KNOWLEDGE ACQUISITION AND ITS INTEGRATION WITH DATA PROCESSING TECHNOLOGIES

Brian R Gaines
Knowledge Science Institute
University of Calgary
Calgary, Alberta, Canada T2N 1N4
gaines@cpsc.ucalgary.ca

Abstract: Tools and techniques for knowledge acquisition for knowledge based systems need to be integrated with the overall system and not treated as separate components. The variety of sources of knowledge, representations and applications within the system, and user roles, makes such integration complex. The complexity raises many issues not generally thought of as part of knowledge acquisition yet fundamentally significant in extending and integrating acquisition tools. This paper gives an analysis of the structure of knowledge-based systems as it relates to the integration of acquisition. The analysis results in a number of design principles and separation derived systemically from consideration of the distinctions involved.

INTRODUCTION

At the time of the first AAAI Workshop on Knowledge Acquisition for Knowledge Based Systems in Banff, November 1986, the literature on knowledge acquisition was sparse, scattered in conference publications, and difficult to access. The 1986 Workshop and the ensuing journal and book publication of some forty papers representing the state of the art world-wide changed that situation (Gaines & Boose 1988, Boose & Gaines 1988). The inclusion of sessions on knowledge acquisition in many conferences, the European Workshops on Knowledge Acquisition for Knowledge Based Systems in London and Reading, Bonn and Paris, and the later AAAI Banff Workshops, have changed the situation even further. We now have a world wide network of communicating researchers aware of one another's activities and an open, widely disseminated and readily available literature on knowledge acquisition.

The problem now is not so much to access research and experience in knowledge acquisition but to make sense of the wide ranging material available and, in particular, to apply the results to improve effectiveness in the development of knowledge-based systems. There are a number of impediments to such understanding and application:

- A diversity of techniques and tools that overlap in their applications and where it is not clear whether they are competitive alternatives or complementary partners
- Lack of variety, detail and evaluation in the case histories of applications of the techniques and tools
- Lack of access to the techniques and tools outside the narrow research communities originating them
- Lack of standardization in the knowledge representation resulting from the techniques and tools making it difficult to integrate them and to interface them to existing knowledge based systems
- Lack of standardization in the forms of data required by the techniques and tools making it difficult to apply them in the same situation and compare them
- Lack of standardization in the user interfaces to the interactive tools making it difficult to integrate them in an effective environment for human-computer interaction
- Lack of portability in the run-time environments required by the tools making it difficult to integrate them with other systems

These are not unreasonable problems at this stage of development of knowledge based systems. We are in the midst not only of the evolution of a rapidly developing technology but also in a revolution of our understanding of the nature of knowledge and knowledge processes in society and technology. Premature standardization resulting in over-rigid systems would be inappropriate and a major impediment to progress. However, we have to balance the need for integration, dissemination and application of knowledge acquisition techniques and tools against the dangers of rigidity. This paper addresses the issues involved, the evolutionary trends, the key requirements for integration and the practical approaches that may be taken.

TRENDS IN KNOWLEDGE ENGINEERING

The basic model for knowledge engineering has been that the knowledge engineer mediates between the expert and knowledge base, eliciting knowledge from the expert, encoding it for the knowledge base, and refining it in collaboration with the expert to achieve acceptable performance. Figure 1 shows this basic model with manual acquisition of knowledge from an expert followed by interactive application of the knowledge with multiple clients through a knowledge-based system shell. There are many problems with this basic model such as the assumption that knowledge engineers are available in sufficient numbers and capable of performing the intermediary role (Shaw & Gaines 1987), and the problems of actually achieving the knowledge transfers involved (Gaines 1987). Understanding and communication are quite distinct skills from performance, and individuals may have widely differing capabilities in each of these three. The practitioner, the scientist and the teacher are three distinct roles that are developed in differing degrees in differing individuals. In knowledge acquisition it may be necessary to build a framework for the understanding and communication of expertise which did not previously exist.

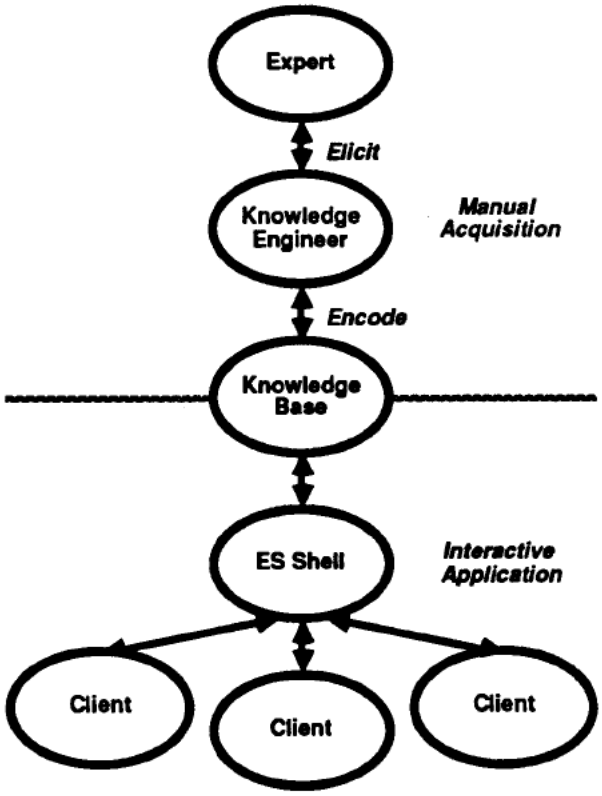

Fig.1 Basic model of manual knowledge acquisition

Figure 2 is a detailed expansion of Figure 1 showing the long chain of information flow between expert performance of a task and client performance of the same task supported by an expert system.

