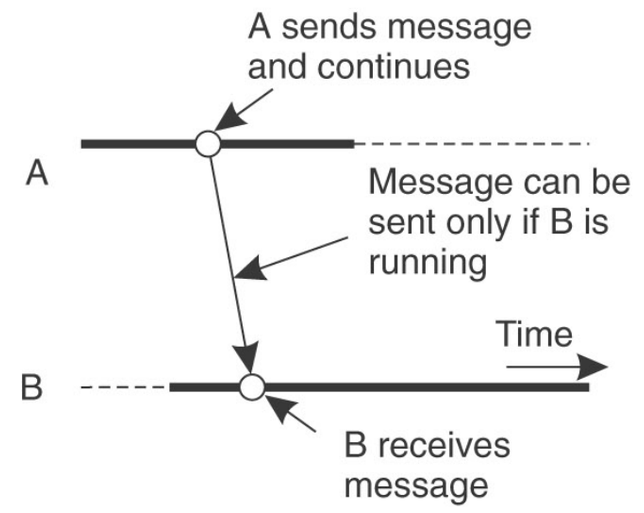
- Transient asynchronous communication
  - UDP
  - One-way RPC
- Transient synchronous communication
  - Sender is blocked until the message is stored in a local buffer at the receiving host
  - Sender receives ack
  - Sender continues
    - Asynchronous RPC



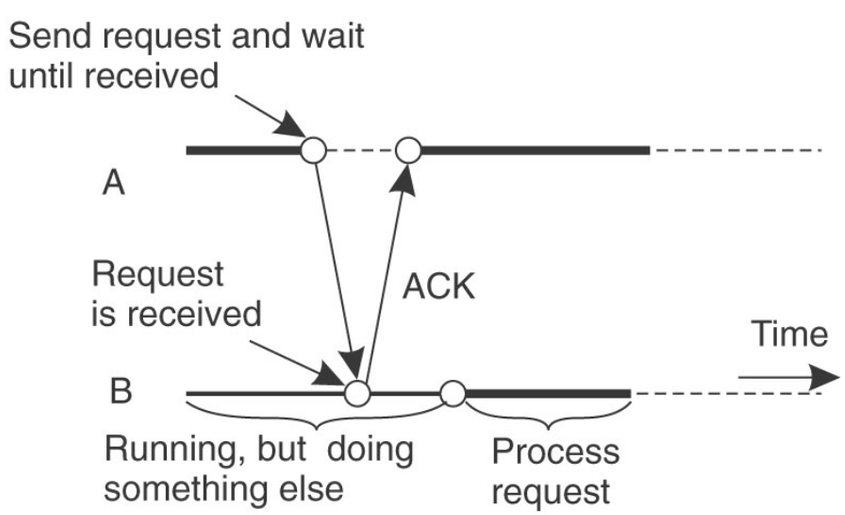
(a)



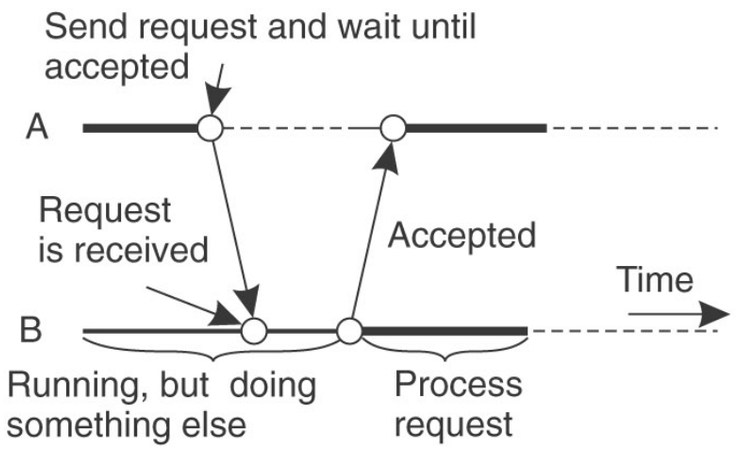
(b)



