<u>Tutor</u> - this system focuses upon the complete educational task - which includes the teaching of both the factual information and the conceptual relationships needed to relate the facts to their use. This role is the most difficult and requires the most "human-like" qualities - understanding of the subject and the student, and how best to teach the material.

No actual training system will ever be completely identifiable as fulfilling one of the above roles. We now list four types of structures which can contain elements of the above roles.

1. <u>review/refresh</u> - to exercise and remind the trained user of appropriate information. This might consist of a question and answer dialogue in the context of solving particular problems, with advice upon techniques to apply at each stage.

2. <u>drill device</u> - to provide practice in the knowledge and reasoning methods taught in the course work. The student will solve problems presented by the system, and receive help, examination and criticism.

3. <u>skill enhancement</u> - to augment the student's basic understanding with the expert's conception of the rules used in the domain; and to train the student in the formal reasoning methods. This system will present new information and lead students to learn new problem solving skills.

4. <u>experimental laboratory</u> - to provide a framework supporting problem solving through simulation of specific example systems. The expert tutor system can be used for explaining the effects occurring in the simulation. The student will be able to make changes to models of physical systems, and hence learn the system's behavior in a risk-free environment.

Structures 1 and 3 contain the greatest elements of the Adviser role. Similarly, structures 1 and 4 associate with the Demonstrator role, structures 1 and 2 with the Examiner role, structures 1,2 and 4 with the Exerciser role and structures 1,2 and 3 with the Tutor role. Of course, any actual implementation will favor some roles over others. At present, there is little research or theory to guide designing systems to meet particular goals.

Structures 1, 2 and 3 are reasonably similar and are likely to be satisfied by the same or roughly similar systems. Structures 2 and 3 are likely to be of maximal utility. Structure 4 is likely to be the hardest to implement, due to the need to have a domain simulator.

4.2 Potential nature of student-machine interactions (Concepts of use)

In this section, we discuss some of the alternative concepts for student-machine interactions. We also include a discussion of the types of input into the system that the student can make, and overall principles related to the interactions.

There are degrees of interaction between the student and tutorial system, from fully passive, to cooperative discussion. Some possibilities are:

problem explication - the expert system takes a particular case and explains its reasoning and solution in appropriate detail. The student is largely passive, except to request more detailed explanations. This system resembles the typical frame-based computer-aided instruction systems.

problem exploration - the student examines the reasoning process used by the system on a particular case. In this instance, the student can choose the next topic to be presented. The student may be able to ask how and why particular actions were taken, and what would have been the effect of changing various input data. Aspects of this type of interaction were researched in the SOPHIE and MYCIN/TEIRESIAS systems.

student solves problem with passive system criticism - the student attempts to solve a problem in a similar way to the expert system. The system can check the student's deductions for correctness by comparing them with its own solution. It can provide hints about what to do next and may report what it attempted to do at the current point in its solution. This approach resembles that used in the BIP system.

student solves the problem with active system criticism - much like the above, except that the system also attempts to induce what reasoning the student is applying and which sub-problems the student is solving. The GUIDON process resembles this somewhat (it also helps to direct the problem solving). The SOPHIE system also encompasses some of this behavior, in that the system calculates what deductions are feasible given the student's current knowledge.

system queries the student - the system may ask the student for a complete problem analysis, analysis of a subproblem, the appropriate next step or for a particular deduced fact. (The student must do the reasoning independent of the computer system.) It compares the student's responses to its solutions and knowledge base. Both EXAMINER and GUIDON have some of these abilities.

There are also varying degrees of language use in the discourse between student and machine. The major possibilities are:

command word or prompted list - the student enters his/her inputs from a list of acceptable command words or phrases (such as "tell more"). This list may be constant over the whole dialogue, or the system may issue a list (ie., multiple choice) at appropriate moments. (GUIDON)

stylized English - the input is in English sentences, but they must conform to some specified (perhaps loosely) sentence structures. Multiple sentences must conform to a discourse structure. (ACE)

<u>keyword based semantic parsers</u> (Bur79b) - the system makes strong assumptions about what must have been said, based upon its knowledge of the subject domain and what certain key words and phrases mean. This system is capable of resolving some pronoun and other implicit references, as well as some spelling corrections. (SOPHIE)

<u>limited domain discourse analysis</u> (Bob77) - the power of this approach is similar to that of the keyword based semantic parser. In this case, the systems are augmented with models of the syntactic structure of natural language, semantic models of word meaning and discourse models (that is, models of how people communicate and about what). In consequence, they are less likely to mis-interpret a student's input, at the cost of being more rigid about what is acceptable input, and being more expensive to develop.

<u>full natural language understanding</u> - This would require the ability to understand changes of topic, to accept grammatically mis-formed sentences, to resolve the references of the various pronouns and noun phrases in the input, as well as to infer the implied information generally left out of normal discourse. No examples of such systems exist.

Lastly, we comment on desirable principles in a teaching system from the point of view of the student-machine interaction. They are:

1. The system should be responsible for a focussed examination of the issues. That is, it should not consider topics randomly, nor should it allow much deviation from the topic at hand. This serves to avoid confusing the student.

2. The system should be able to explain issues at various levels of detail, upon request from the student.

3. The system should cover the major issues, conclusions and rules used first, so as to highlight the basic elements of the solution.

4. The student should have active involvement in the dialogue process.

5. The student should receive feedback from the system on his/her performance.

6. Whatever language interface is chosen, it should be designed so that the student can easily learn its use, and thus be free to concentrate upon the subject material.

4.3 What features should be present in an ideal expert training system?

The ideal training system would be an infinitely wise, knowledgeable, and patient human being, but, as they are not usually available, we will have to settle for an inferior system. We list below the requirements needed in an ideal system, by category. Then, we consider which requirements are of importance for each of the five roles and four structures discussed in section 4.1. This consideration is summarized below in table 3.

A. Structural constraints

A1. There should be an underlying expert system capable of solving a class of problems at an expert level, and which embodies valuable expert knowledge.

A2. There should be an explicit overall educational strategy.

A3. The reasoning process used by the system should be suitable for use by a human being

A4. The expert system must be capable of analyzing and simulating models of domain processes or problems

B. Represented and accessible knowledge

B1. The knowledge base used by the expert system should be explicitly represented and be explainable to the student.

B2. The reasoning methods used by the expert system should be explicitly represented and explainable to the student.

B3. The educational methods used by the tutorial system should be explicitly represented and explainable to the student.

B4. The system should have a model of the student's factual and reasoning process knowledge.

B5. The system's domain model should be explicit and explainable.

B6. The system's problem solution should be explicit and explainable.

B7. The meta-level problem solving knowledge (ie., knowledge about what strategies to adopt and techniques to consider) should be explicitly represented and explainable to the student.

C. Tutorial methods

C1. The system should both accept and provide sample problems for analysis.

C2. The system should allow simultaneous solution of problems by both the student and the system.

C3. The system should train the student in both factual and procedural knowledge.

C4. The system should evaluate the student's level of understanding based on explicit questioning and implicit analysis of behavior.

C5. The tutor must be capable of providing both complete problems and appropriate subproblems for solution.

C6. The tutor must only provide information appropriate to the student's current level of understanding.

C7. The system can criticize the student's inputs.

C8. The system can induce the principles leading to the student's errors (an extremely advanced capability).

C9. The knowledge to be presented should be organized (ie., in a syllabus).

<u>Man-machine interaction</u>

D1. A mixed-initiative dialog structure should be used. But, as Bunderson (Bun69) has shown that system control of the session in learning situations gives better performance, the tutorial system should be responsible for directing the flow of the conversation.

D2. The student should be allowed to ask "What if ..." questions, and the system should be able to examine the changes in its problem analysis.

D3. The system should be queryable in a reasonable subset of natural English.

D4. The student can select the level of detail presented.

D5. The student has various aids available, such as hints and outright help.

	ROLES					STRUCTURES			
Reqt	Adv	Exer	Tut	Exam	Demo	Drill	Exper	Rev	Skill
A1	X		X		X		X		X
A2		X	X	X		X		X	X
A3	X		X		X			X	X
A4		X	X	X			X		
B1	X		X		X		X	X	X
B2	X	X	X	X	X			X	X
B3		X	X			X		X	X
B4	X	X		X	X	X		X	X
B5	X		X		X	X	X	X	X
B6	X	X	X	X	X		X	X	X
B7	X	X	X		X			X	X
C1	X		X	X	X	X	X		
C2		X	X			X		X	X
C3	X	X	X		X			X	X
C4	X	X	X	X	X			X	X
C5		X	X	X		X			X
C6	X		X		X		X	X	X
C7		X	X	X		X		X	X
C8			X	X		X			
C9			X		X			X	X
D1	X		X		X			X	X
D2	X		X		X		X		
D3	X	X	X	X	X	X	X	X	X
D4	X		X		X		X	X	X
D5		X	X	X		X		X	X

Table 3 - Consideration of requirements by role and structure

4.4 Applicable domains

This section discusses selecting an expertise domain as a basis for an expert training system. Also discussed are the high level attributes common to previously developed expert systems.