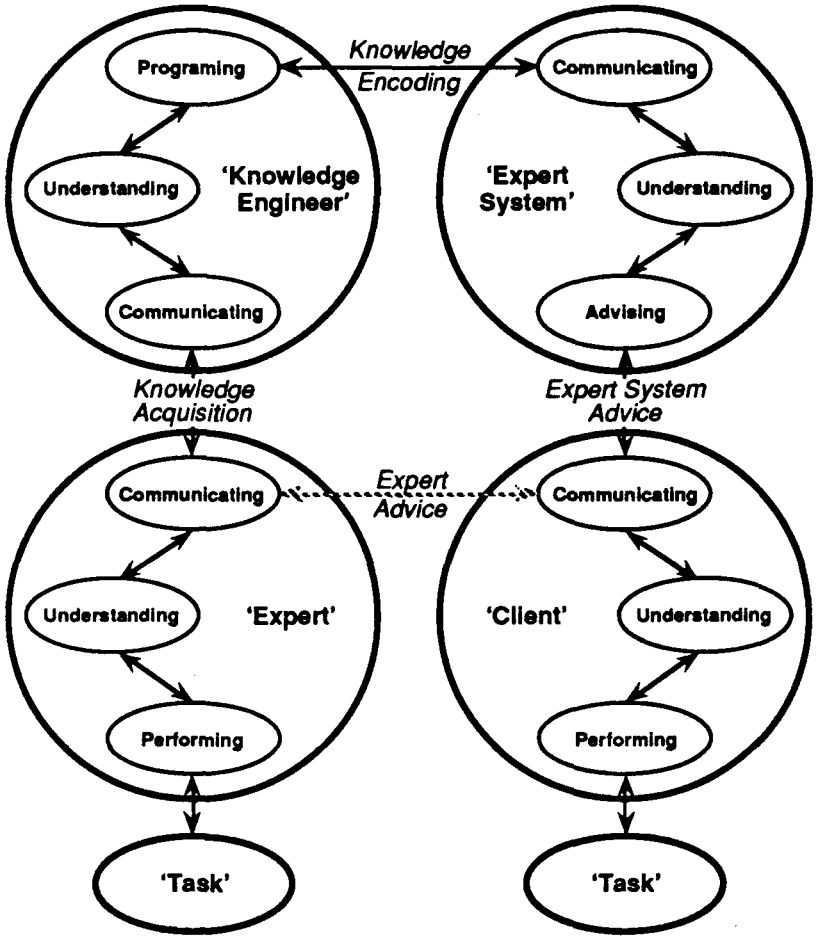

Fig.2 Information flow between expert performance and client performance

The person in the role of an 'expert' is shown as having three associated roles, or internal processes: performing the task; understanding the basis of the performance; and communicating the basis of the performance—the 'practitioner', 'scientist' and 'teacher' roles.

The person in the role of a 'knowledge engineer' is shown as having three similar roles: communicating about the basis of the performance with the expert; understanding the basis of the performance at a level sufficient to specify it for a computer; programing this specification in a form suitable for communication to an expert system on a computer.

The computer in the role of an 'expert system' is shown as having three major internal processes: communicating about the basis of the performance with the knowledge engineer; 'understanding' the basis of the performance in terms of its knowledge representation and inference processes at a level sufficient to generate advice for a client; and advising on specific task performance through communication with a client.

The person in the role of a 'client' is shown as having three roles similar to those of the expert: communicating about the basis of specific performance with the expert system; understanding this at a level sufficient to carry out the performance; performing the task based on the advice of the expert system.

The shaded arrow between the communication roles in the 'expert' and 'client' indicates the advisory role of the expert that is being emulated in part through the 'expert system.' The actual flow of information involved in this advice consists of: 'knowledge acquisition' by the knowledge engineer from the expert; 'knowledge encoding' by the knowledge engineer into the expert system; and then 'expert system advice' from the expert system to the client.

These problems have led to the basic model of knowledge engineering shown in Figure 1 being greatly extended by the introduction of interactive knowledge acquisition and encoding tools allowing the expert to transfer knowledge directly to the system without an intermediary (Boose & Gaines 1988) as shown in Figure 3. It should be noted that such systems support the roles of knowledge engineers and increase their productivity rather than replacing them entirely. The tools can greatly reduce the need for the knowledge engineer to act as an intermediary, but, in most applications, they leave a substantial role for the knowledge engineer.

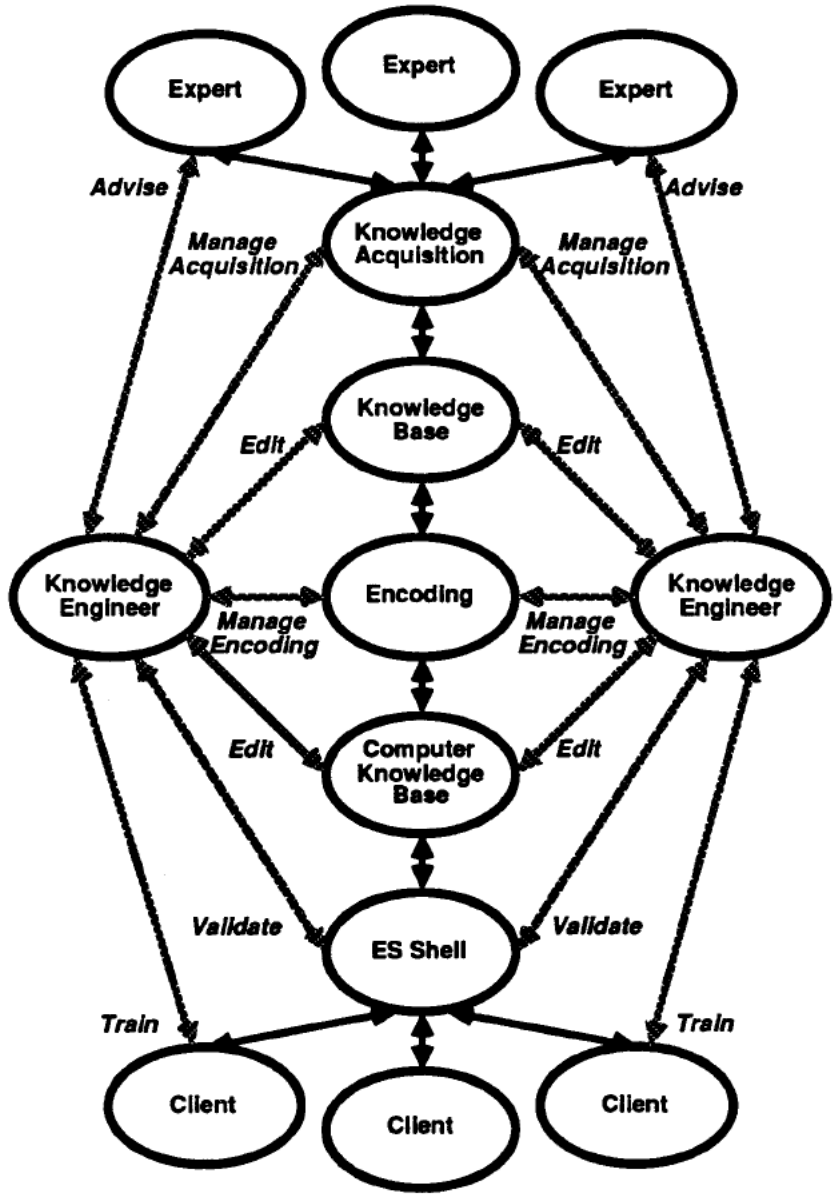

Fig.3 Knowledge engineers' roles in interactive knowledge acquisition

As shown in Figure 3, knowledge engineers can:

- Advise the expert on the process of interactive knowledge elicitation
- Manage the interactive knowledge acquisition tools, setting them up appropriately
- Edit the unencoded knowledge base in collaboration with the expert
- Manage the knowledge encoding tools, setting them up appropriately
- Edit the encoded knowledge base in collaboration with the expert
- Validate the application of the knowledge base in collaboration with the expert
- Train the clients in the effective use of the knowledge base in collaboration with the expert by developing operational and training procedures

This use of interactive elicitation can be combined with manual elicitation and with the use of the interactive tools by the knowledge engineer rather than, or in addition to, the expert. The knowledge engineer can:

- Directly elicit knowledge from the expert
- Use the interaction elicitation tools to enter knowledge into the knowledge base

Figure 3 shows multiple knowledge engineers since the tasks above may require the effort of more than one person, and some specialization may be appropriate. Multiple experts are also shown since it is rare for one person to have all the knowledge required, and, even if this were so, comparative elicitation from multiple experts is itself a valuable knowledge elicitation technique (Boose 1987, Shaw & Woodward 1988).

