KNOWLEDGE ACQUISITION FOR EXPERT SYSTEMS

Brian R. Gaines

Application of expert systems is impeded by the knowledge engineering bottleneck. This has led to the development of rapid prototyping tools such as PLANET [10], ETS [1], MORE [7], SALT [8], MOLE [3], KITTEN [9], AQUINAS [2] and KSSO [5] which expedite knowledge acquisition and transfer [4,6]. These have proved practically useful but the range of applications of existing tools is limited, and it is not clear what limitations are inherent in the prototyping tools, what arise from the shells and their knowledge representation and inferencing procedures, and what arise from our lack of understanding of the underlying processes of knowledge acquisition and transfer. There has also been very little analysis of the developmental processes for knowledge-based system in relation to well-established techniques for system specification, rapid prototyping, evaluation, full-scale development, documentation and maintenance of conventional data-processing systems.

FUNCTIONALITY REQUIRED IN EXPERT SYSTEMS

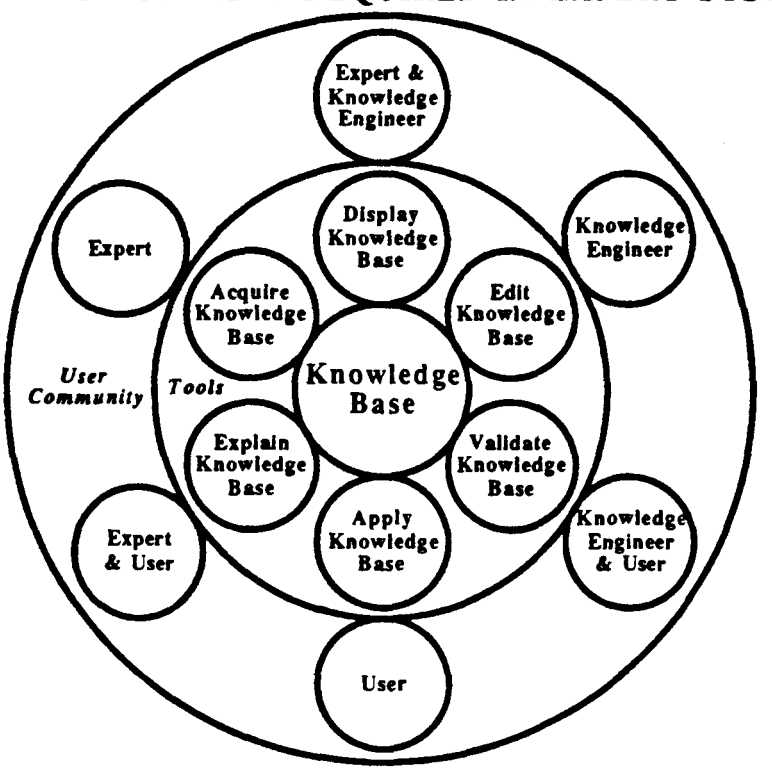

Fig.1 Functionality of expert systems

In considering the development of expert systems, it is important to realize that they are complex integrated systems involving a variety of tools and associated user environments, and that they also involve different categories of operator. Fig. 1 shows the functionality required in an expert system:

• *Knowledge base*: the central ring shows the knowledge base of facts and inference rules that is the heart of an expert system. This extends the facilities of a conventional database by enabling the storage not only of facts but also of rules that enable further facts to be derived. These rules may both be explicitly expressed (production rules) and implicit in the conceptual structure in which the facts are expressed (objects, frames).

• *Expert system shell*: the middle ring shows the expert system shell that provides computational facilities for applying the knowledge base to user decision support:

—the *application system* at the bottom tests the user's hypotheses or seeks to satisfy his goals by deriving the consequences of facts about a particular situation reported by the user, using the general facts and inference rules supplied by the expert and edited by the knowledge engineer;

—the *explanation system* to the lower left answers queries about the way in which facts have been derived in terms of what information and inference rules have been used;

—the *acquisition system* to the upper left provides tools for interviewing the expert to obtain his vocabulary, and general facts and inference procedures in terms of it;

—the *display system* at the top provides tools for presenting the knowledge base in an understandable form, and the relations between the facts and inference procedures it contains;

—the *edit system* to the upper right provides tools for editing the knowledge base while maintaining its integrity in terms of the vocabulary, variables and operations used;

—the *validation system* to the lower right provides tools for checking the knowledge base against specific case histories with known consequences.