The first question is: for what types of training are these systems to be designed? This also relates to the envisaged type and expected number of students. In the most extreme of characterizations, there are two classes of students and systems. The first class is for a large number of students trained to a basic proficiency level in what might be considered a quasi-technical domain (such as automotive repair). This would have the benefits of both low (per student) development cost and a uniform national educational base. However, there are also a large number of trained experts, who could perhaps teach the knowledge more effectively. The second class is for a small number of students trained to expert performance levels in a highly technical domain (such as chemical structure analysis). Of course, there are intermediate situations, such as general medicine, which are somewhat less technical and have a wider base of applicability. (The medical field is, however, very large, and existing expert systems consider only very narrow specialities.)

The next question is: what is it that the system is supposed to teach? Certainly, the facts associated with the domain and the expert's knowledge of its relevant features are important. We think that the most important aspect is the type of reasoning that an expert uses in the solution of a problem. This poses the domain specific questions: What features does the expert look for first? What models of the domain are used? What is important at various times in the process? In essence, we would like the student to learn the decision-making process that the expert uses. This is especially important when dealing with heuristic domains, that is, those domains in which the expertise is largely intuitive. (There are only suggestions of theories on how this type of reasoning is done.) The other domains, those with formal theories, are likely candidates for automation in the next decade anyway.

So, what are the general type of domains which have a high dependence upon heuristic reasoning? Based upon the expert systems listed in section 2.3, the general types of intuition captured are:

trouble shooting/diagnosis - recognition of some failed process based upon observable symptoms (**MYCIN**)

identification/classification/evaluation - recognition of an object or situation based upon evidence (**PROSPECTOR**)

sequential planning - the recognition of the goals and the interactions of the constraints in various phases of a plan (**MOLGEN**)

organization - the spatial or temporal arrangement of objects/actions to meet externally imposed, logical and domain specific requirements (**R1**)

monitoring - insuring that a process conforms to acceptable models of that process (**VM**)

advice - suggestions directed to helping the user meet some goals

(SACON)

Lastly, we list some domains that are likely to be suitable areas for productive expert training systems. These areas are largely suggested by existing expert systems. They are:

quasi-technical - equipment repair (mechanical, automotive), electrical wiring, plumbing, farming, ranching

technical - medical, veterinary medicine, equipment repair (electrical, electronic), laboratory analysis (chemical, biological), exploration (mineral), structural analysis (electronic, mechanical, architectural)

4.5 Evaluation of alternative training system concepts

We now consider alternative concepts of expert and training systems, from the perspective of usefulness and ease of development. This discussion must be reasonably sketchy - there are many alternatives and a full analysis should consider each alternative as a part in a full system. Further, there are degrees of each capability. Below, we comment upon the general requirements associated with each alternative, and express our preferences among them. We note that not all of our preferences are compatible. Perhaps the major value of this section is as a guideline to the evaluation of and commentary upon training system concepts.

In section 2.5, we covered the distinctions between different types of expert systems. Based upon that discussion, we list below our preferences for the types. The principle focus of these preferences is upon an expert system whose performance can be matched by a trained human being. Factors which tend to rule this possibility out are: dependence upon complex analytic theories, large tables of data, unnatural reasoning structures or extensive searching among alternatives. Unfortunately, it is just these features which make most expert systems usable. (This is largely due to the well-understood abilities of a computer, such as speed and accurate bookkeeping, being used to replace the poorly understood abilities of an expert, such as judgement and insight. These allow him/her to simplify problems, isolate key elements, quickly induce possible solutions and avoid unlikely alternatives.)

heuristic models - as theory based models are probably best learned formally, although a training system can be used for providing drill
shallow models - of a simpler nature and less formal
weak models - as strong ones again depend more upon theory and less upon experience
reasoning based - as humans can not expect to duplicate a machine's computational performance
alternative based - so the students learn the major aspects of problems
categories - all seem appropriate, with diagnosis and advice the most likely, due to less of a dependence upon computational power

Not all of the potential training systems need an expert system whose reasoning can be learned and used by a human being. In particular, versions of the Examiner and Exerciser role only require a system capable of solving problems, not also explaining their solution. Hence, the actual suitability of any expert system will depend upon its intended use.

We now consider the training system alternatives. We initially focus upon the five roles, as described in section 4.1. We are somewhat limited in that we have not actually specified any particular system embodying the roles, hence will just consider general requirements. This discussion is a supplement to the requirements listed in section 4.3.

Adviser - this system has the greatest interface difficulties, in that it has to induce what the user needs advice on, which can cover a broad range of topics and viewpoints. Further, the advice might range from strategic suggestions, to comments on specific problems. At present, any usable system will need to be severely restricted in its range, perhaps to just a particular problem class seen from just a particular perspective. (This leads to the possibility of a whole class of Adviser programs covering a general subject area, the use of which is the speciality of a further Adviser program.) These systems will place heavy requirements upon the underlying expert systems, in that the Advisers need to access considerable (explicit) information regarding the domain's model, facts and methods.

The cost of an Adviser system is likely to be high, both for development and pre-use training. In development, the problems stem from acquiring two forms of expertise - on problem understanding and advice generation. (The problem understanding may be eliminated if the system constrains its area of expertise sufficiently.) The user pre-training will cover both problem identification (ie., is it suitable for the Adviser) and formulation (into a model suitable for use). The usefulness of Adviser programs, whether covering general topics or for specific techniques, is likely to be high. Unfortunately little research has been done on such systems (SACON,ADVISOR).

Demonstrator - this system is more easily developed, in that it merely demonstrates the solutions to problems solved by the associated expert system. The major problems consist of isolating and organizing significant information about the domain and problem solution, supplying appropriate detail and abstracting relevant meta-level information (such as motivations). Of course, much of this depends upon the expert system having this knowledge explicitly available. Not much research has been done on the Demonstrator role, but what remains is probably not excessive. Such a system would be easily used by a student, who may control the direction, pace and level of presentation, but who would otherwise be passive. This is also likely to be one of the more inexpensive roles to implement.

Examiner - the requirements upon this role vary according to the degree of its abilities. Simpler systems can merely present problems, and analyze simple answers (such as numbers or names). More complicated systems will accept more sophisticated answers, including relationships and descriptions, and will criticize the solutions on partial correctness or completeness grounds. A further capability is deducing the source of error. A simple system is likely to be highly constrained in its user interfacing, perhaps with inputs from a prompted list. An expert system will be needed to generate the solutions and records of deductions. Not much research has been done on Examiners in the context of expert systems, but for simpler systems, the CAI research on drill systems is applicable. The cost and difficulty of an Examiner system will vary, but is expected to be less than a Tutor or Adviser. It is not clear whether the Examiner role is useful in an isolated context - ie., whether the student has much to gain from examinations without instruction or criticism.

Exerciser - this can be one of the easier roles, depending upon the degree of interaction with the student. The system would require some knowledge of the student's abilities, and of curriculum structures, so that it could select appropriate problems. The amount of criticism and guidance given during the problem session will depend upon the amount of input provided by the student. The system can have a range of diagnostic abilities for analyzing erroneous solutions. The major difficulty involves recognizing what the student has solved correctly. Another factor is the degree to which an underlying simulator for the domain is needed - which depends upon whether the problem being solved is static (as in a medical case) or subject to experimental changes (as in an electronic circuit problem). The development and user training costs are likely to be lower than the Adviser type systems; yet, they can still reasonably satisfy training needs. Simpler forms of this role, such as drill systems, can be implemented without extreme difficulty and with only small amounts of innovation.

Tutor - this is the most comprehensive role. It covers teaching factual, structural and procedural concepts, and subsumes the other roles. Its tutorial requirements are substantial, and include: deduction of the student's development level, recognition of errors, deduction of the basis for misunderstandings, selection of appropriate material and selection of appropriate presentation techniques. It also requires significant attention to psychology - to match the dialogue to the general abilities and current mental state of the student, and to normal conversational postulates. To support these abilities, the role requires a full expert system, with accessible factual and reasoning knowledge databases. Research is not yet sufficiently advanced to allow a human expert more than a modicum of success with this role, let alone to provide for the development of a comprehensive machine-based tutor. However, the value of such a system is high, as a result of its well-rounded approach. The development cost of such systems will also be high, until further research succeeds in outlining their principles.

