Distributed Systems: Concepts and Design

Edition 3

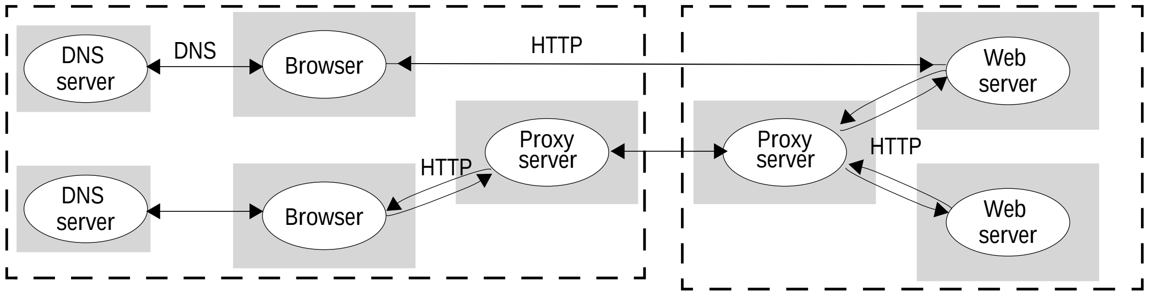
By George Coulouris, Jean Dollimore and Tim Kindberg
Addison-Wesley, ©Pearson Education 2001.

Chapter 2 Exercise Solutions

2.1 Describe and illustrate the client-server architecture of one or more major Internet applications (for example the Web, email or netnews).

2.1 Ans.

Web:



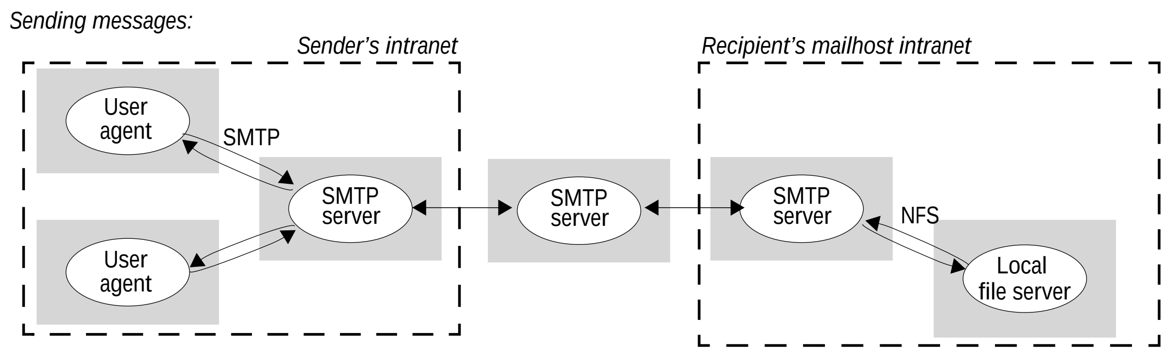
Browsers are clients of Domain Name Servers (DNS) and web servers (HTTP). Some intranets are configured to interpose a Proxy server. Proxy servers fulfil several purposes – when they are located at the same site as the client, they reduce network delays and network traffic. When they are at the same site as the server, they form a security checkpoint (see pp. 107 and 271) and they can reduce load on the server.

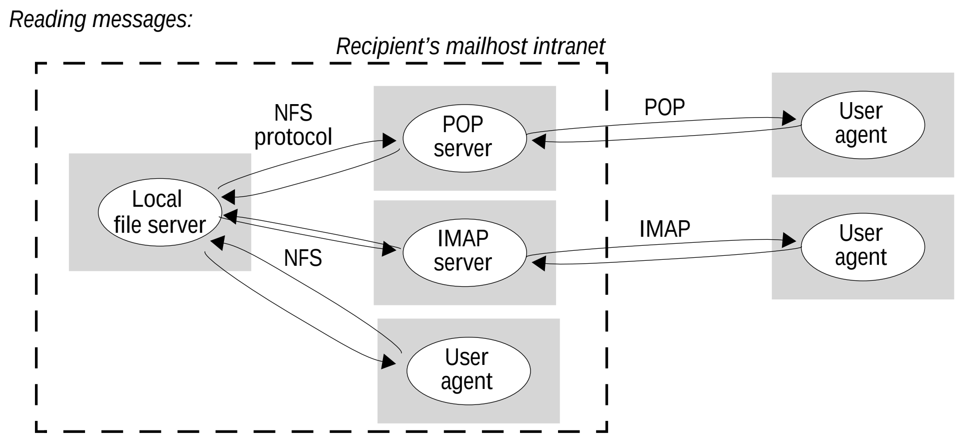
N.B. DNS servers are also involved in all of the application architectures described below, but they ore omitted from the discussion for clarity.