Validation is discussed in more detail in a later section. It is shown in Figure 3 as a global test of the shell in operation with the knowledge base, that is of overall inferential performance. However, validation may also be seen as a local feature of each step of the knowledge engineers' activities: the experts' proper use of the tools needs validation; the operation of the tools themselves needs validation; the resultant knowledge base needs validation; and so on. Attention to quality control through validating each step of the knowledge acquisition process is key to effective system development.

Another trend in knowledge-based systems is that the distinction between 'experts', 'knowledge engineers' and 'clients' is often one of variable roles rather than hard and fast job definitions. The expert may be a major client for the system, or may play a major part in knowledge engineering, or both. Some clients may customize or update the system in the role of knowledge engineers or extend its capabilities in the role of experts.

VIRTUAL MACHINE ARCHITECTURE OF KNOWLEDGE-BASED SYSTEMS

The significance of the knowledge acquisition system and knowledge-based system support environment in providing tools for the knowledge engineer is apparent in the discussion above. The quality of these systems is very important in the development of knowledge-based systems, particularly in supporting the interaction between knowledge engineers and experts. The relations between these systems, the shell, and other supporting technology is best seen by putting them in the context of the complete virtual machine hierarchy for knowledge-based systems. Figure 4 shows significant levels of this hierarchy:

- *Machine Architecture*: at the lowest level of the hierarchy is the machine on which the knowledge-based 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.
- *Operating System*: at the second 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.

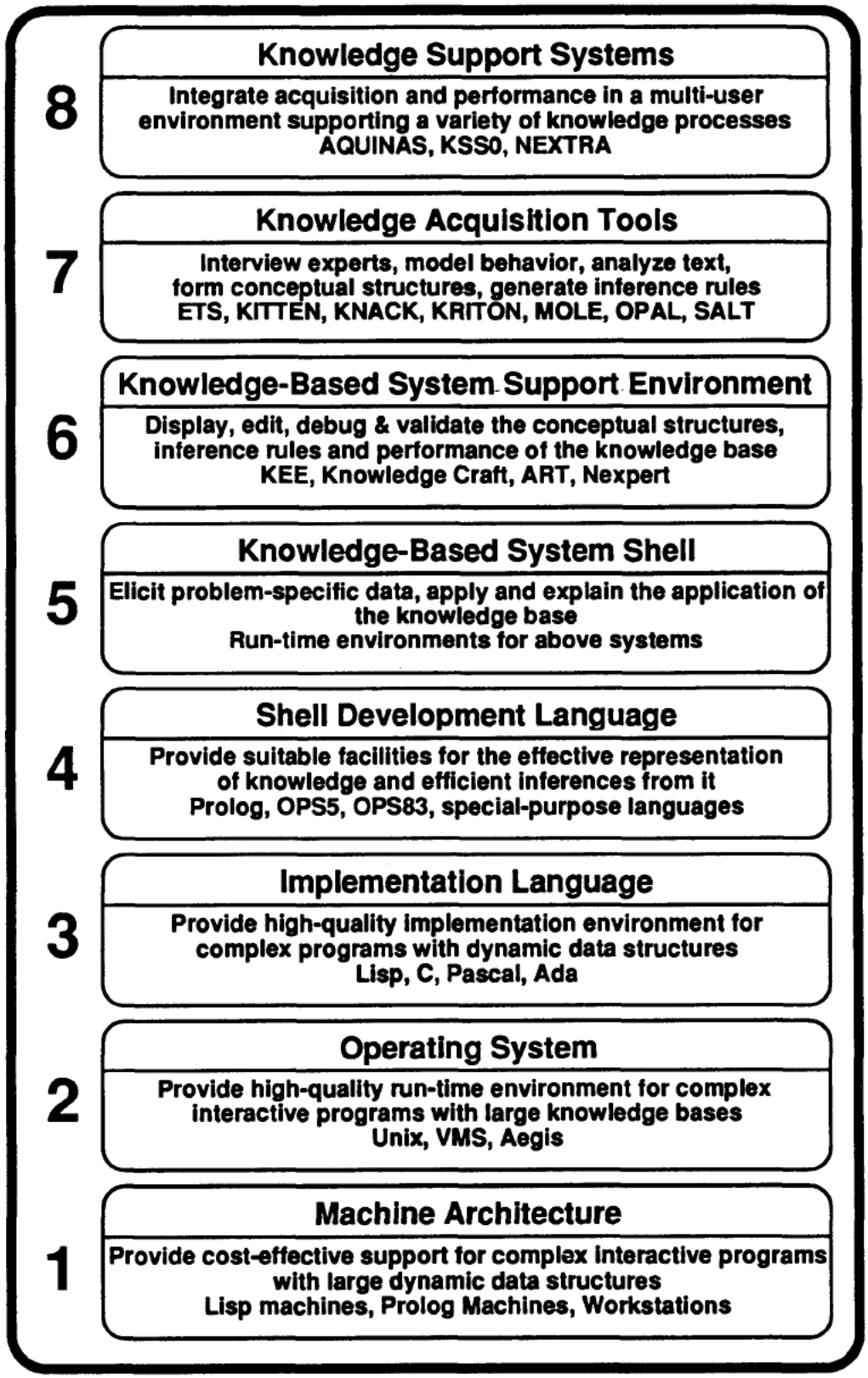

Fig.4 Virtual machine hierarchy for knowledge-based systems

Implementation Language: at the third 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. In particular the object-oriented versions of these languages are well-suited to knowledge-based system implementation.

- *Knowledge-Based System Shell Development Language*: at the fourth level of the hierarchy is the language in which the knowledge-based system shell is written, generally a special-purpose environment for coping with knowledge representation and inference.
- *Knowledge-Based System Shell*: at the fifth level of the hierarchy is the expert system shell as a run-time environment 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.
- *Knowledge-Based System Support Environment*: at the sixth level of the hierarchy is the equivalent of the Application Programing Support Environment (APSE) in conventional systems. The Knowledge-Based System Support Environment provides editing, browsing and debugging tools for the knowledge base. This level encompasses the major differences between the high-quality knowledge-based system shells that support major system development through extensive support tools, and the simpler shells that do not.
- *Knowledge Acquisition Tools*: at the seventh level 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. Automation of systems analysis is more significant for knowledge-based systems than for conventional data processing because they tend to be targeted on narrow areas of in-depth specialist knowledge, rather than broad general problems. In conventional terms, knowledge-based systems are satisfying the requirement for highly customized data processing based on idiosyncratic operations rather than general algorithms.
- *Knowledge Support Systems*: at the top of the hierarchy represent the objectives of recent research in knowledge acquisition that focuses on the integration of acquisition and performance systems to provide an environment supporting a wide variety of knowledge processes.