Overall, we feel that the Tutor role is the most desirable, and the hardest to develop, in that its capabilities are substantial and much research remains. The Adviser type system is the most immediately useful, but would also be difficult to develop. The Demonstrator and Exerciser roles seem to have the best mix of utility, supporting research and ease of development. Lastly, the Examiner system's usefulness varies directly with its capabilities - which, in turn, affect its development cost.

We now consider the structures into which to embed the roles. These are the four structures discussed in section 4.1.

review/refresh - this structure will probably achieve the greatest initial usage, in that it presumes previously trained users who merely need a review of particular information and methods. This simplification allows the system to use a presentation format. Difficulties involve the selection of appropriate material and presentation methods. The type of problems solved can be pre-selected by the user. An underlying expert, with natural reasoning methods, would be needed to initially generate the solutions, but would not be needed after that (although **GUIDON** uses **MYCIN** to constantly evaluate what the student could conclude given his/her current factual knowledge). This type of system would not need much further research and would not be as expensive, nor as difficult to construct, as the other structures.

drill device - this structure is also a likely early success, though there is little research on how they might be constructed. One problem concerns how to present complex problems for interactive solution over a computer-based medium. A second problem is developing methods for the computer to track, evaluate and criticize the student's solution, and to provide assistance when needed. These difficulties relate to the complexity of the presented problems, as when the student cannot solve the problem in just a few well-understood steps (as in an arithmetic drill program). The BIP system succeeds at this task, but only because of a curriculum of specially chosen simple problems. (It also aims at beginning, rather than expert level, computer programming students.) Additional complications depend upon the degree of criticism provided. All told, this structure is likely to be moderately difficult to develop – depending upon the problem level and amount of feedback provided.

skill enhancement – this is the most sophisticated structure, in that it has to both select and present new information. This is difficult to do automatically (ie., without specially tailoring each dialogue). It further requires more flexibility, in that the student should give feedback regarding his/her degree of understanding. A major difficulty is presenting the information effectively, which partly depends upon isolating key factors and relationships to previously learned knowledge. A sophisticated student model is needed to support these abilities. There is little research on the use of this structure at expert performance levels. The GUIDON research takes the view that the enhancement is based on adding, to the student's knowledge, the facts and decision procedures that were used by the expert in particular situations. This requires substantial support from the underlying expert system (explanation of reasoning) and the capability to detect gaps in the student's understanding relative to that of the experts. This structure still needs further research, and even then, is likely to be expensive and difficult to develop. However, the rewards are also greatest with this structure, because of its active instruction of the student.

experimental laboratory – this structure is the easiest to develop, in that it requires fewer tutorial capabilities: no student modelling, no error diagnosis and no material presentation. Depending upon the system goals, it might present problems or suggest hints on solution approaches. The burden of innovation would rest upon the student, and the system would be largely passive. The requirements upon the underlying expert system are greater, though. Most of the expert systems described in section 2.3 do not have enough knowledge to support simulation, let alone actually do the simulation. A few, such as SACON, are capable of analyzing problems given to them. Simulation is usually performed by programs quite unlike the expert systems (like the SPICE electronics simulator used by the SOPHIE system). A further problem is that simulation can only be provided for domains with functional theories, such as electronics, which relates structures to behaviors. Expert systems consulting on electronics have their main expertise on the design and organization of devices, rather than the actual electronics. Though this is all a bit pessimistic, the development of selected experimental laboratory systems, including the underlying simulation programs, should be reasonable easy. Further, they are likely to be useful.

We now summarize our position on the structures we have considered. The skill enhancement structure is likely to be the most valuable, yet most difficult, to develop. The drill device needs further research, for use in a complex problem solving context. The experimental laboratory is useful and easier to develop, but requires capabilities not usually associated with typical expert systems. Lastly, the review/refresh structure is the best initial candidate, in that it requires simpler student interactions and can use the output of existing expert systems.

In section 4.2, we listed five degrees of interaction, from passive explication of problems to active query of the student. In general, the difficulty and cost of the development increases in accordance with the degree of active involvement by the tutorial system. This is because the more active levels require more sophisticated student models and reasoning about the student's actions, intentions and abilities. An ideal system would contain elements of all five approaches, for use in different tutorial contexts. We suggest that the latter two cases, "active system criticism" and "system queries", are still subjects for considerable research. The other three cases, "problem explication", "problem exploration", and "passive system criticism" are more suitable for use in current training systems.

Lastly, we comment upon input interfaces. In section 4.2, we listed five types of interface between the student and tutorial system. The "full natural language" system can be eliminated, in that it does not exist yet. Further, the "stylized English" should be eliminated, in that the student would have to learn almost as much to use this interface as a totally artificial interface. The "command word" approach is the usual choice, in that it is easy to implement, but also requires more user pre-training. The "semantic parser"s allow natural English, but accept simpler sentences and occasionally make mistakes. The "limited domain" system actually understands English within the limited domain, but requires a sophisticated program and a substantial development effort. (Further, the more complicated the questions that the students can ask, the more sophisticated the question-answering machinery must be. This machinery is separate from the input interface, which merely "understands" the question.) All told, we favor the "command word" approach for a simple, low cost solution. If occasional failings are acceptable, then the "semantic parser" approach is worth considering, and could be developed in several man-years of effort. Though the "limited domain" system is more attractive, it is likely to require 5 man-years (specialist timo) to develop per system.

4.6 Benefits of expert training systems

In this section, we discuss the benefits associated with expert training systems. These benefits are accrued during both development and use, and apply to both the student and to the associated domain-specific professional groups.

uniform educational base - because the factual knowledge of the domain is explicitly encoded as the rules used by the underlying expert system, all of the students have access to this same set of knowledge. Further, all students have experience with the same training process (at least during use of the training system). Altogether, these should contribute to producing trained students with a more uniform minimum level of competence.

experience in inference and decision making - as well as learning the factual information in the domain, the students gain experience in its use. The students learn a formal reasoning model by emulation of the training system's reasoning process (hopefully equivalent to the expert's). The student learns what types of information can be deduced, on the basis of what evidence, in support of what goals and at what time. These are exceedingly practical and important issues.

experience with expert level performance - by using the training system, the student becomes familiar with the level of performance attainable by human experts. This is important, so that the student has both a practical sense of the maximum performance attainable, and also a model for minimum performance. Only occasionally can the student reach performance levels beyond what he/she has direct experience with, and there are not sufficient human experts to tutor all of the students. An expert training system will help solve this problem.

antiseptic, but realistic drill - the underlying domain expert program can provide the student with experience about the domain and the solution of its problems without risk. For example, a student can gain experience about medical cases, varying his approach, reasoning process or actions without risk to any patient. If the underlying expert program also has a realistic simulation component, then the student has even greater freedom to experiment.

experimental work - the student can vary the data used as system inputs to see how the solutions change. This provides experience with the sensitivity of the reasoning process.

There are also several benefits to the professional group, as a whole, that arise from the development of expert systems and expert training systems. These include:

organized domain knowledge - the expert system cannot operate without some form of domain model and the domain knowledge itself. The process of developing the system requires that the knowledge used by the experts be organized and explicitly recorded in some formal language. This process alone may contribute to an advance in the field. Further, this encoding then provides a basis for the experts to communicate and discuss their knowledge with each other.

experimental work - given a formal reasoning system (the expert system) and the set of domain knowledge (the rules), it is then possible to test the effects of modifying any of the rules upon the performance of the system. This experimentation can help improve rules to which the system is overly sensitive, to help identify major factors in the decision making process, and to improve the rules upon which the decisions are being made. This knowledge is then usable by the experts in their own decision making process.

<u>reduced</u> <u>expert</u> <u>workload</u> - the increased number of trained expert-level performers and the expert systems themselves will reduce the problem solving demands made upon the experts. In addition, the use of expert training systems should also reduce their teaching demands. Altogether, these should reduce workload demands and increase the amount of applied expertise. (At present, this is expected to pose little threat of job redundancy, because, in many fields, there is likely to be a greater demand for expertise than there are available experts.)

5. Procedure For Developing An Expert Training System

This section contains general discussion on the process of creating an expert training system. It includes the development of both the expert and tutorial elements of the system. The discussion considers the overall development process from both the human involvement and the product created points of view.

