Title: Distributed Applications (05.05.2014)
Date: Mon May 05 09:17:00 CEST 2014
Duration: 46:16 min
Pages: 14

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.

**Terms and definitions**
- Concepts for client-server applications
- Processing of service requests
- File service
- Time service

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

**Name service**

**LDAP - Lightweight Directory Access Protocol**

**Failure tolerant services**

---

**Architecture of distributed systems**

**Issues**
- This section focuses on the following issues
  - Discussion of basic aspects of distributed systems.
  - Transparency as a key concept of distributed systems.
  - How do distributed components cooperate? Thus, we discuss models of cooperation among components of distributed applications.
  - What is the client-server model?

**System Models**

**Transparency**

**Paradigms for distributed applications**

**Client-server model**

**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 service operations. 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).**
1. Client interface (import interface)
   - It represents the server within the client;
   - It prepares parameters and sends the request message to the server;
   - It prepares the interpretation of the result that is extracted from the answer message submitted by the server.

2. Server interface (export interface)
   - It represents all potential clients within the server;
   - It accepts client requests; interprets the parameters; prepares results;
   - It invokes the respective service operation;
   - It prepares and sends the answer message containing the result of the service operation.

### Concepts for client-server applications

<table>
<thead>
<tr>
<th>Client</th>
<th>Presentation execution</th>
<th>Presentation execution</th>
<th>Presentation execution with local database</th>
<th>Presentation execution database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server database</td>
<td>Presentation database</td>
<td>Execution database</td>
<td>Execution (with local database)</td>
<td>Database</td>
</tr>
<tr>
<td>Case 1</td>
<td>Case 2</td>
<td>Case 2</td>
<td>Case 3</td>
<td>Case 4</td>
</tr>
</tbody>
</table>

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.

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

- No parallel processing of requests, which results in the following disadvantages:
  - 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 is expensive;
Synchronization of access to shared persistent data;
Parallel processing of several applications 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.

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

Stateful server subsystems manage state information about their clients.

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

Definition: A file service [Svoboda 1984] provides (remote) centralized data storage facilities to clients distributed among a network. The server deals with bulk data storage, high performance computation, collecting/managing large amounts of data. The 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. A mechanism to detect wrong hint and seek up-to-date information.

Distinction between stateless and stateful

Stateless server
Stateful server