Chapter 14. Implementation Issues for Distributed Systems

Table of Contents

Introduction to Implementation Issues for Distributed Systems

Context

This unit follows on from the previous one by considering how Distributed Systems may be implemented, and what follows from adopting certain styles of implementation.

Introduction

The case can be made that the Internet provides successful support for a Distributed System. The best example of a guise in which it appears to be a single system is perhaps the World Wide Web, although there are others, including text-based messaging and multi-user domains. The main technology on which the Internet's conversion to a Distributed System has been based is client/server interaction. Servers are an obvious way to share resources, but they also lend themselves to other system requirements such as, through replication, scalability.

Client/server technology has been widely used on the Internet, most notably with the World Wide Web. Depending on how it is implemented, it can have a decided effect on the effectiveness of the underlying network and hence of the Distributed System itself. For example, when implementing a task using client/server interaction, there are different ways to proceed. One way to explain the possibilities is to point out that the computing involved in such a task can be carried out at the client computer (client-side computing) or at the server (server-side computing) or it can be shared between them. If all the computing is done on the client side, so that the server simply acts as a repository of material (even programs) to be delivered to the client, then the network becomes what has been called 'hollow'. (With this metaphor, the network is seen as a 'container' of computing, and all the computing is done on the client machines at the 'edge' of the container.) If it is all done on the servers, the network becomes 'full'. (All the computing is done 'inside' the container.) It can be a problem if the computing is done in the 'wrong' place (on the less-powerful machines) or if the network is not being used to good effect. After all, it does not matter where the computing is done as long as the results appear, and the network is there to allow all the computers to be shared.

The upshot is that client/server methods may not always be the best way to achieve a Distributed System. One alternative is to use agent technology. A mobile agent can take computing or tasks anywhere in the network, allowing resources to be shared to good effect. The technology behind agents will be explained and its application in areas such as electronic commerce explored.

Objectives

At the end of this module, you should be able to:

- describe precisely the client-server pattern of communication;
- explore the implications of the widespread use of this pattern of communication in a networked system;
- appreciate the ideas of client-side computing and server-side computing, and to show the relation to them of Java and CGI.

Client- Server communication.

In parallel with this unit, you should read Part 4 of M Stefik, "Internet Dreams", and Chapter 1 of M Castells, "The Rise of the Network Society".

The client- server pattern of communication has been mentioned before. It is a four-stage process as follows:

1. the client sends a request to the server,
2. the server receives the request and determines a response to it,
3. the server sends the response to the client, and
4. the client receives the response and makes use of it.

Familiar situations in which this pattern of communication takes place include the following:

- On a LAN, when one computer acts as a server to hold resources that are available to the other (client) computers.
- With on-demand services, when the computer of the user making the demand is the client and the on-demand server is the server.
- Using the World Wide Web, when the user's computer running a browser is the client and a Web server is the server.

The primary reason for structuring activity in terms of interactions between clients and servers in all these situations is that it enables resource sharing.

Client-server interaction is used by the Internet in many situations other than the World Wide Web. It is used in the provision of other services and in the operation of the Internet itself. In each case, the main reason is to share resources, but it can also provide much of what is needed for the construction of Distributed Systems.

To Do

Do Review Question 1.

Carry out Activity 1.

Do Review Questions 2, 3 and 4.

Implementing a Distributed System

One way of implementing a Distributed System, then, is to make use of clients and servers. Shareable resources will be held on servers, and the client computers will be able to access and share these resources.

Now, a certain amount of computation needs to be carried out at some stage of the performance of any task carried out on a computer network. For example, a Java applet on a Web page needs to be run,

the results of a calculation made using the software held on a server need to be computed, compressed files need to be un-compressed before they can be used, and so on.

By and large, as long as the computation is performed, it makes no difference to the user where it is performed. In general, the computation can be located in different ways. At the extremes, the computation can take place either on the client computers (on the client side) or on the servers (on the server side). Some mixture of the two is also possible. But, even though the way computation is distributed makes little or no difference to the user, it can make a considerable difference to the efficiency and effectiveness of the network.

To Do

Do Review Questions 5, 6, 7, 8 and 9.

Client-side computing and 'hollowness'.

It has been said in the past that the Internet is 'hollow'. It was hollow in the sense that much of the computing took place at the 'edge' of the network (on the client machines) while comparatively little was done within the network (on the servers).

The metaphorical usage of the word 'hollow' employed here treats the Internet (a computer network) as a 'container' for computing. The Internet's servers are seen as being 'inside' the container, while the client machines at its 'edge'. The idea that the computers inside the container carry out no computing is denoted by saying that the container is 'hollow'.

