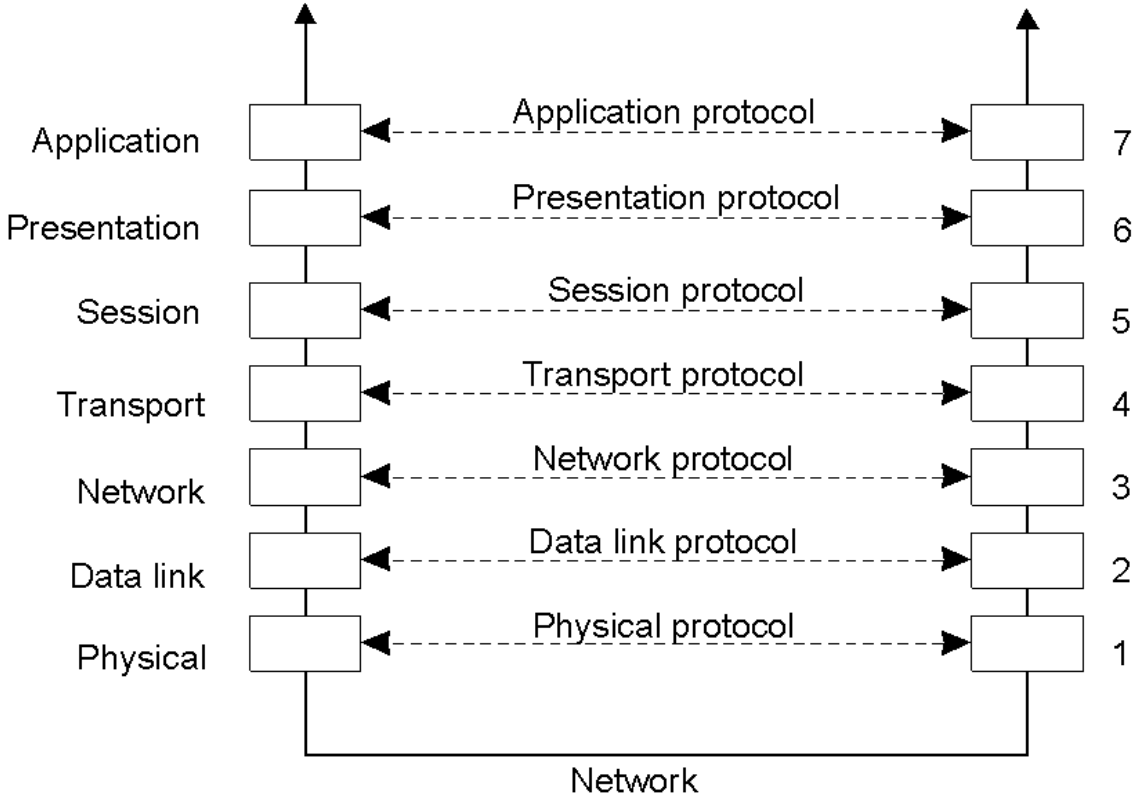
Communication

Chapter 2

Software de Rede

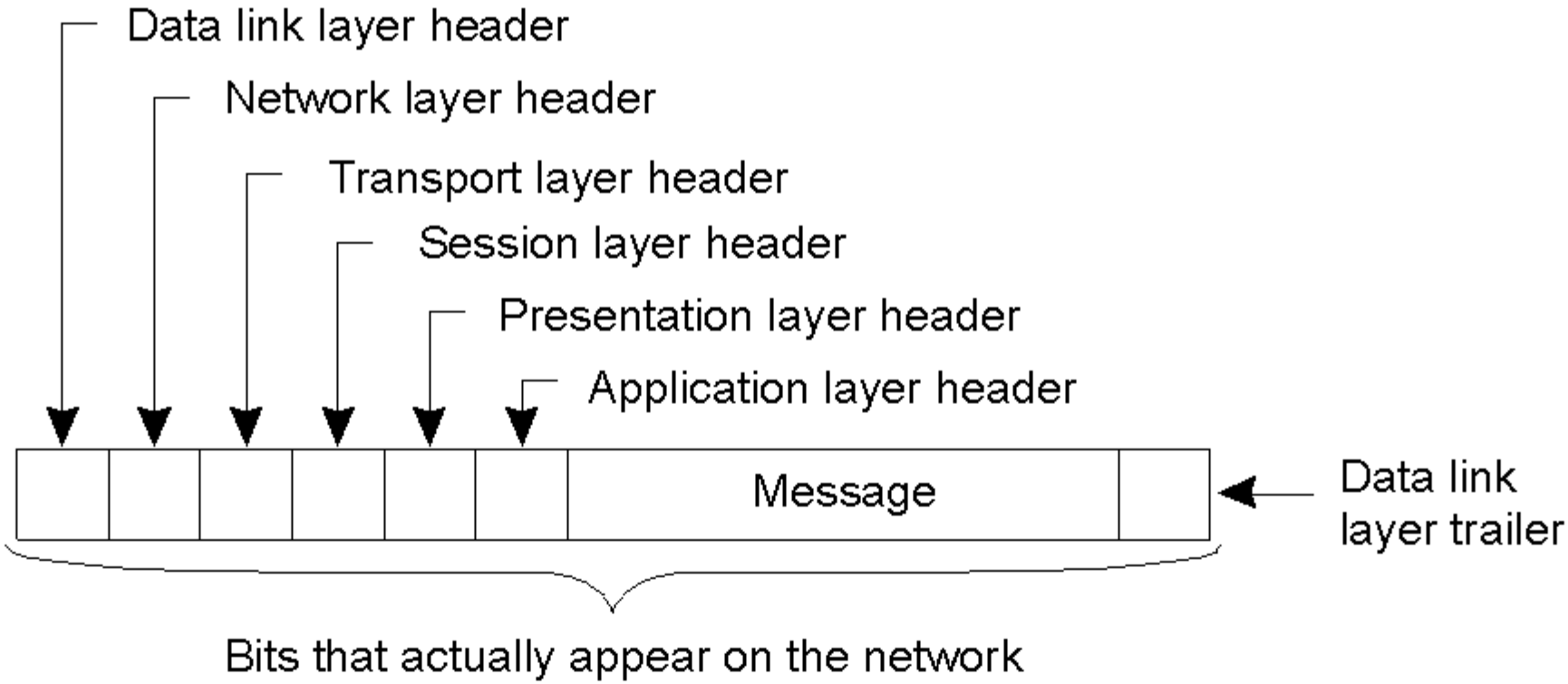
- Troca de Mensagens: Pilha de Protocolos -

Layered Protocols (1)



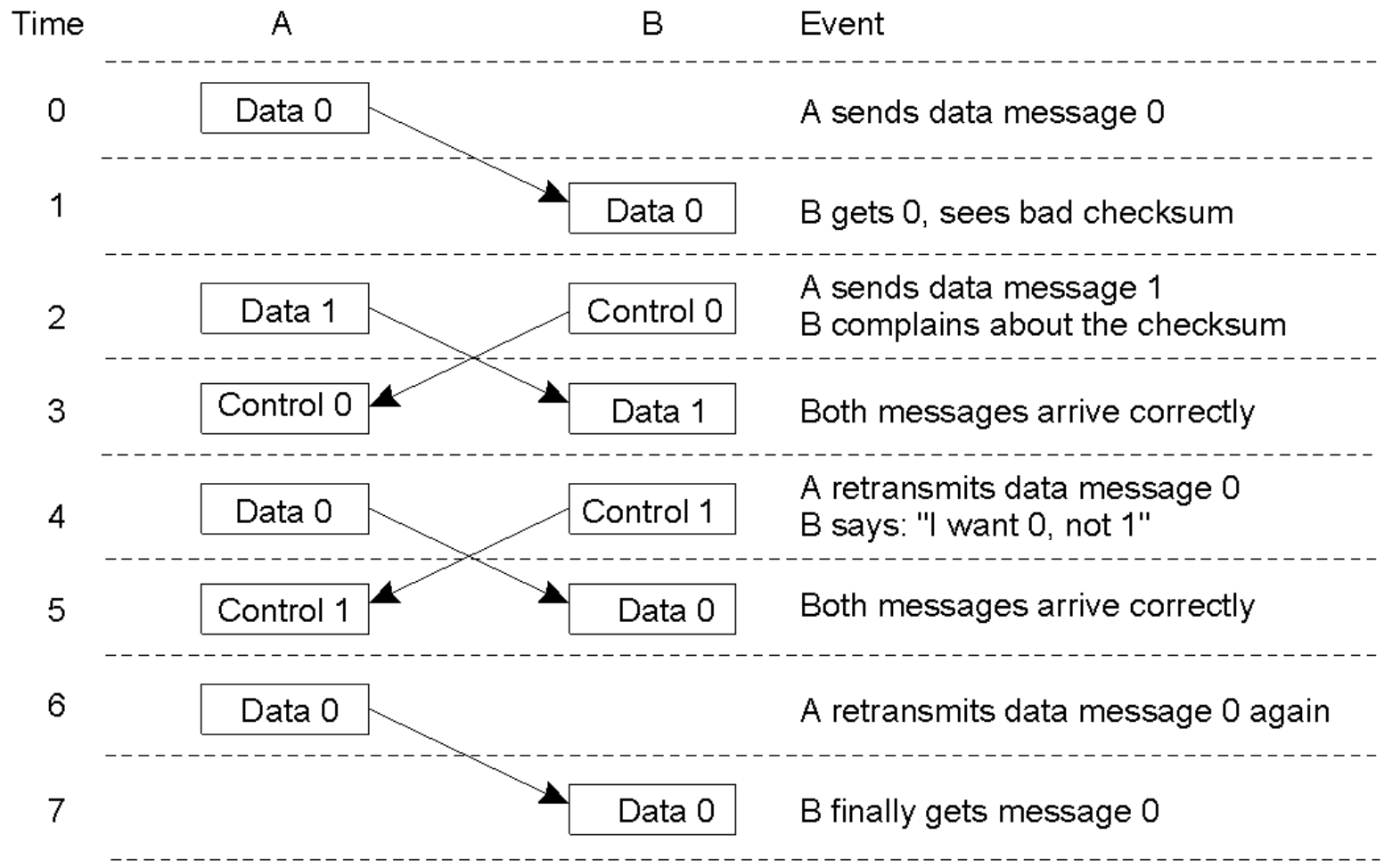
Layers, interfaces, and protocols in the OSI model.