5.1 Choice of domains

In this section, we discuss criteria relevant to the selection of domains suitable for use in expert training systems. These criteria cover the requirements of both the underlying expert system and of the tutorial system. The criteria are:

type of domain - requires technical expertise. Further, the problem solving abilities should depend upon a heuristic/intuitional base supported by a collection of specific, problem solving rules. This is not to say that the knowledge base should be incorrect nor lacking in decision making capability, rather the knowledge should be that obtained from experience. Domains with extensive analytic models or domains whose expert problem solving depends upon the computational powers of the computer are not well suited as bases for similar capabilities in humans. This does not imply that all areas with high performance expert systems are unsuitable. In some cases, the complexities of the use of the performance program form the basis for the expert's knowledge. This suggests the development of expert systems whose function is to provide advice on the use of the performance programs. (ie., SACON, ADVISOR).

availability of experts - the development of the underlying expert system requires having experts with the desired knowledge available.

range of applicability - there must be sufficient applications of both the expert system and the training system to justify the expense of its development and use. This indicates that likely areas are those with a large number of practitioners at sub-expert performance levels (ie., medicine) or areas with high capital expenditures or risk (ie., mineral exploration, nuclear reactor control).

reliability of judgements - the knowledge used in the reasoning process must be well developed, so as to allow experts to solve problems with a high likelihood of success.

user group - there should be sufficient students capable of using the training system. This requires a pool of potential students who have an adequate technical level to benefit from the training.

system validation - the domain should have adequate data to compare the expert system's performance, and further, the trained students', with that of the human experts. This entails domains which have little risk and cost involved in the validation.

5.2 Selection and training of human experts

In this section, we discuss guidelines for the selection and training of experts. These experts are obviously needed in order to provide the knowledge and judgement used in the construction and validation of the expert systems. There is also a larger context in which the expert's judgements are needed. Even though the expert system provides solutions to problems, the interpretation and evaluation of the solutions is another problem. The human expert already knows how to do this, but the student will not. The sorts of knowledge that the expert has about the solution (meta-knowledge) include alternative solutions, the significance of various aspects of the solution, the likely validity of the results, and how to apply the results. This knowledge is as important as the factual and procedural knowledge, and will also have to be passed on to the student. (But how this can be done has not yet been researched.)

The selection of experts for any task is not well understood - it is largely an intuitive task (and, as such, may form a base for a future expert system). We detail below some of the specific requirements on the type of experts needed for the task:

<u>explicit knowledge of the subject</u> - the expert has to not only have the knowledge needed to do the task, but also have some knowledge of what that knowledge is. Intuitive understanding of the problem is not sufficient, as the knowledge will need to be transcribed into some explicit form, so that the expert system can use it. For example, a medical specialist has to both recognize the relationship between the symptom and the infection, and to be able to state it.

<u>general understanding of formal reasoning</u> - it is necessary for the expert to generally understand the nature of the reasoning performed by the expert system. By the nature of computer processes, all reasoning must be done according to some formal procedure. The expert and assisting knowledge engineers have to encode the expert's knowledge into the chosen formal structure, and hence both must understand the nature of this reasoning.

<u>general familiarity with computers</u> - the expert must understand the conflict between mechanical faithfulness and human limitations (causing errors and conceptual failures). This is also needed for understanding machine and system limitations, and general computer capabilities.

<u>general understanding of models</u> - In order for the expert system to reason about a domain, it must have a model of that domain (explicit or implicit). Perhaps the most significant single function of the expert is the creation of that model. (A model consists of a set of primitive structures connected into a whole by a set of defined interactions.) So, the expert must know what a model is and how it differs in behavior from the actual system being modeled.

In addition to the general skills mentioned above, the expert will need to have a few specific skills directly related to the particular expert system. These will have to be learned at the time. They are:

<u>knowledge of the particular reasoning methods</u> - the expert must appreciate the type, structure and order of the reasoning that the system executes, because his encoded knowledge will have to be fashioned to

suit the system.

<u>knowledge of the models supported</u> – the expert, in conjunction with the knowledge engineer, will have to create a model of the domain. This includes separating the analytic (theory based) portions from the heuristic (experience based) portions.

<u>knowledge of rule formats</u> – the expert will have to know the format into which the knowledge is to be represented. Certainly, the knowledge engineers can help with the precise details of the encoding, but is the experts themselves who have to figure out how to structure any particular chunk of information into a computationally useful rule.

It seems likely that the majority of experts in technical fields will be close to meeting the general requirements given above. Training the experts to adequate levels will probably not be too difficult, due to the collaboration of the knowledge engineers during the model formation and knowledge acquisition period.

5.3 Knowledge acquisition

In this section, we briefly outline the process of acquiring the knowledge used as the basis for the expert systems performance.

The first major stage is the model design. In this stage, the items that need to be decided are:

1. what the expert system will do - that is, what actions and outputs for what inputs

2. what are the primitive concepts of the domain - these form the foundation concepts from which reasoning proceeds. In the medical domain, typical concepts would be particular symptoms, infections, disease types, organism types and therapies. At this point, it is also necessary to decide the structure of the concepts. In the CRYSALIS system, there is a three level structure of basic concepts (atomic, superatomic and stereotypic), representing the conceptual levels for viewing the entire problem (deciding the molecular structure from x-ray crystallographic data). The rules in the knowledge base are then responsible for making conclusions about and based upon data elements at the different levels, including the raw data level.

3. which aspects of the expert system's performance are to be based on theory (thus using mainly computational models) and which are to be based on reasoning (from the expert's experience) - This affects what knowledge is "visible" and what is embedded into special purpose computer programs. Simple theory-based knowledge may be visible, as in the case of facts, by direct representation (as in "The possible atomic weights of oxygen are ...").

4. What is the structure of the domain model - **MYCIN** uses a simple model relating infections, sites and responsible organisms. These items are based in an object-centered representation, whereby each object ("infection") has various attributes ("sites") each of which has values ("stomach"). Other systems (eg., **R1**) use more complicated models, involving detailed relationships between the components and the total constraints of the problem structure.

5. selection of reasoning structure - the most commonly used reasoning structure is the causal form, in which the knowledge is encoded in a set of rules of the form: "If A is determined, then conclude B". The use of these rules varies with the application. In some instances (**MYCIN**-like systems), they are used in a goal-directed fashion, in which one attempts to demonstrate desired conclusions (B) by attempting to find the evidence (A). Another form is the data-directed mode, in which the reasoning starts from the evidence (A) and leads to any appropriate conclusions (B). In both cases, the procedure is iterative. A third method consists of constructing a causal network from the full set of inter-relating rules, and uses a probability based (Bayesian or other) scheme to adjust the likelihood of any particular conclusions, based upon observed evidence.
   A second consideration is whether the conclusions are to be exact, or just to be based upon some sort of likelihood value

(either theoretically or heuristically motivated). This decision depends upon the type of domain and the nature of conclusions the expert makes. The R1 system can make absolute decisions by nature of its domain: the evidence is precise (the customer's specifications) and the rules well-specified (system engineering principles). The **MYCIN** system uses attached certainty factors to reflect the heuristic nature of the expert's knowledge and judgement implicit in medical diagnosis.

6. <u>What is the rule format used for representing the knowledge</u> – the logical structure, and primitive symbols used for capturing any particular domain concept are chosen. This process is an art, in that sufficient symbols have to be chosen to represent all distinguishable conclusions, without including so much detail as to overwhelm the computational mechanism. Then, a representation has to chosen to explicitly record the causal relationships between the concepts. In the case of **MYCIN**, the system goals required linking the symptoms to the infections, without requiring the unnecessary detail of the underlying biomedical disease process.

The next major stage is the actual knowledge acquisition phase. This consists of:

1. <u>interviewing the expert</u> – to acquire the knowledge. This also involves the formation of the domain model.

2. <u>rule encoding</u> – casting the knowledge into the formal language used by the expert system.

3. <u>consistency checking</u> – detection of contradictory conclusions.

4. <u>completeness checking</u> – determining features which cannot be concluded, because of lack of rules.

Lastly, there is a phase of validation, testing and tuning. This is discussed more completely in section 5.8. It should be pointed out that all of these phases are not distinct, and have an iterative flavor, especially during the acquisition and testing phases.

5.4 Training system development

In this section, we discuss the work needed to develop the training system program which interfaces with the expert system and student. We focus mainly upon the decisions that need to be made (as compared to the actual options), and the stages of the process.