Dr Gaines is Director of the Knowledge Science Institute, University of Calgary, Calgary, Alberta, CANADA T2N 1N4

• *Operators*: the outer ring shows the three classes of operator of expert systems:
—*users* come to the system for advice on specific problems in a particular domain, supplying it with specific facts and their hypotheses about consequences or goals;
—*experts* come to the system to transfer their knowledge about the problem domain, supplying it with generally applicable facts and inference procedures;
—*knowledge engineers* act as intermediaries between the expert and the system, helping him to encode his knowledge and validate the operation of the resultant expert system.

Most current expert system shells:
• offer a limited knowledge base represented in terms of production rules and frames;
• have weak application capabilities limited to deriving consequences and testing assertions;
• give simple explanations in terms of the facts and rules used;
• have no integrated acquisition capabilities;
• have limited capabilities to display the knowledge base, sometimes in graphical form;
• have interfaces to the knowledge base editor requiring understanding of internal data structures;
• have limited validation capabilities and no integrated database of case histories.

THE VIRTUAL MACHINE HIERARCHY FOR EXPERT SYSTEM?

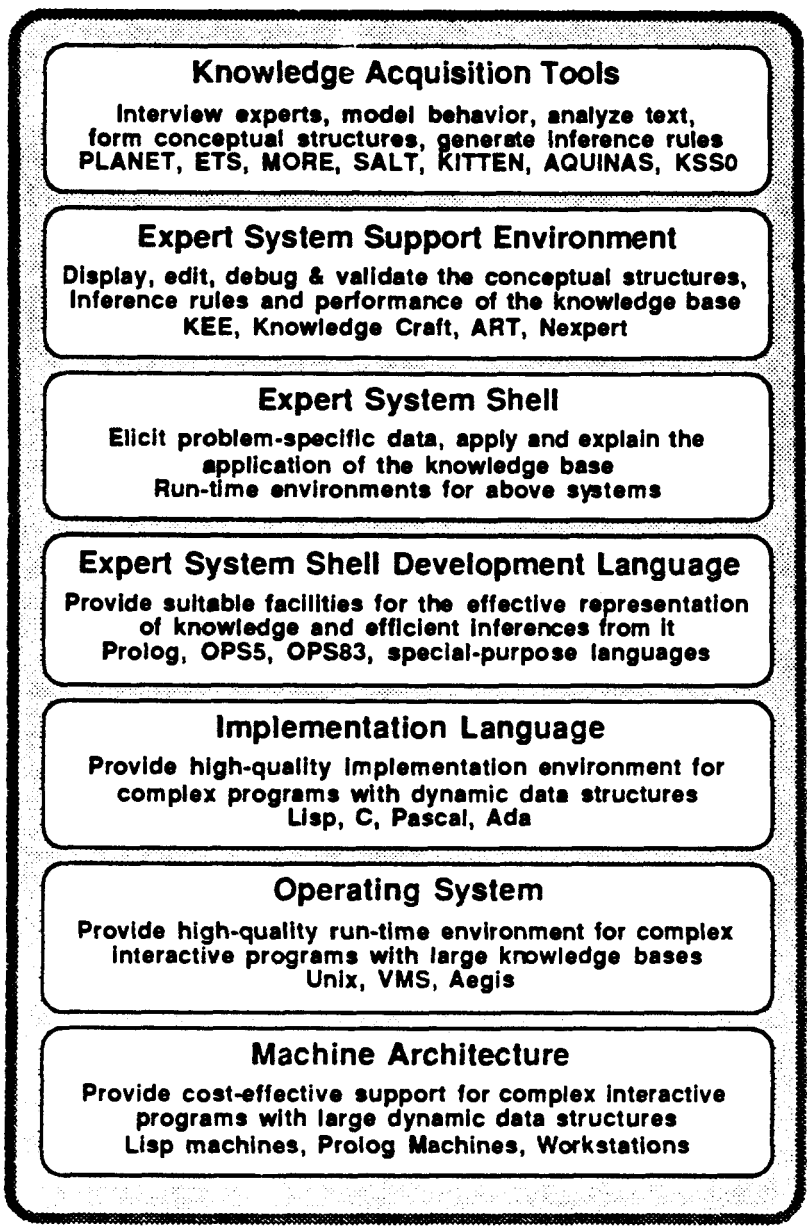

Fig.2 The VM hierarchy underlying expert systems

Knowledge-based systems are at the top of the *virtual machine hierarchy* which has developed in computing, and there are many layers of technology between an expert system and the computer on which it is running. In theory, system developers should not need to know about the lower levels of the hierarchy—machine architectures, operating systems and implementation languages are remote from knowledge processing. However, in practice, these lower levels are the foundations on which systems are built and any defects in them can undermine the functionality of the upper levels.

Figure 2 shows significant levels of the virtual machine hierarchy for knowledge-based systems:
• *Knowledge Acquisition Tools*: at the top of the hierarchy are the tools for automating the systems analysis for expert systems, through automatic interview procedures, modeling actual expert behavior, and analysis of knowledge in textual form.
• *Expert System Support Environment*: at the second level of the hierarchy is the equivalent of the Application Programming Support Environment (APSE) in conventional systems. The Expert System Support Environment (ESSE) has already been described in terms of Figure 1.
• *Expert System Shell*: at the third level of the hierarchy is the run-time system that elicits problem-specific information from the user, provides advice based on its knowledge base, and explains that advice in as much detail as required.

- *Expert System Shell Development Language*: at the fourth level of the hierarchy is the language in which the expert system shell is written, generally a special-purpose environment for coping with knowledge representation and inference.
- *Implementation Language*: at the fifth level of the hierarchy is the implementation language which actually interfaces to the computer. This tended to be Lisp in the early days of expert systems but as speed and space efficiency have become significant, and knowledge representation has become better understood, other languages that support dynamic data structures such as C and Pascal have become widely used.
- *Operating System*: at the sixth level of the hierarchy is the operating system within which the implementation runs. This needs to provide good interfaces to other programs, large databases and communications.