Layered Protocols (2)



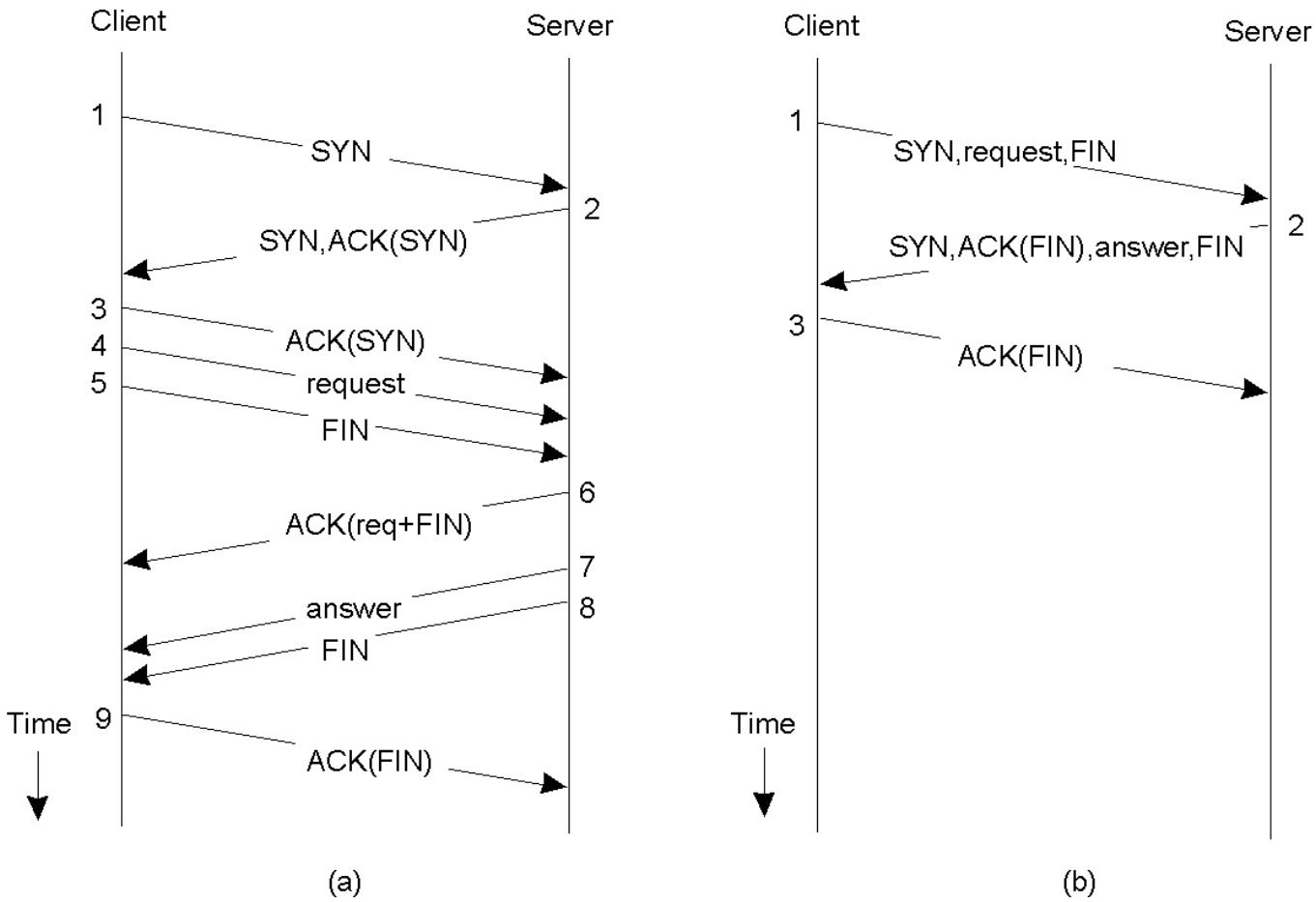
A typical message as it appears on the network.

Data Link Layer



Discussion between a receiver and a sender in the data link layer.

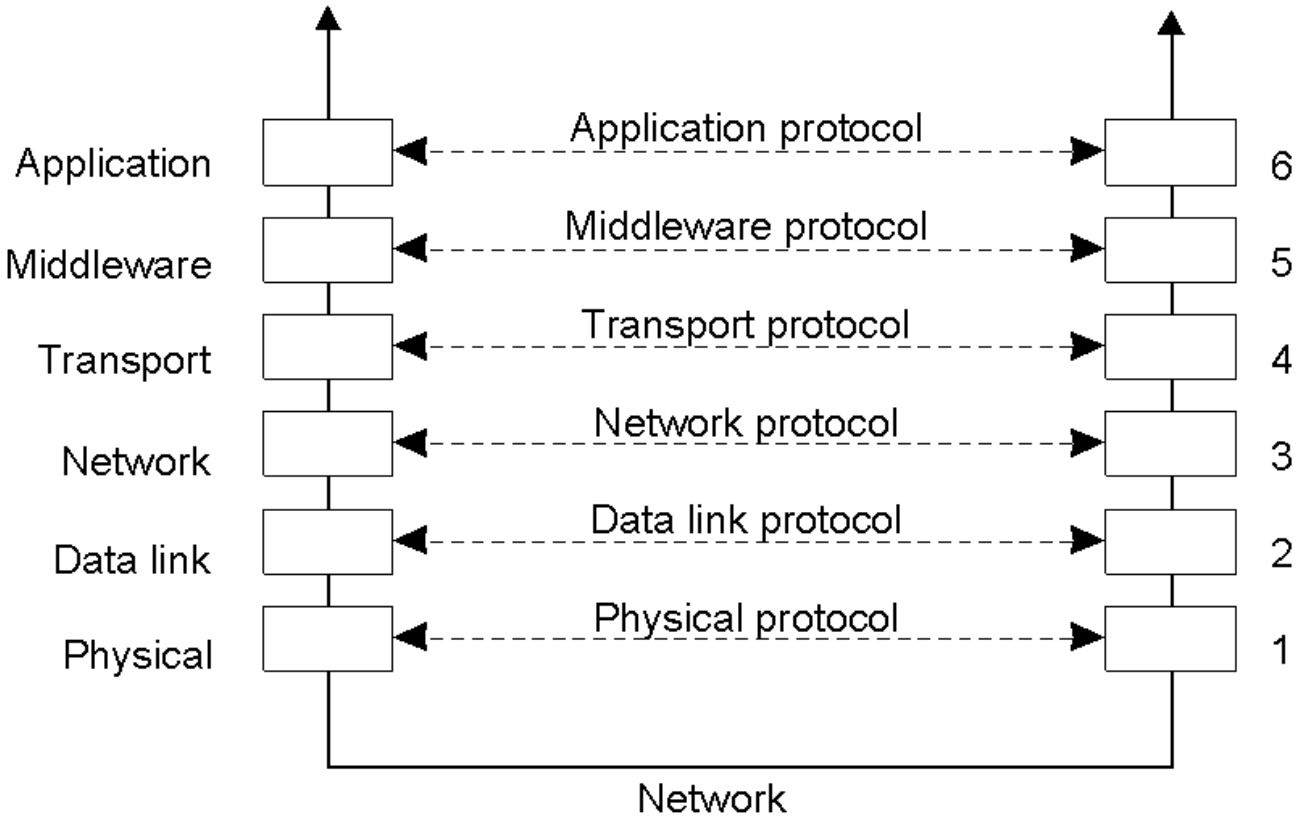
Client-Server TCP



a) Normal operation of TCP.

b) Transactional TCP.

Middleware Protocols



An adapted reference model for networked communication.

