Picture Programs

Programming a computer may eventually be as simple as sketching a diagram or drawing a flowchart

By IVARS PETERSON

A peek into a typical computer program generally reveals little more than a seemingly random array of words and symbols. A closer look shows patterns and perhaps some meaningful order in the way these words are listed. But one of the best ways to get a sense of a program's function is to look at a flowchart that diagrams the way in which the program does its job.

A visual trip along a flowchart's lines takes a user from step to step, from choice to choice, until a particular task is completed. Translated into a conventional computer language like FORTRAN, this picture is literally worth a thousand words or more. But if creating a flowchart were to become the *only* step in writing a computer program, it would eliminate the tedium of translating a complex concept into myriad lines of code.

A few computer scientists are starting to take this possibility seriously. Not only would a "visual language" make computer programs simpler to write, but it would also make them easier to understand. A quick glance would be enough to show a user what's going on. In essence, with a visual programming language, what you want is what you see is what you get.

A panel discussion at the recent National Computer Conference in Chicago highlighted some of the ways in which researchers are beginning to develop visual languages. It also focused on the problems that must be overcome before such languages can become widely used.

The growing availability of computer terminals or personal work stations that handle graphic symbols is one of the forces driving interest in visual programming. Already, users can design "objects," such as business forms or mechanical devices, by assembling simple graphic "building blocks" supplied by the computer. The finished picture, made up of these pieces, as a whole represents one type of visual program.

This kind of scheme works very well when the object being designed has an obvious, direct representation. Laying out an invoice, for instance, simply means putting the appropriate components in the right places on a picture of an invoice. The problem is much more difficult when visual programming is used to represent something abstract—a time sequence, related bits of knowledge, conditional statements. It becomes necessary to devise visual metaphors for picturing these ideas.

"Inventing suitable visual representations is the key problem," says Robert J.K. Jacob of the Naval Research Laboratory in Washington, D.C. "It's hard to think of good pictures to use." And, he adds, the choice of representation makes a big difference in how well the language suits a given application.

Jacob is now experimenting with "state transition diagrams," a pencil-and-paper tool widely used by computer scientists to describe algorithms or computational procedures. He sees an extension of these diagrams as a potential visual programming language.

One of the main virtues of state diagram notation, says Jacob, is that it shows precisely what the user can do at each point in a dialogue between the user and a computer and what its effect will be. Moreover, a computer, using state diagrams as part of a system for providing help to users, can even answer user questions such as: "What can I do next?" "Where am I?" and "How can I do...?"

In general, state diagrams, usually drawn as branching chains of circles linked by arrows, tell users what happens for all possible inputs. The diagrams also clearly indicate what a user can do to switch from one "state" to another in which the results for a certain input may be different.

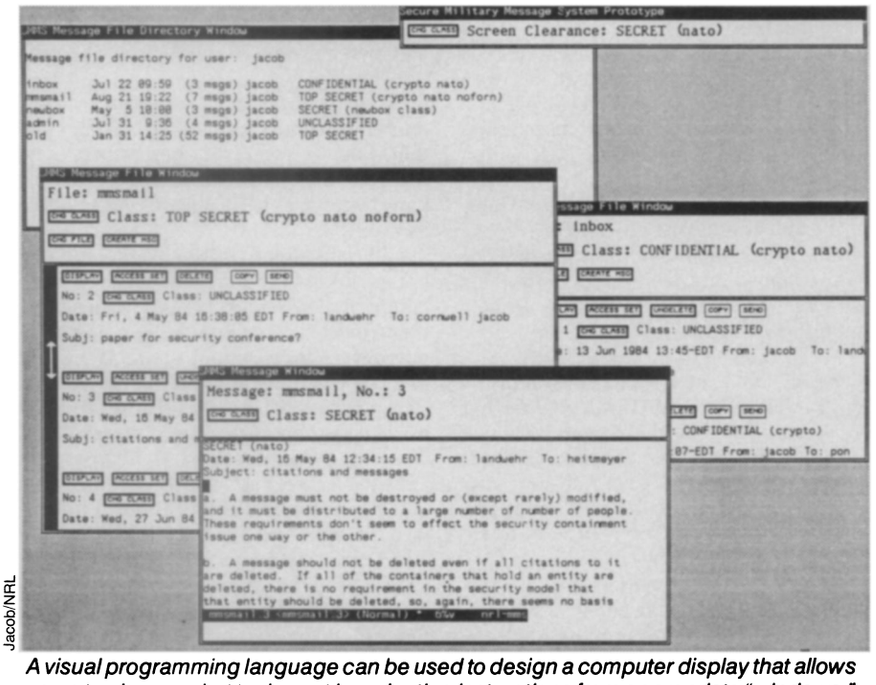
When this notation is used, writing a computer program turns into drawing the appropriate state diagrams, which the computer understands and implements. Editing a diagram automatically alters the program.