The hierarchy of Figure 4 highlights some of the problem that system managers often face in establishing development capabilities for knowledge-based—should they use Lisp machines?—should they train their staff in Lisp or Prolog?—how does a knowledge-based system shell compare with existing programing languages?—what is knowledge engineering and how does it differ from systems analysis? The problem is that 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 knowledge-based 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.

The virtual machine hierarchy for computing is very complex with many branches. Figure 4 focuses on a path through the hierarchy that links the basic hardware and software to knowledge-based systems and knowledge acquisition issues. It omits other parts of the hierarchy that raise other significant integration issues for knowledge-based systems. The remainder of this paper treats these integration issues in a principled fashion from which design rules may be derived.

INTEGRATION ISSUES IN THE USER COMMUNITY

The analysis of Figures 1 through 4 and the discussion of the previous sections raises a number of major integration issues in knowledge acquisition for knowledge based systems. Figure 5 shows the issues related to user roles in terms of the major distinctions involved.

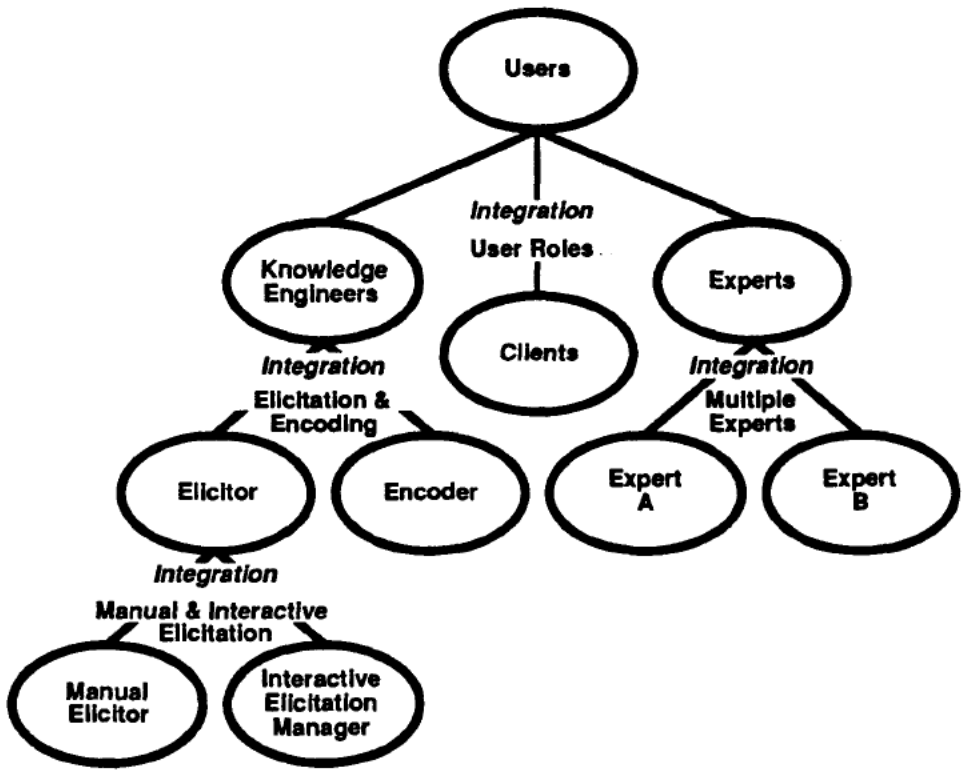

Fig.5 Major distinctions in user roles

In most current systems the knowledge acquisition tools, support tools and applications interface were designed separately and operate in very different ways on the assumption that they will be used by experts, knowledge engineers and clients, respectively. However, this diversity can be very confusing if users are flexible in their roles and switch between them. In addition, the complexity of the knowledge engineering role suggests that a variety of specializations will develop in building complex systems that will result in further roles and role switching. This leads to the design requirement:

Integration of User Roles: The user interface to a knowledge based system should be uniform and consistent across all user roles including expert, knowledge engineer and client.

However, it is important to note some features of the differentiation of roles:

Differentiation of Expert and Client Roles: The user interface to the clients should take into account the need to ensure that they use the terms in the same way as the expert. This may involve training procedures or the use of a different vocabulary.

Differentiation of Expert and Knowledge Engineer Roles: The user interface to the experts should structured to minimize the amount of knowledge required to operate it. Experts should be able to go beyond this minimal knowledge and operate in a knowledge engineering role, but should not be *required* to do so. The distinction between expert and knowledge engineer roles should be one of conceptual frameworks, not one of technologies.

The switch from manual to interactive elicitation has focused attention on the significance of the encoding process previously carried out tacitly by the knowledge engineer. Indeed, one of the problems in manual knowledge engineering may have been confusion between the knowledge representation meaningful to the expert and that usable by the expert system shell. A strong case has been made in manual elicitation for the use of mediating knowledge representations that are meaningful to the expert with encoding for the shell being treated as a separate phase (Johnson 1987). Such a differentiation is analogous to that between high level languages and machine code in conventional data processing. It may be seen as a weakness of current shell support environments that a sufficiently high level of knowledge representation is not available. The transition from Lisp code through OPS5 production rules through frames to generalized objects may be seen as the evolution of higher level knowledge representations.

The sheer variety of possible representations of knowledge that are meaningful to people makes it unlikely that the gap between a natural representation in elicitation and a computable one after encoding will ever be completely closed. Hence, as before, design requirements capturing both integration and differentiation are required:

Integration of Elicitation and Encoding: The knowledge representation processible by the shell in a knowledge based system should be as directly meaningful as possible to an expert.

Differentiation of Elicitation and Encoding: Intermediate knowledge representations should be used in elicitation for a knowledge based system if the shell representation is not directly meaningful to an expert.

The knowledge engineer's interviewing role is significant even when interactive elicitation is being used. The knowledge engineer gains information in discussion that may be used to advise the expert in the use of the tools, used to set up particular tools for elicitation, or loaded directly into the system. The system should support these various processes where appropriate:

Integration of Manual and Interactive Elicitation: Make provision for the knowledge engineer to use knowledge elicited from the expert in a variety of ways to enhance the interactive elicitation process.

And it should not be assumed that completely automatic elicitation is always possible:

Differentiation of Manual and Interactive Elicitation: The reduction in load on knowledge engineers through interactive elicitation should be used to enable them to be more effective in ensuring that knowledge not captured interactively is still transferred to the system.

Elicitation from multiple experts raises many problems of dealing with different vocabularies, contradictions, and so on. The system must make provision, as does AQUINAS (Boose & Bradshaw 1987) both for integrating diverse sources of knowledge and for making contradictory sources of advice available if required. The design requirements are:

Integration of Multiple Experts: Provide tools for analyzing and expressing similarities and differences between different experts' knowledge to enable it to be combined when similar and a consensus to be negotiated when different.

Differentiation of Multiple Experts: When a consensus is not reached the system should be able to use contradictory sources of knowledge in generating advice by showing dissenting opinions.