Mensagens Síncronas

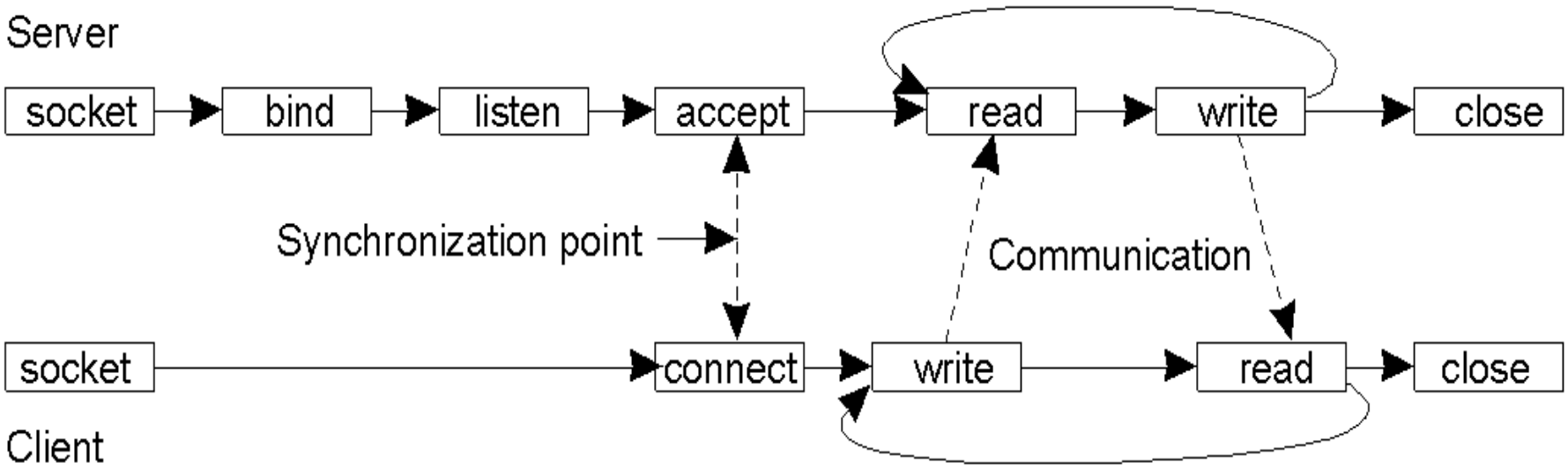
- API para Soquetes -

Berkeley Sockets (1)

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

Socket primitives for TCP/IP.

Berkeley Sockets (2)

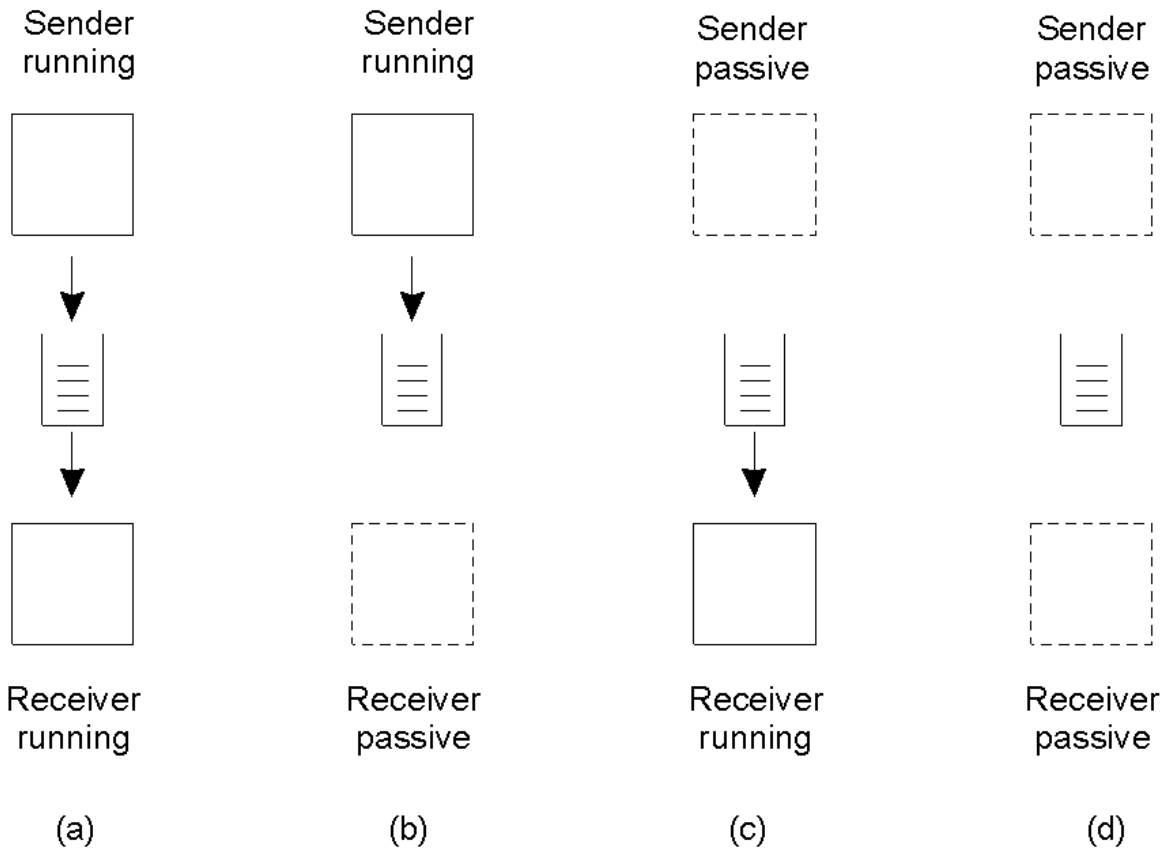


Connection-oriented communication pattern using sockets.

Mensagens Assíncronas

- Servidores de Fila -

Message-Queuing Model (1)



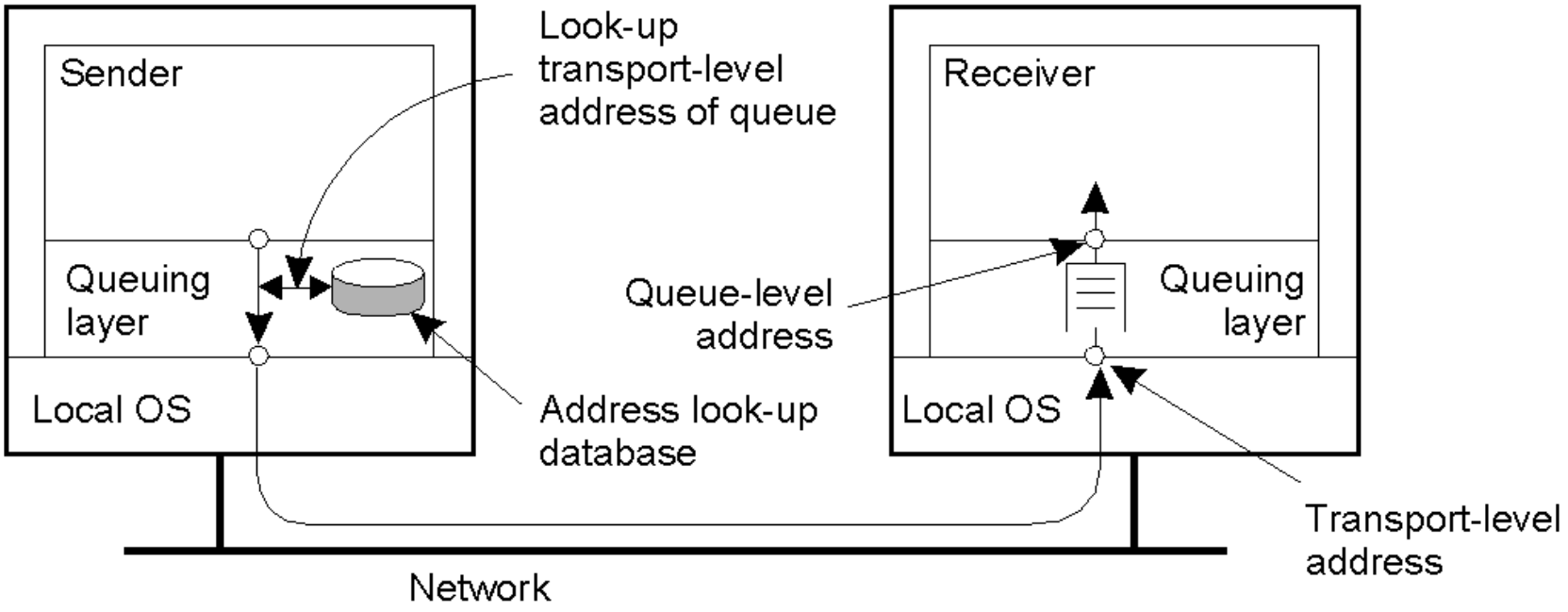
Four combinations for loosely-coupled communications using queues.

Message-Queuing Model (2)

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.