Email:

Sending messages: User Agent (the user's mail composing program) is a client of a local SMTP server and passes each outgoing message to the SMTP server for delivery. The local SMTP server uses mail routing tables to determine a route for each message and then forwards the message to the next SMTP server on the chosen route. Each SMTP server similarly processes and forwards each incoming message unless the domain name in the message address matches the local domain. In the latter case, it attempts to deliver the message to local recipient by storing it in a mailbox file on a local disk or file server.

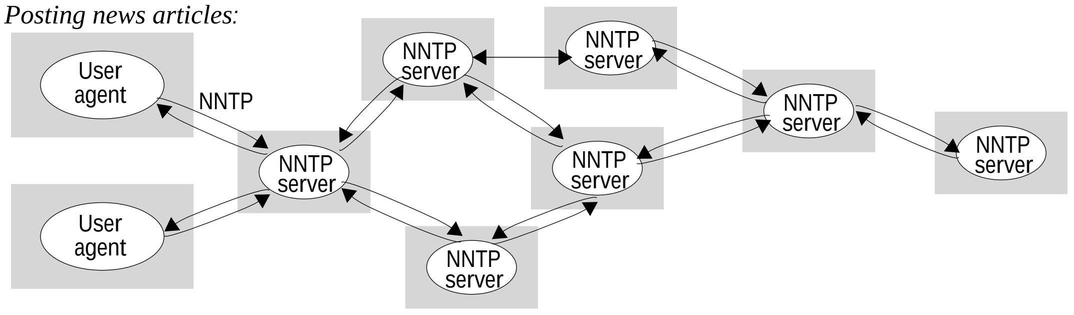
Reading messages: User Agent (the user's mail reading program) is *either* a client of the local file server or a client of a mail delivery server such as a POP or IMAP server. In the former case, the User Agent reads messages directly form the mailbox file in which they were placed during the message delivery. (Exampes of such user agents are the UNIX *mail* and *pine* commands.) In the latter case, the User Agent requests information about the contents of the user's mailbox file from a POP or IMAP server and receives messages from those servers for presentation to the user. POP and IMAP are protocols specifically designed to support mail access over wide areas and slow network connections, so a user can continue to access her home mailbox while travelling.



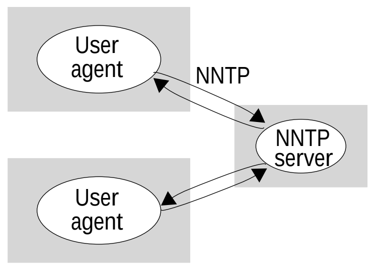


Netnews:

Posting news articles:



Browsing/reading articles:



Posting news articles: User Agent (the user's news composing program) is a client of a local NNTP server and passes each outgoing article to the NNTP server for delivery. Each article is assigned a unique identifier. Each NNTP server holds a list of other NNTP servers for which it is a newsfeed – they are registered to receive articles from it. It periodically contacts each of the registered servers, delivers any new articles to them and requests any that they have which it has not (using the articles' unique id's to determine which they are). To ensure delivery of every article to every Netnews destination, there must be a path of newsfeed connections from that reaches every NNTP server.

Browsing/reading articles: User Agent (the user's news reading program) is a client of a local NNTP server. The User Agent requests updates for all of the newsgroups to which the user subscribes and presents them to the user.

2.2 For the applications discussed in Exercise 2.1 state how the servers cooperate in providing a service.

2.2 Ans.

Web: Web servers cooperate with Proxy servers to minimize network traffic and latency. Responsibility for consistency is taken by the proxy servers - they check the modification dates of pages frequently with the originating web server.

Mail: SMTP servers do not necessarily hold mail delivery routing tables to all destinations. Instead, they simply route messages addressed to unknown destinations to another server that is likely to have the relevant tables.

Netnews: All NNTP servers cooperate in the manner described above to provide the newsfeed mechanism.

2.3 How do the applications discussed in Exercise 2.1 involve the partitioning and/or replication (or caching) of data amongst servers?

2.3 Ans.

Web: Web page masters are held in a file system at a single server. The information on the web as a whole is therefore partitioned amongst many web servers.

Replication is not a part of the web protocols, but a heavily-used web site may provide several servers with identical copies of the relevant file system using one of the well-known means for replicating slowly-changing data (Chapter 14). HTTP requests can be multiplexed amongst the identical servers using the (fairly basic) DNS load sharing mechanism described on page 169. In addition, web proxy servers support replication through the use of cached replicas of recently-used pages and browsers support replication by maintaining a local cache of recently accessed pages.

Mail: Messages are stored only at their destinations. That is, the mail service is based mainly on partitioning, although a message to multiple recipients is replicated at several destinations.

