Designing Systems from the Inside

Mike Bostock

Mike Bostock, who was co-ordinator of the project which gave rise to the computer control program Logicator, describes why it was developed and how it can contribute to systems work within technology

Logicator was an idea that came to light nearly ten years ago. In those days, before the National Curriculum, there were pockets of educationists who recognised that computers had much to contribute to the subject of technology in schools and who worked together to develop ways to make it possible to bring more IT practices into technology departments. It was recognised that the sort of technology that pupils would increasingly be meeting all around them — at the cashpoint, point-of-sale systems, with video recorders and word processors — was a different technology to that which they met in a department called 'CDT'. In there, technology was largely making things out of wood, holding a saw straight and getting a good finish — in essence, craftsmanship. This was important, but it was perceived that designing and making needed to be taught in the context of evolving technologies in the real world.

What these enthusiasts recognised was the importance of *systems* as a subject of study and something worthy of designing and making. This was always a problematic topic because systems were something that had to be programmed, and this was the blackest of arts, only understood by the strangest of people (usually bearded) who called themselves teachers of Computer Studies. Yet the goal was irresistible because systems were becoming more significant than artefacts and the essentially craft-based activities associated with their creation. Designing and making systems would offer richness and challenge to the brightest student, providing scope for activities that could shape tomorrow's engineers from the moment they realised their intellectual and practical skills could unite in the achievement of making a system. Even if it were just a programmed buggy, one could sense this to be the beginning of something relevant and something distinctly important to the nation.

What is a system?

A system is an artefact that has a function, often electro-mechanical, sometimes purely electronic. It can carry out tasks by referring to a set of rules, or conditional sequences, that are held inside its memory. To make a system one has to build two things: the physical device and the program that controls its function. They are both as products, although conventionally the physical product has been more highly valued as a product over the computer program. The reason for this is that the computer program is usually unintelligible. One can only deduce its effectiveness by watching the system work. What was required was a way to make programs that could be seen and understood by humans as well as by machines.

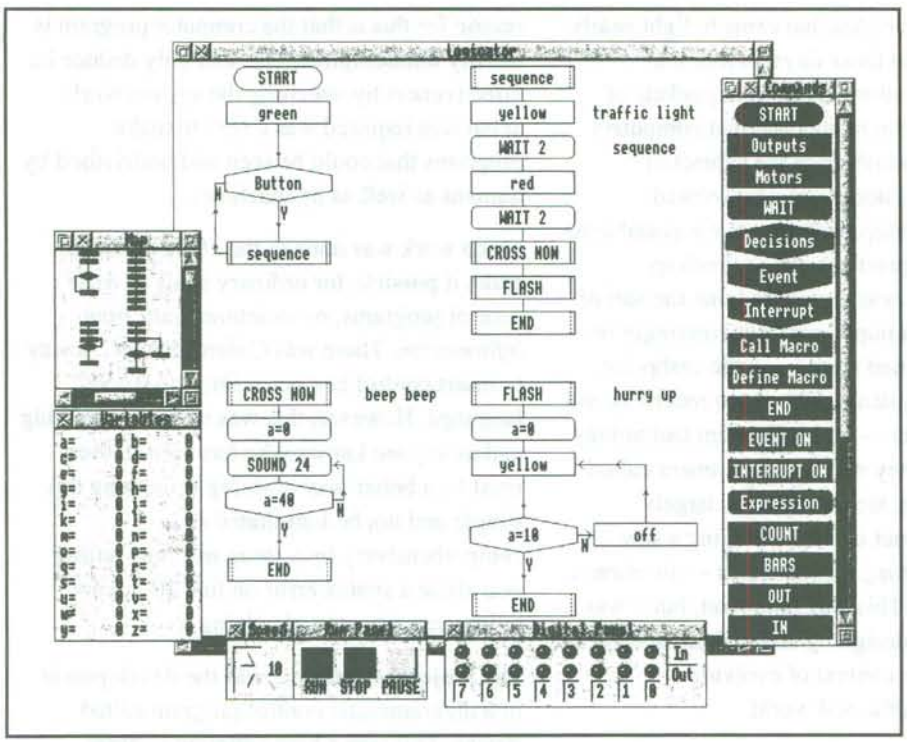
Much work was done in the 1980s to try to make it possible for ordinary souls to write control programs, or *structures built from information.* There was Control BASIC, a way to insert control keywords into the BASIC language. However, this was still programming and as anyone knows who has tried it, there must be a better way of doing something this simple and not be humiliated so comprehensively by a smug machine telling you about a syntax error on line 20 but not telling you any way of solving it.

The project of that time was the development of a diagrammatic control program called Bricks. This was a brave attempt to draw blocks on the screen into which there were short series of keywords that turned on motors and detected the state of sensors. Building the control program involved stacking the bricks together. This approach was hampered by the lack of memory on the BBC micro but otherwise pointed the way forward.

With windowing systems and increased memory in modern machines, it has become possible to rethink the ideas of visual programming languages as applied to developing control programs or *system algorithms.* The idea was born of a friendly program that would let you draw and test logical structures without having to write low-level code. The medium of the flowchart had to be the right one to use because people who are not programmers use flowcharts to explain an operation that is dynamic or conditional. For if one is going to get involved with creating instructions that are not predictable but depend on conditions that constantly vary, then the flowchart is one of the best ways to represent it, and to explain it to someone else.