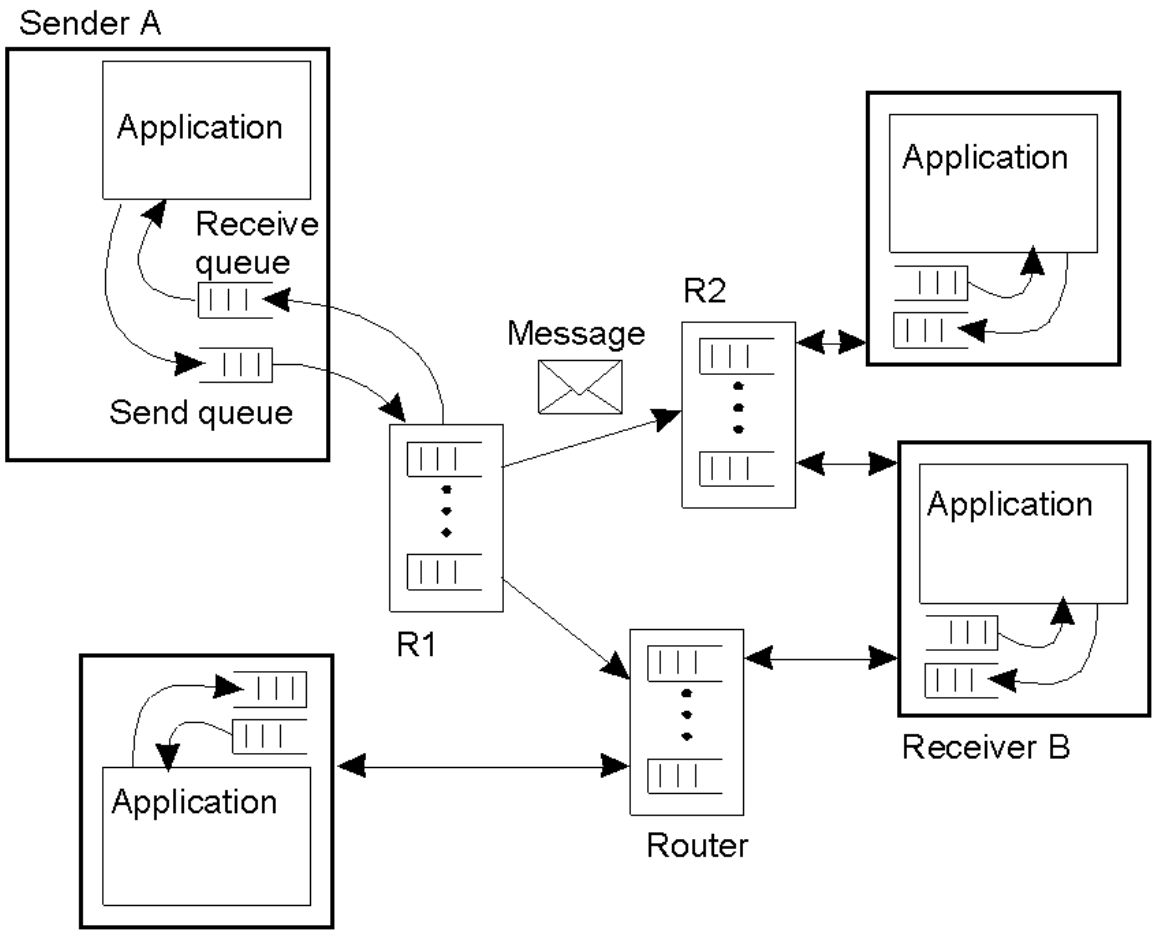
Basic interface to a queue in a message-queuing system.

General Architecture of a Message-Queuing System (1)



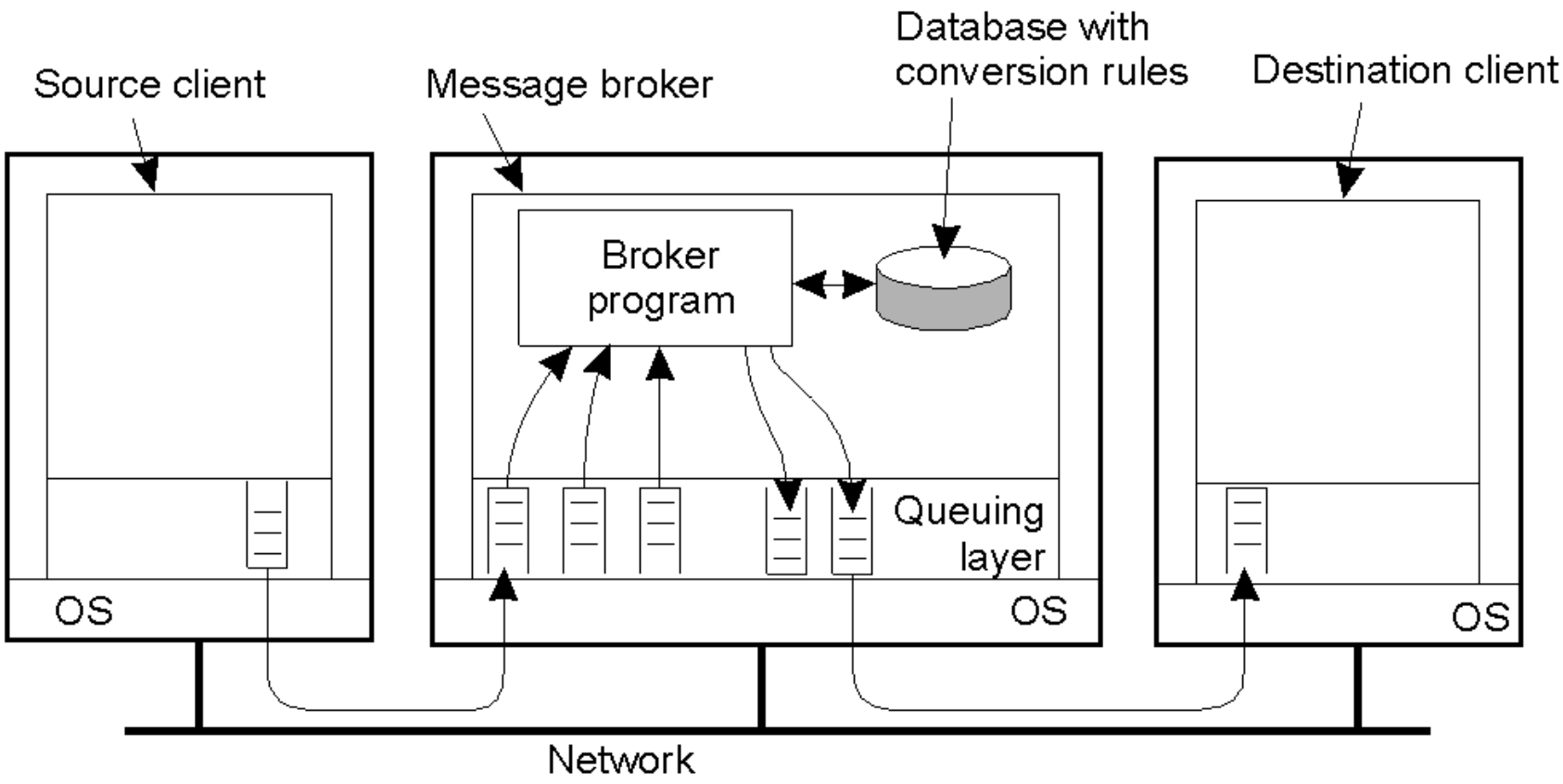
The relationship between queue-level addressing and network-level addressing.

General Architecture of a Message-Queuing System (2)



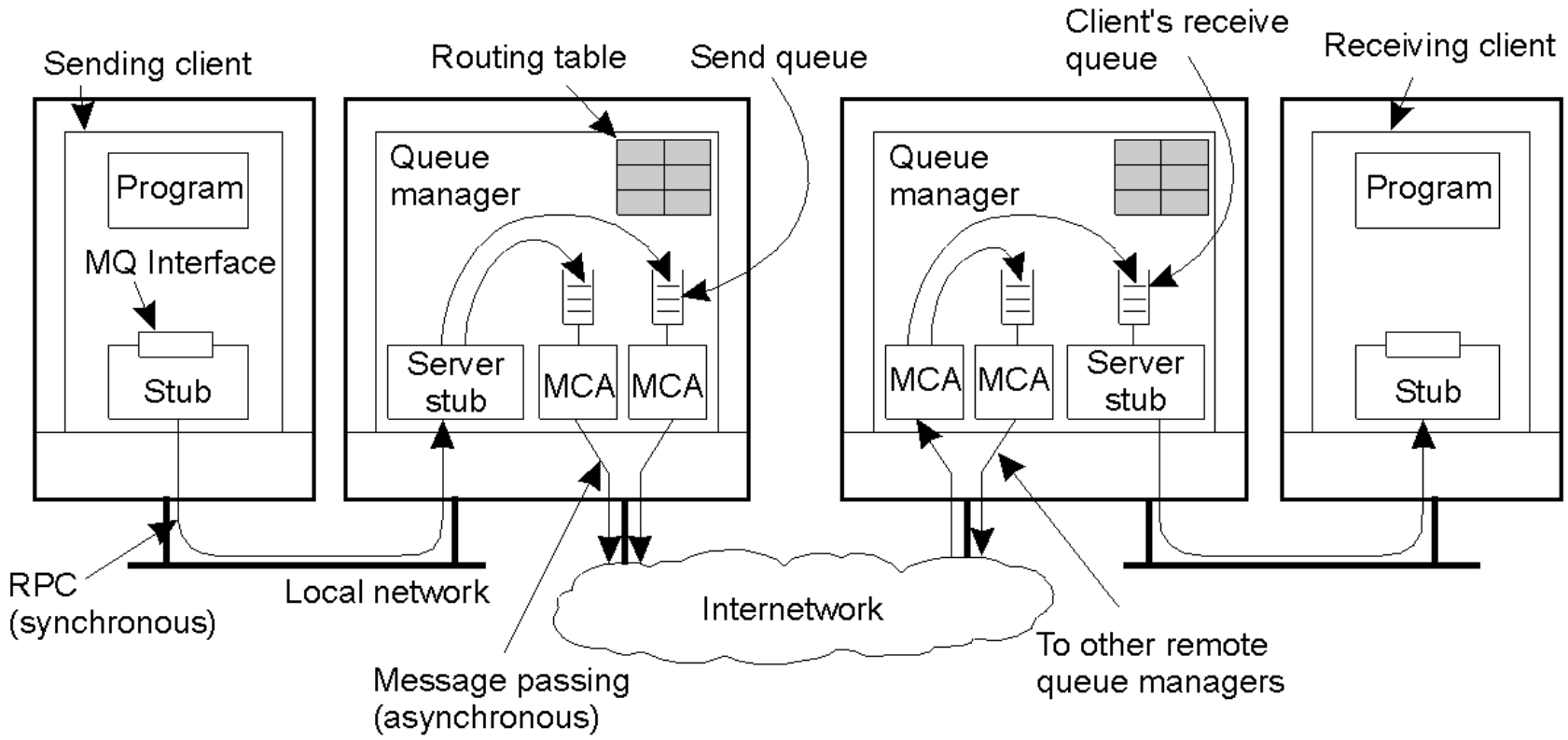
The general organization of a message-queuing system with routers.

