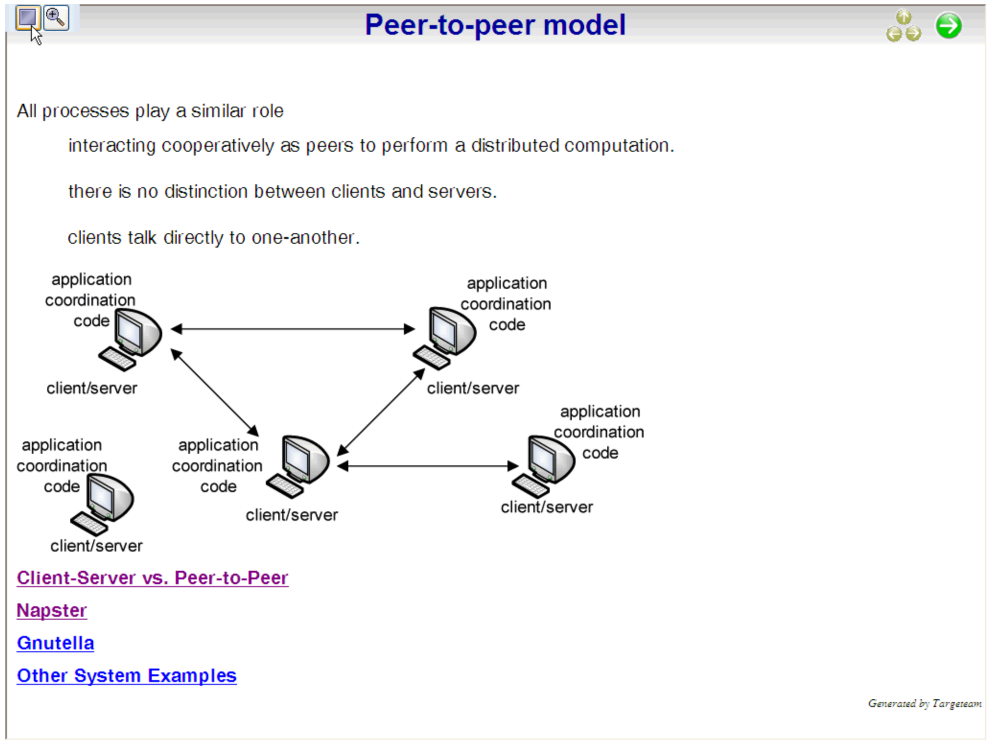
Script generated by TTT

Title: Distributed_Applications (08.05.2012)

Date: Tue May 08 14:30:13 CEST 2012

Duration: 91:04 min

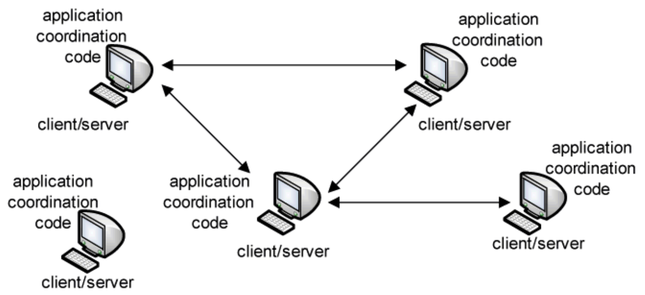
Pages: 25



Peer-to-peer model

All processes play a similar role

- interacting cooperatively as peers to perform a distributed computation.
- there is no distinction between clients and servers.
- clients talk directly to one-another.



application coordination code
client/server
application coordination code
client/server
application coordination code
client/server
application coordination code
client/server
application coordination code
client/server