The overall process consists of seven stages, as listed below. System development proceeds through each of them roughly sequentially (with possible iterations to improve or correct performance). This development is no different from that of any other computer program, but here we discuss it from the training systems viewpoint. Stages 1-4 are discussed in this section, and stages 5-7 are discussed in section 5.8. The stages are:

1. Specification - selection of the system's overall function and methods

2. Design - selection of representations and algorithms for performing the desired functions

3. Implementation - the encoding of data and methods into tutorial rules and procedures

4. Testing - ensuring that the tutorial system executes its functions according to design, without error

5. Validation - ensuring that the tutorial system functions according to specifications

6. Tuning - improve the system performance

7. Evaluation - assessing the effectiveness of the system (ie., does it meet its goals)

The specification stage is the most important part of the whole process, in that it is during this phase that the overall system concept is chosen. To meet this goal, we have identified decisions that need to be made:

1. selection of goals - the overall conception of the function of the system. The result of this decision corresponds to one of the roles identified and discussed in section 4.1. Possible goals for this system include:
   giving advice on solving problems (Adviser role)
   demonstrating example solutions (Demonstrator role)
   assessing the student's knowledge (Examiner role)
   providing experience with solving problems (Exerciser role)
   teaching domain facts and concepts (Tutor role)

2. selection of framework - how the system will attempt to meet the goals selected above. The frameworks are also discussed in section 4.1. Possibilities include:
   review/refresh - examination of sample problems and methods
   drill device - provides problems for student to solve
   skill enhancement - factual and procedural knowledge presentation
   experimental laboratory - simulated environment for directed

experimentation

3. selection of tutorial methods - how to select topics, how to select and present relevant information, how to model, assess and criticize the student, what alternative actions are available and their criteria for selection, how to control and focus the interaction, and student capabilities. This topic will require the most consideration, in that it covers the system's detailed behavior and is most responsible for implementing the decisions of items 1 and 2.

4. selection of dialogue structure - the nature of the interaction (ie., who has what capabilities) and how to control the dialogue to meet the conflicting requirements of tutorial goals, student needs and desires, and naturalness of interaction.

5. selection of accessible knowledge - what aspects of the domain knowledge, reasoning methods, problem solution, student model and tutorial methods should be explicitly represented and accessible by the user or tutorial program.

The design stage comes next. In this stage, we select techniques and structures that will result in a system whose performance meets its specifications. At the top levels of this process, we select the major components of the system, their functions, and their interrelationships and interfaces. At the detailed levels, we consider actual data and program structures. The major elements of this stage are:

1. selection of tutorial methods representations - how we represent the program's decision making rules. They may be explicit (ie., production rules) or implicit (ie., computer program subroutines).

2. selection of interfaces - how does the training system communicate with the user and associated expert system

3. selection of underlying expert system - whether a problem solver, adviser, analyst or simulator. This also includes considering what capabilities and accessible knowledge are needed to support the tutorial goals.

4. design of actual programs - to segregate the overall functions into identifiable components and to specify their organization and interfaces.

The implementation stage is perhaps the most lengthy stage. It requires the development of both the procedural knowledge (ie., computer programs) and the explicit knowledge encoding (ie., production rules).

The testing stage is similar to that discussed in section 5.8, except that here the focus is upon the tutorial system methods. The goal of this stage is to ensure that the programs execute correctly over their range of inputs. This also includes ensuring that the deductions made using the explicitly recorded knowledge are correct. The notion of correctness here is "performing according to design". This mainly means that the system must make the right responses to the student's actions at the right times.

The remaining stages, validation, tuning and evaluation, are discussed in section 5.8.

5.5 System frameworks (Use of existing systems)

As we have seen, the construction of an expert system is a substantial enterprise. A majority of the effort lies in the development of the software framework used to support the expert level reasoning. In consequence, researchers have concentrated upon development of expert system frameworks suitable for use with a variety of knowledge bases. The intent of these systems is to allow the knowledge engineer to re-use a set of common facilities, and just concentrate his/her efforts upon the development of the knowledge base.

These systems are called the domain-independent expert systems, and will save considerable system development time. Each domain-independent system encapsulates a particular view of the expert system structure - such as a particular reasoning model, knowledge representation and selection of support facilities. Many of the differences between systems arise from the intended use of the system, from the interactive consultation focus of **MYCIN**, to the largely independent data interpretation of **GAMMA**.

The major components generally included in domain-independent expert systems are:

<u>inference engine</u> - reasons about the problem using the domain knowledge
<u>control program</u> - selects among the current potential actions
<u>input interface</u> - analyzes the user's inputs
<u>explanation program</u> - answers questions about the knowledge base and specific problem solutions
<u>knowledge base maintenance</u> - supports examination and editing of the domain specific knowledge. It may be able to do consistency checking and to evaluate rule changes against previous problem solutions.
<u>working memory structure</u> - maintains the current problem solution context

The construction and use of domain-independent expert systems is still a research issue. Until there has been much more use of the developed facilities, there is not likely to be any convergence to an agreed upon set of capabilities. Commercially viable systems are not likely to be available for 5 to 10 years.

We conclude this section with a brief description of 6 domain-independent systems. These systems have arisen from a variety of sources. Two, the **EMYCIN** and **HEARSAY-III** systems, are generalizations of existing expert systems. **EXPERT**, while reasonably general, has been designed for mainly medical consultation domains. The **XPRT** system is actually more of a specialized programming language which the knowledge engineer can use to assemble a complete system. On the other hand, the **AGE** system not only provides the prepackaged components for systems, but also guides the knowledge engineer in the actual construction. The **AL/X** system is a probabilistic reasoning system, somewhat based upon two other expert systems, **MYCIN** and **PROSPECTOR**, and aims at being commercially usable.

<u>EMYCIN</u> - this is the abstraction of the facilities of the **MYCIN** system. It uses a discrete logic production rule based reasoning structure, with certainty factors, focussed about a goal-directed AND/OR solution tree. It has the full set of components listed above. This system appears to be best

suited for applications which require substantial alternative based reasoning concluding about a small set of previously known objects. It also appears to be most usable for consultation type expert systems.

HEARSAY III - the abstraction of the facilities of the HEARSAY II system. Its major features are the control program and its working memory structure. The type of reasoning is unconstrained, and the working memory is a hierarchically structured blackboard. Further, provision is made for partitioning the domain-specific knowledge into separate knowledge sources, representing the notion of cooperating experts. This system appears to be best suited for problems requiring substantial data analysis, in the context of richly articulated models and multiple sources of domain knowledge.

EXPERT - a system designed for mainly consultation type domains, such as medical diagnosis. This system attempts to emphasize categorical, rather than probabilistic reasoning, because this stimulates better predictability and correctability. Data can be input in a variety of formats: multiple choice, checklist, numerical values, and yes/no, and can be grouped into questionnaires. The system differentiates between data values (findings) and all intermediate or final conclusions (hypotheses). EXPERT contains versions of all the major components listed above. This system appears to be best suited for problems depending upon incomplete knowledge, substantial heuristic reasoning about a few conclusions, and interactive contexts.

AL/X - this is a PASCAL based domain-independent expert system which will execute on a mini-computer system when using a moderate rule set size (150 rules). Its inference engine is based upon the propagation of probabilistic inferences in a network of interrelated hypotheses and deductions. The system evaluates the effects of any data upon all possible conclusions, resulting in the conclusions of greatest likelihood. Its working memory structure is the hypothesis network. AL/X uses a command menu based input and control language, and provides some explanation facilities. The system appears to be best suited for applications in which all possible conclusions are known, as well as the mechanisms for making these conclusions, but in which the actual reasoning supported is overly complex or tedious for a human expert.

AGE - this system is more of an expert on building expert systems. It has a selection of alternative forms of the major components of an expert system, and manages them in a manner suitable for combination into a complete system. AGE offers three major advantages - the components are prepackaged, the user can experiment with alternative configurations, and AGE leads the knowledge engineer through the development process (keeping track of the current development status). The alternative structures include: problem solving structures, knowledge representations, reasoning structures and control structures. This system offers a great deal of promise, but is, as yet, still a research system.

XPRT - this system provides a framework in which to develop sophisticated reasoning programs. It has a declarative knowledge representation, an underlying inference mechanism, and facilities for controlling the inferences. It is not a domain-independent expert system, rather more of a reasoning system upon which it is convenient to develop specific expert systems.

5.6 Computing hardware bases

This section briefly describes the computer systems which are the best candidates for use in expert training system applications. We include machines used in the past, as well as machines with good potential in the near future. Additional information can be found in papers by Rae (Rae80) and Boley (Bol80).

