Client-server systems

Introduction

Database + Interface with user

Good \[\Rightarrow\] Better

Interface with user

Translation protocol

Database and retrieval “engine”

“Client-server”
Client-server systems: description

- A “client” is a program that runs on the computer which you access in the first place (often your desktop PC or an online access computer).

- Each client provides an interface to each of the “services” (databases, online files, e-mail, …) that are made available by other systems, which are called “servers.”

Client-server computing architecture

<table>
<thead>
<tr>
<th>Client computer</th>
<th>Server computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>• is often managed by a user of the server’s services, who is not a computer expert</td>
<td>• is often managed by a computer expert</td>
</tr>
<tr>
<td>• sends message(s) / requests to the server</td>
<td>• on which information and/or a service resides</td>
</tr>
<tr>
<td>• on which results are prepared and displayed</td>
<td>• on which information is prepared for the client</td>
</tr>
</tbody>
</table>

Client-server protocol
Client-server database systems: description

- The user specifies which database(s) to query and formulates a query, using the client software.
- The client software then connects to the database(s) and submits the query, in a structure suitable for communication between client and server.
- The server retrieves data from the database, orders these, and returns these to the client.
- The client processes the incoming data, and presents them to the user.

Client-server database systems: software involved

Diagram showing the software components:
- **Client software module** = User interface
- **Server software module** = Indexing + retrieval

Arrows indicating the interaction between user and data, according to the specific client-server protocol.
Client-server database systems: computers that may be involved

- User’s personal computer
  - Queries
  - Buyer of data
  - Consumer

- Client computer
  - Client program to formulate queries
  - Intermediary
  - Waiter in restaurant

- Data and retrieval engine computer
  - Data / Information
  - Seller of data
  - Cook

Client-server systems: specific cases (Part 1)

- 1 computer, 1 client/server system
  Example: a client-server database system used on a personal computer

- 2 computers, 1 client/server system!
  client software at user’s personal computer
  + server software at remote server computer
  (= most common situation)
  Examples: telnet, WWW, e-mail, …
Client-server systems: specific cases (Part 2)

- 2 computers, 2 client/server systems
  Example: telnet to another computer and on that computer another client/server process

Client-server systems: specific cases (Part 3)

- 3 computers, 2 client/server systems
  Example: telnet to another computer and on that computer start of another client/server process with a 3rd computer
**---

Client-server systems: “thin” and “fat” clients

- Dumb terminal *(not really a client)*
- X- or ICA-terminal *(thin client)*
- Network computer: NC or NetPC *(thin or fat client)*
- PC *(fat client)*

---

Increasing
- cost of hardware
- cost of maintenance *(cost of ownership)*
- complexity
- host/server independence and possible customization

---

Client-server systems: dumb terminals as clients

- Wired to a central computer *(mainframe or minicomputer)*.
- Not really a client in a client-server system, because programs originate and execute on the central computer, and not on the terminal.
- Text-based; only poor graphics built with characters.
Client-server systems: X- or ICA-terminals as clients

- Networked to a server computer.
- Programs originate and execute on the server computer; graphics execute on the terminal.
- Example of a “thin” client.

Client-server systems: NC network computers as clients

- Networked to a server computer.
- Programs originate on the server computer, and execute on the server or on the terminal (= the NC network computer, in this case).
- Example of a “thin” client.
Client-server systems: 
P PCs as clients

• In the case of a client-server system: networked to a server computer.
• Programs usually originate and execute on the terminal (= PC, in this case), but may also originate and execute on the server computer.
• A “fat” client.

Client-server systems: 
thin clients

Thin clients can work best in the following situations:
• When older, text-based terminals are replaced.
• When users work with a limited set of programs.
• When users share desktops.
• When remote users are difficult to support.
• When the applications revolve around remote data instead of local data.
• When data security is most important.
Client-server systems

Benefits and problems

?? Question ??

Which benefits offers the client-server architecture to the user?
Client-server systems: benefits for the user of the system (Part 1)

😊 The user needs to know a smaller number of user interfaces, when more than one server can be accessed by the same client, compared with the case when every system offers its own interface.

Client-server systems: benefits for the user of the system (Part 2)

😊 When more than 1 client is available for a type of server, then the user can make a choice and work with the interface offered by that particular client software, which is well adapted to his/her situation, knowledge and experience.
Client-server systems: benefits for the distributor of data

The distributor needs to be concerned less with the user interface; instead, his/her server only has to be compatible with the “important” client programs, that is, with one or several important client-server protocols.

Due to the benefits for the user (sometimes = buyer), the number of users / consumers / buyers will grow.

Client-server systems: benefits for database system management

The user interface and some other functions can be processed by the (micro)computer of the user, when this executes the client retrieval software. This makes the workload lighter

» of the database computer
  (Which executes the data server software)

» of the data communication network
  (Which must transfer less data concerned with the user interface)
Client-server systems: problem with client programs for Windows

- The preferences, history, bookmark, mail and news files of the Windows client programs on the microcomputer can and should be adapted by every user. This is a problem when these files are stored on the hard disk on the client microcomputer, when more than 1 person uses this same microcomputer.
- Solution: install the system so that personal files can easily be stored in a personal directory on a drive in the local network.

Client-server systems: programs compatible with protocols

- The client-server protocols all have one part concerning the client and one concerning the server.
- Programs used at the client as well as at the server side can be compatible to a certain extent with such a client-server protocol.
- Compatibility is in most real cases not a full YES or NO.

<table>
<thead>
<tr>
<th>Protocol: Programs:</th>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Server</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Client-server systems

Examples

?? Question ??

Which client-server protocols do you know?
### Client-server systems: examples in the Internet

- nfs
- telnet
- ftp
- gopher
- HTTP (WWW)
- Z39.50
- E-mail SMTP
- E-mail POP
- E-mail IMAP
- News NNTP

### Question

Which client-server client *programs* do you know?
** Question **

How do most computer services or information and communication services think about the client-server approach? What are the consequences for their work and workload?

---

Client-server systems

Trends
How is the client-server approach evolving?

***-Examples

Client-server systems: trends

• From few central server computers (mainframes for instance) with dumb terminals,
  to more server computers with a diversity of “thin” to “fat” client computers.
• From Internet client software dedicated to a particular server,
  to more applications of more generic client software.
  
  *(Example: incorporation of functions/servers in the WWW accessible with 1 big multipurpose, generic WWW client program.)*
• You are free to copy, distribute, display this work under the following conditions:

  » Attribution:
  
  *You must mention the author.*

  » Noncommercial:
  
  *You may not use this work for commercial purposes.*

  » No Derivative Works:
  
  *You may not change, modify, alter, transform, or build upon this work.*

• For any reuse or distribution, you must make clear to others the license terms of this work.