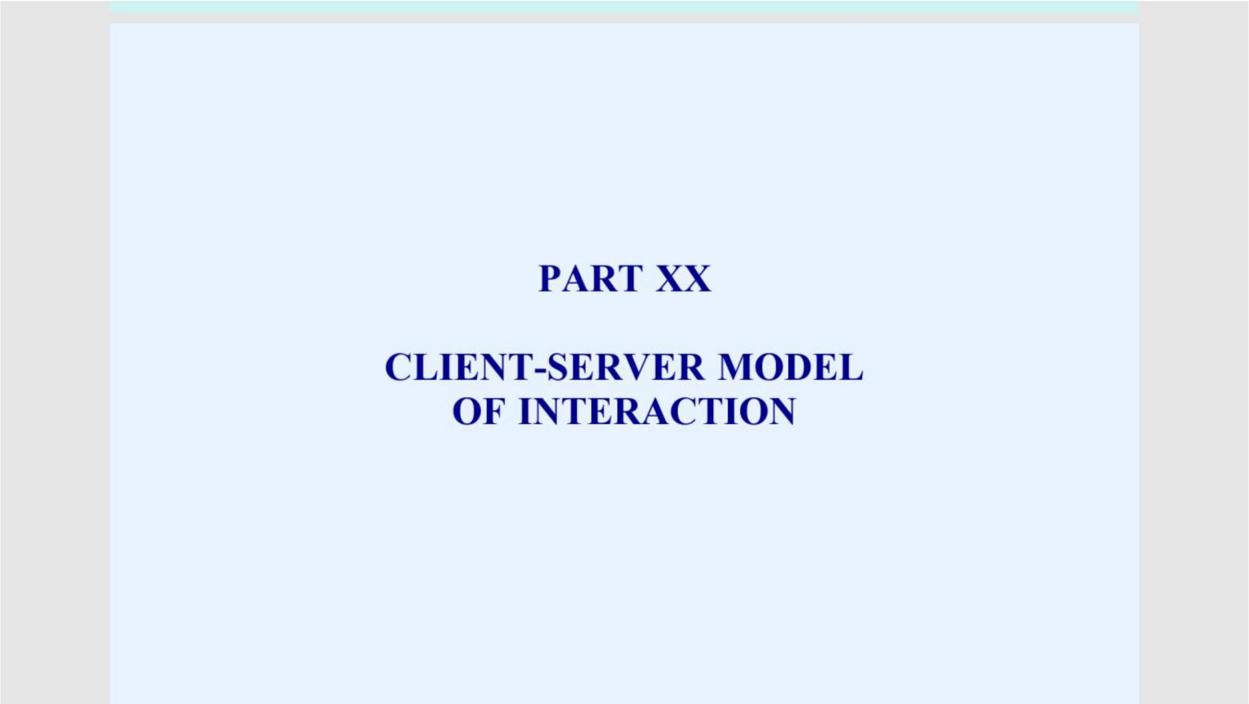
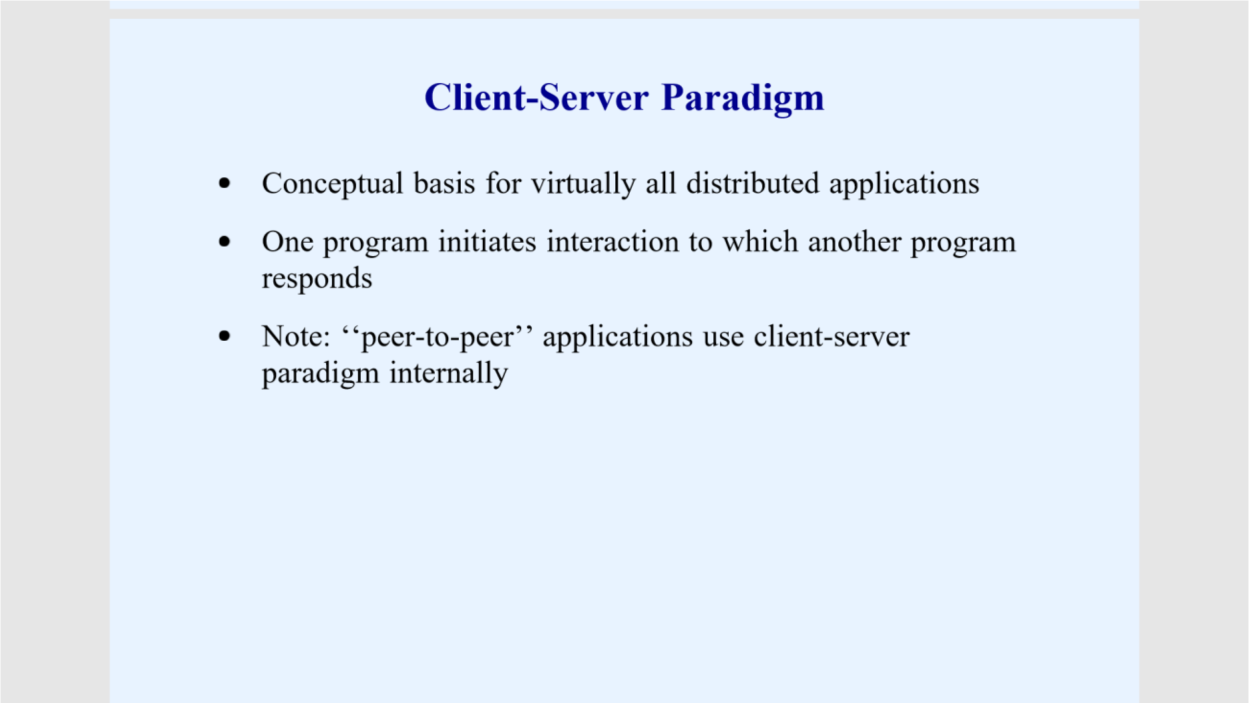


