USER-SERVER DISTRIBUTED COMPUTING

R.M. Needham

Rapporteur : J.P. Aspden

These two lectures concern a style of providing computing facilities which is beginning to receive substantial attention. It depends upon a particular class of digital communication system which is often known as the local area network. We therefore start by some discussion of the properties of local area networks and of the ways in which they contrast with more usual communication systems.

At one end of the scale of communication systems there are wide flung networks like the ARPANET which span countries or continents at modest bandwidths (say kilobits) and which have to take error control measures of a variety fairly standard in telecommunications. A high degree of reliability is achieved in systems like this by arranging for retransmission in the case of error; for rerouting in case some part of the network goes down; and by careful control of the order in which material is sent. It is also necessary to have fairly elaborate flow control facilities to avoid particular transactions clogging parts of the network. In general a lot of the technical aspects of the design of such networks are concerned with bringing their reliability up to that of the computers which are connected to them. At the other end of the scale one has communications systems which are very much faster and are, indeed, often hardly thought of as communication systems at all which connect the various components of a multiprocessor computer or of a multicomputer system, being highly specialised to the particular purpose in hand. Local area networks come in between. They typically exist in a building or on a site by contrast with the one extreme where the network exists in a country and the other where it exists in a box or at best a room. The characteristic of the local area network is that it has very high bandwidths say from 1 megabit upwards and that there is a certain amount of control over what is attached to it. Again this is intermediate between the two examples used; the wide area network has to be hospitable to almost anything and the very high performance bus will only cater for things specifically designed for it. The local area network is not designed to serve every computer that could possibly exist but it certainly has some latitude. The best known example of a local area network is the Xerox Ethernet, a piece of which is simply a contention bus made of coaxial cable which transmits packets from place to place at a rate of 3 megabits. Different pieces are connected by means of gateway computers. In order to send a packet, the sending station listens to the net until it appears to be quiet and then commences to send a packet. It is possible that more than one station will come to the conclusion that the communication medium is free at about the same time in which case the communication will be corrupted. The transceivers are able to detect this, and transmission is then aborted and retried later. The underlying technology is well known, and will not be gone into further here.

A more recent example of a local area communication system is the Cambridge Ring in which the communications medium is a ring carrying bits serially at a rate of 10 MHz. The basic unit of communication is a packet which contains 16 data bits but which consumes rather more than twice that number of bits transmitted. The Cambridge Ring works on the empty packet principle in which a station wishing to transmit waits until an empty packet appears on the ring which it may then use for the transmission required. The connection of a computer to the Cambridge Ring has three components. Firstly there is a repeater through which the ring passes; it has the primary function of regenerating signals, but it also finishes a connection to another unit known as the station which contains the logic for transmission and reception. All repeaters are identical and all stations are identical except for the station identification which is a number in the range 1 to 254. Finally, there is what is known as the access box which is the particular logic concerned with interfacing the station to the computer or other device which it serves. A station has a selection register which determines whether it is prepared to receive from all other stations, or from one particular other station, or from nowhere. The station-to-station bandwidth of the ring is about 1 megabit, although several transactions at about that rate can proceed at the same time. Again the engineering details of the Cambridge Ring are not pertinent to the present discussion and they may be found in the literature.

Since local communications systems are liable to be connected to a considerable number of computers or other devices, it is desirable that the interfaces to them should be as cheap as possible. This is the more important because they are likely to be used for connecting objects which are not of themselves extremely expensive. The general principle appears to be that one should keep the control logic simple even if this means not exploiting all of the bandwidth which may, in principle, be possible, and it will undoubtedly happen before long that local communications systems will be available in which the repeaters and stations, or whatever performs their function - we may perhaps say that part of the logic which is the same for every connected device - are available on one or two purpose designed chips. This will not of itself reduce the interfacing cost to a trivial sum because the access box still has to be made; this, by definition, is particular to the connected device. It is another aspect to the design of such communication systems that simple access boxes should usually be possible.

