Distributed Systems
Principles and Paradigms

Chapter 11
(version 28th November 2001)

Maarten van Steen

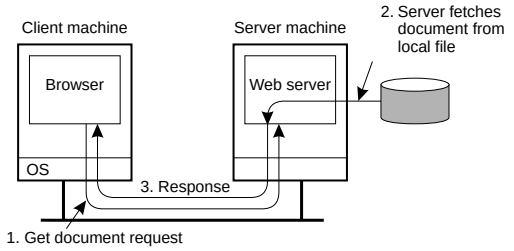
Vrije Universiteit Amsterdam, Faculty of Science
Dept. Mathematics and Computer Science
Room R4.20. Tel: (020) 444 7784
E-mail:steen@cs.vu.nl, URL: www.cs.vu.nl/∼steen/

Distributed Document-Based Systems

- World Wide Web
- Lotus Notes

WWW: Overview

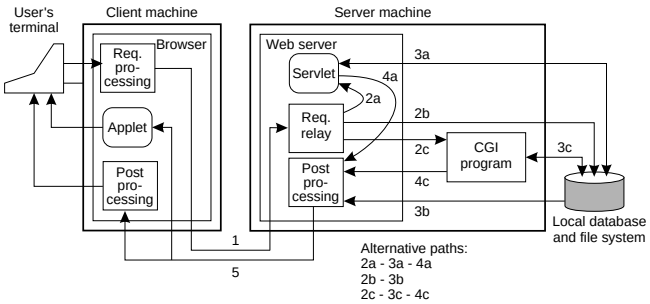
Essence: The WWW is a huge client-server system with millions of servers; each server hosting thousands of **hyperlinked** documents:



- Documents are generally represented in text (plain text, HTML, XML)
- Alternative types: images, audio, video, but also applications (PDF, PS)
- Documents may contain scripts that are executed by the client-side software

Extensions to Basic Model

Issue: Simple documents are not enough – we need a whole range of mechanisms to get information to a client



Communication (1/2)

Essence: Communication in the Web is generally based on HTTP; a relatively simple client-server transfer protocol having the following request messages:

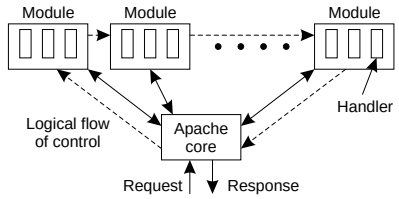
Operation	Description
Head	Request to return the header of a document
Get	Request to return a document to the client
Put	Request to store a document
Post	Provide data that are to be added to a document (collection)
Delete	Request to delete a document

Communication (2/2)

Header	C/S	Contents
Accept	C	The type of documents the client can handle
Accept-Charset	C	The character sets are acceptable for the client
Accept-Encoding	C	The document encodings the client can handle
Accept-Language	C	The natural language the client can handle
Authorization	C	A list of the client's credentials
WWW-Authenticate	S	Security challenge the client should respond to
Date	C+S	Date and time the message was sent
ETag	S	The tags associated with the returned document
Expires	S	The time for how long the response remains valid
From	C	The client's e-mail address
Host	C	The TCP address of the document's server
If-Match	C	The tags the document should have
If-None-Match	C	The tags the document should not have
If-Modified-Since	C	Tells the server to return a document only if it has been modified since the specified time
If-Unmodified-Since	C	Tells the server to return a document only if it has not been modified since the specified time
Last-Modified	S	The time the returned document was last modified
Location	S	A document reference to which the client should redirect its request
Referer	C	Refers to client's most recently requested document
Upgrade	C+S	The application protocol sender wants to switch to
Warning	C+S	Information about status of the data in the message

WWW Servers

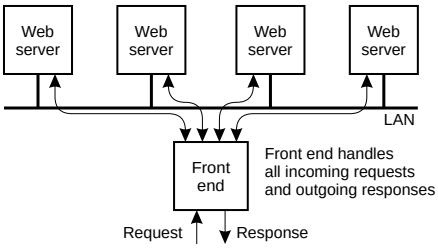
Important: The majority of Web servers is a configured **Apache server**, which breaks down each HTTP request handling into eight phases. This approach allows flexible configuration of servers.



