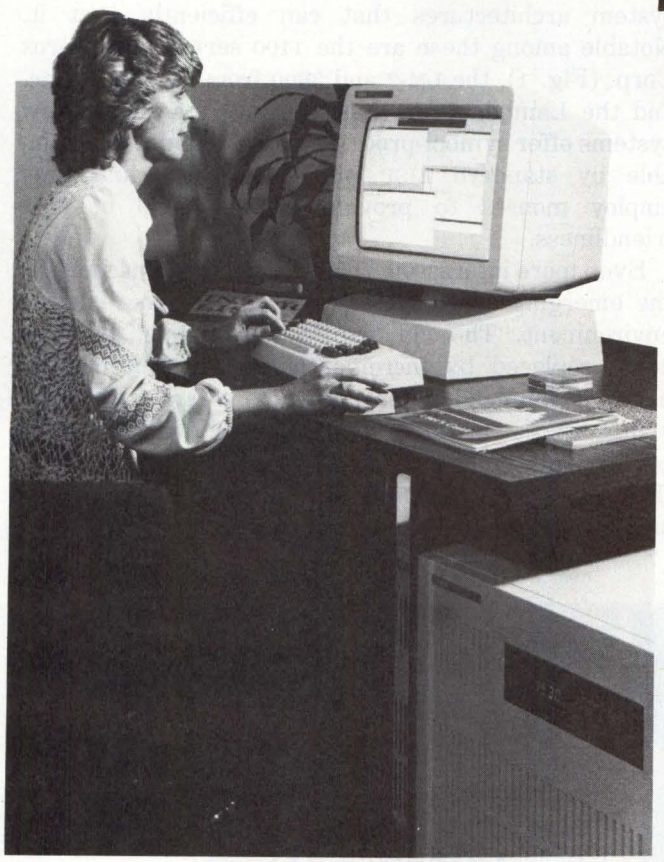
Computers simulate human experts

STEVEN K. ROBERTS, Words'Worth Inc.



'Knowledge engineering' addresses applications previously considered inappropriate for computerization

Fig. 1. The Xerox 1100 computer system *provides an interactive LISP programming environment with 8M bytes of virtual address space. Standard features include a 23M-byte disk, a network interface and a 1024- × 808-pixel display. A three-button mouse (under operator's right hand) supplements a 64-key keyboard.*

The 1982 conference of the American Association for Artificial Intelligence demonstrated that artificial intelligence, traditionally an academic field, is finally starting to attract the attention of bottom-line-oriented business managers. The percentage of attendees from private industry was higher than ever, and the exhibition included several real products complete with specification sheets and model numbers.

Although artificial intelligence is a diverse field, including natural-language processing and computer vision, most progress in AI has been achieved by using computers to simulate human experts in narrowly defined subjects. Using heuristic approaches and novel programming environments, these "expert systems" constitute an exciting and profitable new class of computer applications.

Three kinds of knowledge

Most computer systems represent a distillation of task-specific knowledge that can be implemented as a set of programs and data structures. For example, in a simple industrial-control system, all the necessary "knowledge" might be stored in the form of a few values in a program table. But for a machine to be intelligent, it must incorporate not only factual knowledge, but also heuristic and "meta-knowledge."

Heuristic knowledge consists of intuitions, judgment rules, pet theories and inference procedures that, in concert with factual knowledge about a subject, allow a human expert to exhibit intelligent problem-solving behavior. While it has a vaguely procedural flavor about it, heuristic knowledge is much more abstract than what is generally called "software."

Meta-knowledge is even more general and abstract. It is perhaps best described as a nonsymbolic awareness of how to think—how to approach and organize an unfamiliar problem so that both heuristic and factual knowledge are effectively applied.

The more heuristic knowledge a system has, the less it must search for a specific datum. The less searching it

must do, the less bogged-down it will be when confronted with a highly associative task such as visual pattern recognition. The more meta-knowledge a system has, the more it can learn from experience and refine its behavior.

The essence of knowledge engineering is the integration of factual, heuristic and meta-knowledge into a practical system. There are various approaches, ranging from collections of "production rules" (IF-THEN structures) to relatively passive "semantic networks" consisting of knowledge entities whose interrelationships are defined by lists of properties and linkages. Whatever the approach, a key issue is the naturalness of expression in both the "teaching" of the system and in its subsequent application.