PART XX

CLIENT-SERVER MODEL OF INTERACTION



Client-Server Paradigm

- Conceptual basis for virtually all distributed applications
- One program initiates interaction to which another program responds
- Note: ‘‘peer-to-peer’’ applications use client-server paradigm internally

Definitions

- Client
  - Any application program
  - Contacts a server
  - Forms and sends a request
  - Awaits a response
- Server
  - Usually a specialized program that offers a service
  - Awaits a request
  - Computes an answer
  - Issues a response

Server Persistence

A server starts execution before interaction begins and (usually) continues to accept requests and send responses without ever terminating. A client is any program that makes a request and awaits a response; it (usually) terminates after using a server a finite number of times.

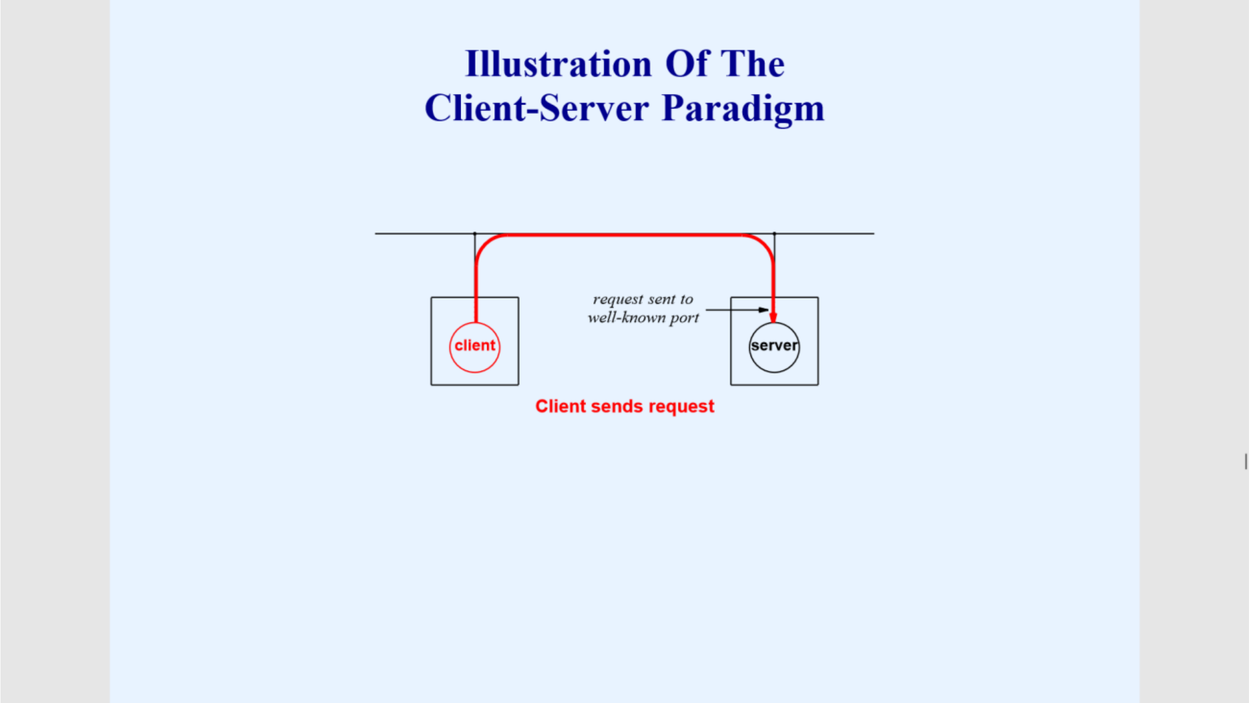


Illustration Of The Client-Server Paradigm

Client sends request

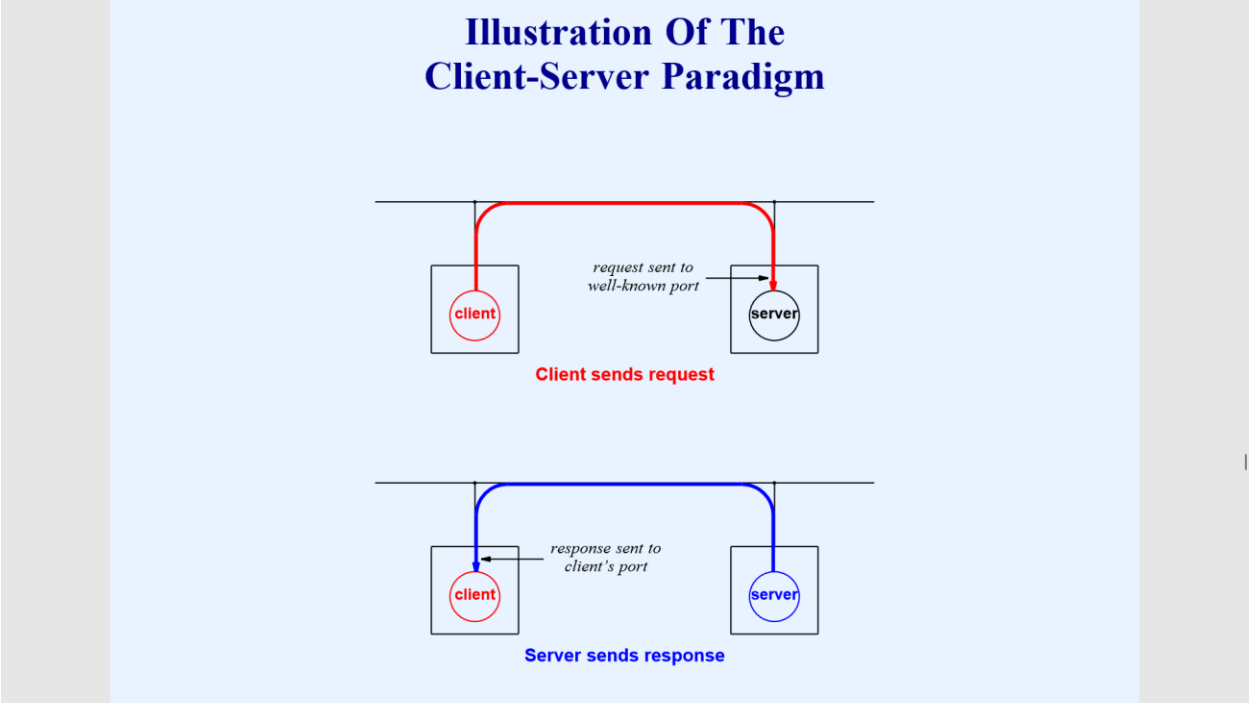


Illustration Of The Client-Server Paradigm

Client sends request

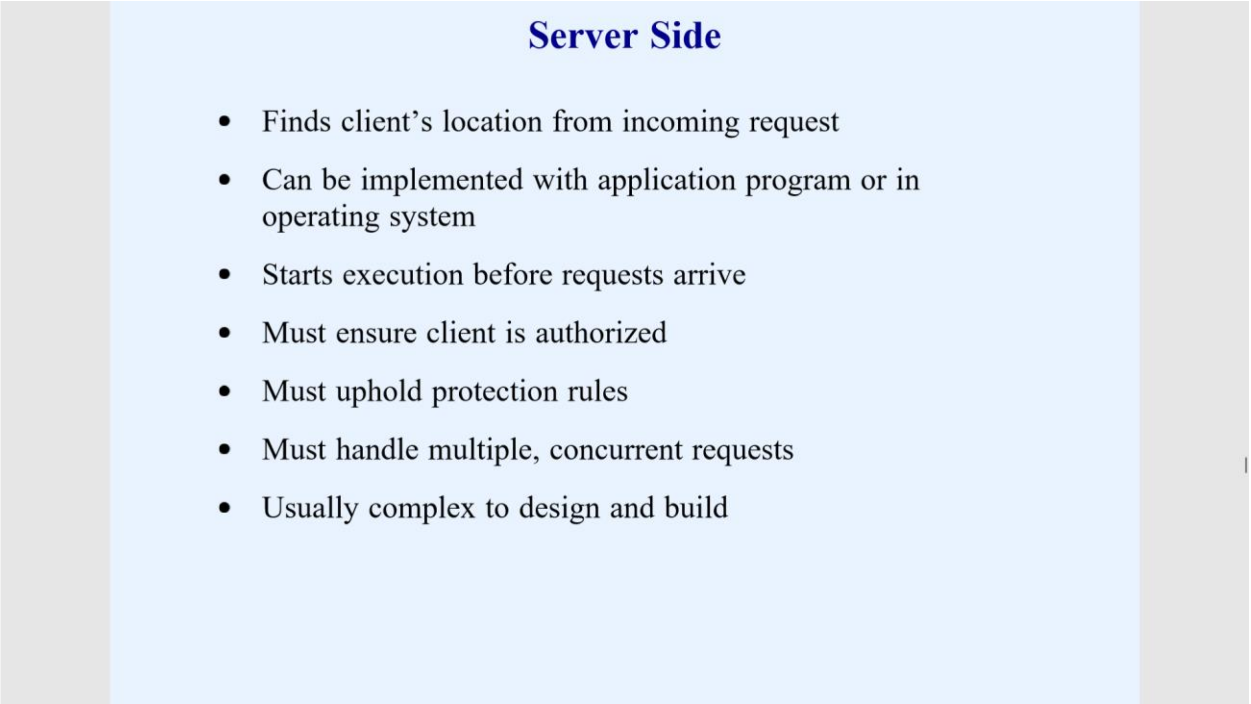
Server sends response

Use Of Protocol Ports

A server waits for requests at a well-known port that has been reserved for the service it offers. A client allocates an arbitrary, unused, nonreserved port for its communication.