Where Logicator came in

Logicator is both a drawing program and a programming medium. Its drawing functions, including 'drag and drop' and 'block move' are easily recognised by anyone who has used a simple drawing program. But underneath the surface of the drawing the program creates fast, computer-readable code that can control



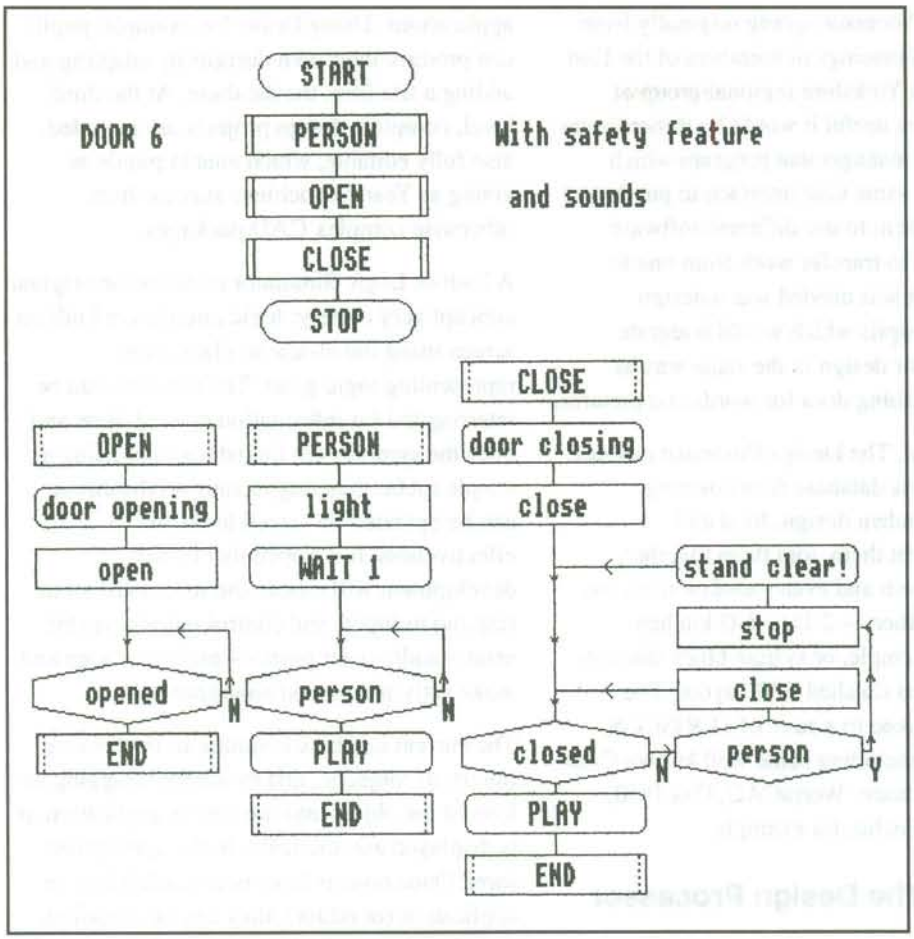
Flowchart of pelican crossing

motors, lights and sensors and operate the most demanding system, or model of a system. The flowchart can then be run as if it were a computer program by pressing a Start button. Program flow is shown by following a highlighter around the flowchart. Logical problems can be deduced from observing program flow and easily corrected by redrawing the offending section. Because the algorithm is a physical drawing rather than a list of commands whose function may not be obvious, it is now possible to get both teacher and learner involved in discussing the design efficiency of a particular flowchart.

Well, that's the theory. The acid test of this technique is 'does it encourage progression in systems work?'. This is an important question because currently many would feel that there are few examples of progression to be seen in schools within the medium of computer control. The best examples of computer control work in primary schools produce some nice models and about six Logo-style control procedures. The best examples in secondary schools produce . . .well, it would be quite hard to find more advanced examples than this.

The reason for this has been referred to in a previous article, but in essence the nub of the problem is that control programs like BASIC tend to be one-dimensional, like a list. Decision-making control programs that use several or maybe multiple, sensors can be more efficiently designed and communicated in the two-dimensional world of the flowchart because one can design the algorithm sideways as well as up and down.

Mike Pooley leads a team of staff who have developed a control technology room in the technology department at The Hedley Walter School in Brentwood, Essex. The department uses Logicator running with Acorn RISC PCs in conjunction with Smart Box interfaces and system models that include an automatic door,



Above: Flowsheet for an automatic sliding door: the door is activated when a light beam is broken and includes a safety feature on closing; it can also play sampled sounds such as 'Stand clear!'

a sensing buggy and a card reader. The system has been chosen because the school is committed to up-to-date practices and techniques that are close to those in industry and manufacturing. This has involved a heavy investment in IT and control equipment but enables the department to teach designing and making using a range of materials and to place IT capability within a relevant context.

Mike now uses Logicator across all the year groups, starting with Year 7: 'By Key Stage 4 we are hoping that students will be able to design and build systems from different materials and link the use of Logicator with other work in technology such as electronics'. He also hopes that as pupils develop capability in all areas of the subject they will be able to design and build programmed systems as part of the complete design cycle. In effect pupils are designing systems from the inside: by putting the functionality of the system first and treating it as a design exercise where the medium is the flowchart. Logicator is an idea that has taken some time to arrive but now looks set to make an important contribution to promoting progressive systems work in technology.

Opposite and right: pupils from Hedley Walter School tackling control