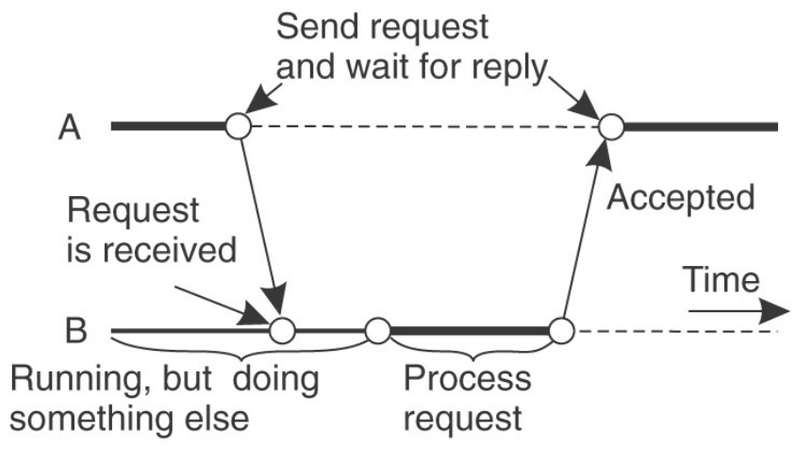
(c)



(d)



(e)



(f)

(a) persistent asynchronous
(b) persistent synchronous
(c) transient asynchronous
(d) receipt based transient synchronous
(e) delivery based transient synchronous
(f) response based transient synchronous

Persistence and Synchronization

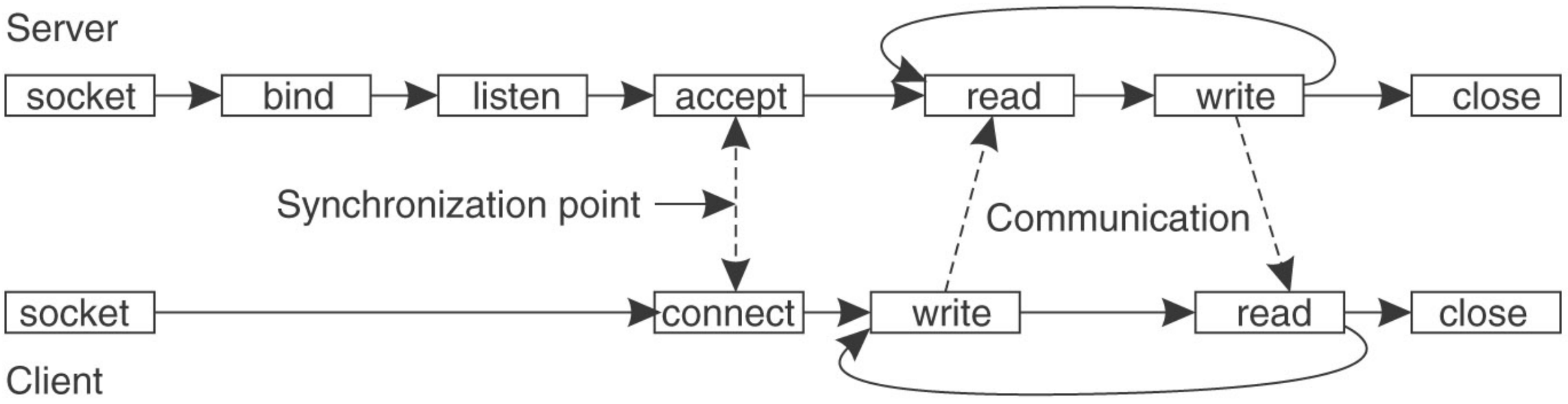
- Persistent communication is necessary in large-scale and widely dispersed interconnected networks
- Persistency is useful when dealing with failure

Message Oriented Transient Communication

- Berkeley UNIX sockets interface

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 Oriented Transient Communication



Message Oriented Transient Communication

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 is none
MPI_irecv	Check if there is an incoming message, but do not block