The particular characteristics using this type of communication system stem from the physical properties and also from some observations about reliability. By comparison with long-distance communication, the chances of corruption of individual bits of a communication may be made extremely low. Indeed, the principle cause of the failure of a communication is likely to be failure of the computers involved rather than failure of the communications mechanism. This may sound slightly odd, but one has to remember that the computers involved are under the control of people who switch them on and off or who interfere with the software running in them, and it is this sort of failure which is referred to rather than hardware malfunction. It is also characteristic of local communication systems that they tend to have plenty of capacity and can conveniently be used for a wide variety of traffic

from single characters to extensive blocks of data. The key observation is that the communication rates which they attain are comparable with the internal data rates of operating systems for shared computers. It is thus feasible to use the communication system for the kind of purpose which is traditionally associated with components of a multi-user operating system.

It accordingly becomes possible to make use of separate computers connected by a local communication system to perform a number of the functions which one traditionally associates with an operating system. That part of an operating system which is concerned with the management of line printers may be replaced by a printing server, that part which handles terminals by a terminal server, and so on. This apparently simple observation increases in importance when we realise that the area servers may perform their services in response to requests from a considerable number of clients. The same printing server may handle printing on behalf of more than one main processor. We think of printing as a function which is to be supported for the community in general rather than for any particular machine.

Various questions have to be asked about this approach. Just which of the functions of a traditional operating system can be split off in this way to good effect? What additional constraints, if any, are imposed by the requirements to provide a particular function for multiple clients? What are the effects of a possibly floating population of machines?

When one has a system involving large numbers of machines, such as the Xerox Ethernet, it is likely that the physical layout will change from time to time as machines are moved around, and this physical movement is likely to be accompanied by a corresponding change in the network address of that machine. For this reason it is extremely unwise to write programs in terms of literal addresses, and some form of late binding to addresses is needed. The time scale over which these changes happen is one of hours and days rather than milliseconds and seconds, and, given this kind of time scale, a common solution to the problem of late binding is name lookup. This can be done by providing one or more reference servers to supply information such as the current whereabouts of the file server. It is implied by this that one must have early binding to the address of the name lookup server, but early binding to just one address should be acceptable. An often voiced complaint about this method is its alleged inefficiency, but it has been shown that in local networks it is easy to implement very simple protocols, particularly when the problem is one of an enquiry nature. For instance, the Xerox System for name lookup is implemented by a single-packet-out, single-packet-in protocol, which can hardly be regarded as a sigificant overhead.

The prime consequence of providing a service for multiple machines is that the interface to it has to be specified in a uniform and neutral manner. The ability to do this is perhaps the major determiner of whether or not particular functions should be split off. Examples arise when we consider systems for processing character information. A server which ran a line printer would be a good example. The part of a general purpose operating system which manages a line printer quite commonly takes its data in some

kind of stream format which is particular to the computer in question. Although one could put the material into some supposedly neutral form such as a simple ASCII string it is not usual to do so. If we are replacing the printer by a printing server which works for everybody then there are two choices. Either everybody when addressing it must put the material to be printed into some highly standardised form, or the printing server itself must have a reasonably wide knowledge of formats which are likely to arise. In the latter case it is necessary that every document carries a sufficient indication with it as to what representation is being used for it. There is some discussion of this general question in a paper by Andrew Birrell and myself in the July issue of Operating Systems Review. It is there pointed out that although one might have supposed that the representation of text is becoming rapidly standard, in fact the reverse is the case because of the proliferation of special formats generated by text processing and layout systems.

It has been assumed in what was just said that the way in which one makes use of a printing server is to pass to it the material to be printed. This presupposes that the printing server is equipped with a local buffering or spooling system, and indeed this is a perfectly respectable way to proceed. It is, however, not the only way since another of the functions which can be exported from an operating system is the filing function. This is an action which can be achieved in a variety of ways, which will now be briefly discussed.