Message Brokers



The general organization of a message broker in a message-queuing system.

Example: IBM MQSeries



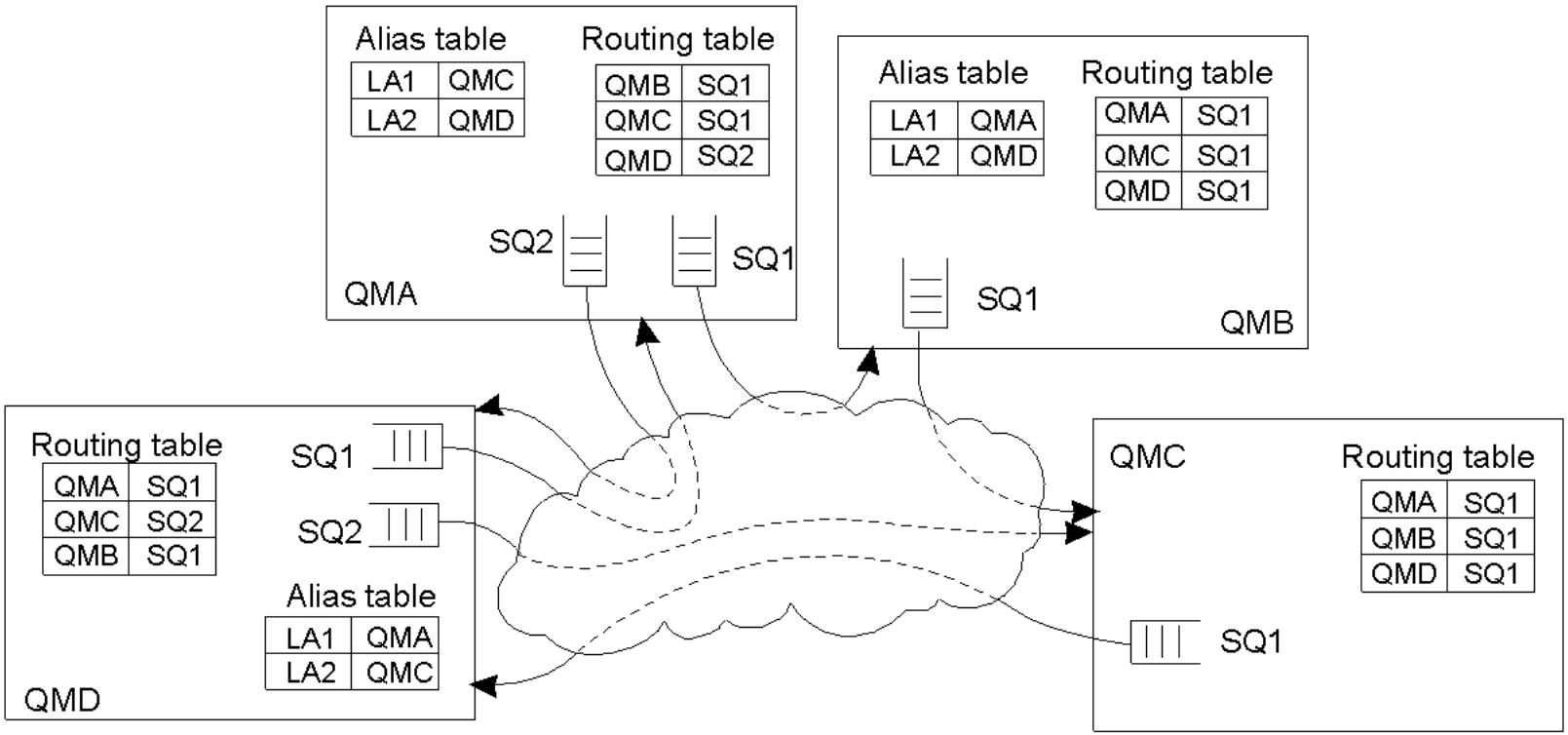
General organization of IBM's MQSeries message-queuing system.

Channels

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

Some attributes associated with message channel agents.

Message Transfer (1)



The general organization of an MQSeries queuing network using routing tables and aliases.

Message Transfer (2)

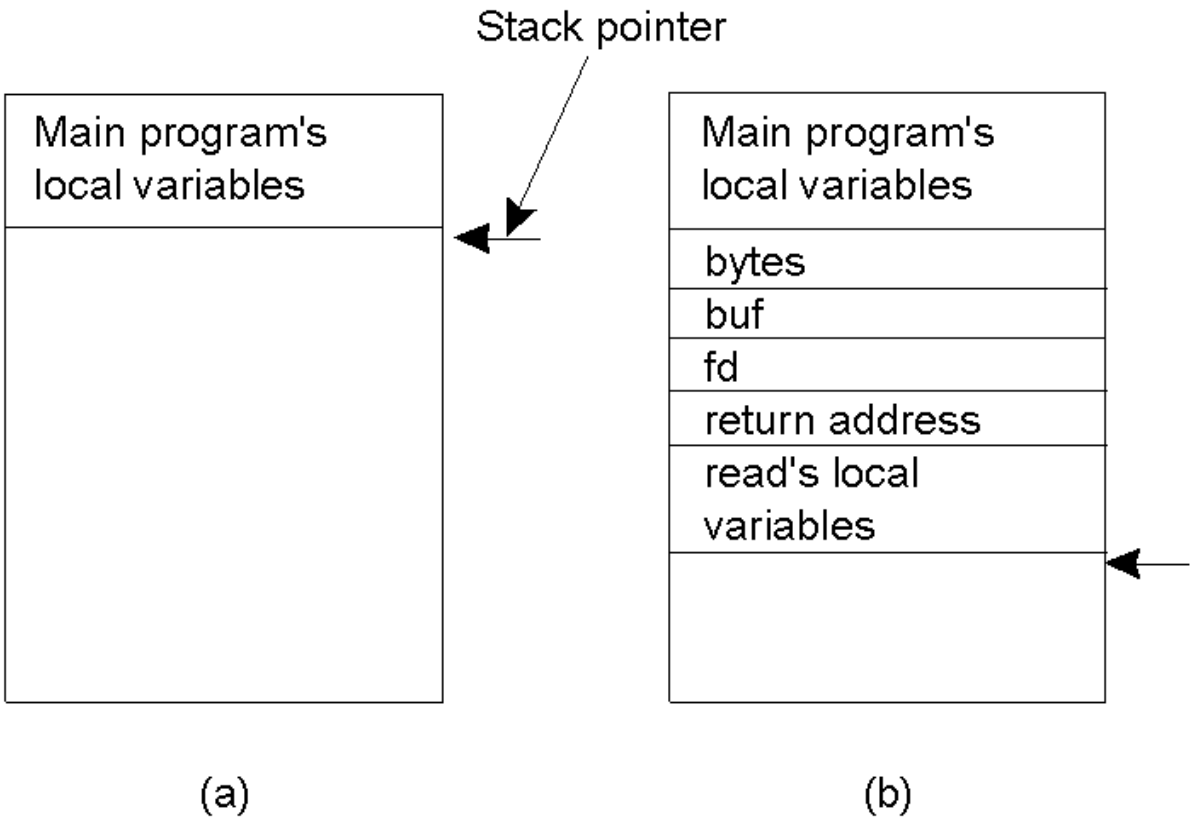
Primitive	Description
MQopen	Open a (possibly remote) queue
MQclose	Close a queue
MQput	Put a message into an opened queue
MQget	Get a message from a (local) queue

Primitives available in an IBM MQSeries MQI

Chamada Remota de Procedimentos

- Stubs & Skeletons -

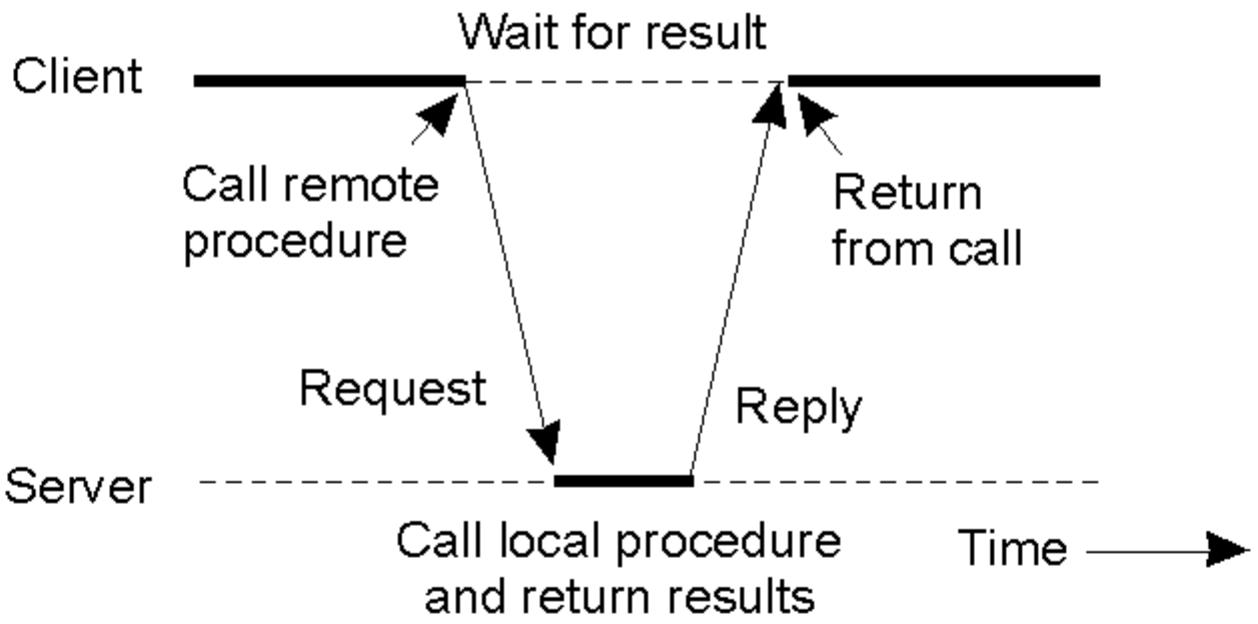
Conventional Procedure Call



a) Parameter passing in a local procedure call: the stack before the call to read