- *Machine Architecture*: at the lowest level of the hierarchy is the machine on which the expert system runs. This is really irrelevant, but use of Lisp in the early days of expert systems has focused attention on Lisp machines. Nowadays any good workstation is adequate.

PROBLEMS IN EXPERT SYSTEM DEVELOPMENT

The preceding section have given an overview of expert systems and an indication of some of the problems that may arise in their development. Many of these relate to the knowledge engineering bottleneck but there are many others that are not direct outcomes, and the knowledge engineering problems in themselves have many facets.

There are what may be termed the problems of *professionalism* in establishing prototyping facilities:

- *Theory*: the lack of scientific and engineering foundations for knowledge-based systems is a major impediment to the development of professional standards for expert system development. We understand remarkably little about the nature of expertise even though it one of the most valued commodities in our society. We also have little fundamental understanding of knowledge, the ways in which it is represented in human thought processes, the ways in which it is used in human problem solving, and the ways in which it is transmitted between people.
- *Techniques*: the lack of well-established techniques for knowledge engineering is a major problem for new development teams. There is some folklore, generally mutually contradictory, a few rules of thumb based on experience, but so far no in-depth studies of what is actually done and how successful it has been.
- *Tools*: the lack of commercially available tools for knowledge acquisition and transfer is also a major problem. There are now many research reports of such tools but even those developed industrially have not yet been developed into products.
- *Training*: the training situation is improving all the time, largely because it has become the major secondary industry to the supply of expert system shells. A range of courses is being offered by commercial and university organizations. It is still a problem to get good quality courses targeted to specific requirements and on the schedules required.

Knowledge acquisition and transfer processes in human society are complex and poorly understood. Neurology, psychology, linguistics, education, sociology, anthropology, philosophy and systems theory all have significant contributions to make. However, integrating them, disambiguating jargon, and combining different objectives and perspectives are major problems.

There is a variety of short-term problems arising from these weaknesses in foundations:

- *System Specification*: the formal specification of software that has made software engineering a professional discipline which can ensure quality within well-defined budgets and schedules is notably lacking for expert system developments. We seem at the stage of conventional software development of twenty years ago. This should change as expert system developments move out of the laboratories into the hands of experienced software development groups trained in software management. There is little in the literature to support and expedite this change currently, and we need to develop specification disciplines for expert systems.
- *Access to Expertise*: the problems of getting experts to make the time commitments necessary to the development of expert systems should never be underestimated. Successful projects inevitably seem to have involved the experts in such a way that they become personally committed to the success of the system.
- *Acquisition of Skills*: communicating the basis of skilled performance between people, and between people and computers, is very difficult.