Jacob has used a primitive version of this idea to design and specify how a user interacts with a computer in several prototype systems. It forms the basis for a military message system, for example, in which the user can manipulate information displayed in several "windows" on a video screen. Details of his scheme appear in the August IEEE COMPUTER, a special issue devoted to visual programming.

"Visual languages are difficult to build, implement and write," says Jacob. His own system may be as much as five years away from general use.

Moshe M. Zloof of M.M. Zloof, Inc., in Dobbs Ferry, N.Y., emphasizes the usefulness of being able to see a computer program as a whole. "A computer programmer builds a mental image of a program as he or she reads it," he says. People who don't know how to program a computer have much more trouble building such an image. However, a program in the form of pictures lets them grasp much bigger chunks at a time.

This is the principle behind Zloof's "Office-by-example" software developed for IBM. It can be used, for instance, to put

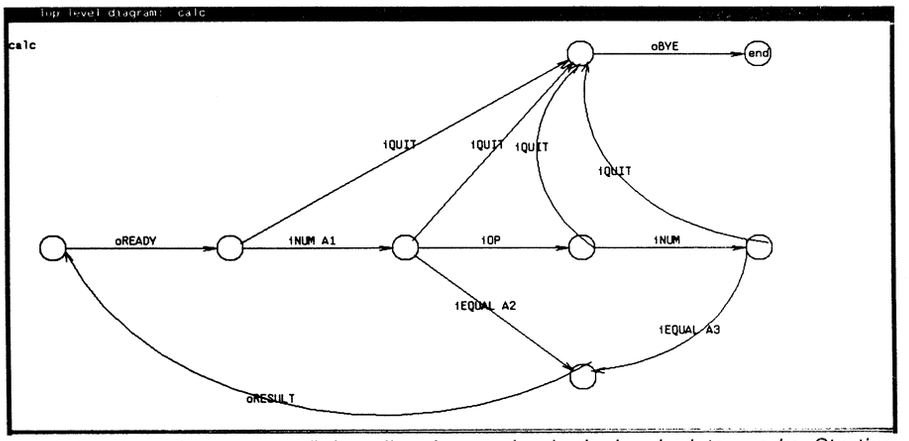


Jacob/NRL

A visual programming language can be used to design a computer display that allows users to choose what to do next by selecting instructions from appropriate "windows."

together invoices from information on orders and prices. All the necessary procedures are represented on a video display and can be easily modified and manipulated to construct the required form. Zloof is now developing a "Database-by-example" that uses a similar visual form to retrieve information from computer files.

This visual approach gets complicated, however, when a single picture can't tell the whole story. "You can easily lose track of what you're doing," says Zloof. There may be limits to how cluttered a single picture can get and to how many different pictures a person can handle.



Illustrations: Jacob/NRL

This "state transition diagram" describes how a simple desk calculator works. Starting at the circle farthest to the left, the diagram illustrates the choices at every step in performing a simple computation like adding together two numbers.

Margaret A. Khorfhage of Trammel Crow Co. in Dallas argues that stylized pictures take time to read and grasp. People have to be taught to understand the symbols, she says. Although good "icons" can sometimes be created for nouns, other parts of speech, like verbs, tend to lose out.

Khorfhage wonders how much of the current surge of interest in visual languages is just a fad. "Can we communicate effectively without a visual language?" she asks.