b) The stack while the called procedure is active

Client and Server Stubs

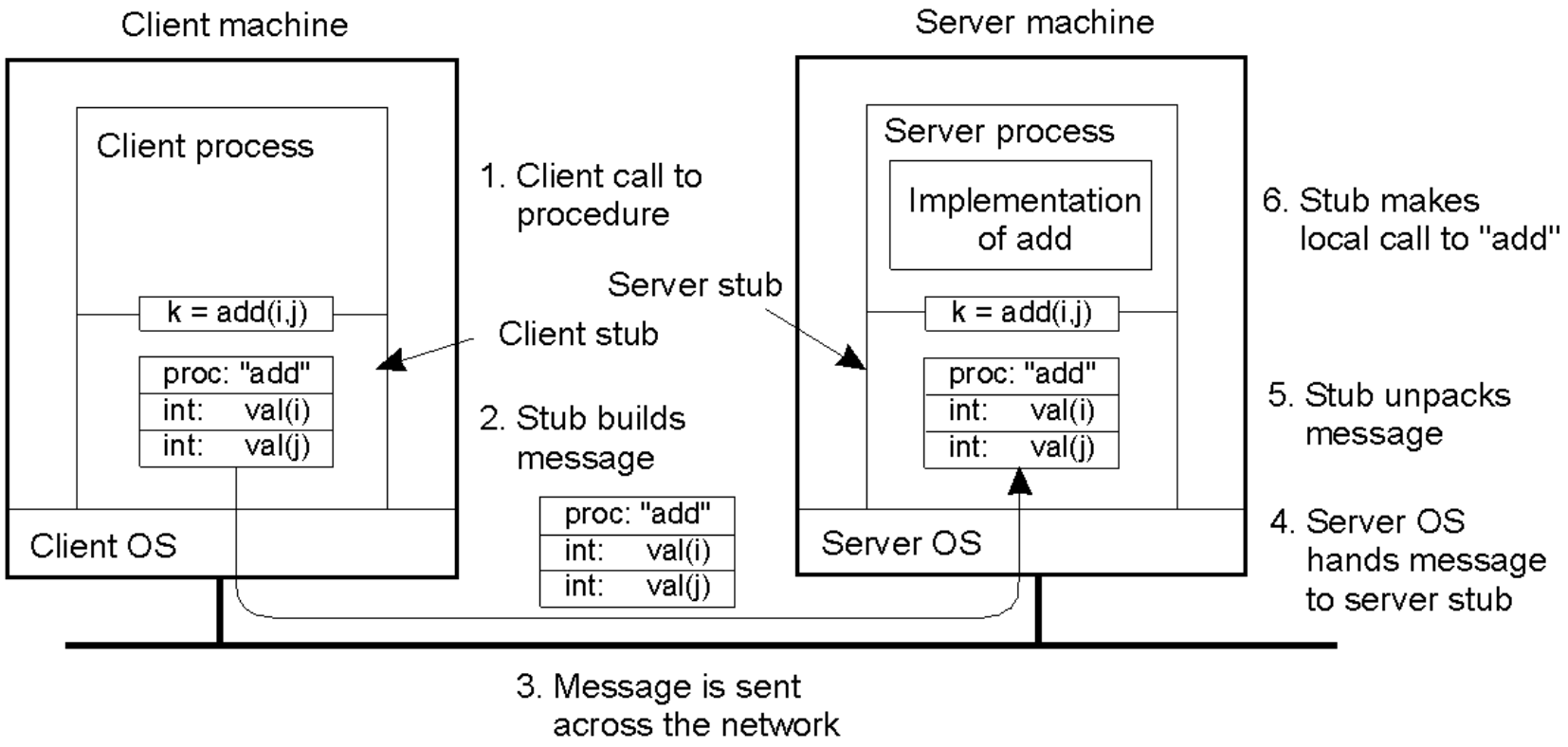


Principle of RPC between a client and server program.

Steps of a Remote Procedure Call

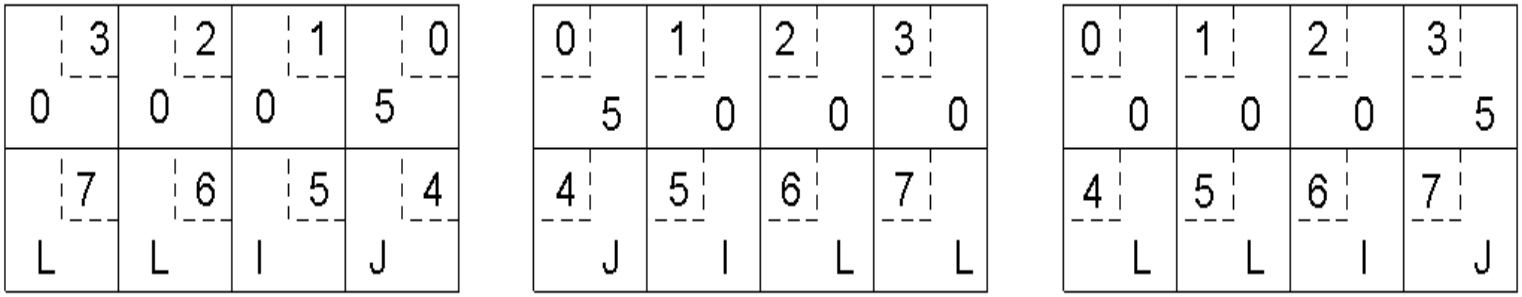
1. Client procedure calls client stub in normal way
2. Client stub builds message, calls local OS
3. Client's OS sends message to remote OS
4. Remote OS gives message to server stub
5. Server stub unpacks parameters, calls server
6. Server does work, returns result to the stub
7. Server stub packs it in message, calls local OS
8. Server's OS sends message to client's OS
9. Client's OS gives message to client stub
10. Stub unpacks result, returns to client

Passing Value Parameters (1)



Steps involved in doing remote computation through RPC

Passing Value Parameters (2)



a) Original message on the Pentium

b) The message after receipt on the SPARC

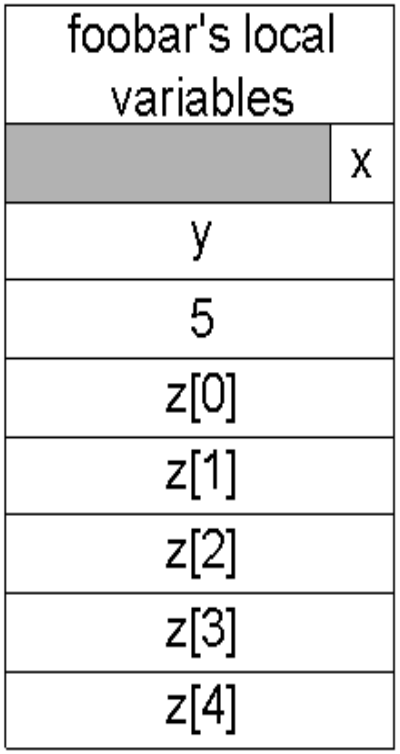
c) The message after being inverted. The little numbers in boxes indicate the address of each byte

Parameter Specification and Stub Generation

a) A procedure
b) The corresponding message.