1. Resolving document reference to local file name
2. Client authentication
3. Client access control
4. Request access control
5. MIME type determination of the response
6. General phase for handling leftovers
7. Transmission of the response
8. Logging data on the processing of the request

Server Clusters (1/2)

Essence: To improve performance and availability, WWWW servers are often clustered in a way that is transparent to clients:



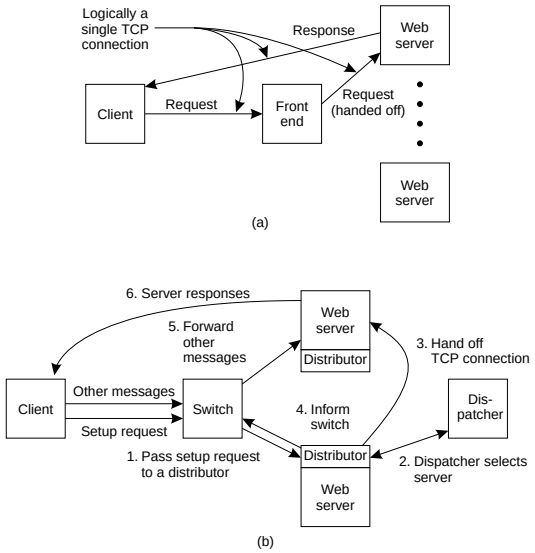
Problem: The front end may easily get overloaded, so that special measures need to be taken.

Transport-layer switching: Front end simply passes the TCP request to one of the servers, taking some performance metric into account.

Content-aware distribution: Front end reads the content of the HTTP request and then selects the best server.

Server Clusters (2/2)

Question: Why can content-aware distribution be so much better?



Naming: URL

URL: Uniform Resource Locator tells how and where to access a resource.

Scheme	Host name	Pathname
http ://	www.cs.vu.nl	/home/steen/mbox

(a)

Scheme	Host name	Port	Pathname
http ://	www.cs.vu.nl :	80	/home/steen/mbox

(b)

Scheme	Host name	Port	Pathname
http ://	130.37.24.11 :	80	/home/steen/mbox

(c)

Examples:

http	HTTP	http://www.cs.vu.nl:80/globe
ftp	FTP	ftp://ftp.cs.vu.nl/pub/minix/README
file	Local file	file:/edu/book/work/chp/11/11
data	Inline data	data:text/plain;charset=iso-8859-7, %e1%e2%e3
telnet	Remote login	telnet://flits.cs.vu.nl
tel	Telephone	tel:+31201234567
modem	Modem	modem:+31201234567;type=v32

Synchronization: WebDAV

Problem: There is a growing need for collaborative auditing of Web documents, but bare-bones HTTP can't help here. Solution: Web Distributed Authoring and Versioning.

- Supports exclusive and shared write locks, which operate on entire documents
- A lock is passed by means of a lock token; the server registers the client(s) holding the lock
- Clients modify the document locally and *post* it back to the server along with the lock token

Note: There is no specific support for crashed clients holding a lock.

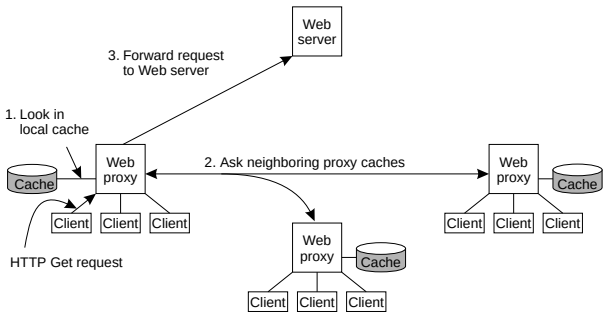
Web Proxy Caching

Basic idea: Sites install a separate **proxy server** that handles all outgoing requests. Proxies subsequently cache incoming documents. Cache-consistency protocols:

- Always verify validity by contacting server
- Age-based consistency:

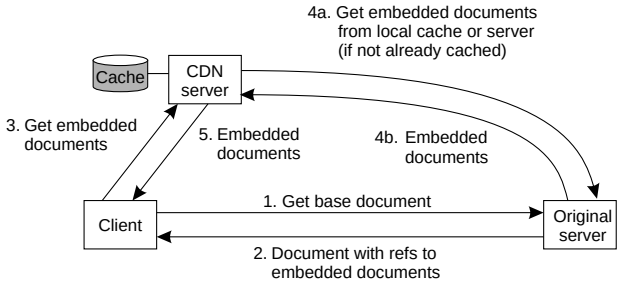
$$T_{expire} = \alpha \cdot (T_{cached} - T_{last\_modified}) + T_{cached}$$

- Cooperative caching, by which you first check your neighbors on a cache miss:



Server Replication

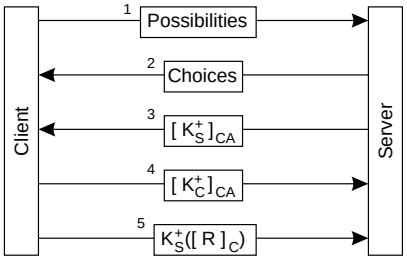
Content Delivery Network: CDNs act as Web hosting services to replicate documents across the Internet providing their customers guarantees on high availability and performance (example: Akamai).



Question: How would consistency be maintained in this system?

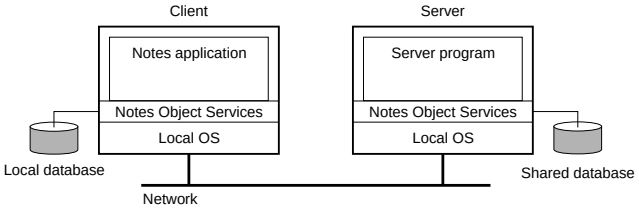
Security: TLS (SSL)

Transport Layer Security: Modern version of the the Secure Socket Layer (SSL), which "sits" between transport layer and application protocols. Relatively simple protocol that can support mutual authentication using certificates:



Lotus Notes: Overview

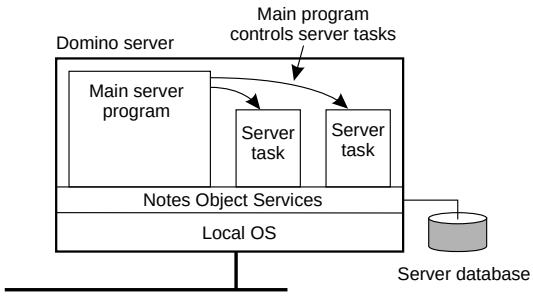
Basics: All documents take the form of **notes**, which are collected in databases. A note is essentially a list of **items**.



Note type	Category	Description
Document	Data	A user-oriented document such as a Web page
Form	Design	Structure for creating, editing, and viewing a document
Field	Design	Defines a field shared between a form and subforms
View	Design	Structure for displaying a collection of documents
ACL	Administration	Contains an access control list for the database
ReplFormula	Administration	Describes the replication of the database

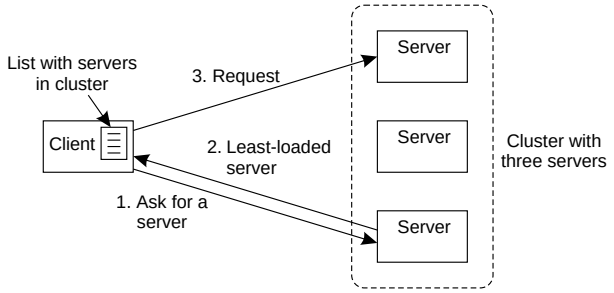
Domino Server

Essence: A straightforward server design, in which a main server controls various server tasks, spawned as separate processes running on top of NOS:



Server Clusters

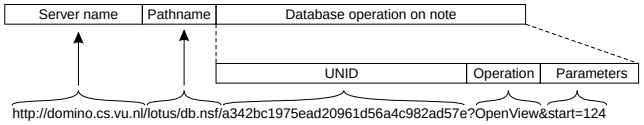
Essence: Simple approach – client contacts a known server and gets a list of servers in that cluster, along with a selection of the currently least-loaded one.



