DOING THE WORK

Now we have a (fairly) clear idea of what we want to do with the information in the system and, in general terms, how we expect to want to do it, we have just about half completed our investigation into the system requirements. The other half is just as important, though, for simply being able to manage data standing still and moving about doesn't achieve the prime goal of computing which we have been pursuing throughout our studies – to produce results as instructed. The data are indispensable, but until we can change them into results we haven't completed the job. We must therefore direct our attention to this question of changing data into results.

> *Our estimate that we have completed half the task was not merely a wild guess or rhetorical device. It turns out that there is a significant measure of dualism between data and processing; in many cases, useful data structures have counterparts in ways of organising the processing. While we don't claim any exact correspondence, there is enough resemblance to suggest that the topics might well be about the same size.*

There is one significant difficulty in coming to grips with this notion of activities carried on within computers. In everyday life, we are reasonably well acquainted with data and how they are managed; we all use timetables, telephone directories, letters, books, and many other forms of collected information. The information isn't just localised; there are tables of contents and indices which give us ways of finding items without reading from the beginning every time, we use references in the text to point to other parts of a document, or to other documents, and many other such devices. (Now we call them hyperlinks and some people pretend have only just been invented.) We accept all this as a matter of course, and we have a good practical idea of how it all works.

In contrast, we have nothing like so clear an idea of how *activities* can be organised, how they fit together, how they interact, and so on, particularly at fine levels of detail. We speculate that this is because, until very recently, we have never had to worry about such things; if we wanted to carry out some complicated job, we would guess how it should be done, then delegate each part of it to someone skilled in the required sort of work. Because there were so many people involved, every part of the task was controlled by a highly intelligent, adaptable, and resourceful processor which could communicate fluently with the other processors, so if our original guess wasn't quite right, or if it didn't cater for some unforeseen circumstances, a solution would be found by natural cooperative processes.

It is only recently that we have begun to entrust comparatively complex sequences of operations to machines which are not equipped with such intelligence and resource, and have therefore had to face the question of describing the required activities completely and precisely down to the last detail. It is even more recently that computers have made it possible to carry out mechanically activities at the level of complexity we find in operating systems, and the skills required to organise such systems do not come naturally. This is seen even in elementary computer programming classes, where many people have great difficulty in analysing even very simple tasks sufficiently well to encode them as computer programmes.

In an operating system, we must address the problem of carrying out many complex operations in ways which (to satisfy our requirement for a functional system) do not interfere with each other, and with the additional complication that we do not know beforehand just what these activities will be. Effectively to deal with such questions, we need a clear idea of what we mean when we talk about activities in a computer system, and that is what we shall try to develop in the rest of this section.

WHAT IS WORK ?

We implied above that work could be regarded as "changing data into results"; in fact, that's rather too specific for our purposes, because it doesn't cover everything that it should. Work does not always require data and results, at least in the sense of information recorded in files. Work might be behaviour. Consider a computer set up as a teaching machine; there is nothing directly corresponding to the traditional idea of "results". In this case, the "result" is the way in which the machine interacts with the pupil. There are no data in the traditional sense of the word, where it implies the special information you want to give to the computer for this run as opposed to the programme which is always the same; instead, the special information comes through the interface from the pupil, and the programme must deal with it as it arrives. (If you are not enthusiastic about teaching machines, think about games.)

We shall therefore define work, for present purposes, as *following instructions,* which are expressed in some way and must be represented in the computer in some form which it can use. We shall for the time being regard it as someone else's job to make sure that the instructions which are followed do contribute to the eventual aim of doing work as instructed. It is not unreasonable to expect that the instructions will commonly come in organised collections of some sort, and we shall call these collections *programmes.* There might also be data and results, which, though their functions are different, must also have representations of some sort in the machine; we have already discussed how the system must manage information, and to do that trick we invented *files* (for information that's standing still) and *streams* (for information in motion).

To talk about the behaviour of computers, we follow an analogous path, and invent *processes*. We have to be careful how we define processes, because there are many things which they are not. For example, they are not programmes; they are not results; they are not computers in action. A process is an abstraction of the notion of "doing work as instructed" which we have emphasised throughout. It requires instructions; it requires some agent capable of doing the work; it might require certain information, and it should produce work done, which might be embodied in results – but it is none of these. It is an activity – the performance of the instructions by the agent.