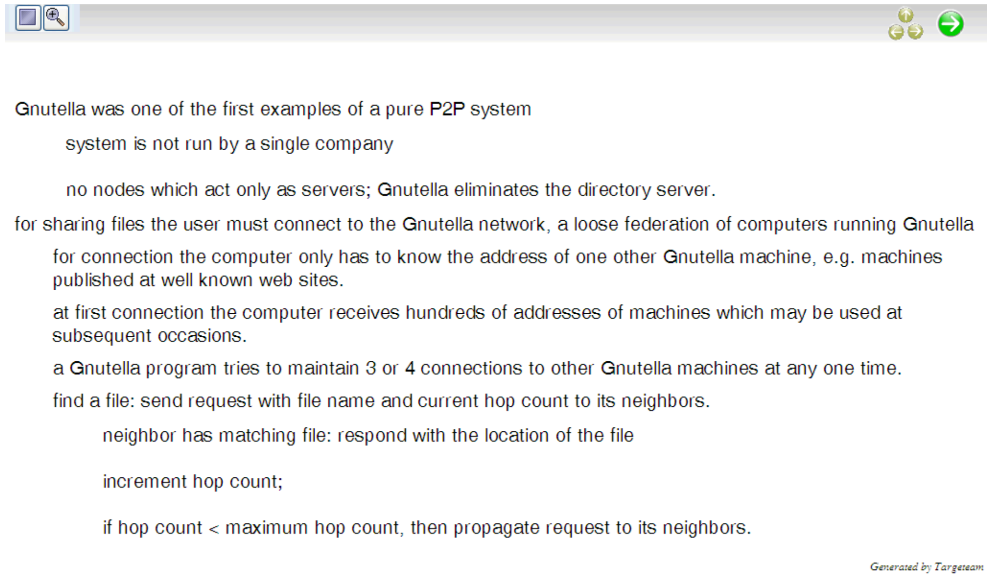
Client-Server vs. Peer-to-Peer

Napster

Gnutella

Other System Examples

Generated by Targeteam



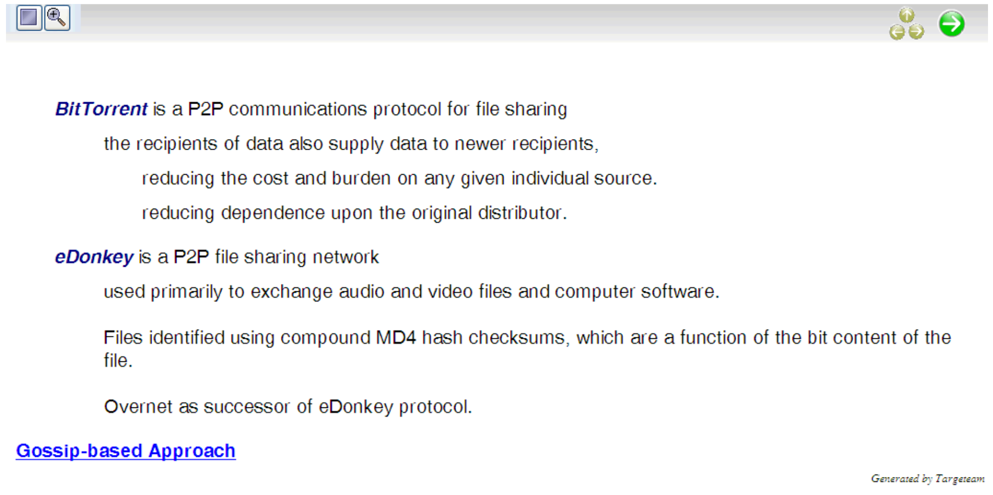
Gnutella was one of the first examples of a pure P2P system

- system is not run by a single company
- no nodes which act only as servers; Gnutella eliminates the directory server.

for sharing files the user must connect to the Gnutella network, a loose federation of computers running Gnutella

- for connection the computer only has to know the address of one other Gnutella machine, e.g. machines published at well known web sites.
- at first connection the computer receives hundreds of addresses of machines which may be used at subsequent occasions.
- a Gnutella program tries to maintain 3 or 4 connections to other Gnutella machines at any one time.
- find a file: send request with file name and current hop count to its neighbors.
  - neighbor has matching file: respond with the location of the file
  - increment hop count;
  - if hop count < maximum hop count, then propagate request to its neighbors.