Question: What happens if the initial server is too busy or down?

Naming

Issue: Lotus is database oriented, and therefore is much tailored to support directory services (and searches) instead of plain name resolution (as in traditional naming services). There is support for URLs:



Identifiers:

Identifier	Scope	Description
Universal ID	World	Globally unique identifier assigned to each note
Originator ID	World	Identifier for a note, but includes history information
Database ID	Server	Time-dependent identifier for a database
Note ID	Database	Identifier of a note relative to a database instance
Replica ID	World	Timestamp used to identify the same copies of a database

Replication

Connection documents: Special notes describing exactly when, how, and what to replicate. Servers have replication tasks that are responsible for carrying out replication schemes:

Scheme	Description
Pull-push	A replicator task pulls updates in from a target server, and pushes its own updates to that target as well
Pull-pull	A replicator task pulls in updates from a target server, and responds to update fetch requests from that target
Push-only	A replicator task only pushes its own updates to a target server, but does not pull in any updates from the target
Pull-only	A replicator only pulls in updates from a target server, but does not push any of its own updates to that target

Note: This scheme comes very close to the epidemic protocols from Chp. 6. To remove notes, **deletion stubs** are used, similar to death certificates in epidemic protocols.

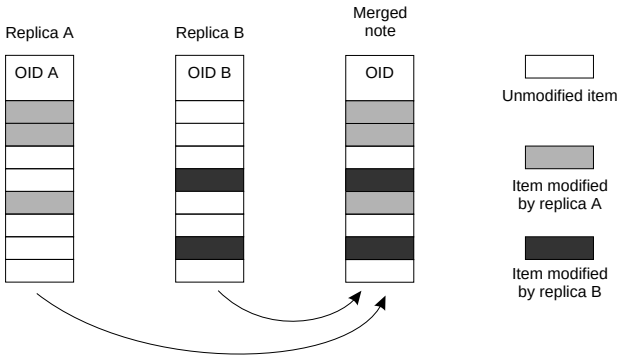
Conflict Resolution (1/2)

Problem: Notes allows concurrent modifications to replicated notes, but follows an optimistic approach (assuming that write shares do not occur often). Here's where originator IDs come in (= UNID + sequence number & timestamp).

Solution: Conflicts are detected by comparing OIDs: if they are different while their UNID is the same, we may have a potential conflict. Updates (per copy) are recorded in history lists.

- When an item is modified, the note's sequence number is incremented and credited to the item
- One list is subset of the other $\Rightarrow$ update to longest list
- Two lists the same until sequence number $k \Rightarrow$ merge copies only if modifications took place on different items.

Conflict Resolution (2/2)

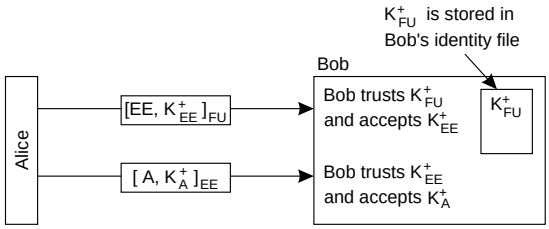


All other cases: There is a nonresolvable conflict; declare one the winner and let the users solve it.

Security

Essence: Notes uses public-key cryptography for setting secure channels. Crucial becomes the validation of public keys.

Example: Alice works in the CS department of the Franeker University (FU); Bob in the EE department. They share the public key for FU.



Finally: Having databases around, Lotus Notes has extensive access control mechanisms. See book and references for details.