DISTRIBUTED SYSTEMS

First people bought lots of microcomputers. Then, because very few people work in total isolation, they began to connect the microcomputers together. There was nothing specially new about that – people had been connecting larger machines together for a long time, and the resulting networks were very useful for exchanging information, electronic mail, archiving, and other such purposes. Generally, though, if you started work on one computer, any others accessible through a network would all look like additional devices. (The first networks almost always assumed that the object at the other end of a communications line was a terminal – sometimes a very fast terminal – because they already had software to handle terminals, and a computer is usually versatile enough to make itself look like anything you like provided that your view is along a piece of wire.)

But once you've started this sort of thing, it grows. If you only want to refer to a file on a distant machine, it's silly to have to copy it all. It might even be impossible, if the file is a large database to which you wish to refer on your small machine. And copying files about is also undesirable from the security point of view. So why not devise ways to use a remote file from your local computer ?

Well, now we can use files from other computers. Here's this file on another computer which I want to use – but it happens to be a code file. Why can't I execute it just as I can a local executable file ?

But it's silly having to bring the code all the way to my processor to execute – why can't I execute it on the remote machine ?

And there's this useful piece of software on yet another machine which I'd like to build into my programme. Do I really have to copy it to my machine ? If I do, it will get out of date – and, anyway, it's much better to leave it there where it can use all the data it need conveniently. (Here's the client-server model again.)

Look, why do I have to bother about where all these bits of code and data are ? That's a purely mechanical job – let the computers handle it.

And that, as they say, is how distributed systems are born. They are much more than the original networks from which they grew, and a fully integrated distributed system does act as a single computer system. To make this work, you need an operating system which in some sense encompasses the whole hardware system, processors, network, and all. Such systems are still being developed, and it's probably realistic to say that no one really knows how best to do it. Nevertheless, you can get distributed systems now which will do a fair job of running a lot of microcomputers as a single system, and there's no doubt that distributed computing will be a common way of working in the rather near future.