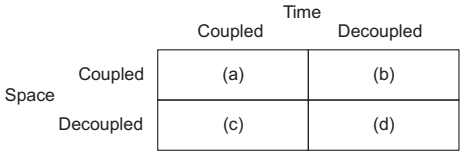
Department Computer Science	**Distributed Systems**
VU University	**19.12.2008**

1 Distributing data and processes may help to address size scalability, but may easily introduce geographical scalability. Give a well-known example to illustrate this point, as well as a solution. *5pt*

A well-known example is DNS by which the complete name space is distributed such that name resolution is also distributed. However, DNS may easily suffer from geographical scalability as many long-haul connections need to be crossed during name resolution. The solution to that problem is extensive result caching, which works because DNS data is mostly only read.

2a Processes can be decoupled in time and space. For each combination of decoupling (see figure), characterize the type of distributed system through an example. *5pt*

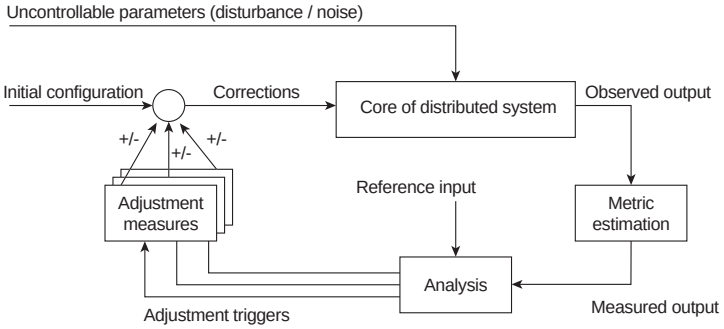


(a) Distributed systems in which processes communicate only through TCP connections. (b) Systems based on e-mail communication. (c) Group communication using multicast addressing, such as in event-based publish-subscribe systems. (d) Systems deploying shared data spaces in which subject-tagged messages are stored in a database.

2b Explain the concept of interceptors as used in middleware and why they can be useful. *5pt*

An interceptor allows an application to break in the usual flow of control between an application, middleware, and the underlying communication network. A request-level interceptor is used for modifying the request as sent between an application and the middleware; a message-level interceptor is used to modify the requests sent between middleware and transport-level communication interface. Interceptors are used to transparently modify the request-reply policy as deployed by the middleware. As such, they are useful as a mechanism to modify the behavior of the middleware to satisfy the needs of applications without having to modify those applications.

2c Explain how a feedback control loop works by providing an example of a distributed system that fits the following figure. Explain each component, as well as each connection to/from a component. *10pt*

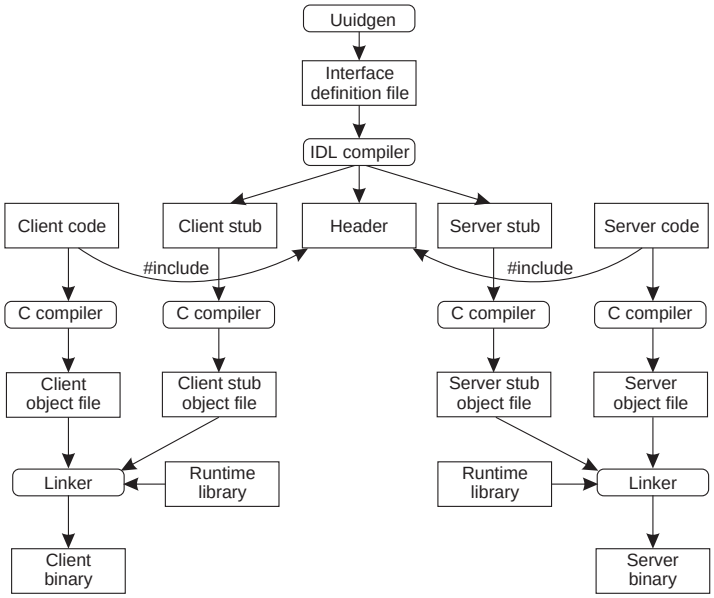


See book. A good example is a web hosting service such as Globule, which supports automatic replication of Web pages.

3a Unlike local pointers, having systemwide object references helps to improve access transparency in RPCs. How can such object references be implemented? *Hint: think of how Java realizes remote method invocations.* *5pt*

Taking Java as our example, we can simply take a complete client-side stub as an object reference. The stub contains the contact address of the remote object, but because the stub is a piece of code that can be interpreted by any Java virtual machine, it can be copied and moved between different processes. In this way, it is indeed a systemwide object reference.