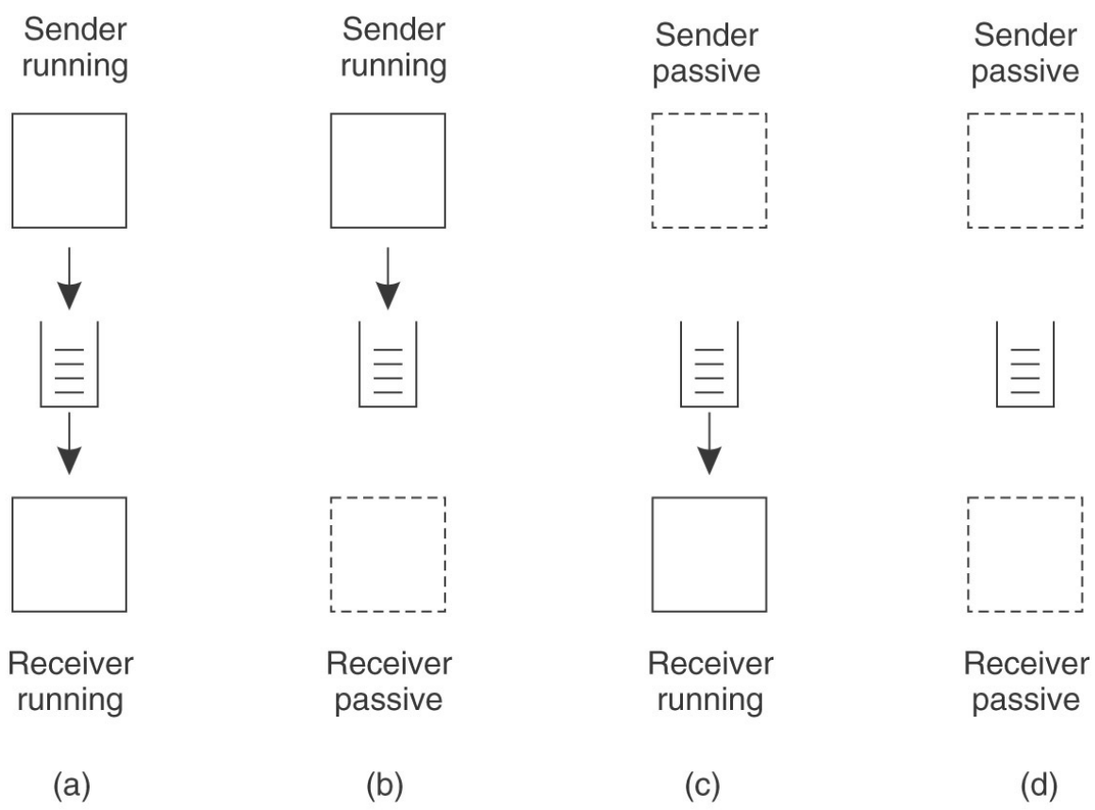
Message Oriented Persistent Communication

- Important class of middleware services:
  - **Message Queuing Systems**
  - **Message Oriented Middleware (MOM)**

- support persistent asynchronous communication
  - offer intermediate-term storage capacity for messages

Message Oriented Persistent Communication

- Message Queueing Model
  - Applications communicate by inserting messages in specific queues
  - For sender:
    - Guarantee that message will be eventually inserted into the recipients queue