The suggested necessary requirements for such a machine are:

speed - 1 microsecond instruction time maximum
address space - 18+ bit addressing minimum
main memory space - 256K bytes minimum
file store - 5000K words storage minimum
high level language - **PROLOG**, **LISP** or **PASCAL** preferred

The present generation of mini-computers and micro-computers (ie., DIGITAL PDP 11, Data General Nova, Intel 8086) cannot meet these requirements. However, several currently available multi-user machines provide the minimum facilities. They include:

DIGITAL Equipment Corporation DECsystem 10 or DECsystem 20 - mainframe computer systems

DIGITAL Equipment Corporation VAX 11/750 - a large minicomputer system

Prime Computer, Inc, Prime 550 - a large minicomputer system

The preferred requirements for a machine are a high resolution graphics display (eg. 800*1000 pixels), a pointing device (eg. mouse) and an interprocesser communication network facility (for monitoring student progress, and future capabilities). Since the high resolution display makes heavy demands on the processor, these requirements are best serviced by a single user machine. Available machines include:

PERQ machine, manufactured by Three Rivers Computer Corporation - a sophisticated single user computer system

Massachusetts Institute of Technology LISP machine, manufactured by LISP Machines Inc or Symbolics Inc - a sophisticated single user machine based about the **LISP** programming language

DOLPHIN, manufactured by XEROX Electro-optical Systems - a single user LISP machine

DOMAIN, manufactured by Apollo Computer Inc - a single user microprocessor based machine

A summary of the features of the listed computer systems is given in table 4.

MACHINE	SAMPLE COST	FAST	ADDRESS	MEMORY	LANGUAGE	FILE	DISPLAY	SINGLE USER	NETWORK
PDP 10	$300000	Y	Y	Y	Y	Y	O	N	U
VAX 11/780	$100000	Y	Y	Y	Y	Y	O	N	U
PRIME	$105000	Y	Y	Y	N	Y	O	N	U
PERQ	$25000	Y	Y	Y	Y	Y	Y	Y	Y
LISP	$125000	Y	Y	Y	Y	Y	Y	Y	Y
DOLPHIN	$60000	???	Y	Y	Y	Y	Y	Y	Y
DOMAIN	$45000	???	Y	Y	Y	Y	Y	Y	Y

Table 4 - Summary of features of likely computer hardware
Y - meets requirements
N - doesn't meet requirements
O - optional purchase
U - unnecessary in this system

5.7 Software development

In this section, we overview the software engineering aspects of developing an expert training system. In this instance, we are considering the framework which supports the knowledge and reasoning, as separate from the knowledge itself, whose acquisition was discussed in section 5.3.

There are four major components to a tutorial system, interacting roughly as shown in figure 7 below. These components are:

1. inference engine - the component which does the expert reasoning, given the knowledge base and current data.

2. input interface - the component which makes sense of what the user has input. (The sense will depend upon the current discourse context.) Its complexity depends upon the degree of natural language input desired.

3. question answerer - the component which searches through the system's data structures to locate or reason out answers to the user's questions.

4. tutorial system - the component which directs the student's interaction with the system, so as to fulfill its tutorial goals.

Each of these components is significant, and represents a development effort of 1-3 man years each (estimated), based upon their general complexity.

There has been research aimed at generating domain independent versions of the inference engines, as discussed in section 5.5. Further, though there are only few examples of research on tutorial systems in the expert system context (section 3.3), it might be possible to develop a tutorial system base which is relatively domain independent. That is, it would be aware of the structure of the domain, about which it was tutoring, and of the form of the student model, but not of the actual contents of the domain and model. In any case, a domain-independent tutor would have to be tightly coupled to a specific domain-independent expert system. These considerations are all speculative, given the current state of research. (The **GUIDON** system was built to meet these requirements in conjunction with the **EMYCIN** system, but was found to be more effective with certain types of domains (Cla79c)).

The input interface could be either the simplest (given prompted keyword input) or the most complex (given reasonably natural language input) component in the system. A full natural language system is likely to require substantial re-development for each new domain, because each subject has much implicit structure and knowledge, all of which is needed for understanding, but little of which is expressed in the actual utterance.

An intermediate level input interface would be one like the **CHAT-80** system (War81). This system is designed to understand and answer English questions about the contents of a class of databases. The domain-specific vocabulary and the contents of the database are input according to the requirements of the specific problem. This system is not directly usable in the tutorial application, as the types of queries asked by the student do not fit the model used in **CHAT-80** (quantified relational database queries).

Further, the system is designed for answering questions, and not for understanding the student's answers to the training system's questions. None the less, this research indicates the possibility of reasonably simple, yet flexible and efficient input interfaces.

The question answering component could perhaps be made relatively domain-independent, as existing domain-independent expert systems have a similar component (eg. EMYCIN). The general technique would need to be extended to include access to the tutorial elements of the system.

All told, a working system could probably be developed, based upon the work of previous systems, for about 5-10 man-years of effort. Of course, given that such a construction is relatively original, and is intended for real use, (as compared to the previously described systems, which were research projects), this estimate is not reliable. Further, due to the original nature of the work, it is likely that many technical problems will be encountered, as well as deficiencies in the actual final performance. The authors do feel, though, that, despite the risks, some such system is within the reach of current software engineering, knowledge engineering and computer-aided educational techniques.

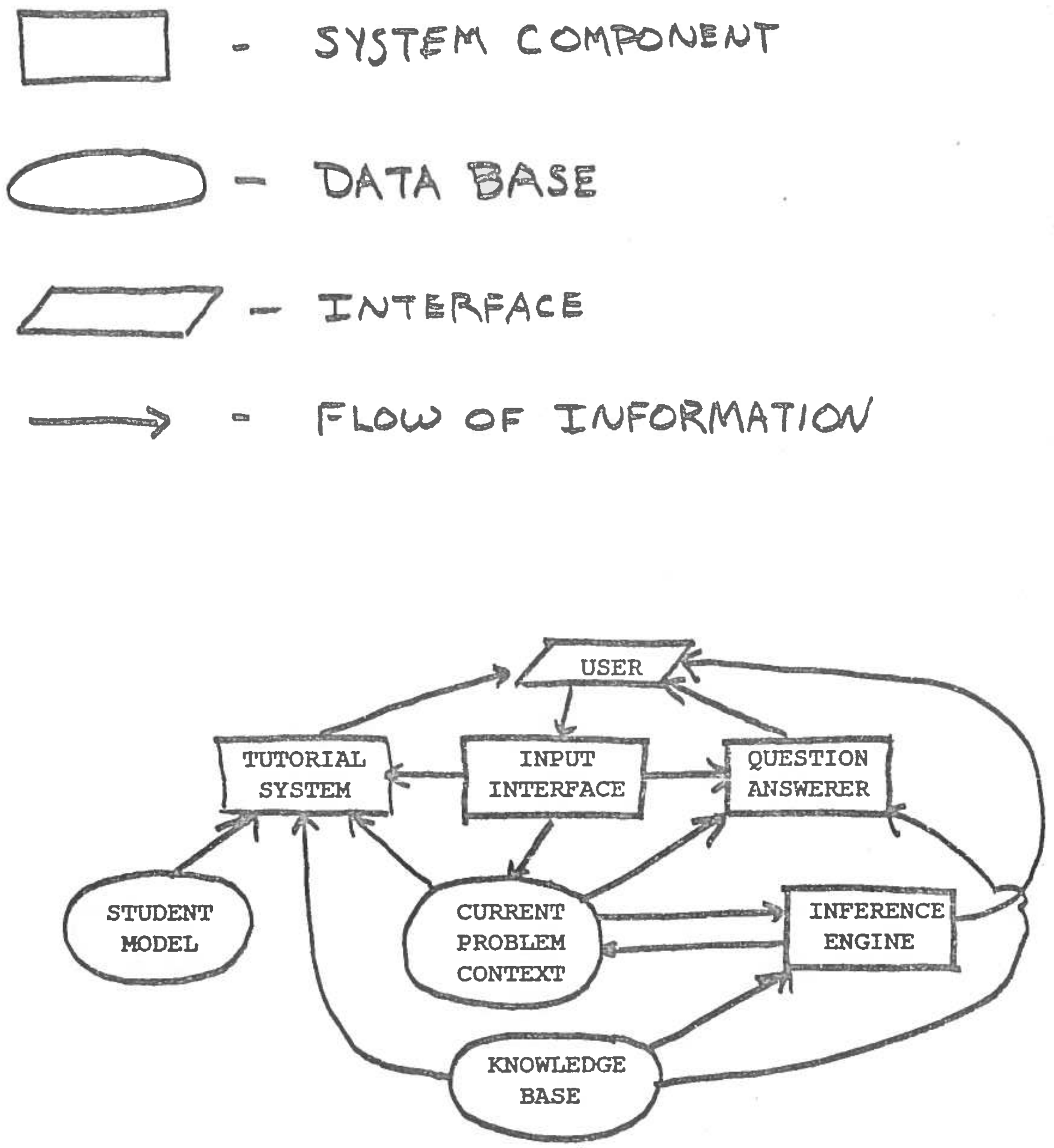

Figure 7 - hypothetical expert training system structure

5.8 Testing, validation, tuning and evaluation

In this section, we present a discussion of the final stages of expert training system development. In these stages, we presume that the software forming the underlying (domain-independent) system is complete and executing properly. Rather, we address issues relating to the performance of the system while using the domain specific knowledge.

testing and validation - In these phases, the issue is ensuring that the system is giving correct solutions to the problems presented to it. Moreover, it should achieve solutions based upon correct reasoning. The testing phase focuses mainly about the knowledge engineer ensuring that the knowledge was correctly encoded and is used correctly by the inference engine. The validation phase focuses mainly about the domain experts ensuring that the system achieves its results based upon correct reasoning, that the system solves an adequate range of problems, and that minor perturbations in the problem data cause only minor perturbations in the solutions.