3b The figure below shows how an RPC system works in practice. Explain what is in the runtime library. *5pt*

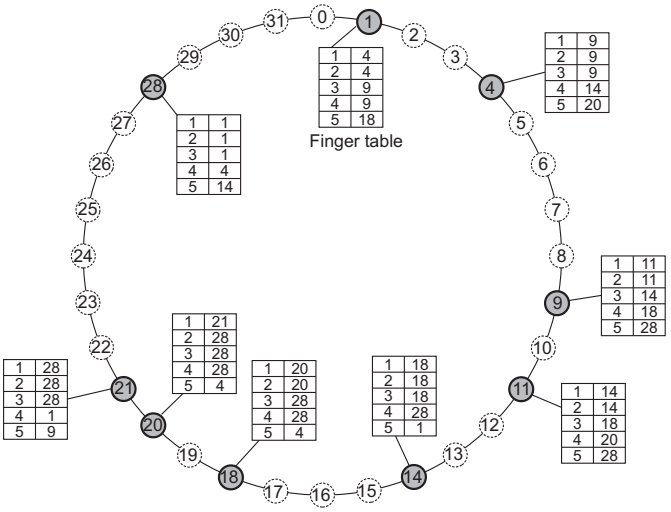


Typically parameterizable calls to the underlying transport-level interface, but notably also library routines for converting data structures to host-independent representations.

3c In RPC, a client needs to **bind** to a server. What does this mean and how can it be realized? *5pt*

Binding in this case means that the client makes all the necessary preparations to allow it to call procedures maintained by the server. It is accomplished by first having the server register, one way or the other, to which network-level and transport-level address it is accepting incoming requests. This can be done through a separate, well-known directory server. A client asks for this contact information, after which it can, for example, set up a TCP connection to the appropriate server.

4a Explain how name resolution works in Chord by resolving $k = 30$ starting from node 21 in the following example. Do the same for $k = 19$ from 21. *5pt*



(1) $21 \rightarrow 28 \rightarrow 1$. *(2)* $21 \rightarrow 9 \rightarrow 18 \rightarrow 20$.

4b In Chord, the finger table entry $FT_p[i]$ of peer p is equal to $succ(p + 2^{i-1})$. Explain how Chord's finger tables can be extended to incorporate proximity routing. *5pt*

There is no reason why p can't just keep a whole number of references to nodes in the range $[p + 2^{i-1}, p + 2^i - 1]$. *In that case, when it is required to look up a key k, it can decide to route that request to the peer with the smallest id* $\geq k$ *that it knows, but which is also closest to itself.*

4c As in any other naming system, it is possible to look up up a key in Chord recursively or iteratively. Explain the differences, as well as some advantages and disadvantages of either approach. *5pt*

With recursive lookups, a message is forwarded from peer to peer until it reaches its destination. In contrast, with an iterative lookup, the requester is returned the next peer it should ask for the key. One can argue that in the case of Chord, iterative lookups are much better: recursive lookups do not have the advantage of proximity-awareness and are also more vulnerable to security attacks. On the other hand, if the response follows the same route as the recursive lookup, replication can be done moer effectively.

4d Give two approaches to using Chord for implementing a distributed file system. *5pt*

The simplest one is by hashing a file name fn to $h(fn)$ and storing the file at the peer with the smallest id $\geq h(fn)$. Alternatively, we can also store blocks at peers instead of whole files. For example, a block with content d can be stored, together with its hash $h(d)$ at the peer with id $\geq h(d)$.

5a Explain how a blocking primary-backup protocol works, as well as its nonblocking variant. Why is a primary-backup protocol in which the primary moves to the location of the writer, never blocking? *5pt*

See Figures 7-20 and 7-21, respectively. The variant in which the primary moves to the writer is always nonblocking as there is otherwise hardly any advantage in having such a variant. With moving the primary, it becomes possible to efficiently execute a series of updates locally.

5b Sketch the design of a simple, centralized consistency protocol for active replication that realizes sequential consistency. *5pt*

Simply install a sequencer: when an operation needs to be carried out at multiple replicas, first fetch a sequence number from the sequencer and then forward the operation to the replicas.

6a One can argue that NFS is not really a file system. Explain why. *5pt*

NFS is actually a protocol and its implementation to make an existing local file system available to remote clients. This is best illustrated by Figure 11-2.

6b Mention two different measures that were designed into NFS version 4 in order to let it operate better in wide-area networks. *5pt*

(1) Moving to a stateful design by allowing clients to locally cache files an operate on them. (2) Supporting compound procedures by which multiple requests could be sent in a single message.

7a Explain how Akamai uses standard Web-caching techniques to effectively implement server-initiated replication of Web pages. *5pt*

In essence, you need to explain Figure 12-20.

7b Explain how a content-aware cache works in edge-server systems that support replication of Web applications. *5pt*

With a content-aware cache, the client's edge server assumes that queries can be classified according to a limited number of so-called templates. Each template is essentially the same as a prototype function declaration. When a query with template T is issued for the first time, the server's response is stored locally. A next time, the edge server can execute a so-called containment check to see whether the query addresses a subset of what was requested before. If so, the response can be looked up locally.

***Grading:** The final grade is calculated by accumulating the scores per question (maximum: 90 points), and adding 10 bonus points. The maximum total is therefore 100 points.*