The distinctions shown in Figure 5 can be extended further. For example, there may be major differences between clients making a differentiation between interfaces appropriate. However, the major issues relevant to knowledge acquisition are captured in the design considerations discussed.

INTEGRATION ISSUES IN THE KNOWLEDGE-BASED SYSTEM SUPPORT ENVIRONMENT

The significance of the knowledge-based system support environment shown in Figure 4 in providing tools for the knowledge engineer is apparent in the discussion above. The quality of this environment is very important in the development of knowledge-based systems, particularly in supporting the interaction between knowledge engineers and experts. Figure 6 shows some of the major issues in the support environment in terms of the distinctions involved.

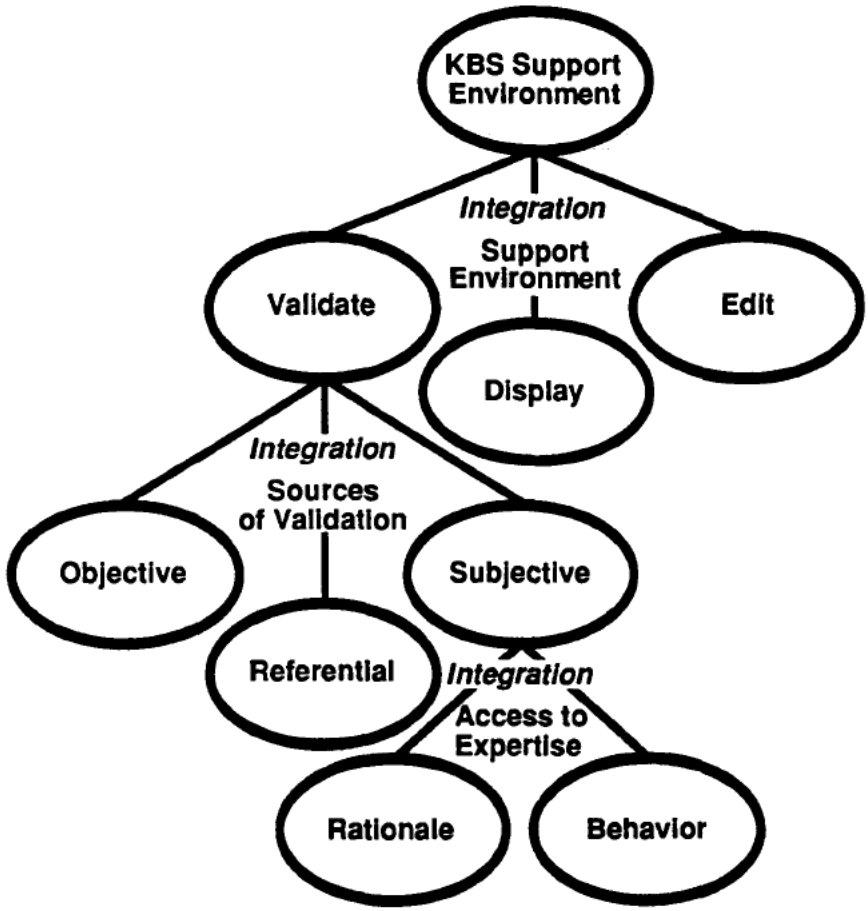

Fig.6 Major distinctions in the expert system suppport environment

The basic tools required are those for editing, displaying and validating the knowledge base. It is important that these tools present a uniform and consistent view of the knowledge base and that they aid consistency maintenance in the knowledge base during editing. The design requirements are:

Integration of Support Environment: Provide tools for editing, displaying and validating the knowledge base which are mutually self-consistent and, if possible, automatically propagate edits to maintain consistency or, otherwise, show clearly where parts of the knowledge base have become inconsistent.

The limited converse is that different perspectives should continue to be available.

Differentiation of Support Environment: Support whatever variety of perspectives on the knowledge base is necessary to ensure users in different roles have access that they find meaningful.

Validation is a complex topic in its own right (Shaw & Woodward 1988, Benbasat & Dhaliwal 1988), and it is important that a variety of tools are provided to support different forms of validation. The objective validation of the knowledge base in terms of its performance, its subjective validation in comparison with experts, and its referential validation in comparison with other knowledge sources should be supported. Objective validation may require significant case histories that can be collected only from an operational system in actual applications. Referential validation may require deep analysis of derivations of the knowledge base from other knowledge bases. There is a natural separation in subjective validation between experts' direct agreement that their rationale corresponds to that of the knowledge base, and observed correspondence between their behavior and that of the system based on the knowledge base. Both types of validation should be supported. The design requirements are:

Integration of Sources of Validation: Provide tools for validating the knowledge base both in terms of its absolute performance and in comparison to experts.

Objective validation may require significant case histories that can be collected only from an operational system in actual applications.

Differentiation of Sources of Validation: The tools for subjective and objective validation may need to be substantially different and operate over different timescales.

The validation of knowledge-based systems has all the dimensions of normal scientific validation together with those concerned specifically with the emulation of expertise. As shown in Figure 7, the system can be validated through four primary pathways:

- Through its correspondence to actual events checked against empirical data;
- Through its efficacy in applications checked against actual applications;
- Through the adequacy of its foundations checked against other disciplines corresponding to the derivation of presuppositions from other disciplines—this is sometimes termed 'deep knowledge';
- Through the comprehensibility of explanations given checked against the relevant reference community—corresponding to explanation and communicative action.

Fig.7 Dimensions of validation of knowledge-based systems

The expert systems paradigm introduces the two additional pathways shown on the right of Figure 7 by taking into account the possibility that there may be "experts" who are competent in applications and explanations:

- Through its correspondence to expert performance checked against the behavior of experts in applications
- Through its correspondence to expert knowledge checked against the rationale of experts in communicating with their reference community

This diagram may be iterated a number of times to show the source of the knowledge structures of experts from empirical data and other disciplines, introducing considerations of the validity of experts and reference communities. Similarly, it may be iterated to show the way in which other disciplines in turn are validated against their own empirical data, deeper foundations, applications and reference communities. There is no intrinsic conflict between the additional pathways of validation introduced and the positivist viewpoint of science emphasizing the primacy of empirical data. However, there is a major increase in the realism of this model in modeling the actual processes of knowledge acquisition rather than the normative criteria adopted in 'science' as ultimate arbitrators.

The differentiation in subjective validation between experts' direct agreement with the knowledge base and correspondence between their behavior and that of the system based on the knowledge base leads to further design requirements:

Integration of Knowledge and Performance Evaluation: Provide tools for enabling experts to evaluate the knowledge base directly and for comparing system and expert behavior in advising clients.

It is possible for there to be contradictions between these two forms of evaluation (Michalski & Chilausky 1980).

Differentiation of Knowledge and Performance Evaluation: Take into account that there may be contradictions between experts' evaluations of knowledge and the resulting correspondence in performance.