For both tutorial and expert systems, this phase requires a detailed examination of the reasoning processes associated with particular problems and their solutions. It further requires an adequate number of test cases, largely due to current software technology (as the domain knowledge is entered as unstructured independent rules, which allows unpredictable interactions). The phase consists of having the experts and knowledge engineers examine, in detail, the output of the system, so as to check its reasoning. In some cases, right/wrong evaluation, can be made (as in computer configuration layout); in others, there is only a subjective determination (as in medical diagnosis). The result of these two stages is a system which gives acceptable results, based upon valid reasoning. These two phases have been categorized together because they occur reasonably simultaneously, with both the knowledge engineers and the experts cooperating.

tuning - In many domains, there is not just one correct solution, rather a range, among which one or more may be preferred. Expert systems built on such domains often use certainty factors or ratings associated with deductions, and rank solutions in accordance to the final certainty values. It would be ideal if the expert system were to rank its alternative decisions in the same order and with the same weight as the expert. Tuning requires examination of the set of deductions made by the system, to determine rules whose evidence is being considered too strongly or weakly. It may also require additions of new support rules, based upon other evidence initially omitted by the expert. This is obviously an iterative process.

The tutorial system would also require tuning. The system behavior would be observed in training sessions to determine the effectiveness of various strategies, and appropriateness of actions at various stages of the dialog. This process is much more difficult than the expert system tuning, due to lack of analytic theories of educational methods. (The decision of what to do next, by human tutors, is usually motivated by intuition coupled with general educational philosophies). Though the effect of the changed tutoring rules on the system's behavior may be dramatic, the change in tutoring effectiveness is hard to evaluate.

In addition to the substantive aspects of system tuning, there are more practical matters. These include decreasing response time, reducing program size, transfer to more cost-effective machinery. Various cosmetic improvements are also useful, such as improving the coherence and appropriateness of system responses, improving explanations and incorporation of graphics output. These points address some of the difficulties involved in producing a system for public use based upon a research system.

<u>Evaluation</u> - of both the expert and tutorial systems.

Evaluating the expert system is probably easiest, and prior research has been done (Gas80). The usual approach is to compare the solutions of both the expert system and human experts over a range of problems. This does require some procedure for comparison, which depends upon the domain. Factors include: correctness, which, in exact domains, is easy; completeness (covering all facets of the problem); elegance (as in circuit board layout) and cost-effectiveness (cheap solutions). The general goal of this stage is to show that the system is indeed performing at an expert level.

Evaluating the tutorial system is more difficult; even evaluating human tutorial methods is difficult. Factors which need to be considered include both cost and performance. The degree to which the student acquires knowledge of the facts and methods of the domain must be measured. Further, it must be asked whether the problems and solution methods are realistic enough to actually provide the student with usable experience, and is the student benefiting. Another question is, are there any reductions in the amount of time spent in training or the demands upon the experts. (The Stanford BIP project (Bar76) found both the experimental and control groups achieved similar performance, but the test group had 25% more programming experience).

An evaluation of the overall cost-effectiveness of the training is also needed.

6. Deployment Considerations

This section discusses some factors affecting the actual training system use.

6.1 User training preparation

In this section, we discuss the preparation that the user needs prior to starting to use the expert training system.

There are two peripheral topics, unrelated to the actual subject domain. They are:

mechanics of system usage - this includes the basic issues of general interactive computer usage, such as use of the keyboard functions, starting the tutor program and what to do when something goes wrong. This probably also requires some education on the nature of computers, so as to develop a feel for the source, and hence the limitations, of its abilities.

use of the input interface - this is the program which is used by the student to communicate to the training system. The type of preparation depends upon the input interface chosen. If a natural language interface is chosen, the student will have to learn its vocabulary and limitations (lack of grammatical features and sentence complexities). Further, he/she will have to learn to paraphrase inputs and to recognize mis-interpretations. If a keyword command input interface is used, then the list of valid commands, parameters and options will have to be learned. Both of these interfaces will require substantial preparation, with the keyword approach requiring the greatest effort.

It is not appropriate to use the expert system to teach the basic elements of the domain knowledge. This is due to the underlying expert system focusing more upon performance issues (use of the knowledge) than upon competence issues (having the knowledge). It will probably be more economical to learn the basics via other mechanisms, such as readings and lectures.

It will also be necessary to teach the student how the expert system solves problems before the student actually uses the system. Otherwise, we believe that the system's choice of actions and decisions will be mysterious, and not convey any information. Once the student understands the goals of the system, then it will be easier to see how any particular action helps achieve those goals. Three topics of this pre-training are:

domain model - the student will learn how the domain elements are structured for problem solving. This includes the types of object descriptions (if multiple views are used, as in a hierarchical structure) and the relationships that can exist between elements at each level and between levels. For example, if the student were working with CRYSALIS system model, then he/she would have to learn the submodels at the atomic, structural component (ie., amino acid) and complete molecule level. The student would also have to learn the interlevel relationships, such as which components have which atoms, and how the model records this information.

<u>domain knowledge structure</u> - the student should be familiar with most of the rules used by the system. This is partly due to the efficiencies of learning the rules directly, and partly due to the need to learn the knowledge in the form that it will be used. (Clancey, in developing GUIDON (Cla79c) classified the rules into levels of difficulty. They were then opportunistically tutored whenever GUIDON felt the student was sufficiently advanced for the rule.)

<u>reasoning structure/problem solving method</u> - the student needs to know how (ie., the steps) the expert system is going to solve the problem and by what sorts of techniques. The actual use of the system should provide the intuitive understanding of this process, but some explicit study of the methods beforehand should be useful.

6.2 Instructor training preparation

In this section, we overview the training that the instructors (or demonstrators) will need in order to support the users during their training.

To begin, the instructor will need all of the training that the user receives, but in greater detail. Further, he/she will need enough practice with the system that its features become familiar.

The second major requirement involves the use of the stored case problems. The instructor must have solved all of the problems, while using the expert training system. This is so he/she can help or give advice to the student, or explain the various aspects of the system's behavior. When actually involved in the course, the instructor will have to select problems for the student's exercises, hence must know the type and purpose of each problem. (More sophisticated ICAI systems may be able to do this selection based upon the student and curriculum models, but we believe that this technique is not yet adequately developed for practical use.)

The instructor will probably be responsible for recognizing system failures. These may be catastrophic failures, such as computer or software breakdowns, or merely errors in reasoning on the part of the expert system. All major errors will have to be reported somewhere, so that they can be fixed. This will require training in the recognition, recording and reporting of major errors, and the rectification of minor errors.

The system will probably keep some sort of logging information abstracted from the student's sessions, although the amount of detail kept will depend upon the goals of the collection. One possible use is for the instructor's review of the actual sessions, to analyze the student's performance and deficiencies. This will require training in the interpretation of the session log.

7. Summary and Conclusions

This section summarizes the discussion developed in the body of this paper. It's purpose is to answer a few specific questions raised by the funding agency, and to identify some other important issues.

7.1 Short answers to specific questions

In this section, we answer questions specifically raised by MSC/TSD. The answers are summaries of more extensive discussions taken from the body of the paper. The appropriate sections are referenced.

1. What is involved in developing a methodology for acquiring knowledge that could be used in various domains? (2.6, 5.2, 5.3)

The technique used most effectively for knowledge acquisition is interviewing. The knowledge engineer, as interviewer, attempts to isolate, clarify and record the domain knowledge of the expert. Before this process begins, the two (engineer and expert) have to agree upon a model or model schema for the domain, so as to clarify the relevant aspects of the problem. Also, they have to agree upon the manner in which the problem solving is to be accomplished by the expert system. Given these three steps, the knowledge then has to be encoded into a chosen formal language. Lastly, there is an iterative phase of knowledge debugging and improvement (Fei79).

A highly relevant issue is the type of skills the chosen experts should have. We believe the major skills needed are: (1) an explicit understanding of the knowledge and procedures used in solving problems from the domain, and (2) an ability to articulate personal problem solving capability.