Message Oriented Persistent Communication

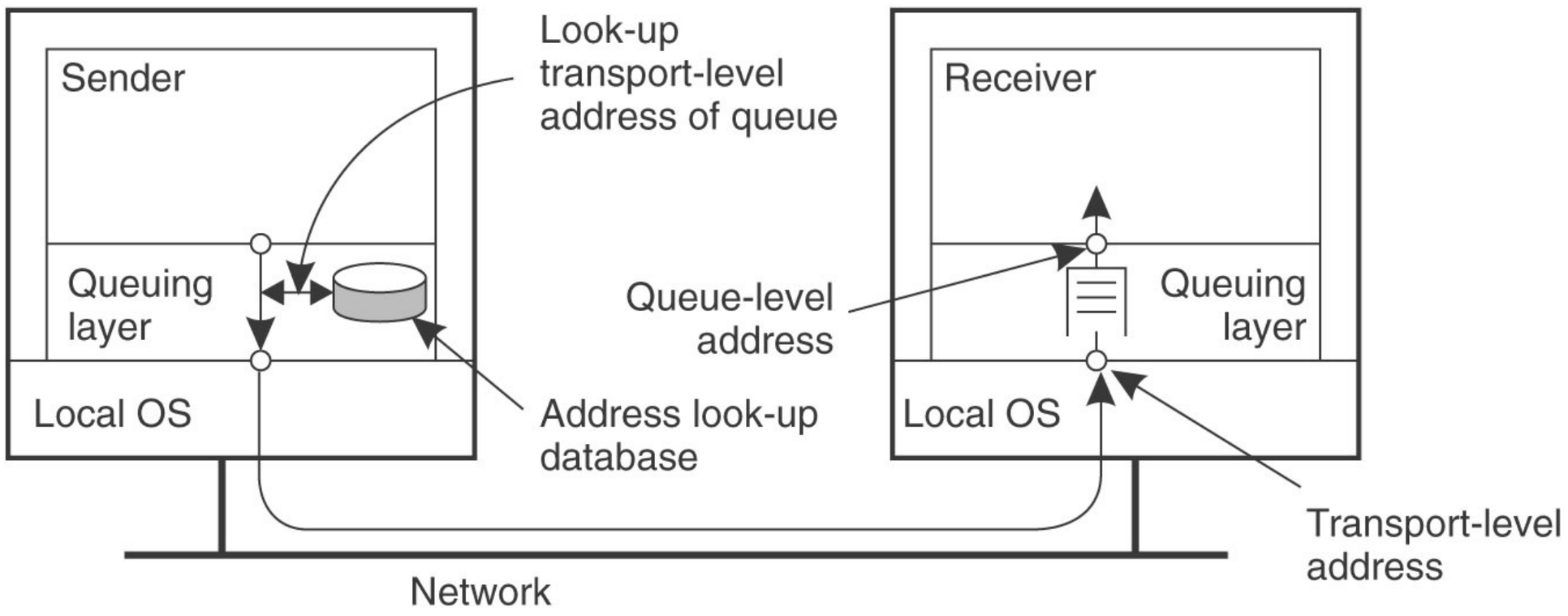
Primitive	Meaning
Put	Append a message to a specified queue
Get	Block until the specified queue is nonempty, and remove the first message
Poll	Check a specified queue for messages, and remove the first. Never block
Notify	Install a handler to be called when a message is put into the specified queue

Message Oriented Persistent Communication

- Message queuing system
  - Many allow installing a handler as a *callback* function
    - Invoked when message is put into the queue
  - Allow daemon etc to monitor queue for incoming messages