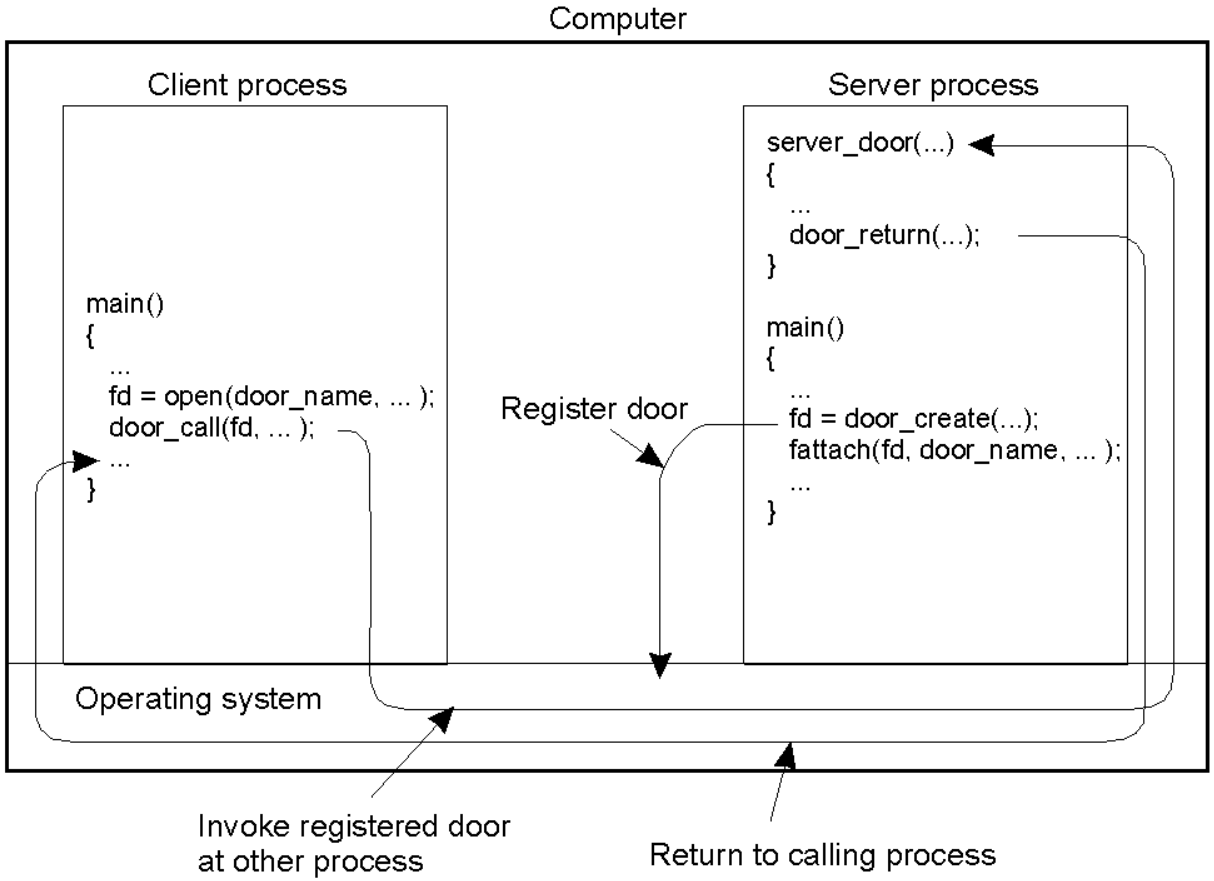
```
foobar( char x; float y; int z[5] )
{
   ....
}
```

(a)



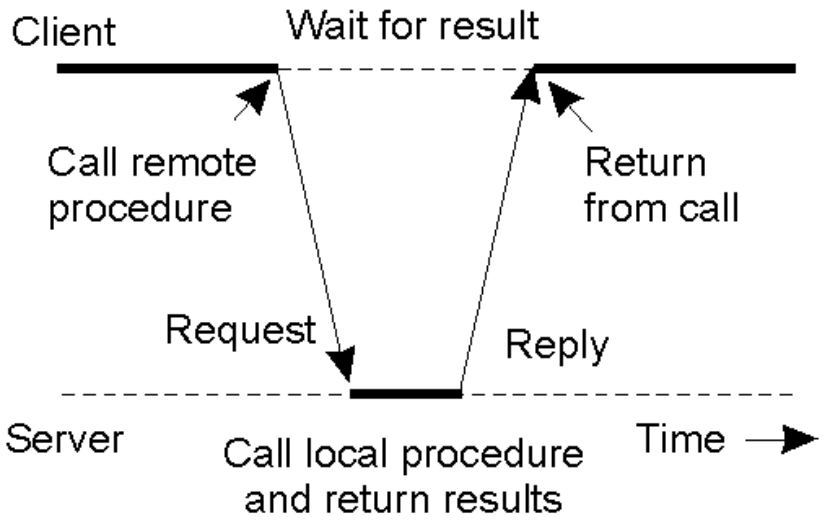
(b)

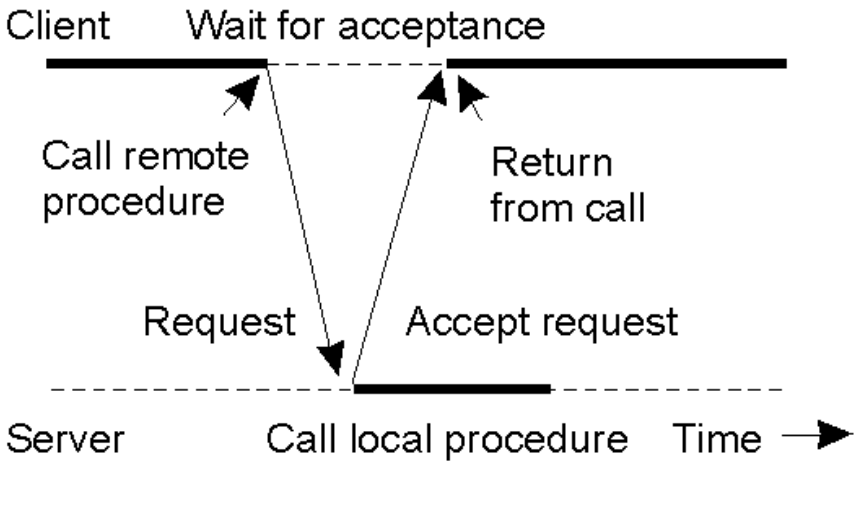
Doors



The principle of using doors as IPC mechanism.

Asynchronous RPC (1)