The hollowness has occurred because so much Internet activity, and notably that on the World Wide Web, is based on client-server interaction. The servers 'inside' the network are used to store and communicate resources. The resources are sent to the users' computers at the 'edge', where computation takes place. Java applets on Web pages illustrate the problem in that, although they have been stored on servers, they are not usually executed until they have been sent to a client.

At present, Java seems to have become the language of choice for implementing client-side computations.

To Do

Do Review Questions 10, 11 and 12.

Carry out Activity 2.

Why is hollowness a problem?

By and large, a computation can be executed by any machine on the network. Also, the servers 'inside' the network are, in general, more powerful that the clients at the 'edge'. As long as users receive the results of their computational tasks at their client machines, it makes no difference to them where the computation actually took place. Also, it is clearly more effective to carry out the computations on the computers that are best at doing them. For these reasons, hollowness can be a sign of the ineffective and wasteful use of a network's resources.

We can note that the reason for Java's platform-independence is precisely to permit the mobility that comes from being able to run a Java program on any computer. Any network that is hollow is simply not taking advantage of what Java and similar languages have to offer.

Another relevant issue is that to take advantage of the opportunity for concurrency that is offered by a distributed system, a task must be divided into sub-tasks so that some of them can be run on different computers at the same time. But hollow networks make it impossible to capitalise on this opportunity by insisting that each task is carried out entirely on a client machine.

So, hollowness is a problem because it impedes the effective and efficient sharing of the computing resources of the network, and also because it prevents the realisation of the full potential of computer networks.

To Do

Do Review Questions 13 and 14.

How can hollowness be avoided?

Clearly, hollowness can be avoided by causing computation to take place 'inside' the network. We will distinguish two ways of doing this. One involves the use of standard methods of encouraging or enforcing server-side computing. The other involves the use of agents.

Server-side computing can be achieved by implementing client-server interaction in such a way that any computation involved takes place on a server. This can be done by ensuring that a client instigating a task sends its request, along with any necessary data, to a server possessing the program needed to accomplish the task. The server can then run its program and return the result. It can also be done by having the client instigate a task by sending a server not only its request and data but also the program necessary for the task. This is facilitated by writing programs in a platform-independent language like Java.

If clients and servers are to exchange programs and results, a set of standards is needed that describes how the client is to prepare the data that it sends to the server, so that the server will be able to decode and act on it correctly. Such a standard exists in the form of the Common Gateway Interface (CGI). CGI applications are often scripted in Perl.

Another way of ensuring that computation is carried out within the network is to enable the users' computers send agents into the network to perform tasks on their behalf and to return with the results.

Agents are often launched from, or acquired from, sites created specially for that purpose. Tasks typical of those currently performed by agents include:

- Locating the on-line store selling a specific CD at the lowest price,
- Retrieving Web documents meeting given criteria, and
- Deleting 'junk' e-mail

The use of either client-side computing or agents can ensure that a network not only becomes not 'hollow' but also that it behaves, in effect, as one large shared computer.

To Do

Do Review Questions 15, 16 and 17.

Carry out Activities 3 and 4.

What is an agent?

This section looks at what agents are in a little more detail. An agent can be defined, initially at least, as a message sent into the network to carry out some task on behalf of its despatcher. An agent may proceed by carrying a program capable of performing the task to a site that is prepared to execute it, or it may search for a site that already has the resources necessary to perform the task. Once despatched, an agent is a packet essentially indistinguishable from any other within the network. Its payload is either a program to be executed at the destination, or a description of a task to be performed.

In fact, agents should be clearly identified. One way to do this is to have a field in the packet format, the contents of which identify the packet as being of the agent type. Then, when an agent arrives at

its destination, the destination computer can decide whether to accept it. The computer may not be prepared to accept an agent which, after all, will use it for some purpose not to its owner's benefit.

A little thought will show that agents are not so different to the messages sent in achieving client-side computation. Both are messages sent into the network to perform some task. Agents, though, ought to be more 'intelligent', in the sense that they should achieve their task with some independence whereas a client-side computation will be specifically directed to an identified server.

To Do

Do Review Questions 18, 19 and 20.

From 'pull' to 'push'.

The use of agents can bring about a general change in network usage from 'pull', where the users 'pull' the information and resources they need out of the network, to 'push', where the network 'pushes' out to the users the information and resources they require. From the users' point of view, the essential difference between 'push' and 'pull' is that with the former they have to do the work, while with the latter they employ an agent to do the work on their behalf.

The metaphor used here sees the Internet as a 'container' of information and resources. The contents of the container can be 'pulled' out from the outside or 'pushed' out from the inside.