Netnews: Each group is replicated only at sites requiring it.

2.4 A search engine is a web server that responds to client requests to search in its stored indexes and (concurrently) runs several web crawler tasks to build and update the indexes. What are the requirements for synchronization between these concurrent activities?

2.4 Ans.

The crawler tasks could build partial indexes to new pages incrementally, then merge them with the active index (including deleting invalid references). This merging operation could be done on an off-line copy. Finally, the environment for processing client requests is changed to access the new index. The latter might need some concurrency control, but in principle it is just a change to one reference to the index which should be atomic.

2.5 Suggest some applications for the peer process model, distinguishing between cases when the state of all peers needs to be identical and cases that demand less consistency.

2.5 Ans.

Cooperative work (groupware) applications that provide a peer process near to each user.

Applications that need to present all users with identical state - shared whiteboard, shared view of a textual discussion

Less consistency: where a group of users are working on a shared document, but different users access different parts or perhaps one user locks part of the document and the others are shown the new version when it is ready.

Some services are effectively groups of peer processes to provide availability or fault tolerance. If they partition data then they don't need to keep consistent at all. If they replicate then they do.

2.6 List the types of local resource that are vulnerable to an attack by an untrusted program that is downloaded from a remote site and run in a local computer.

2.6 Ans.

Objects in the file system e.g. files, directories can be read/written/created/deleted using the rights of the local user who runs the program.

Network communication - the program might attempt to create sockets, connect to them, send messages etc.

Access to printers.

It may also impersonate the user in various ways, for example, sending/receiving email

---

2.7 Give some examples of applications where the use of mobile code is beneficial.

2.7 Ans.

Doing computation close to the user, as in Applets example

Enhancing browser- as described on page 70 e.g. to allow server initiated communication.

Cases where objects are sent to a process and the code is required to make them usable. (e.g. as in RMI in Chapter 5)

2.8 What factors affect the responsiveness of an application that accesses shared data managed by a server? Describe remedies that are available and discuss their usefulness.

2.8 Ans.

When the client accesses a server, it makes an invocation of an operation in a server running in a remote computer. The following can affect responsiveness:

1. server overloaded;
2. latency in exchanging request and reply messages (due to layers of OS and middleware software in client and server);
3. load on network.

The use of caching helps with all of the above problems. In particular client caching reduces all of them. Proxy server caches help with (1). Replication of the service also helps with 1. The use of lightweight communication protocols helps with (2).

2.9 Distinguish between buffering and caching.

2.9 Ans.

Buffering: a technique for storing data transmitted from a sending process to a receiving process in local memory or secondary (disk) storage until the receiving process is ready to consume it. For example, when reading data from a file or transmitting messages through a network, it is beneficial to handle it in large blocks. The blocks are held in buffer storage in the receiving process' memory space. The buffer is released when the data has been consumed by the process.

Caching: a technique for optimizing access to remote data objects by holding a copy of them in local memory or secondary (disk) storage. Accesses to parts of the remote object are translated into accesses to the corresponding parts of the local copy. Unlike buffering, the local copy may be retained as long as there is local memory available to hold it. A cache management algorithm and a release strategy are needed to manage the use of the memory allocated to the cache. (If we interpret the word 'remote' in the sense of 'further from the processor', then this definition is valid not only for client caches in distributed systems but also for disk block caches in operating systems and processor caches in cpu chips.)

2.10 Give some examples of faults in hardware and software that can/cannot be tolerated by the use of redundancy in a distributed system. To what extent does the use of redundancy in the appropriate cases make a system fault-tolerant?

2.10 Ans.

- Hardware faults - processors, disks, network connections can use redundancy e.g. run process on multiple computers, write to two disks, have two separate routes in the network available.
- Software bugs, crashes. Redundancy is no good with bugs because they will be replicated. Replicated processes help with crashes which may be due to bugs in unrelated parts of the system. Retransmitted messages help with lost messages.

Redundancy makes faults less likely to occur. e.g. if the probability of failure in a single component is p then the probability of a single independent failure in k replicas is p^k.

2.11 Consider a simple server that carries out client requests without accessing other servers. Explain why it is generally not possible to set a limit on the time taken by such a server to respond to a client request. What would need to be done to make the server able to execute requests within a bounded time? Is this a practical option?

2.11 Ans.

The rate of arrival of client requests is unpredictable.
If the server uses threads to execute the requests concurrently, it may not be able to allocate sufficient time to a particular request within any given time limit.
If the server queues the request and carries them out one at a time, they may wait in the queue for an unlimited amount of time.

To execute requests within bounded time, limit the number of clients to suit its capacity. To deal with more clients, use a server with more processors. After that, (or instead) replicate the service....

