Seminar - intro

Distributed systems - student version

Dr Leonardo Mostarda,
School of Science and Technology,
Camerino University, Italy
Version 2.0, 1 March 2013

Question 1

What is the role of middleware in a distributed system?

Solution 1

To enhance the distribution transparency that is missing in network operating systems. In other words, middleware aims at improving the single-system view that a distributed system should have.

Question 2

Explain what is meant by (distribution) transparency, and give examples of different types of transparency.

1 Solution 2

Distribution transparency is the phenomenon by which distribution aspects in a system are hidden from users and applications. Examples include access transparency, location transparency, migration transparency, relocation transparency, replication transparency, concurrency transparency, failure transparency, and persistence transparency.

Question 3

Why is it sometimes so hard to hide the occurrence and recovery from failures in a distributed system?

Solution 3

It is generally impossible to detect whether a server is actually down, or that it is simply slow in responding. Consequently, a system may have to report that a service is not available, although, in fact, the server is just slow.

Question 4

Why is it not always a good idea to aim at implementing the highest degree of transparency possible?

Solution 4

Aiming at the highest degree of transparency may lead to a considerable loss of performance that users are not willing to accept.

Question 5

Describe precisely what is meant by a scalable system. Scalability can be achieved by applying different techniques. What are these techniques?

Solution 5

A system is scalable with respect to either its number of components, geographical size, or number and size of administrative domains, if it can grow in one or more of these dimensions without an unacceptable loss of performance. Scaling can be achieved through distribution, replication, and caching.

Question 6

When a transaction is aborted, we have said that the world is restored to its previous state, as though the transaction had never happened. We lied. Give an example where resetting the world is impossible.

Solution 6

Any situation in which physical I/O has occurred cannot be reset. For example, if the process has printed some output, the ink cannot be removed from the paper. Also, in a system that controls any kind of industrial process, it is usually impossible to undo work that has been done.

Question 7

We argued that distribution transparency may not be in place for pervasive systems. This statement is not true for all types of transparencies. Give an example.

Solution 7

Think of migration transparency. In many pervasive systems, components are mobile and will need to re-establish connections when moving from one access point to another. Preferably, such handovers should be completely transparent to the user. Likewise, it can be argued that many other types of transparencies should be supported as well. However, what should not be hidden is a user is possibly accessing resources that are directly coupled to the user's current environment.

Question 8

A server program written in one language (for example C++) provides the implementation of a BLOB object that is intended to be accessed by clients that may be written in a different language (for example Java). The client and server computers may have different hardware, but all of them are attached to an internet. Describe the problems that need to be solved to make it possible for a client object to invoke a method on the server object.

Solution 8

As the computers are attached to an internet, we can assume that Internet protocols deal with differences in networks. But the computers may have different hardware - therefore we have to deal with differences of representation of data items in request and reply messages from clients to objects. A common standard will be defined for each type of data item that must be transmitted between the object and its clients.

The computers may run different operating systems, therefore we need to deal with different operations to send and receive messages or to express invocations. Thus at the Java/C++ level a common operation would be used which will be translated to the particular operation according to the operating system it runs on.

We have two different programming languages C++ and Java, they use different representations for data structures such as strings, arrays, records. A common standard will be defined for each type of data structure that must be transmitted between the object and its clients and a way of translating between that data structure and each of the languages.

We may have different implementors, e.g. one for C++ and the other for Java. They will need to agree on the common standards mentioned above and to document them.