INTERACTIVE SYSTEMS

The batch operating systems developed from the preceding monitor systems over a period of many years, but the aim of the development was always the same : to get the maximum possible work out of the processor. In consequence, people using these systems noticed very little change throughout this development. The machines got bigger and faster, the job control language got rather more complicated, and either the turnaround time for a job decreased quite a bit or there seemed to be a lot more people around in the computer centre. But you still expected to prepare a set of punched cards describing your job, relinquish it to the tender mercies of the computer centre staff, and come back some time later to collect the results, should there be any.

When the interactive systems came along, some things changed, but the changes were not so much concerned with what was done but with how it was done. People still did much the same sort of work as they had with the batch systems (good cheap graphics screens were still in the future), but both the way people executed the work and the way the processor handled it had to change.

WHAT'S NEW ABOUT INTERACTIVE SYSTEMS.

This turns out to be much more than a simple switch of input and output devices; the whole emphasis of the system has to change to accommodate a radically new way of working. Some of the differences are :

- At any moment, there are now many independent programmes running. The batch systems used multiprogramming with perhaps 10 to 20 programmes in execution at any moment; to satisfy the same number of customers, it was necessary to run perhaps 100 simultaneous interactive sessions.

- None of the sessions is predictable : anyone can decide to do anything at any moment – so there is no longer any possibility of anything like job queues. Without this predictability, the system has no chance of selecting work which maximises use of all its parts, so the interactive systems are bound to be less efficient than the batch systems.

- There is no control over demand for resources – so if 35 people suddenly decide, more or less simultaneously, to read a file from a tape, you will need 35 tape drives. (Or someone will have to wait – see below.) While that's a fairly unlikely event, there will certainly be considerable fluctuations in demand for all the system's resources – memory, disc, printers, tape drives … – and, once again, there is no way to even out the bumps by judicious selection of jobs to execute.

- There is much more scope for misbehaviour. Someone trying to read other people's files on a batch system had no way of telling whether this activity had been detected, and had to turn up, or at least provide some form of identification, to collect the results. With an interactive system, the criminal has a much better chance of detecting the forces of justice at work, and can retrieve any results without risk of exposure.

- Everyone wants service NOW. Studies suggest that "NOW" means within two or three seconds for simple routine operations, with rather lower expectations for jobs which are perceived to be significantly bigger.

To cater for these changes in work pattern, a system cannot avoid inefficiency. If peaks of demand are to be handled within acceptable times, then enough resources of all sorts must be available. Much of this capacity will be unused for most of the time.

A CHANGE IN ATTITUDE.

Why was this change acceptable ? Partly because it was obscured by the continuing reductions in the cost of hardware; but mainly because the hardware had become so cheap that the software and its development became much more prominent in the total cost of a computer system, and people began to worry more about *efficient use of their staff* than about efficient use of the machinery.

An interesting sidelight on this new view is cast by a phenomenon which appeared in the early days of mainframe interactive systems. A number of software packages were advertised largely on their claims that they would protect you from the operating system – meaning that you wouldn't have to worry about the complicated operating system instruction set any more, as the package just used simple instructions and did all the translation for you.

QUESTIONS.

Who actually benefits from an interactive operating system ? (Remember we're NOT talking about a lot of people using, say, banking transaction terminals : the people at the terminals are controlling the operating system directly.) Who loses ?

How is a batch system different from an interactive system in which we pretend that each terminal is a card reader ?

What's "efficiency" ?