The generic name for a machine on a local communication system whose principle task is to store data is a file server. The justification for the existence of file servers is in part because of the essentially shared nature of much data and in part because of the economies of scale which persist in backing storage. One can have various different conceptions of what a file server is, and it is worth indicating some of these to give some idea of the range of design choices which are possible. At one extreme, it is possible to have a file server which is rather like a multiple access system which has only those commands available which have to do with filing. One can log into it, identify oneself by means of a password, and make use of file management commands. If one wishes to pass some material to a file this is very much like the input command of a multiple access system, and if one wishes to extract the contents of a file this is very like a type command. Naturally enough other commands would be available to examine one's directory and perform similar administrative operations. The natural use for a file server of this sort is in support of local filing systems in prime machines. One would proceed by deciding whether in one's local machine the desired material was available, and if it was not perform the indicated file transfers. On completion of the work a file transfer in the other direction would take place, and this will be the means by which material was made available to colleagues. At the other end of the scale one could have a file server which consisted simply of unorganised backing store being shared between various clients. For example if the file server's disc had, say, 800 cylinders one might divide this into lumps of 50 cylinders and issue these to particular client machines to use as they wished. One would be providing a strict substitute for local discs. If the client machines had sufficient memory that their backing store

transfer requirements were not high, this would not necessarily be a silly thing to do. It is however, a very limited goal since it does not facilitate any sharing or communication of information. At intermediate points there are file server designs which impose varying degrees of organisation upon the data and facilitate to varying degrees communication and cooperation between prime machines. There are a number of design questions here, particularly in connection with parallel access restrictions and interlocks, in which there is scope for a good deal of research and experimentation. It is intended to construct at Cambridge a file server in which the server itself takes the responsibility for managing the continued existence of objects and for interlocks, but where the responsibility for providing conventional directory lookup mechanisms is left with the client. This is made possible by an extension of capability ideas whereby directory managers can hold capabilities for files which will remain valid provided they have been properly registered with the file server.

The subject of capabilities in distributed systems itself justifies a little discussion. We are familiar with the notion of capabilities in single machines which are essentially tickets of permission to do something the integrity of which is protected by features of the hardware architecture of the machine. Essentially, in all known implementations, hardware features are used to enforce a distinction of type so that is impossible for an unauthorised person to manufacture a capability which will work, even if they are able to produce the relevant bit pattern. It is not so easy to use the same implementation technique in a distributive system, though it might be possible to do so by complicating the interfaces of the computers to the communication mechanism. What is possible is to ensure the integrity of capabilities to any desired degree by making valid capabilities an extremely sparse subset of the space in which they are represented. If network capabilities are made, for example, 128 bits long, then unless an unusually large number of objects exists it is extraordinarily unlikely, assuming that the bits are allocated in an intelligent manner, that any attempt to construct a capability out of wholecloth will work. Capabilities in this sense have one minor difference from the hardware protected capabilities of a single machine. In a machine such as the Cambridge CAP or the Plessey system 250 a capability which is of type segment can be utterly relied upon to give access to a segment. (This is not true, however, of proposed systems in which revocation of capabilities is provided.) Network capabilities protected by their sparseness can only be validated by attempting to use them. They have the property that a capability presented as an argument to some function cannot be an illegitimate one but may be a dud one. Capabilities which depend upon sparseness are sometimes regarded as being made by encryption; this is a valid viewpoint if the process of interpreting them involves decryption. It seems in practice more convenient to check for the existence of a capability inside, for example, a file server, without ever actually decrypting it.

Client machines have, in the foregoing, been left to the basic notions. One way of interpreting them is to consider them to be personal machines in the sense that one resides in every office. A user performs computations on his personal machine and relies upon services given around the network for backup in the shape of filing, printing and so on. The best developed instances of user server

distributed computing are of this type. It is also possible to consider the matter in a different way. Instead of having a computer in every office one has a rather considerable number of perhaps more powerful computers in some suitable basement. The communications system is used to connect terminals of a more or less conventional sort to these, and it is the terminals which exist in large numbers. Clearly it is necessary that sufficient computers, or, as they will be called, processing servers, should exist to provide an adequate service to the human users. It would be envisaged that these machines would be shared only sequentially rather than concurrently, and that the gain from having fewer of them than there are users would be that they would be more powerful. Although the cost of discs for electronics is falling rapidly, it may be assumed that more powerful machines will continue to be more expensive than less powerful ones. I believe that the architecture of such machines will present significant opportunities for improvement, since the environment in which they work is simpler than that of traditional computers. They do not have to be concurrently shared and they do not have to act as peripherals for switching centers. One obvious consequence of this is that little attention need be given to problems of preservation and restoration of machine state, which have a substantial influence on the design of ordinary shared computers. Essentially, they would be computers with only one peripheral, that being the connection to the communication system.