INTEGRATION ISSUES IN KNOWLEDGE ACQUISITION

The significant features of the knowledge acquisition component of Figure 3 are best analyzed by considering the natures of the classical and expert system design paradigms. These correspond to a large extent to the differentiation between objective and subjective validation shown in Figure 6. The center column of Figure 8 shows a system for which a decision or control system is to be designed together with sources of knowledge about the system.

The left hand column shows the use of these sources in the classical system design paradigm:

- Knowledge of past case histories is used to select a class of system models
- The information required to discriminate within this class determines how the system should be instrumented for data acquisition
- Data is collected from the system through the instrumentation
- A model is identified from the model class which best fits the data
- This model is used to design a decision or control system for optimal performance

The expert systems paradigm may be seen to be particularly applicable when it is not possible to model the system but there is an alternative source of data available because human operators are able to performance the decision or control task. The right hand column shows the use of knowledge sources in the expert system design paradigm:

- Structured interviewing may be used to acquire knowledge directly from the operators
- Behavior modeling may be used to identify the operators' strategies even if they are unaware of them or give incorrect ones in interviews
- Text analysis may be used with instructional material such as the operators' manuals
- Reasoning by analogy may be used based on the case histories without the data collection of the classical system design paradigm

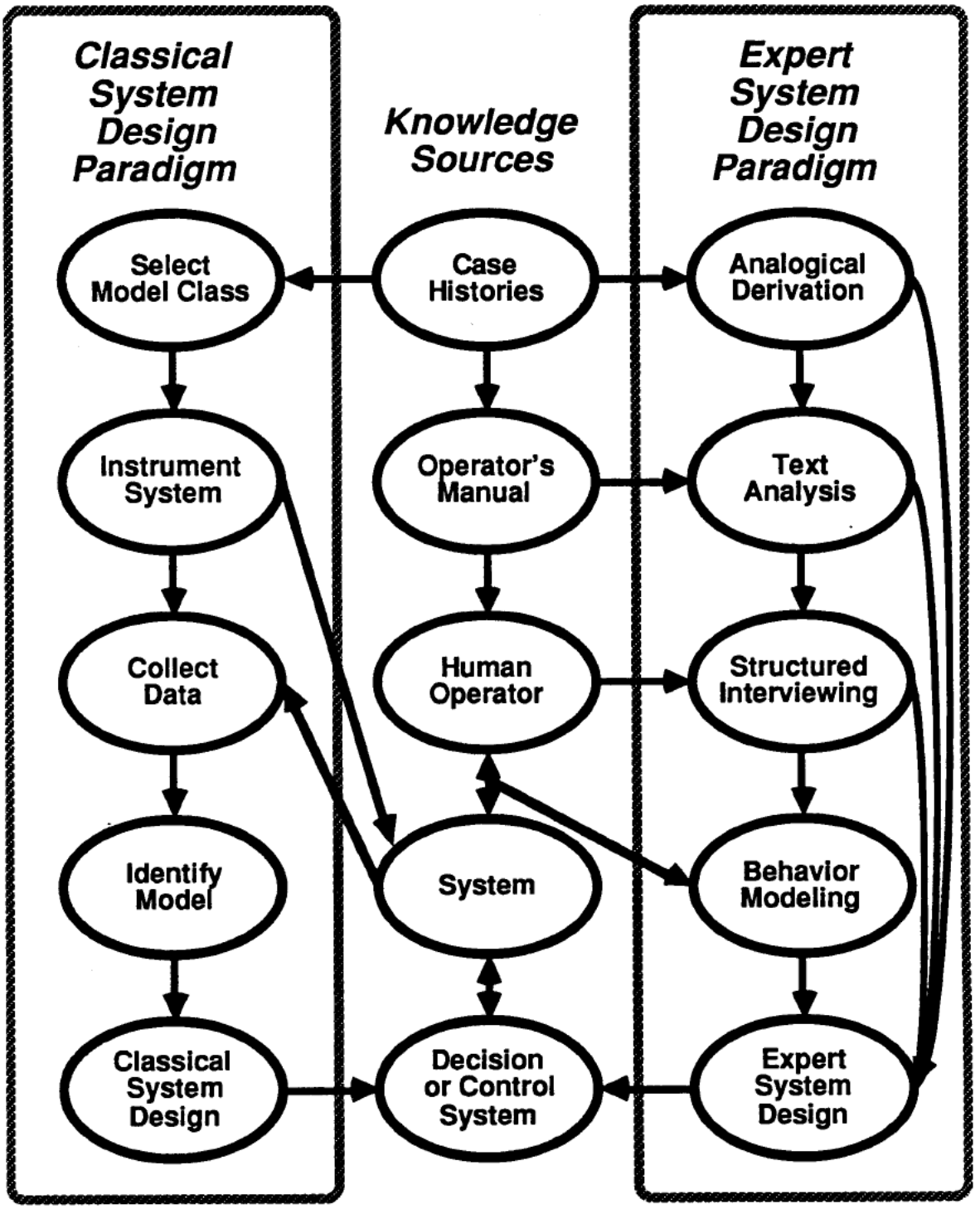

Fig.8 Classical and expert system design paradigms

Note that the classical and expert system design paradigms, and the various acquisition techniques for knowledge-based systems, need not be regarded as competitive. They may all be used in the system design, and it may be that a collection of weak methods is necessary because there is no overall strong method that can be applied. The resultant integration issues are analyzed in Figure 9 in terms of the distinctions involved.

