designers concentrate first on making the system fit the people, rather than assuming that the user will adapt to the system.

In recent years, ergonomists have been studying the way that people react to complex software — the so-called 'user interface'. Through most of the history of computing, the users have been skilled, highly motivated professionals prepared to accept discomfort and inconvenience and willing to attain a level of technical competence as the price of computer power. Today's user, however, is literally the ordinary person with a very limited tolerance of temperamental and demanding machines; if he had to learn a database command language to use a bank cash dispenser, he would be likely to take his business to the building societies instead! And it is not just casual users who have interface problems. Database systems that put megabytes of management information on every office desk are everywhere underused and misused, because extracting the information from them requires fluency in complex languages like SQL. Researchers and users alike hope that the next generation of natural language 'front end' systems will make it possible to communicate with the database or the financial model in plain English. In this type of system, the front end will translate the user's input into the system's command language, possibly prompting the user to specify or amplify his requests, and explaining or expanding the system's responses.

There are many ways to help the user, including the provision of 'help' facilities (see page 526). Artificial intelligence research has led to the development of knowledge-based programs that can explain how they arrive at decisions, and it is thought that these techniques could help to create 'adviser modules' to help and prompt program users. This type of 'intelligent' software raises a philosophical question, however: what is a good explanation, and to whom? The programmer might want an explanation of the data structures and the processes involved in a system, while the business user would prefer a discussion of means to achieve a commercial end. Such distinctions are semantic and linguistic problems for the ergonomists and the computer scientists.

A similarly grey area in the study of the user interface is known to psychologists as 'modelling'. People use mental models (such as national or racial stereotypes) as a way of filling in gaps in their knowledge; most of us would confidently predict a stranger's appearance, views, personality and politics simply from knowing his job — civil servant, say, or conductor (bus or orchestra). Similarly, people approach computers with stereotyped ideas about the machines' power and behaviour, and may see them as more intelligent and knowledgeable than humans performing the same sort of task. When they meet the kind of curt, impersonal responses that many software packages make (especially to incorrect inputs), they may consider the computers rude and unfriendly, even hostile — judgements that are, of course, totally inappropriate to computers. This personalised perception of the machine affects the whole interaction, often leading to inappropriate responses on both sides of the console.

Programmers attempting to instil user-friendliness can exacerbate the problem by introducing features that make the program seem more intelligent than it really is. For example, using cheery prompts like HELLO JOHN, I AM THE SPIRIT OF THE DATABASE may encourage the user to reply in kind, often with catastrophic results and consequent discouragement.

True user-friendliness needs a rather more skilful approach. It means thinking and caring about people, and allowing for their complicated reactions to computers and work; there is more to ergonomic design than tilting the screen and painting the disk drives lilac!

Working Conditions
It is generally acknowledged that people at work are influenced in their performance by gross physical factors such as worktop heights, air temperature and noise levels. However, the subtler effects of colour, light and perception of space are now being recognised as equally important aspects of the work system, especially where computers are used. A systems analyst installing a new system must consider all these factors, in addition to choosing the hardware and software

Flexible FORTH

People can be frustrated and constrained by received ideas just as surely as by the design of their physical environment. FORTH, the language seen by many as a replacement for BASIC, was invented by astronomer Charles Moore at Kitts Peak Observatory in Arizona. Moore was working on controlling telescope movements using FORTRAN programs, but found that language too biased towards pure processing and unsuitable for external control applications. Writing a new language was his solution to the problem. FORTH differs from rigidly structured languages like FORTRAN in allowing the user to create virtually a new dialect of the language to suit each new programming task. The price of FORTH's flexibility is its starkly unfriendly approach — an efficient system isn't necessarily a comfortable one to work with