"It's not a fad," says Adarsh K. Arora of the Gould Research Center in Rolling Meadows, Ill. He points to the increasing use of computer-aided design, spreadsheet programs, windows and other visual aids to help users get what they need out of a computer. All of these may be considered primitive visual programming languages, he says.

Arora suggests that animation may eventually be used to show how objects, as designed on a computer, would work. Visually following flows through a "data structure" or program to see how it behaves could make it easier to track down errors.

"Visual languages are here," says Jacob, "but they have a long way to go. In the long run, a theoretical understanding of visual perception is needed." This would let designers devise more effective graphic symbols for their particular picture programs. □

Letters continued from p. 99

Causal chain challenged

In "Guns or babies?" (SN: 7/20/85, p. 40) a strong favoritism is given to one particular causal explanation of a correlation. While it is likely that this favoritism accurately reports the orientation of whoever was interviewed for the article, I suspect SCIENCE NEWS might concern itself a bit more with alternatives, in the name of objectivity.

When a correlation between two factors is detected, assuming the correlation is not accidental, there can exist one or more of three causal relationships: A causes B, B causes A, and/or A and B are both caused by C. In the recent article, attention was paid to the idea that increased military spending reduces social services funding so that health care or other correlates with low infant mortality suffer (A causes B), and the death rate rises. No alternatives to this causal chain were mentioned or proposed.

If we look for a reversal of this hypothetical causal chain (B causes A), we could speculate that in a society with high infant mortality (a lot of parents losing their children, perhaps due to underdevelopment of local medical or sanitation services or prevalent local disease), there may exist a relatively high amount of personal anger, frustration, fear and disillusionment and a general societal belief that life is essentially more hostile than it may be believed to be elsewhere. These emotions and beliefs may result in society-wide attitudes which are manifest in increased attention to military armament.

Looking for an instance of explanation for the third type of causal chain for this correlation (A and B are both caused by C), we might wonder if militarism and infant mortality could both have their roots in an economy of scarcity, which may exist for ecological or historical reasons, and which results in both insufficient health care resources and an overly strong society-wide sense of the need for self-protection (coupled with genuine fear on the part of the rulership).

For each of these possibilities, the solutions sought in order to improve the situation would be different.

Laurie Spiegel
New York, N.Y.

Alternatives to animal experiments

It is encouraging to hear of the National Academy of Sciences' recommendation to seek alternatives for laboratory animals in experimentation ("Routing out phylogenetic bias," SN: 5/18/85, p. 312). It is not an entirely new idea, though. Sen. Dole (R-Kans.) introduced a bill in early 1984 which, along with its companion bill, called for an amendment to the Animal Welfare Act. Included within the bill were findings that different methods of testing existed, as did the capacity for a data base that would minimize unnecessary duplication of experiments on animals. The bill has been stagnant for close to a year, so it was good to see an interest in the matter again.

However, I became discouraged when I read of a National Research Council report ("Medicine capsules," SN: 6/1/85, p. 345) calling for the opening of a research center to study the effects of deliberate and accidental injuries due to motor vehicle accidents, firearms, jumps, etc. Note: to study, not treat injuries. This implies that wounds would be inflicted upon animals simply to observe — a practice that does take place.

Wouldn't the more logical solution be to focus on these injuries before they occur?

Gregory C. McKinney
Philadelphia, Pa.

WIMPS aren't the bullies

WIMPS (Weakly Interacting Massive Particles) are the cause for the lack of neutrinos emanating from the sun's core, suggest Faulkner and Gilliand (SN: 7/13/85, p. 23).

Recently, Raymond Davis of Brookhaven National Laboratory has presented data that suggest that the sun's neutrino counts vary inversely with the sunspot cycle. Sunspots occur on the sun's visible surface while neutrinos, which are not affected by the magnetic fields of sunspots, supposedly come from the sun's core. Any viable solar theory must explain the dynamic relationship between sunspots and the neutrino count, which the WIMP theory does not.

Stephen Goodfellow
Detroit, Mich.

Afraid I couldn't resist the enclosed thought after reading "Weak sun blamed on WIMPS." Please forgive my irreverence.

Joan Hensley
Bloomington, Ind.

COMPANION TO THE "QUARK" BEING CALLED UP ON THE "SQUARK BOX"