Message Oriented Persistent Communication

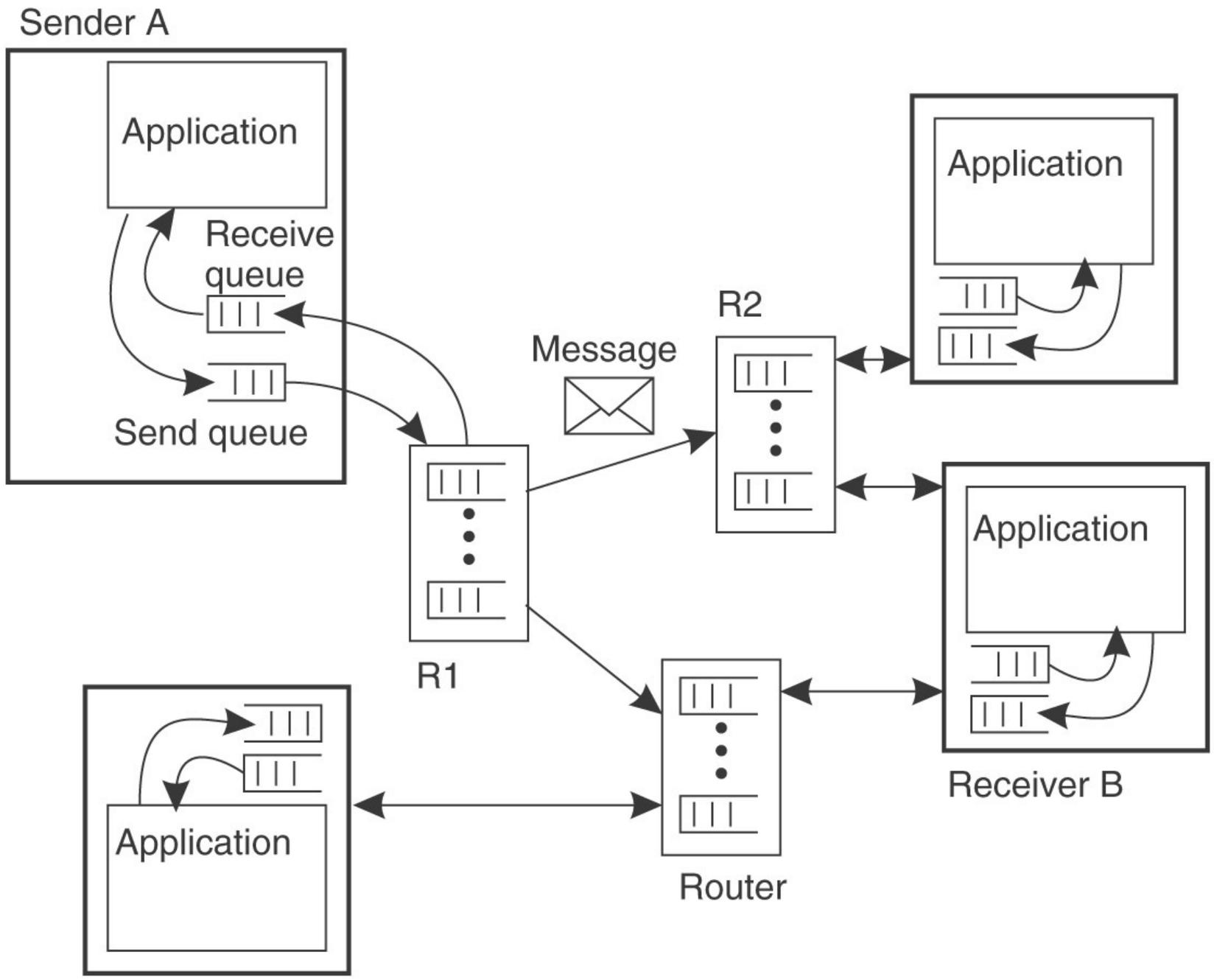
- Message-queueing system architecture
  - Local: on the same machine or the same LAN
  - Queues distributed over network



Message Oriented Persistent Communication

- Message Queueing System Architecture
  - Queues managed by *queue manager*
  - Some act as routers / relays
    - Generate an overlay network
    - Existence simplifies network administration
      - Only routers need to be updated when a queue is moved, inserted, or deleted
- Can also be used to log messages
  - for security, fault tolerance