Novel programming environments

The advent of knowledge processing has spurred new programming environments that facilitate problem solving. A significant development is the resurgence of LISP—one of the oldest surviving computer languages. Developed by John McCarthy of the Massachusetts Institute of Technology's Artificial Intelligence Lab in 1958, LISP has become the most widespread implementation language for intelligent systems. Possessing major philosophical differences from most computer languages, it has spawned development of specialized system architectures that can efficiently host it. Notable among these are the 1100 series from Xerox Corp. (Fig. 1), the LM-2 and 3600 from Symbolics Inc. and the Lambda from LISP Machine Inc. These new systems offer symbol-processing performance unattainable by standard LISP implementations, and they employ mouses to provide a high level of user friendliness.

Even more interesting than the new LISP machines is the emerging concept of an integrated programming environment. The old "edit-compile-debug" cycle is being replaced by incremental compilation, maintenance of command histories to undo errors and display managers to provide dynamic windowing that is conceptually similar to pieces of paper on a desk top (Fig. 2). Such intelligent development tools portend signficant increases in programmer productivity.

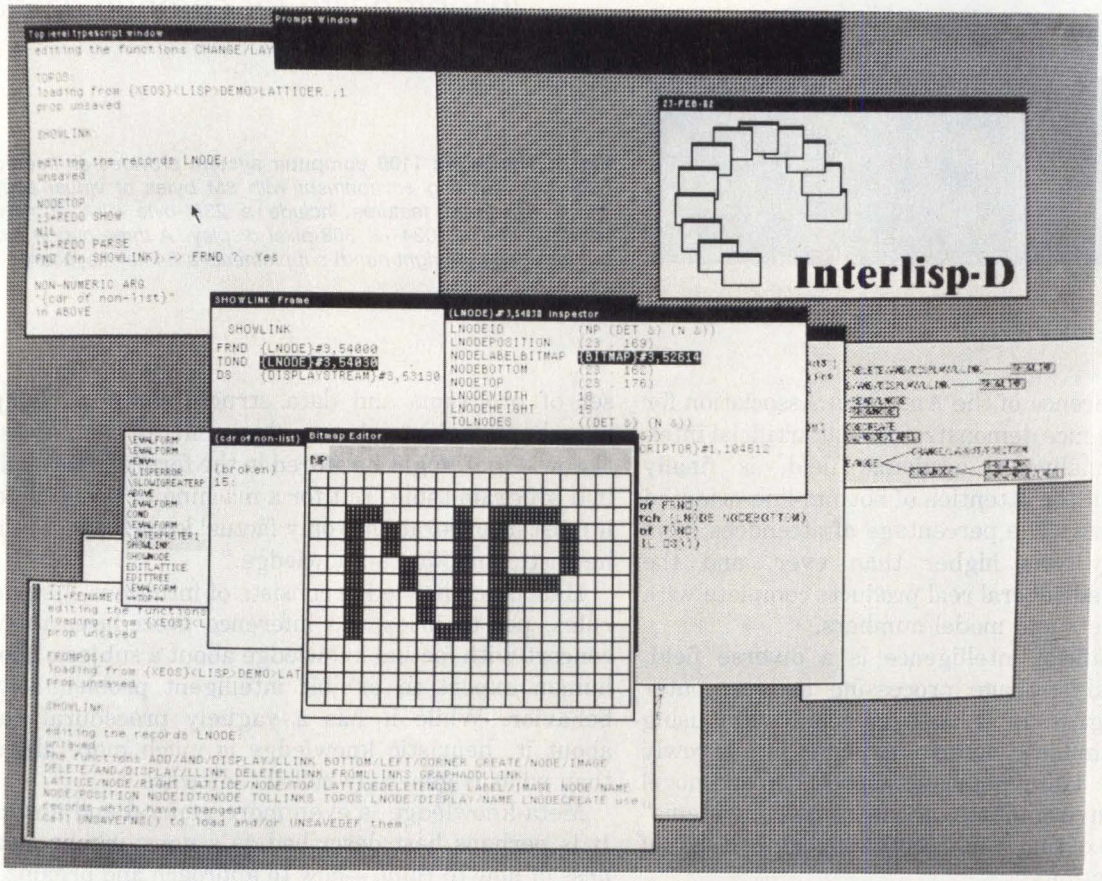


Fig. 2. Display "windowing" on the Xerox 1100 *allows several user tasks to be represented graphically in a way that reveals their interrelationships. Each task can be modified and redisplayed without global recompilation. A record is kept of all commands and their results, so that the user can back up and undo operations without the havoc that might normally be expected.*

Intelligent expert systems

The most practical applications of knowledge processing are in building systems geared to narrowly defined applications, such as medical or electronic diagnosis, that traditionally have required a high level of human expertise.