Generated by Targeteam



BitTorrent is a P2P communications protocol for file sharing

- the recipients of data also supply data to newer recipients,
  - reducing the cost and burden on any given individual source.
  - reducing dependence upon the original distributor.

eDonkey is a P2P file sharing network

- used primarily to exchange audio and video files and computer software.
- Files identified using compound MD4 hash checksums, which are a function of the bit content of the file.
- Overnet as successor of eDonkey protocol.

Gossip-based Approach

Generated by Targeteam

Propagate information in the same way as epidemic diseases spread.

- approach explained informally
  - time t_0 : suppose I know something new
  - time t_1 : I pick a friend and tell him; now 2 people know.
  - time t_2 : we each pick a friend and tell them; now 4 people know
  - time t_3 : ....
- information spreads at exponential rate.
  - due to re-infection information spreads at approx. 1.8^k after k rounds.
- combination of push and pull works best.
- algorithm is quite robust and scalable
  - information travels on exponentially many paths.
  - difficult to slow down.
  - the load of the participating nodes is independent of the system size.
  - information spreads in log(system size) time.
  - network load is linear in system size.

BitTorrent is a P2P communications protocol for file sharing

- the recipients of data also supply data to newer recipients,
  - reducing the cost and burden on any given individual source.
  - reducing dependence upon the original distributor.

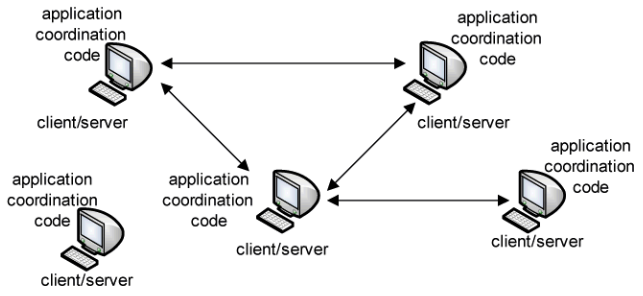
eDonkey is a P2P file sharing network

- used primarily to exchange audio and video files and computer software.
- Files identified using compound MD4 hash checksums, which are a function of the bit content of the file.
- Overnet as successor of eDonkey protocol.

Gossip-based Approach

All processes play a similar role

- interacting cooperatively as peers to perform a distributed computation.
- there is no distinction between clients and servers.
- clients talk directly to one-another.


application coordination code
client/server
application coordination code
client/server
application coordination code
client/server
application coordination code
client/server
application coordination code
client/server

Client-Server vs. Peer-to-Peer

Napster

Gnutella

Other System Examples

Paradigms for distributed applications

certain order for message delivery

messages to a group of recipients: messages arrive in different order, due to different transmission times.

One sender

There are the following ordering schemes:

according to the message arrival on the recipient's side; different receivers can have different message arrival sequences.

according to message sequence number generated by the sender; this approach is sender-dominated.

receiver creates a serialization according its own criteria.

Several senders

If several senders are involved, the following message ordering schemes may be applied:

1. no serialization.
2. loosely-synchronous.

   There is a loosely synchronized global time which provides a consistent time ordering.

3. virtually-synchronous.

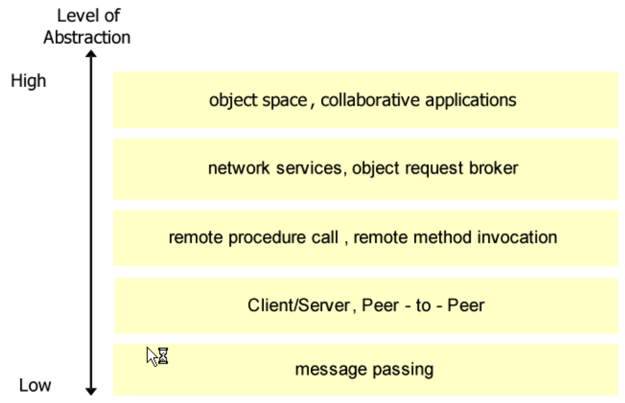
   The message order is determined by causal interdependencies among the messages. For example, a message N has been sent after another message M has been received, i.e. N is potentially dependent on M.