- *Representation of Knowledge*: encoding knowledge within the representation schemes of current shells can present major difficulties. Production rules as used in early expert systems offer very limited facilities for encoding simple knowledge structures at a single level. Frames and their generalization through objects allow conceptual hierarchies to be better expressed. Neither rules nor frames are adequate to encode the procedural or temporal knowledge which is often a key component of expertise.

Rapid prototyping is a significant practical objective and techniques for accelerated initial development without loss of long-term quality are now common in conventional data processing. However, expert system development emphasizes weaknesses in some areas of performance which we tend to take for granted in standard information technology. There are problems with:

- *Validation*: because we can rarely define and quantify expertise, the validation of expert systems can be very difficult. Do we subjectively validate against the opinions of the expert or do we attempt objective validation of performance against performance standards, accepting that the experts may.not themselves satisfy such criteria. How, do we determine whether the knowledge base captures the expert's relevant knowledge, accurately, consistently and completely? There are very few validation studies in the literature.
- *Usability*: the usability of expert systems raises many problems. If we accept the jargon of the expert and the rules of inference based on it, .how do we ensure that the operators will use this jargon in the same way as the expert so that the inferences remain valid. It is such considerations that make nonsense of the popular stereotype that access to encoded expertise can make experts of all of us. Understanding the questions, advice and explanations from an expert system generally requires an operator with the same professional background as the expert.
- *Maintainability*: the superficiality of the knowledge encoded in current expert systems raises major problems of maintenance. Keeping the knowledge base up to date may require an ongoing effort at the same level as the original development, a situation very different from most conventional data processing. The usual problems of extensive maintenance not introducing problems of its own are at least as severe for expert systems as for standard data processing.
- *Upgradability*: many expert systems, and the shells supporting them have been designed as stand-alone packages with little scope for integration into large-scale data processing environments. These systems will be difficult to upgrade and integrate with other software.

REFERENCES

[1] Boose, J.H. (1986). Rapid acquisition and combination of knowledge from multiple experts in the same domain. **Future Computing Systems 1**(2), 191-216.

[2] Boose, J.H. & Bradshaw, J.M. (1987). Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for expert systems. **International Journal of Man-Machine Studies 26**(1) (January).

[3] Eshelman, L., Ehret, D., McDermott, J. & Tan, M. (1987). MOLE: A tenacious knowledge acquisition tool. Carnegie-Mellon University. **International Journal of Man-Machine Studies 26**(1) (January).

[4] Gaines, B.R. (1986). Foundations of knowledge engineering. Bramer, M.A., Ed. **Research and Development in Expert Systems III**. pp.13-24. Cambridge: University Press.

[5] Gaines, B.R. (1987). Rapid prototyping for expert systems. Oliff, M., Ed. **Proceedings of International Conference on Expert Systems and the Leading Edge in Production Planning and Control**. pp.213-241. University of South Carolina.

[6] Gaines, B.R. (1987). An overview of knowledge acquisition and transfer. **International Journal of Man-Machine Studies 26**(4) (April).

[7] Kahn, G., Nowlan, S. & McDermott, J. (1985). MORE: an intelligent knowledge acquisition tool. **Proceedings of the Ninth International Joint Conference on Artificial Intelligence**. pp. 581-584. California: Morgan Kaufmann.

[8] Marcus, S., McDermott, J. & Wang, T. (1985). Knowledge acquisition for constructive systems. **Proceedings of the Ninth International Joint Conference on Artificial Intelligence** pp. 637-639. California: Morgan Kaufmann.

[9] Shaw, M.L.G. & Gaines, B.R. (1986). Advances in interactive knowledge engineering. Bramer, M.A., Ed. **Research and Development in Expert Systems III**. pp.111-122. Cambridge: University Press.

[10] Shaw, M.L.G. & Gaines, B.R. (1986). Interactive elicitation of knowledge from experts. **Future Computing Systems, 1**(2), 151-190.