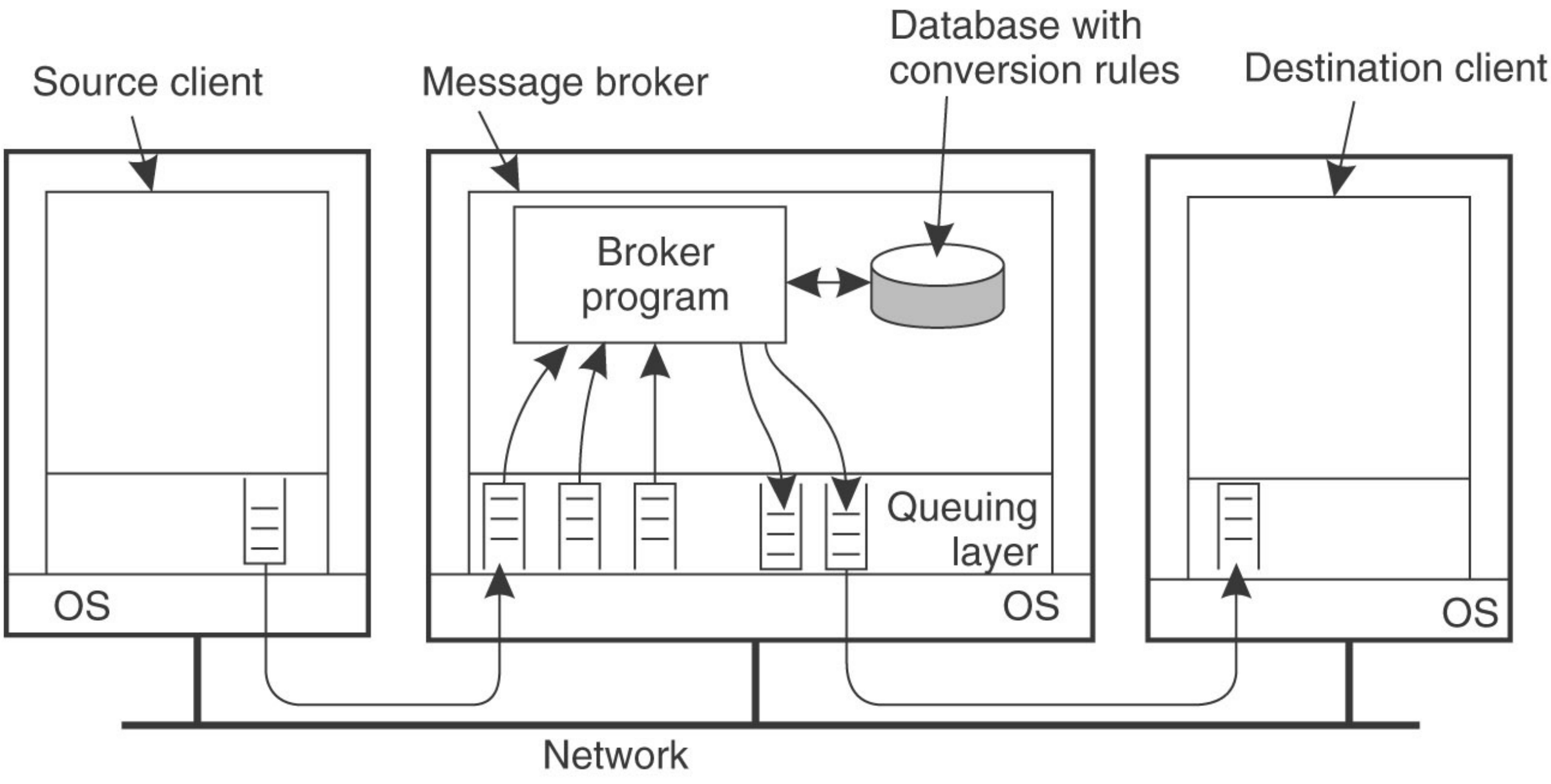
Message Oriented Persistent Communication



Message Oriented Persistent Communication

- Message Brokers
  - Integrate existing and new applications
    - Applications need to understand messages
      - Sender can format outgoing messages according to the receiver
      - or: Use a common message format
      - or: **Message brokers** convert incoming messages to a format understood by receiver