It was mentioned at the outset that local communication networks have point-to-point bandwidths from say, one megabit upwards. It seems likely that developments will go in the direction of having very much higher bandwidths, which should lead to certain simplifications. Provided that the communication system can be so designed so that it continues to be useful for modest chunks of traffic as well as very large ones (which may prove to be a substantial challenge in their design) then we may readily attain the point where the communication system can occupy the entire memory bandwidth of the computer it is connected to with further simplification to the architecture. This high bandwidth will be put to use partly in avoiding the use of any local peripherals, even fast ones, and partly to give effect to a general simplification of protocols by deliberately using the bandwidth inefficiently.

Discussion

Referring to a point Dr. Needham had made earlier about there being a 'cultural difference' between wide area and local area networks, Dr. Tanenbaum suggested that this was not the case, proposing that the principal difference was one of bandwidth, and the fact that wide area networks were using existing telephone equipment which was gradually being improved. Dr. Needham contested this view, adding that he did not regard cultural differences as being permanent. If one was setting up a local network and one consulted, say, an ARPA expert, one would be in danger of receiving advice which was invalid, since it would be unlikely that this expert would appreciate the differences between the two types of network. Mr. Shelness reinforced the view that a cultural difference did exist, with respect to the amount of centralised control which was kept over the network. Due to the geographical separation of users of a wide area network it is necessary to maintain a large degree of centralised control, whereas in a local network one can afford to allow users much more control over their use of the network. He quoted an example of a network where the use of some devices entailed their physical movement from one machine to another, and in this case, contention for that device was at the level of human interaction, and noticing that the device was not being used. He felt that it would be a good thing if this principle could be retained, such that contention for a device takes place at the device itself, rather than in some centralised name server.

In reply to a question on who would have the authority to change the names and addresses in the name server, Dr. Needham felt that it ought to be in someones office in order to protect it. If it was to reside on a shared machine, then this machine should be better protected than most.

Dr. Coulouris challenged the fact that binding of names to addresses would only change at a slow rate, and suggested that it should be possible dynamically to distribute processors and other services in order to use optimally the bandwidth of the network, since bandwidth is a limiting factor on the performance of the network. Dr. Needham doubted whether bandwidth was such a limiting factor, and went on to point out that most of the computers on the network which provide a service are special in some way; for instance, the file server would have several disk drives attached. This type of process can not be switched at a rapid rate, since switching would require physical movement of the disk drives. He added that there were few jobs which required only computation without needing some other physical resource. However, it was admitted that there was still a great deal of uncertainty about solutions to problems such as these, and the need for experimentation in this area was stressed.

Mr. Shelness said that he had had some experience of a file server, which was originally built to provide service at one of the extremes which had been outlined, namely as a support for other file systems. It was now being used for all three functions, as support, as a file system in its own right, and for memory swapping, and this suggested that it was impossible in practice to restrict the use of such a server to just one of these three uses.

Professor Whitfield, referring to an earlier point about accounting in a shared file server that client machines should handle their own accounting, felt that in the wider context of a more public ring it would be difficult to do the accounting anywhere but in the file server itself. Dr. Needham explained that what he would like to see in the situation of a file server being used by a collection of file systems would entail the file server only performing accounting between the different systems, and allowing each individual system to manage its own resources in its own way. He qualified this by suggesting that it was not obvious how this should be done, since general naming networks, while having flexible naming systems, present many problems in accounting. This is one of two known doubts why this type of file server may prove to be of no use in practice. The other is that the way in which physical management of the disk is done may not suit all clients. In the context of the Cambridge ring this is not a problem, since however cleverly disk accesses are arranged to obtain rapid transfer rates, the network will only distribute the information at a rate of 1 megabit, although in a higher bandwidth network this could prove to be a serious problem.