'Pull' and 'push' can be compared in these ways:

- Users need to be familiar with the system and its organisation, and experienced with its use to be able to 'pull' from it exactly what they need with proficiency. On the other hand, beginners and technophobes can have what they want 'pushed' out to them as easily as anyone else.
- Users who know what they want need to express their requirements each time they 'pull' something out, whereas with 'pushing' one statement of requirements can lead to any number of 'pushes'.
- Viewing a network and its resources as a market place of available resources, 'pulling' resources from the network corresponds to the real-world activity of going to the market to get what you want. Having resources 'pushed' to you corresponds to sending a description of your needs to the market and having what you asked for delivered to you. The latter is less sociable, but more convenient and also more open to consumer influence and so to the development of a consumer-driven market place.

To Do

Carry out Activity 5.

Activities>

Activity 1 - client -server interaction

Find examples of other Internet applications other than the World Wide Web that make use of client-server interaction. Also, find examples from within the operation of the Internet itself where client-server interaction is employed.

You can find a discussion of this activity at the end of the chapter.

Activity 2 - LANs

Find out if it is the case that on a typical LAN such as an Ethernet with PCs attached and software stored on the server, the software is actually run by the client machines. Can you find an example of a network where all the computing is performed on the server side?

You can find a discussion of this activity at the end of the chapter.

Activity 3 - junk e-mail

Contrast and compare the effectiveness of the deletion of junk e-mail when it is done as a client-side activity, as a server-side activity, and by agents.

You can find a discussion of this activity at the end of the chapter.

Activity 4 - agents

Use and experiment with a representative selection of the agents available on the Internet. Then assess their usability, utility and the extent to which they support innovation.

Activity 5 - pull-push

Explain the sense in which 'pull' is prescriptive and 'push' is descriptive, that 'push' is more economical than 'push' but more likely to lead to inertia, and that 'push' may allow users more influence than 'pull'.

You can find a discussion of this activity at the end of the chapter.

Review Questions

Review Question 1

What resources are typically being shared in each of the examples given above?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 2

Which of the properties of a Distributed System covered in chapter 13 can be provided by making use of interaction between clients and a single server?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 3

Which of the properties of a Distributed System covered in unit 13 can be provided by making use of interaction between clients and two or more replicas of the same server?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 4

The only property not mentioned in the answers to Review Questions 2 and 3 is openness. Can openness be supported in either case?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 5

Why, in general, is the location of the computing required to achieve a task a matter of indifference to the user?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 6

Suppose that the user sees an animation when a Java applet executes. What happens when the necessary computation takes place on the client side? What happens when the computation takes place on the server side?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 7

Suppose payroll software is held on a server which has a processor far more powerful than that on any client machine. After the input data has been entered at a client machine, would it be more sensible to perform the necessary computation at the client machine or the server?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 8

When compressed files are held on a server, should the computation needed to decompress a retrieved file be done on the client side or the server side?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 9

Should the computation needed to decrypt an encrypted file be done on the client side or the server side?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 10

When a network that makes extensive use of client-server computing is 'hollow' is the use of the client-server pattern of interaction entirely to blame?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 11

Does the creation of networks that are not hollow require the abandonment of client-server computing?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 12

Can you think of a word that is opposite in meaning to 'hollow' that could be used to characterise networks implemented using client-server computing with the computation done on the server side?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 13

How does network hollowness cause conflict with the efficient use of network resources?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 14

How does network hollowness cause conflict with the natural capabilities of a distributed system?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 15

What techniques can be used to ensure that a network is not 'hollow'?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 16

What are the relationships between client-side computing, server-side computing, CGI, Java and Perl?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 17

How would you launch an agent?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 18

What does it mean to say that an agent is 'autonomous'?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 19

What is the difference between the requirement on agents that carry programs into the network to be executed and that on agents that carry tasks into the network to be performed?

You can find an answer/comment for this review question at the end of the chapter.

Review Question 20

What is the essential difference between the ways that computation is instigated within the network by server-side computing and by agents?

You can find an answer/comment for this review question at the end of the chapter.

Discussion Topics

1. The widespread use of client-server communications in implementing distributed systems on a network can make the network 'hollow' if the computations are performed on the client side. If the computations are performed on the server side, the network will not be hollow. Techniques for client-side computing include the so-called 'function shipping' (see Coulouris), and the use of a Common Gateway Interface (CGI) as a standard for the exchange of the information needed in a computation between a client and a server. Are these two approaches essentially the same? Is there any essential difference between them and a primitive (unintelligent) agent that is explicitly directed to its destination?

2. What would an agent have to do to be considered intelligent? How could it be made to behave in such a way?

3. It has been said that, although 'push' technology brings convenience, its extensive use can also be dangerous in that it has the potential to 'blinker' its users. What does this mean? Is it so?