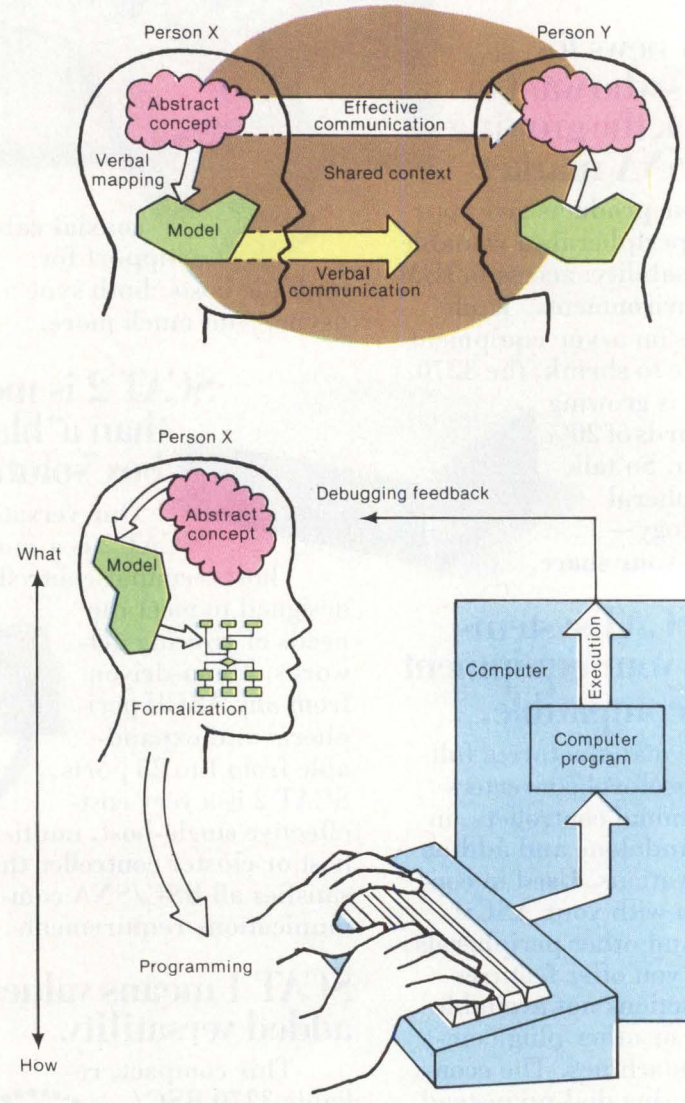
Expert systems employ knowledge processing that differs dramatically from conventional data processing. In conventional data processing, problems are solved through explicit algorithms in a procedural language. For example, database manipulation is performed through predefined operations on a data structure. This approach lends itself well to automating routine processes and organizing predictably interrelated data. But knowledge processing implies a different structural flavor. New facts and relationships can be added without rewriting any code because they take the form of independent packets of information. The distinction between "program" and "data" almost entirely dissolves because, from the standpoint of the application, there is no established sequential procedure. Internal

THE MAN-MACHINE COMMUNICATION PROBLEM

Language is proving to be inseparable from intelligence—a fact that has stymied attempts to develop workable natural-language interfaces and translation programs. The difference between human and human-computer communication is illustrated by the upper and lower figures shown here.

In the upper figure, Person X is expressing an idea to Person Y and is delivering a sequence of words chosen not only for their applicability to the "model" but also for their appropriateness to the listener. The choice of words is made on the basis of the *shared context,* a constantly growing body of knowledge that links the participants. If X is successfully fitting the words to the context, then Y can create a corresponding model of the idea and gradually project it "upward" into abstract conceptual space. New information is added to the shared context throughout the conversation. If X misjudges the shared context, communication falters; if X suddenly changes it, the result is laughter or confusion.

In the lower figure, Person Y has been replaced by a computer. X is now confronted with a tedious task: X must formalize the verbal model and express it in a syntactically and semantically rigid computer language without a shared context. X begins with an idea of *what* X wants the system to do, them dons a formal intellectual straitjacket to tell it exactly *how*. This gives rise to the idea of a *what-how* spectrum, with microcode and assembly language near the bottom *(how)* and human dialog near the top *(what)*. Until a shared context can be created, human-computer communication will continue to be awkward. Thus, AI techniques are prerequisite for a truly friendly system.



to an expert system is an infrastructure of inference procedures, probably written in a symbolic programming language such as LISP. But this infrastructure operates exclusively at the meta-knowledge level. All knowledge associated with the application itself is embodied in a flexible and relatively unstructured database. Given a sufficiently limited application domain, this approach permits a system to draw inferences from its stored knowledge in ways that were not anticipated at the time of its creation. Further, the expert system can explain its conclusions, deal with incomplete input data and know enough about what it *doesn't* know to generate follow-up queries.