The solution may be costly and in some cases keeping the replicas consistent may take up useful processing cycles, reducing those available for executing requests.

2.12 For each of the factors that contribute to the time taken to transmit a message between two processes over a communication channel, state what measures would be needed to set a bound on its contribution to the total time. Why are these measures not provided in current general-purpose distributed systems?

2.12 Ans.

Time taken by OS communication services in the sending and receiving processes - these tasks would need to be guaranteed sufficient processor cycles.
Time taken to access network. The pair of communicating processes would need to be given guaranteed network capacity.
The time to transmit the data is a constant once the network has been accessed.

To provide the above guarantees we would need more resources and associated costs. The guarantees associated with accessing the network can for example be provided with ATM networks, but they are expensive for use as LANs.
To give guarantees for the processes is more complex. For example, for a server to guarantee to receive and send messages within a time limit would mean limiting the number of clients.

2.13 The Network Time Protocol service can be used to synchronize computer clocks. Explain why, even with this service, no guaranteed bound given for the difference between two clocks.

2.13 Ans.

Any client using the ntp service must communicate with it by means of messages passed over a communication channel. If a bound can be set on the time to transmit a message over a communication channel, then the difference between the client's clock and the value supplied by the ntp service would also be bounded. With unbounded message transmission time, clock differences are necessarily unbounded.

2.14 Consider two communication services for use in asynchronous distributed systems. In service A, messages may be lost, duplicated or delayed and checksums apply only to headers. In service B, messages may be lost. delayed or delivered too fast for the recipient to handle them, but those that are delivered arrive order and with the correct contents.

Describe the classes of failure exhibited by each service. Classify their failures according to their effect on the properties of validity and integrity. Can service B be described as a reliable communication service?

2.14 Ans.

Service A can have:

arbitrary failures:

- as checksums do not apply to message bodies, message bodies can corrupted.
- duplicated messages,

omission failures (lost messages).

Because the distributed system in which it is used is asynchronous, it cannot suffer from timing failures.
Validity - is denied by lost messages
Integrity - is denied by corrupted messages and duplicated messages.

Service B can have:

omission failures (lost messages, dropped messages).

Because the distributed system in which it is used is asynchronous, it cannot suffer from timing failures.

It passes the integrity test, but not the validity test, therefore it cannot be called reliable.

2.15 Consider a pair of processes X and Y that use the communication service B from Exercise 2.14 to communicate with one another. Suppose that X is a client and Y a server and that an invocation consists of a request message from X to Y (that carries out the request) followed by a reply message from Y to X. Describe the classes of failure that may be exhibited by an invocation.

2.15 Ans.

An invocation may suffer from the following failures:

- *crash failures*: X or Y may crash. Therefore an invocation may suffer from crash failures.
- *omission failures*: as SB suffers from omission failures the request or reply message may be lost.

2.16 Suppose that a basic disk read can sometimes read values that are different from those written. State the type of failure exhibited by a basic disk read. Suggest how this failure may be masked in order to produce a different benign form of failure. Now suggest how to mask the benign failure.

2.16 Ans.

The basic disk read exhibit arbitrary failures.

This can be masked by using a checksum on each disk block (making it unlikely that wrong values will go undetected) - when an incorrect value is detected, the read returns no value instead of a wrong value - an omission failure.

The omission failures can be masked by replicating each disk block on two independent disks. (Making omission failures unlikely).

.

2.17 Define the integrity property of reliable communication and list all the possible threats to integrity from users and from system components. What measures can be taken to ensure the integrity property in the face of each of these sources of threats

2.17 Ans.

Integrity - the message received is identical to the one sent and no messages are delivered twice.

threats from users:

- injecting spurious messages, replaying old messages, altering messages during transmission

threats from system components:

- messages may get corrupted en route
- messages may be duplicated by communication protocols that retransmit messages.

For threats from users - at the Chapter 2 stage they might just say use secure channels. If they have looked at Chapter 7 they may be able to suggest the use of authentication techniques and nonces.

For threats from system components. Checksums to detect corrupted messages - but then we get a validity problem (dropped message). Duplicated messages can be detected if sequence numbers are attached to messages.

2.18 Describe possible occurrences of each of the main types of security threat (threats to processes, threats to communication channels, denial of service) that might occur in the Internet.

2.18 Ans.

Threats to processes: without authentication of principals and servers, many threats exist. An enemy could access other user's files or mailboxes, or set up 'spoof' servers. E.g. a server could be set up to 'spoof' a bank's service and receive details of user's financial transactions.

Threats to communication channels: IP spoofing - sending requests to servers with a false source address, man-in-the-middle attacks.

Denial of service: flooding a publicly-available service with irrelevant messages.