Answers and Comments

Activity 1

Much file sharing makes use of servers, and FTP can be used to transfer a file from a file server. Mail servers are used as part of the way that e-mail operates. In the operation of the Internet itself, name servers are used for the automatic conversion of names to their corresponding addresses, and terminal servers are used to permit access from so-called 'dumb terminals'.>

Activity 2

Most LANs such as that described do their computing on the client side. Networks based on X-terminal and X-windows do all their computing on the server side: the client machines used in this case are disc-less and contain little or no memory assigned for computing purposes.

Activity 3

A client-side solution allows the junk e-mail to be delivered only to be deleted at the receiving computer. A server-side solution could delete the selected items at the server before they were ever sent. An agent-based solution could delete them within the network.

Activity 5

To 'pull', you need to know how to do what you want to do, that is, you need to be in a position to prescribe your actions. With 'push' you need only to know what you want, that is, you need only be in a position to describe what you want.

With 'pull', the completion of one course of action leads to one result. With 'push', providing one description can cause the delivery of any number of results meeting that description at successive times. The ease of the latter may well lead to a reluctance to change the description. For example, by going to the market each week, one can get what one knows how to get each time. But by once sending a description of what one wants, one can have it delivered each week for ever after, and it may be tempting not to change the description.

By going to the market oneself, one can only get what is available, leaving the producer in control. When sending a description of what one wants, one takes no account of what is available. If a number of people express a demand for something that is not available, the market has an incentive to provide it. But this passes some control to the consumer.

Review Question 1

On a LAN, a server typically provides shared software and shared storage facilities. With on-demand services, the server holds the material that can be demanded, and shared. On the Web, the servers hold the Web pages that can be shared.

Review Question 2

In addition to resource sharing, the use of a single server will ensure transparency through the rapid and automatic use of the client-server interaction, and scalability, at least to the point that the capacity of the server permits.

Review Question 3

Resource sharing and transparency, as before, and improved scalability. In addition, fault tolerance can be provided because a failure on a server or on the route to a server can be covered up. Concurrency will be possible if the activities requested of a server can be partitioned.

Review Question 4

If client-server interaction is to be the basis of a Distributed System based on the network, then the existence of an openly available description of the client-server protocol will ensure openness.

Review Question 5

The user is interested in having a task completed or a result delivered. Where the computation involved in achieving this takes place is a matter of indifference.

Review Question 6

Initially, the Java applet is stored on a Web page at a server. When computation takes place on the client side, after the client has requested the page, it will be delivered and the applet on it will be executed by the client machine. When computation takes place on the server side, after the client has requested the page, the server will execute the applet and deliver not only the page but the results of executing the applet on it. This may cause the server to deliver the basic Web page for display by the browser first, and then to deliver the series of modifications to that page needed to create the animation.

Review Question 7

It is probably more sensible to have the more powerful machine do the computation. This requires the delivery of the input data to the server, which runs the program with this input, and delivers the result to the client. The alternative is to deliver the program to the client so that it can run the program and compute the result itself. The former would not only perform the computation more quickly, but would also (on the basis that the input data file is almost certainly much smaller than the program file) involve less transmission.

Review Question 8

If the purpose of the compression is to reduce the size of files for transmission, decompression should be done by the client. If the purpose of the compression is to reduce the size of the files for storage on the server, decompression could be done by the server.

Review Question 9

On the client side. If the server does the decryption, the file will be transmitted without encryption, which rather defeats the point of encrypting it in the first place.

Review Question 10

No. The problem is the use of client-server interaction coupled with the fact that computation is carried out on the client side.

Review Question 11

Not necessarily. Client-server computing with computing done on the server side would give non-hollow networks.

Review Question 12

How about 'full'?

Review Question 13

By requiring that a particular (client) computing resource is used for a task rather than allowing the most suitable or the most readily available resource on the network to be used.

Review Question 14

By, for example, preventing the possibility of concurrency.

Review Question 15

Server-side computing and agents.

Review Question 16

Client-side computing is often programmed in Java. Server-side computing is often scripted in Perl. CGI provides a standard for the transfer of data from a client to a server.

Review Question 17

Go to a Web site that allows you to customise and launch one.

Review Question 18

Essentially, that once it is launched it controls itself and determines its own actions.

Review Question 19

An agent that carries a program into the network to be executed must find a site with sufficient spare time and computing capacity to execute the program, while an agent that carries a task into the network to be performed must find a site willing and able to perform the task.

Review Question 20

With server-side computing, computation is directed to a server, while an agent must search for a suitable site.