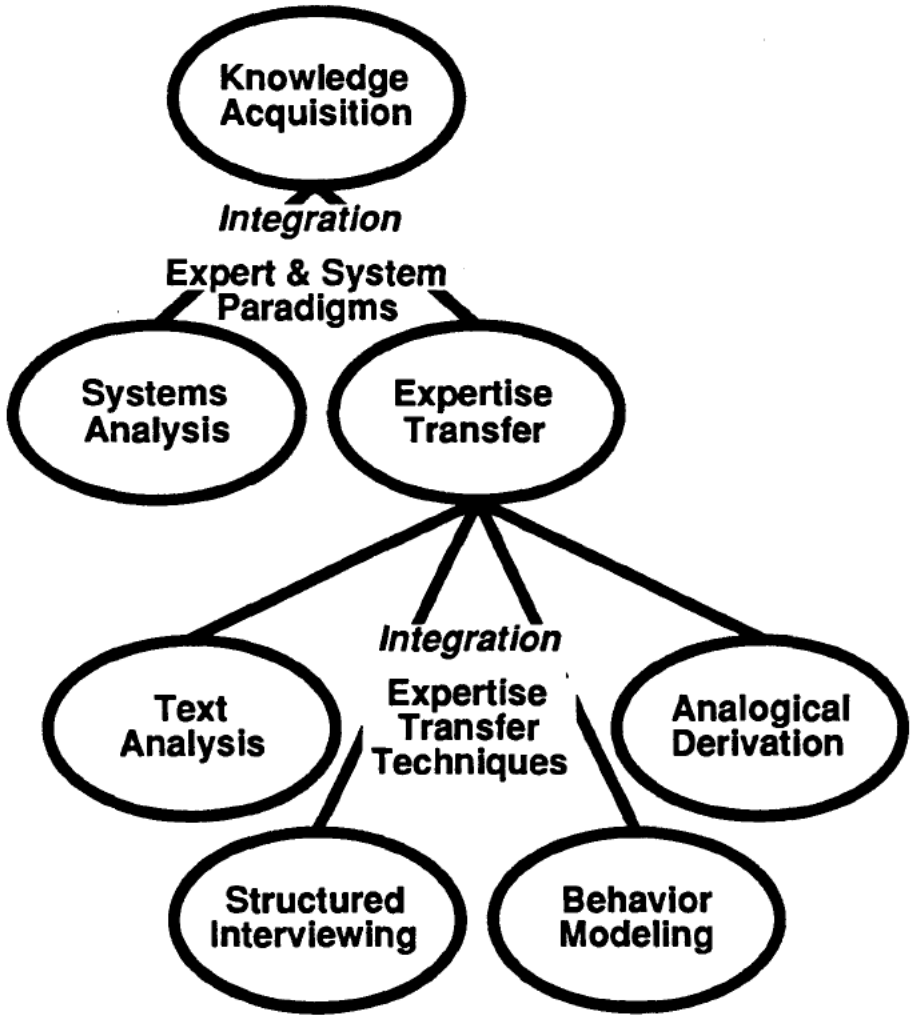

Fig.9 Major distinctions in knowledge acquisition approaches

The integration of the classical systems analysis with expertise transfer techniques is a major task. The primary problems currently are in the widely different forms of knowledge representation usually involved, and these will be noted in a later section in terms of integration of conventional and knowledge-based system components. However, at a knowledge acquisition level the problem can be seen as one of the imbalance in the types of knowledge structures studied in depth to date. There has been an imbalance favoring symbolic knowledge to redress the past emphasis on quantitative methods. This needs redressing with the knowledge structures underlying arithmetic being integrated with those underlying non-arithmetic symbol manipulation. The design requirements are:

Integration of Systems Analysis and Expertise Transfer: Provide knowledge representation that is adequate to encompass quantitative and qualitative data and reasoning.

The success of the expert systems approach should not detract from the long term significance of building deep models of the system itself. The objective performance evaluation of expert systems already discussed can provide data on deeper issues. It corresponds to learning from experience in human experts.

Differentiation of Systems Analysis and Expertise Transfer: Provide for system analysis of performance data from knowledge-based systems in applications to improve their performance and the depth of their models.

The integration of various techniques for expertise transfer is already a theme at the knowledge acquisition workshops. For example, case history and text analysis may be used to provide terms and concepts that aid interviewing and behavior description. The design requirements are:

Integration of Expertise Transfer Techniques: Provide intermediate knowledge representations that allow for transfer between different sources of knowledge.

The limited converse is that the different sources of knowledge, and the possibilities of contradictions between them, should be recognized.

Differentiation of Expertise Transfer Techniques: Keep track of the different sources of knowledge within the system and allow for contradictions in inferences from it.

INTEGRATION ISSUES OVERALL

Figure 10 combines the structures of Figures 5, 6 and 9 into an overall analysis of the distinctions underlying integration issues in knowledge acquisition for knowledge based systems. It raises some additional integration issues.

At the middle of Figure 10 a distinction has been made between the education, application and explanation components of the knowledge-based system shell. The 'educational' aspects of the operation of the shell are important in acquisition because they support the clients in using the terminology and distinctions acquired from the expert in the same way as the expert intends. It is important in acquiring attributes from experts to acquire not only the terminology and exemplary cases but also the decision procedures whereby the values of entities on these attributes are decided. These procedures can be used to develop tutorial material for clients. This will often consist of illustrative examples presented through appropriate media rather than detailed logical procedures in textual form.

The knowledge required for explanation presents similar problems because it may be more extensive and more demanding than that required for application (Clancey 1987). Explanation facilities in most shells are very weak currently but as they develop to take into account the more advanced analyses of the nature of explanation (Hughes 1986) the requirements for explanation-oriented knowledge elicitation will increase.

The design requirements are:

Integration of Education, Application and Explanation: Education and explanation should be treated as a major supporting components to the application and knowledge acquisition should be targeted accordingly.

In many areas such as validation there are major differences between education, application and explanation that need to be taken into account.

Differentiation of Education, Application and Explanation: The need for separate validation of performance in education, application and explanation should be taken into account.