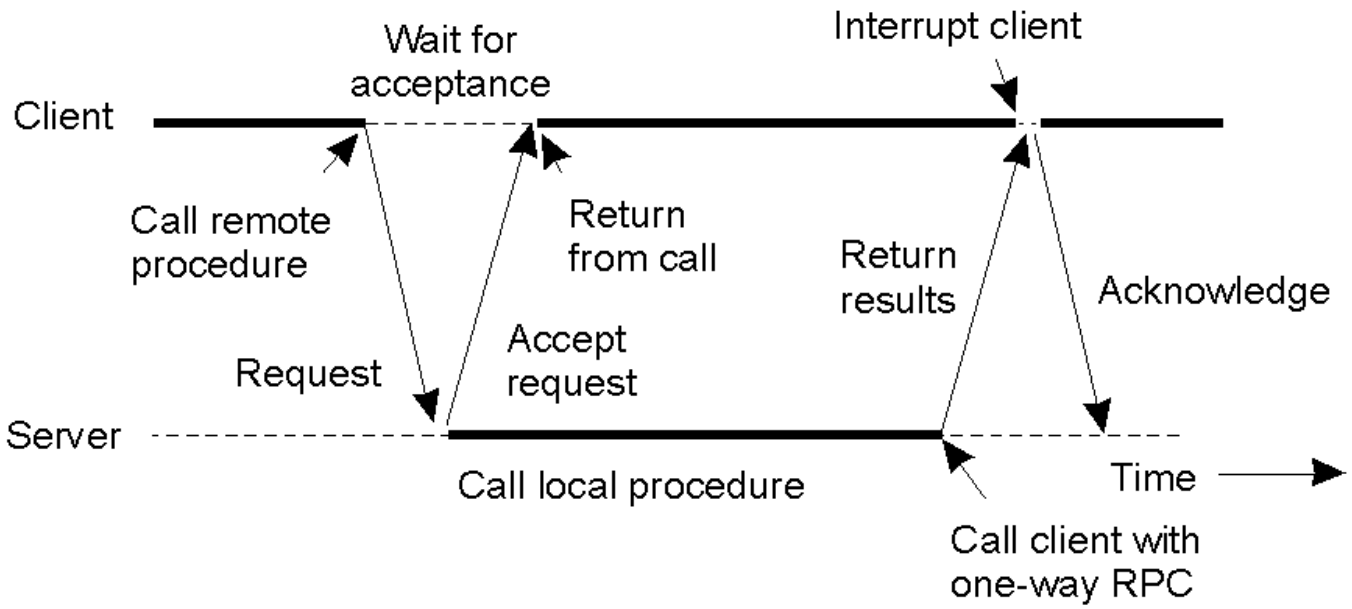




a) The interconnection between client and server in a traditional RPC

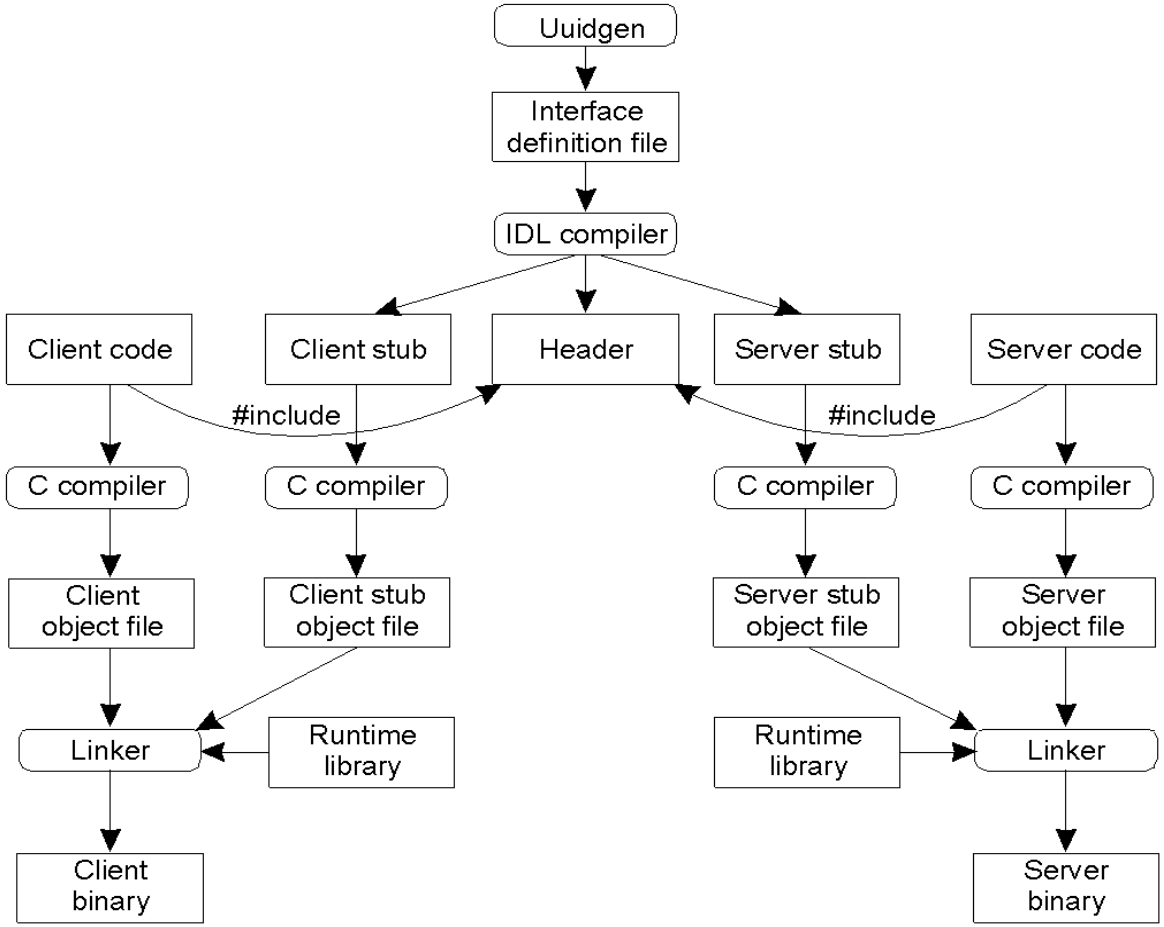
b) The interaction using asynchronous RPC

Asynchronous RPC (2)



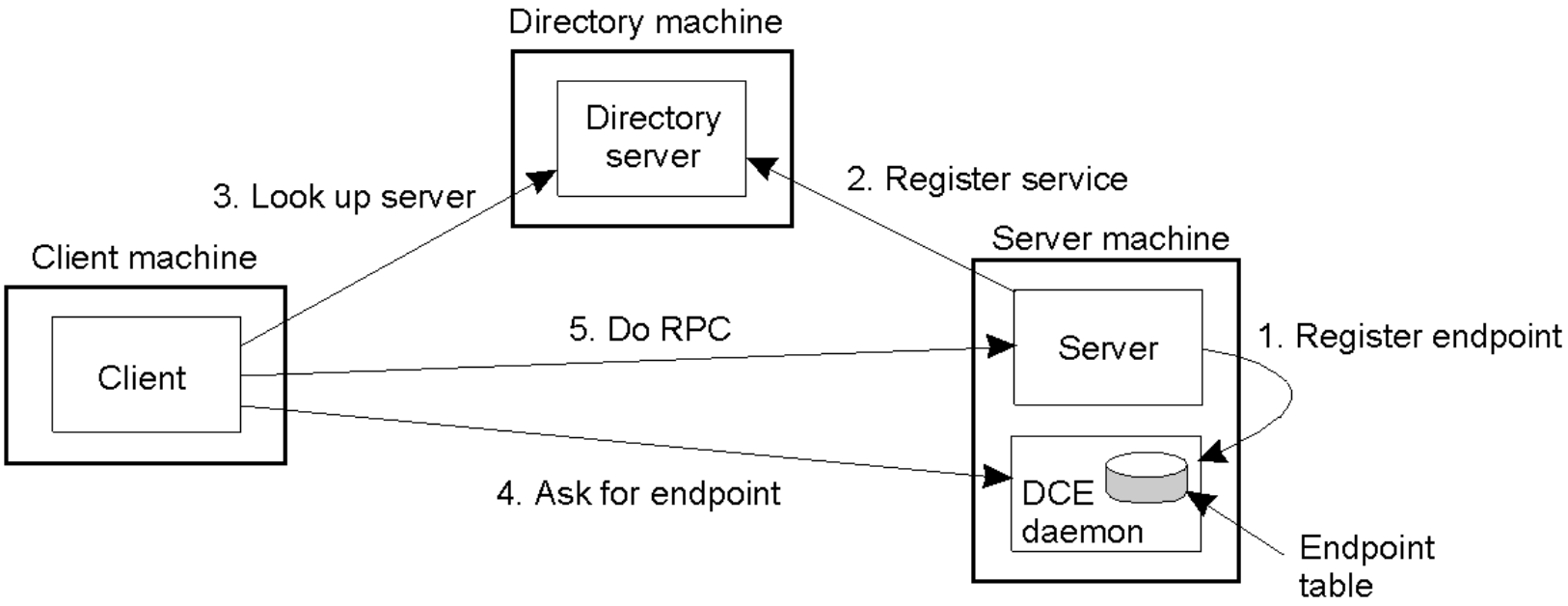
A client and server interacting through two asynchronous RPCs

Writing a Client and a Server



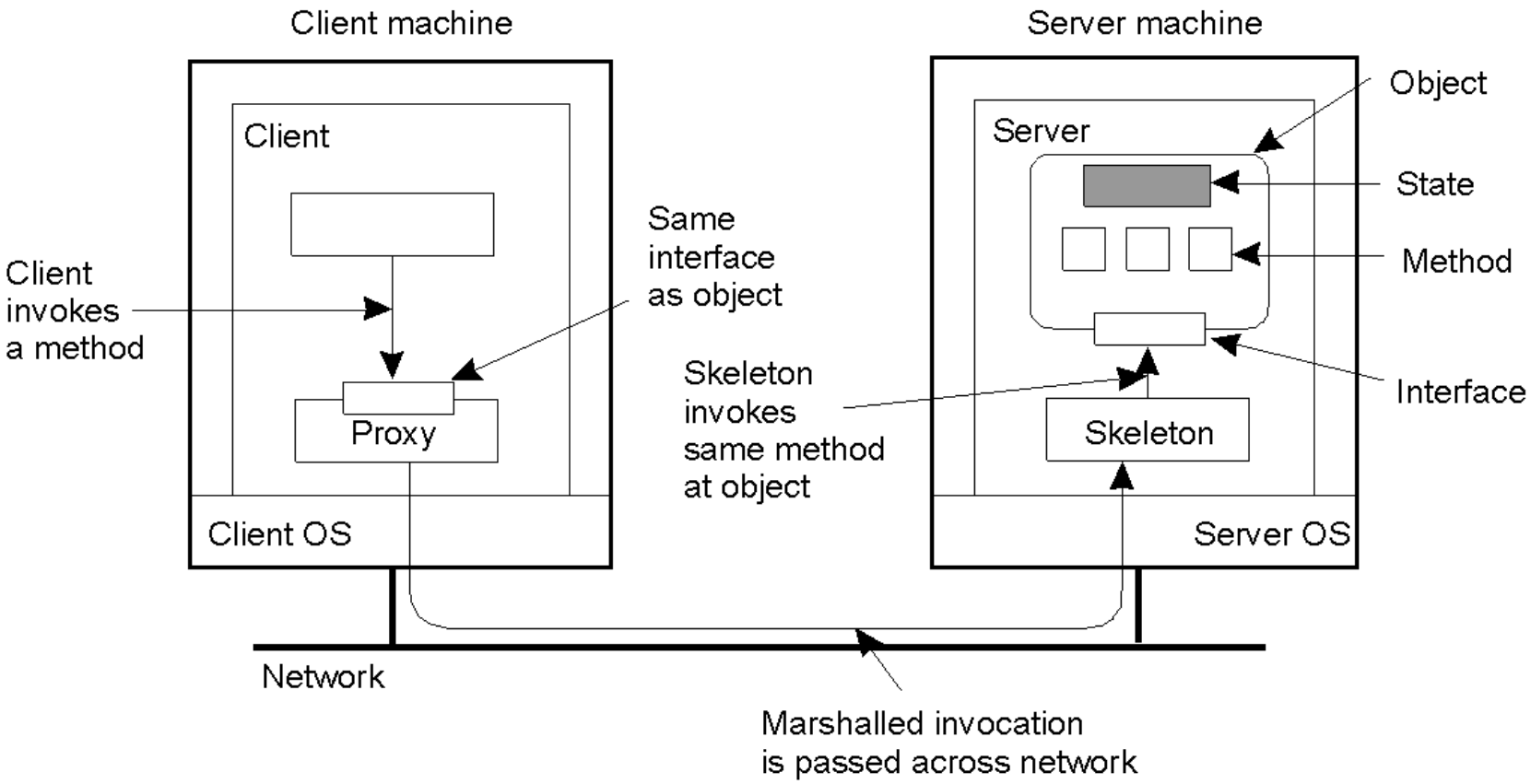
The steps in writing a client and a server in DCE RPC.

Binding a Client to a Server



Client-to-server binding in DCE.

Distributed Objects



Common organization of a remote object with client-side proxy.