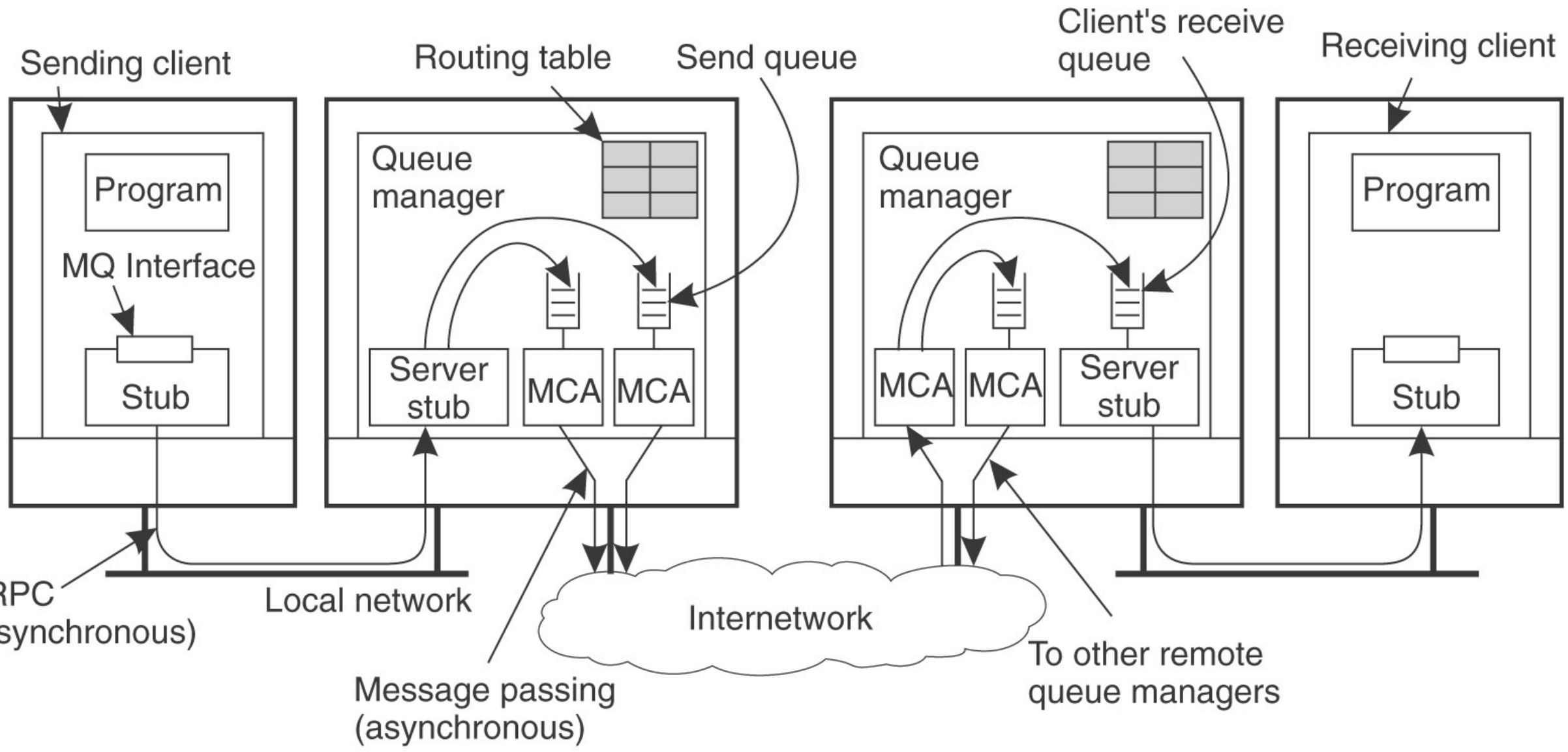
Message Oriented Persistent Communication



Message Oriented Persistent Communication

- Message queuing systems are almost like email servers
  - But:
    - email servers directly support endusers
    - have additional requirements such as automatic message filtering, …

Message Oriented Persistent Communication



Message Oriented Persistent Communication

- IBM MQSeries
  - Queues are managed by queue managers
  - Message channels connect two queue managers
    - unidirectional
    - reliable
    - managed by Message Channel Agent (MCA)
- Queue managers can be linked into the same process as an application
  - Queues are hidden to application
  - Standard interface for messaging

Message Oriented Persistent Communication

- IBM MQSeries
  - Message channels
    - one send queue
    - one receive queue
    - both MCA need to be up for sending a message
  - Initialization:
    - Manually
    - or: application starts MCA
    - or: Send queue triggers when message is put into queue, which starts a handler, to start the sending MCA
    - or: start MCA over the network
  - Termination:
    - Automatic after a specific time without sending messages