4. totally ordered.

   by token: before a sender can send a message, it must request the send token.

   a selected component (the coordinator) determines the order of message delivery for all recipients.

Levels of Abstraction



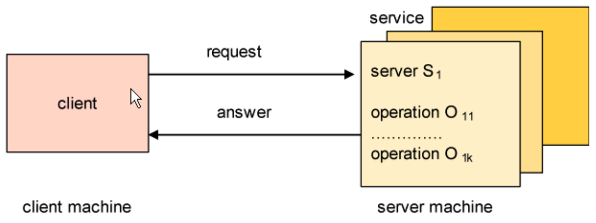
Level of Abstraction
High
Low
object space, collaborative applications
network services, object request broker
remote procedure call, remote method invocation
Client/Server, Peer - to - Peer
message passing

Client-server model

The client-server model implements a sort of ***handshaking principle*** , i.e., a client invokes a server operation, suspends operation (in most of the implementations), and resumes work once the server has fulfilled the requested service.

Definition: A **time service** provides a synchronized system-wide time for all nodes in the network.

Terms and definitions



sender, receiver: pure message exchanging entities.

client, server: entities acting in some specialized protocol.

Client

Definition: A **client** is a process (some say, an application) that runs on a client machine and that typically initiates requests for service operations.

Potential clients are a priori unknown.

Service

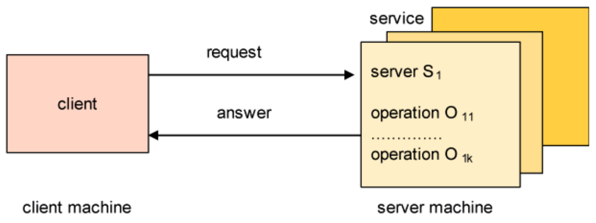
Definition: A **service** is a piece of software that provides a well-defined set of services. This piece of software may run on one or multiple (server) machines.

Server

Definition: A **server** is a subsystem that provides a particular service to a set of a priori unknown clients. A server executes a (piece of) service software on a particular server machine. Obviously, a single server machine can host multiple server subsystems.

A server provides a set of operations (procedures).

Terms and definitions

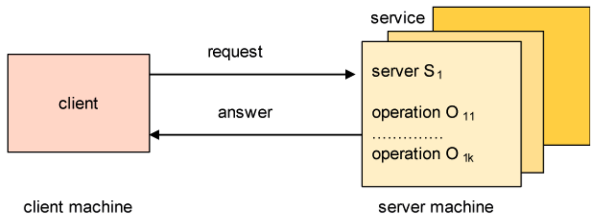


Definitions

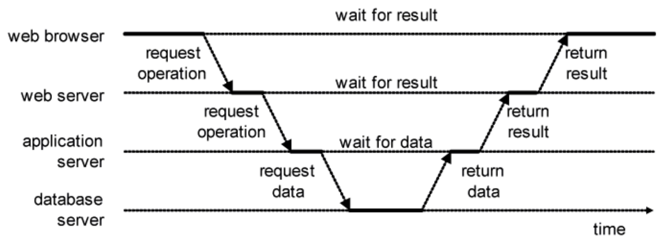
Client-server interfaces

Multitier architectures

Terms and definitions



[Definitions]
[Client-server interfaces]
[Multitier architectures]



Client					
presentation execution	presentation	presentation	presentation execution	presentation execution (with local database)	presentation execution database
Server					
database	presentation execution database	execution database	execution database	execution (with local database)	database
Case 1	Case 2	Case 2	Case 3	Case 3	Case 4

Different cases

Case 1: remote data storage. access, for example, via Sun NFS.