These capabilities suggest a class of applications quite distinct from those normally considered for computerization. Teknowledge Inc. points out, "Knowledge systems are particularly useful in situations where expertise is scarce, different parts of the expertise are distributed among many people or the expertise is simply not available on a reliable or continuing basis."

Also suitable for expert systems are operations in which there is a heavy dependence on experts who are overworked to the point at which considerable delays in production are caused. In engineering, this often occurs when the "resident wizards" spend an inordinate amount of time with repair personnel, sales representatives, potential customers and vendors simply because the experts are the only ones who know a company's

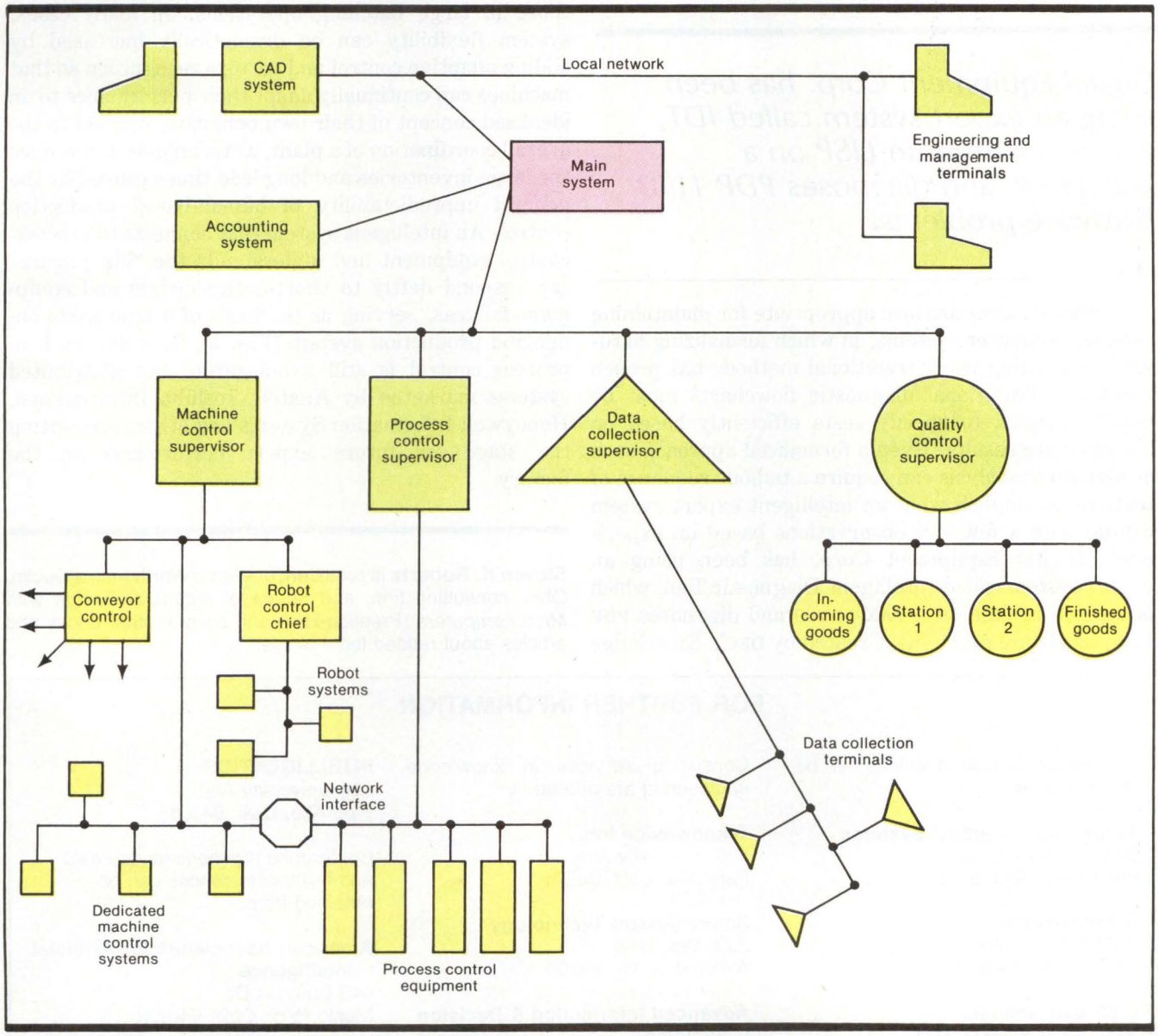


Fig. 3. A factory-wide hierarchical network *contains high-level systems that are isolated from the details of subordinate tasks. The main system, which in many conventional designs carries the entire processing load, serves here only to schedule and control global activity at a relatively abstract level. This is an ideal environment for implementing artificial-intelligence techniques, since the hierarchical network can host real-time "smart" software without compromising process-control response time.*

product well enough to answer everybody's questions.

In electrical engineering, expert systems are becoming justifiable in VLSI design, printed-circuit-board layout, high-level machine diagnostics and customized system configuration. VLSI design is profiting from the application of knowledge-engineering techniques. Work under way in Palo Alto, Calif., at Xerox's Research Center and at Stanford University is enabling a knowledge-based system to become an expert assistant for VLSI designers. Knowledge is derived from the Mead and Conway textbook, *Introduction to VLSI Systems* (Addison-Wesley, 1980), and from examples provided by human practitioners.

Digital Equipment Corp. has been using an expert system called IDT, which is written in LISP on a VAX-11/780 and diagnoses PDP 11/03 hardware problems.

Expert systems are also appropriate for maintaining complex computer systems, in which formalizing hardware debugging using traditional methods has proven awkward. Traditional diagnostic flowcharts must be highly complex to specify tests efficiently based on intermediate results. Even a formalized approach such as signature analysis can require a tedious sequence of tests to accomplish what an intelligent expert system can do with a few key observations based on experience. Digital Equipment Corp. has been using an expert system called Intelligent Diagnostic Tool, which is written in LISP on a VAX-11/780 and diagnoses PDP 11/03 hardware problems. Created by DEC's Knowledge Engineering Group, South Lawrence, Mass., IDT runs tests, interprets the results and suggests specific card replacements to an on-site technician. The system learns from experience, considers human opinions and applies statistical information so that testing is biased toward frequent problems.