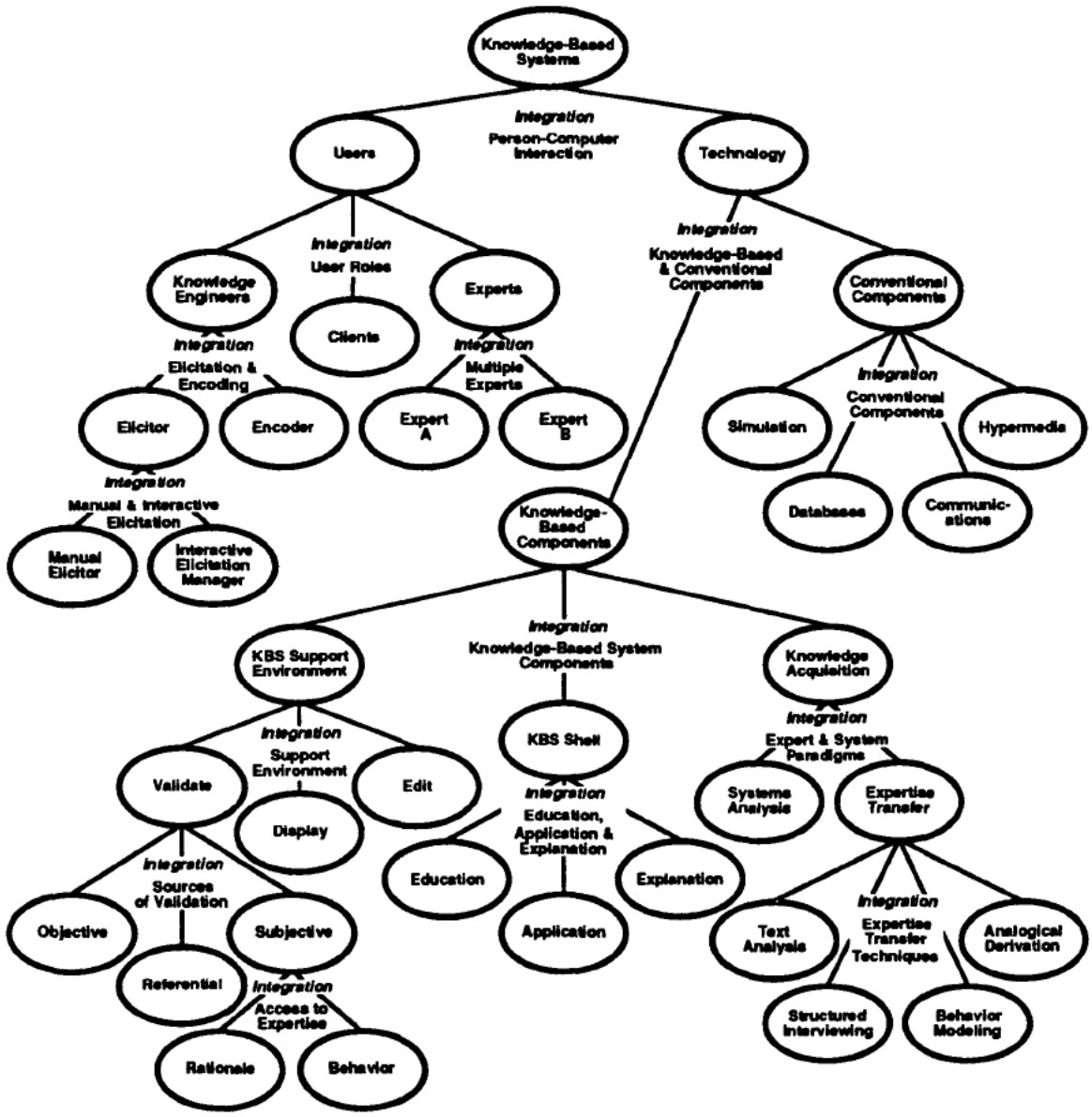

Fig.10 Overall structure of distinctions relating to knowledge acquisition integration in knowledge based systems

At the top right of Figure 10 some of the conventional components of a data processing system have been introduced and a distinction has been made between them and the knowledge-based components. The term 'knowledge' has tended to be used in a specialized way in the knowledge based system context as 'knowledge which is symbolic and computationally processible.' Much significant knowledge does not come within this narrower definition yet is still important in knowledge based systems. This leads to a need to integrate several conventional data processing technologies, for example:

- *Simulation* is important in showing the time sequences of the behavior of dynamic systems
- *Databases* provide a convenient means for storing large quantities of data with efficient retrieval, and can be used to build a variety of object hierarchies in knowledge bases from different perspectives

- *Communications* is important in allowing interaction between experts at remote sites during group elicitation processes
- *Hypermedia* provide a means for storing and retrieving knowledge even if it is not computationally processible and, in particular, provide for knowledge transfer through video material allowing mimicing at a non-verbal level

The design requirements are:

Integration of Knowledge-Based and Conventional Components: Extend the concept of a knowledge-based system to include the knowledge structures of conventional data processing.

The only danger is that some essential features of knowledge based systems are lacking in existing forms of data processing.

Differentiation of Knowledge-Based and Conventional Components: Extend conventional data processing to include features of knowledge based systems such as explanation.

At the top of Figure 14 the final integration is between the users and technology, that is the achievement of effective person-computer interaction. This is a major problem in all systems of the complexity of that shown in Figure 14, and is already apparent in knowledge acquisition even at the level of integration of Figure 11. Intelligent user interface management systems for knowledge acquisition tools has already become a significant research topic (Kitto & Boose 1986). The gap between functionality and usability can be very difficult to bridge, and it is easy to create multi-source, multi-format, multi-window operating environments that present great power but utter confusion to the user (Gaines & Shaw 1986). The design requirements are:

Integration of Users and Technology: Provide a uniform and consistent user interface to the knowledge based system that maximizes the usability of the available functionality.

However, the interface must reflect reality and not be cosmetic.

Differentiation of Users and Technology: Do not use syntactic sugar and other interface tricks to give the user the impression of greater capabilities than exist in the system.

CONCLUSIONS

Tools and techniques for knowledge acquisition for knowledge based systems need to be integrated with the overall system and not treated as separate components. The variety of sources of knowledge, representations and applications within the system, and user roles, makes such integration complex. The complexity raises many issues not generally thought of as part of knowledge acquisition yet fundamentally significant in extending and integrating acquisition tools. This paper has given a preliminary analysis of the structure of knowledge-based systems as it relates to the integration of acquisition. It is intended that this structure be used as a basis for system design and it will be extended to form a 'requirements specification' for knowledge based systems integrating acquisition with other aspects of their operation.

ACKNOWLEDGEMENTS

Financial assistance for this work has been made available by the Natural Sciences and Engineering Research Council of Canada. I am grateful to many colleagues for discussions over the years that have influenced this paper. In particular I would like to thank John Boose, Jeff Bradshaw, Mildred Shaw and Joan Vickers for access to their own research and many stimulating discussions.

REFERENCES

Benbasat, I. & Dhaliwal, J.S. (1988). A framework for the validation of knowledge acquisition. **Proceedings of the Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop**. pp.1-1-1-18. Banff (November).

Boose, J. H. (1987) Rapid Acquisition and Combination of Knowledge from Multiple Experts in the Same Domain. **Future Computing Systems 1**(2) 191-216

Boose, J.H. & Bradshaw, J.M. (1987) Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for knowledge-based systems. **International Journal of Man-Machine Studies 26,** 3-28.

Boose, J.H. & Gaines, B.R., Eds. (1988). **Knowledge Acquisition Tools for Expert Systems**. London, Academic Press.

Clancey, W.J. (1987). From GUIDON to NEOMYCIN and HERACLES in twenty short lessons. van Lamsweerde, A. & Dufour, P. (Eds). **Current Issues in Expert Systems**. pp.79-123. London, Academic Press.

Gaines, B.R. (1987) An overview of knowledge acquisition and transfer. **International Journal of Man-Machine Studies 26**(4), 453-472 (April).

Gaines, B.R. & Boose, J.H., Eds. (1988). **Knowledge Acquisition for Knowledge-Based Systems**. London, Academic Press.

Gaines, B.R. & Shaw, M.L.G. (1986). Foundations of dialog engineering: the development of human-computer interaction part II. **International Journal of Man-Machines Studies, 24**(2), 101-123 (February).

Hughes, S. (1986). Question classification in rule-based systems. Bramer, M.A., Ed. **Research and Development in Expert Systems III**. pp.123-131. Cambridge: University Press.

Johnson, N. (1987). Mediating representations in knowledge elicitation. **Proceedings of the First European Workshop on Knowledge Acquisition for Knowledge Based Systems**. pp.A2-1-10. Department of Computer Science, Reading University.

Kitto, C.M. & Boose, J.H. (1986) Heuristics for expertise transfer: the automatic management of complex knowledge acquisition dialogs. Boose, J.H. & Gaines, B.R. (Eds) **Proceedings of the First AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop**. pp.22-0-22-17. Banff (November).

Michalski, R.S. & Chilausky, R.L. (1980). Knowledge acquisition by encoding expert rules versus computer induction from examples—A case study involving soyabean pathology. **International Journal of Man-Machine Studies 12**, 63-87.

Shaw, M.L.G. & Gaines, B.R. (1987) KITTEN: Knowledge Initiation & Transfer Tools for Experts & Novices. **International Journal of Man-Machine Studies 27**(3), 251-280 (September).

Shaw, M.L.G. & Woodward, J.B. (1988) Validation in a knowledge support system: construing and consistency with multiple experts. **International Journal of Man-Machines Studies 29**, 329-350.