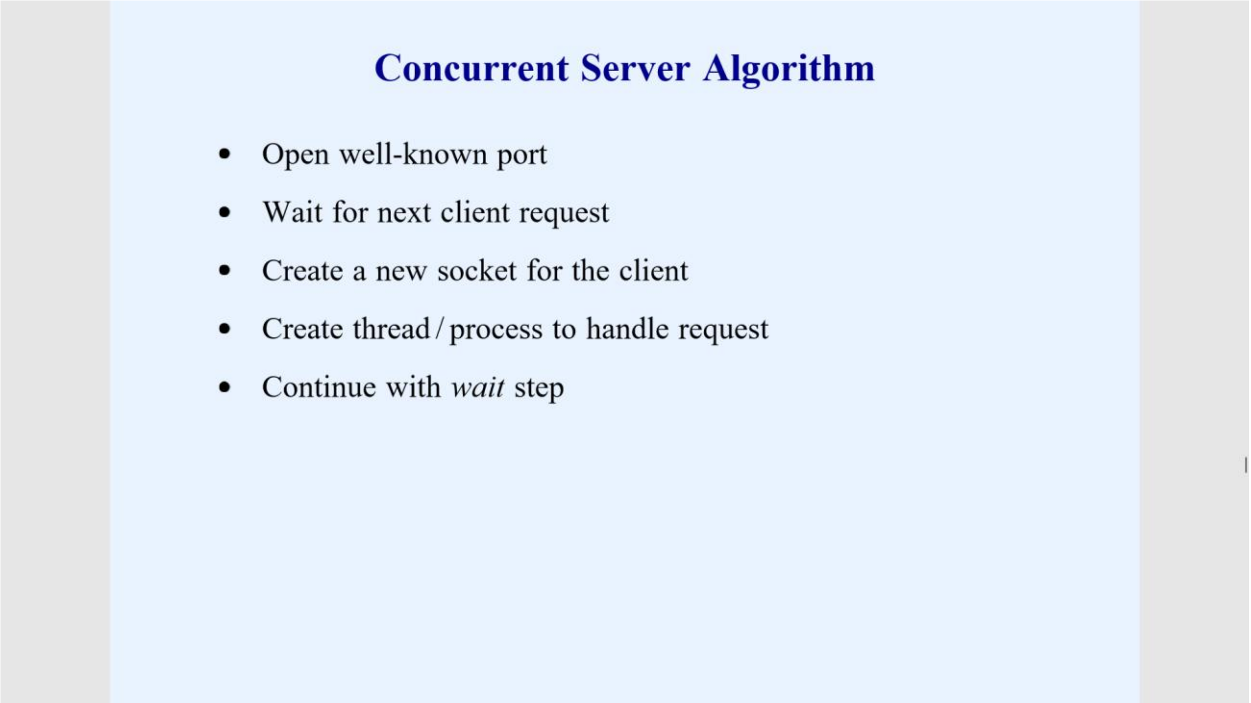
Client Side

- Any application program can become a client
- Must know how to reach the server
  - Server's Internet address
  - Server's protocol port number
- Usually easy to build



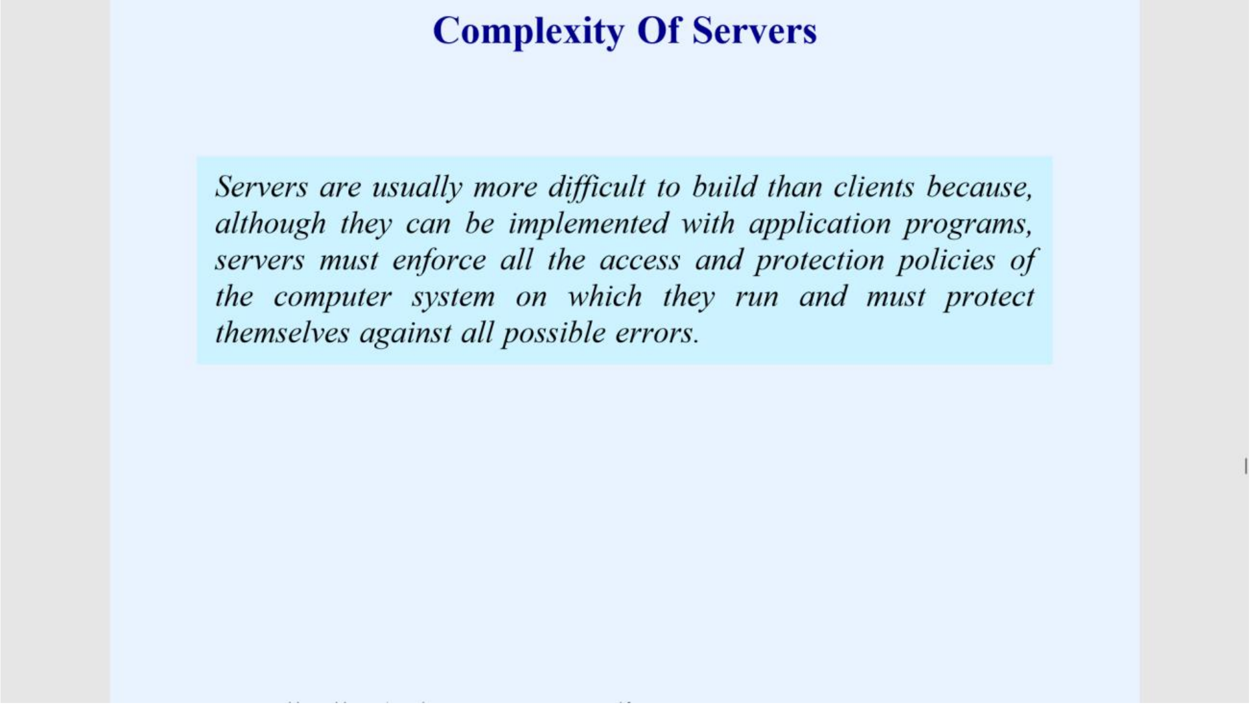
Server Side

- Finds client’s location from incoming request
- Can be implemented with application program or in operating system
- Starts execution before requests arrive
- Must ensure client is authorized
- Must uphold protection rules
- Must handle multiple, concurrent requests
- Usually complex to design and build



Concurrent Server Algorithm

- Open well-known port
- Wait for next client request
- Create a new socket for the client
- Create thread / process to handle request
- Continue with *wait* step



Complexity Of Servers

Servers are usually more difficult to build than clients because, although they can be implemented with application programs, servers must enforce all the access and protection policies of the computer system on which they run and must protect themselves against all possible errors.