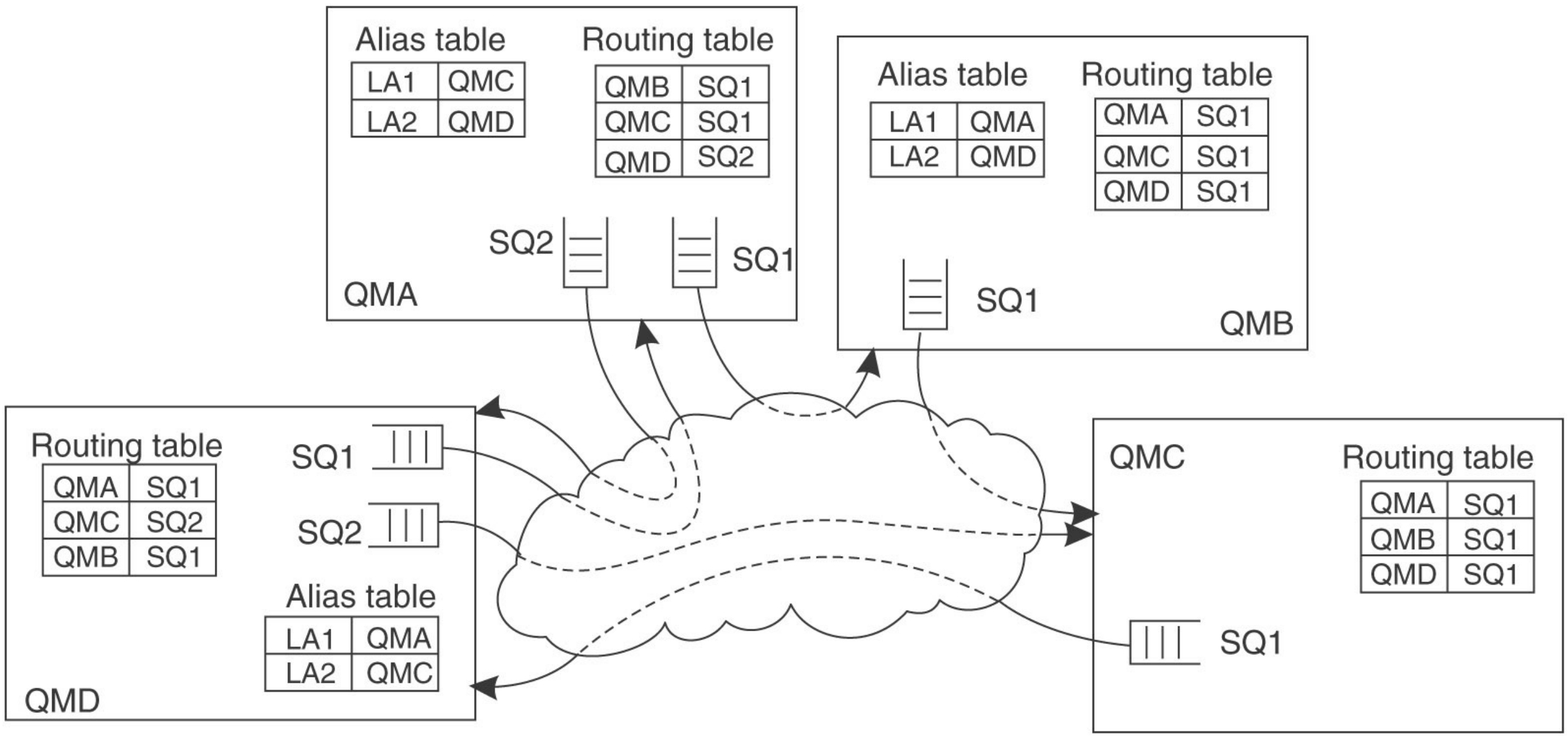
Message Oriented Persistent Communication

Attribute	Description
Transport type	Determines the transport protocol to be used
FIFO delivery	Indicates that messages are to be delivered in the order they are sent
Message length	Maximum length of a single message
Setup retry count	Specifies maximum number of retries to start up the remote MCA
Delivery retries	Maximum times MCA will try to put received message into queue

Attributes for message channel agents

Message Oriented Persistent Communication

Routing tables and aliases for MQSeries



Message Oriented Persistent Communication

Primitive	Description
MQopen	Open a (possibly remote) queue
MQclose	Close a queue
MQput	Put a message into an opened queue
MQget	Get a message from a (local) queue