Other applications suitable for expert systems are those in which large quantities of routine data must be monitored closely by a highly knowledgeable expert for a suitable combination of conditions that may represent a problem. This situation is common in the control of critical industrial processes such as nuclear power plants, turbines and large furnaces. A pattern of activity may suggest imminent failure, even though no alarm conditions exist. Such applications are complicated by the need for quick judgments and responses.

Installations that can profit from knowledge engineering include distributed process-control systems such as those in large batching operations. In many cases, system flexibility can be dramatically increased by adding adaptive control and pattern recognition so that machines can continually adapt their performance to an idealized concept of their own behavior. Applied to the overall coordination of a plant, AI techniques can reduce the large inventories and long lead times caused by the general unpredictability of human-based production control. An intelligent system that connects to process-control equipment and understands the "big picture" can respond deftly to short-notice orders and equipment failures, serving as the basis of a true parts-on-demand production system (Fig. 3). This approach to process control is still avant-garde, but distributed systems marketed by Anatec, Toshiba International, Honeywell Information Systems and others are setting the stage for future expert performance in the factory. □

Steven K. Roberts is president of Words'Worth Inc., a Dublin, Ohio, consulting firm, and author of *Industrial Design with Microcomputers* (Prentice-Hall) and several other books and articles about related technologies.

FOR FURTHER INFORMATION

Information on LISP machines can be obtained from:

Xerox Electro-optical Systems
300 N. Halstead St.
Pasadena, Calif. 91107

Symbolics Inc.
9600 De Soto Ave.
Chatsworth, Calif. 91311

LISP Machine Inc.
3916 S. Sepulveda Blvd.
Culver City, Calif. 90230

Consulting services in knowledge engineering are offered by:

Teknowledge Inc.
525 University Ave.
Palo Alto, Calif. 94301

Smart System Technology
P.O. Box 9164
Alexandria, Va. 22304

Advanced Information & Decision Systems
201 San Antonio Circle, Suite 286
Mountain View, Calif. 94040

INTELLIGENTICS
124 University Ave.
Palo Alto, Calif. 94301

Conference proceedings, journals and further references can be obtained from:

American Association for Artificial Intelligence
445 Burgess Dr.
Menlo Park, Calif. 94025