Case 2: remote presentation (for example X window system).

Case 3: distributed application

cooperative processing among the individual components of an application.

Case 4: distributed data storage

The information is distributed between client and server; information replication is possible.

clients and servers have different life spans; servers manage these requests in a queue.

[Single dedicated server process]

[Cloning of new server processes]

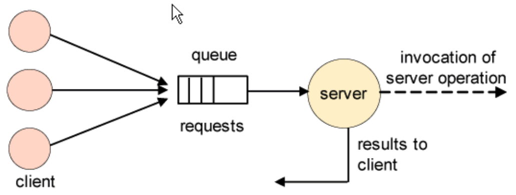
Parallel request processing through threads

This is a variant of the second approach.

Shared address space, i.e. the approach allows shared utilization of variables;

Single dedicated server process

A single dedicated server process is in charge of processing requests for service operations.



no parallel processing of requests, which results in the following disadvantage:

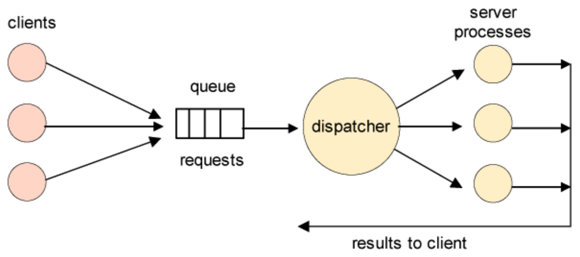
approach may be time consuming.

no interruption of the processing of the current request when a higher prioritized request appears in the queue.

server becomes bottleneck.

Cloning of new server processes

Every incoming request is handled by a new server process.



Cloning of new server processes is expensive;

Synchronization of access to shared persistent data;

Parallel processing of several applications is possible;

Processing of service requests

clients and servers have different life spans; servers manage these requests in a queue.

Single dedicated server process

Cloning of new server processes

Parallel request processing through threads

This is a variant of the second approach.

Shared address space, i.e. the approach allows shared utilization of variables;

Definition: A **file service** [Svobodova 1984] provides (remote) centralized data storage facilities to clients distributed among a network.

server deals with bulk data storage, high performance computation, collecting/managing large amounts of data.

client deals with "attractive" display, quick interaction times.

use of caching to speed up response time. Use of cache "hints" to facilitate cache management

speed up system when hint is correct.

mechanism to detect wrong hint and seek up-to-date information.

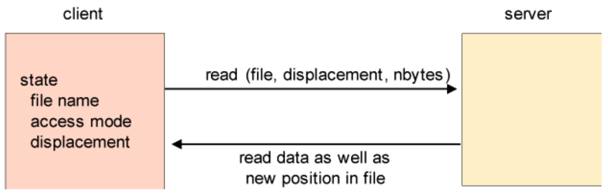
Distinction between stateless and stateful

Stateless server

Stateful server

Stateless server

Stateless server do not manage any state information about their clients; the client must supply all necessary parameters to process the request.

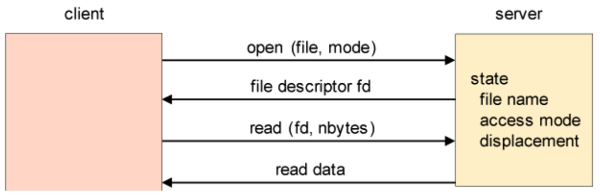


stateless server does not track clients or ensure that cached data stays up to date ⇒ cache refresh is responsibility of the client.

client uses often write-through caching policy.

A crashed server can be restarted without dealing with state reinstallments.

Stateful server subsystems manage state information about their clients.



server tracks its clients and takes actions to keep their cached states up-to-date. Client can trust its cached data ⇒ cache is owned by the server.

As a consequence, programming at the client site becomes less complex.

stateful transactional server architecture: after recovery of server crash an abort message is sent to client.