2. What skills would be required of the "knowledge engineers" who would develop the systems? (2.4, 2.6, 5.3, 5.5, 5.7, 5.8)

The knowledge engineering skills fall into three areas. The first area is an in-depth familiarity with existing expert systems, including the use and modification of the various domain-independent systems. This also includes basic computer programming skills (most probably the LISP programming language). The second area is a good working knowledge of the various techniques of knowledge engineering, including reasoning models, knowledge representation techniques, domain model schemas, computer-based problem solving and the methods for explicitly recording the problem solving process. The final area is concerned with obtaining the domain-specific knowledge from the expert. The interviewer must be a diplomatic listener and questioner, as well as have considerable social skills.

At present, due to lack of interest in Artificial Intelligence research, which largely stems from the negative Lighthill report (Lig73), there is probably no one in the UK who exactly meets these requirements, and only a small number who are relatively close.

3. What estimates can be made of the ratios of development time to actual system's running time? (5.7)

These two figures are difficult to compute. In section 5.7, we estimated that it would probably take 10 man-years of effort to develop a rudimentary expert training system. We also briefly discussed the development of domain-independent expert and tutorial systems, and estimate (here) that it would take only 2-3 man-years of effort to adapt one such base to any specific domain (including knowledge acquisition).

By the use of rule-structured knowledge bases, it is possible for human experts to constantly update the knowledge base as new principles and relationships are discovered. As a result, the knowledge base is likely to remain current over a reasonable time period (10+ years). (However, these considerations ignore the likely technological obsolescence of the computer hardware and software.)

We estimate the execution of such a system will require on the order of 1/2 to 1 minute of real time per input from the student (less for simple questions). Assuming 50 questions in a problem solving session and 25 problems per training program, we estimate 20 hours of computer use per student. Assuming a figure of 100 students (in the UK) per year, and a 10 year use period, this leads to an estimate of 20,000 hours of use (about 10 man-years).

Note, that these figures are roughly comparable to previous experience (Bun70), in which it has taken 300+ man-hours of effort to produce a 1 hour computer-aided instruction session.

<u>4. Are theses ratios likely to change with advances in programming technique?</u>

Yes. The development of usable domain-independent expert and training systems is likely to reduce the development effort to about 2-3 man-years per domain (but this development is not likely for 5-10 years. Also, this is for systems comparable in power to existing systems.) Further, as experience with the use of training systems grows, it will be possible to add additional problems to the tutorial system repertoire, so that the students can be given essentially unlimited system use with only minimal extensions.

<u>5. How feasible is it to run useful expert systems on micro-computers? (5.5, 5.6, 5.7)</u>

Currently, expert systems cannot be run on micro-computers.

The major problems with current micro-computer systems include program size limitations and a lack of adequate development and execution facilities (programming languages included). Within 5 years, these problems are likely to be eliminated through the availability of cheap single user machines, such as the PERQ.

Work is being done on one domain-independent expert system, **AL/X**, to allow it to run on an APPLE II+ micro-computer, with a rule base capability of 150 rules. This system uses a Bayesian probability network for the rule structure and, as such, is probably not well suited for training applications. (Formal mathematical models are easier for computers to manipulate and are thus not well suited for human use.) Even if it was suitable, to implement a training system would involve adding a tutorial capability, a question-answering capability and a natural language interpreting capability. The program for each of these is likely to be larger than the core expert system program. For reasons of size, the **AL/X** system has only elementary forms of the latter two components.

6. What are the practical implications of developing expert systems to act as training aids?

For the learner, it is most likely to provide a basis for examining simulated case studies with the opportunity to see expert decision making abilities in practice. If appropriately arranged, this ought to extend or improve his/her decision making capabilities, while also demonstrating some effects of mis-management or bad decision making in a risk-free environment.

For the population as a whole, it may serve as an introduction to a likely future, where computer experts will offer advice on a variety of subjects (20+ years future).

For the experts, it is likely to reduce the tutorial demands made on their time, without prejudice to their own careers, due to a current general excess of demand for expertise in most all fields.

For computer manufacturers and software engineers, it will serve to extend computer-based technology into a new user domain, thus providing new jobs.

7. What guidelines could be developed for identifying domains that would lend themselves to the development of expert systems? (2.5, 5.1, 5.2)

The major feature of the domain itself is that the problem solving abilities found in those expert in the domain should be based upon heuristic or judgemental knowledge, rather than upon detailed analytic models. This should not be taken as implying inexactness, nor inability to actually solve problems, but rather an ability to solve problems based upon incomplete knowledge of the domain. Domains with extensive theories are best served by systems capable of automatic execution of those theories, especially as the computer provides the computational power and bookkeeping needed for problems which overwhelm humans (Bar80).

Practical considerations include: the number of people already capable of the performance, the requirements for experts, the level of expertise attainable and the value (cost-effectiveness) of computer-based training.

8. To what extent is expert system development likely to be relevant to enhancing understanding of Human Learning processes?

The development of expert systems is not anticipated to have any immediate effect on our understanding of human learning processes. One possible line of development which might is the related effort to build systems which model human performance at a task with a view to understanding difficulties by experimenting with "bugs", "mal rules" and so on (Bro78b,Sle81). Work on expert systems might benefit a great deal by paying attention to these investigations. More immediately, the development and use of expert training systems is likely to provide rich data concerning human problem solving performance based on investigations of strategies actually followed by learners. Also, expert tutors will make good frameworks for experimenting with teaching strategies, such as the active/passive distinction. This is because these can be explicitly represented in the tutorial systems, thus making them suitable for easy modification and replacement.

9. Is the development of expert systems likely to "spin off" any other concepts/techniques that may make a contribution to training? (4.6)

The major contribution is likely to come from the codification of the knowledge and methods used by the experts while problem solving. The students will then be able to use the transcribed information, with or without expert system interaction, to supplement their textbook learning.

The use of formal domain models and formal reasoning methods may be of use to the students, in terms of both the specific models and methods used for any particular domains. The models will help to clarify the relevant aspects of a domain, especially those aspects needed for solving particular problems. The reasoning methods illustrate that effective procedures do exist, and how they are used.

The de-mystification of various domains of expertise is likely to be of benefit to the experts, the students and general public. Also, the formalization is a useful method for helping to locate contradictions and incomplete areas. These would be found when the expert system produces contradictory results or fails to produce any at all.

10. From a training viewpoint, what are likely to be the most profitable directions for work on expert systems to take? (4.1, 4.4, 4.5, 5.5, 5.6)

We interpret this question to mean: "If you were to submit a proposal for a low risk expert training system, what would the general character of this proposal be?

We suggest that the lowest risk system would be a Demonstrator type expert training system.

The student would choose a particular problem, which the training system would then demonstrate. Its explication would focus upon the solution, key factors, reasoning steps and facts used in the solution. The student would be able to select among available topics, request greater detail, ask some simple questions or merely passively follow the presentation. This type of system incorporates the more passive features of the **GUIDON** system, and omits the student modelling and tutorial aspects. Further, the system solves the problem (in the Demonstrator system), rather than the student (as in the **GUIDON** system).

The proposal would select a particular domain and nominate experts who could serve as consultants. The system would use an existing domain-independent expert system, with necessary modifications to convert it to new languages (eg., PROLOG) and/or machines. (eg., VAX 11/780 (DIGITAL) or DECsystem 10 (DIGITAL) with remote terminal connections.) The student interaction would be via simple keyword based inputs. This requires greater student learning, but leads to simpler and less expensive systems.

A system such as this is likely to require 4 man-years to develop (approximately 1 man year each for: framework adaptation, knowledge acquisition, training system addition, and system testing/improvement).

A more sophisticated system, such as Clancey's **GUIDON**, would be a preferable alternative. This system would selectively present new material while guiding a student who is solving a selected case problem. Unfortunately, much research remains before such systems will be successful.

7.2 Other important points

In this section, we address other important points. The format of the presentation is answering additional questions of our choosing. As in section 7.1, appropriate sections are referenced with each question.

1. What research subjects will contribute to enhanced expert training system capabilities? (2.8, 3.2)

We think research in the following subjects will contribute to better expert training systems. The expert system research we suggest is:

natural reasoning - so that the type of reasoning used by expert systems more closely resembles that used by humans.
meta-level strategies - to express the expert's judgements on how to solve the problems, and which issues to address first. (Current expert level knowledge is mainly factual.)
explanation generation - so that the expert systems can better explain what it did, why it did it, and what it knows.
advice systems - inducing the user's needs and suggesting solution approaches.

For training system research, we suggest:

tutorial strategy representation - methods of explicitly recording and using the strategies, and the strategies themselves.
student modelling - deducing and representing the student's knowledge and current reasoning.
dialogue management - for both keeping the dialogue focussed, yet flexible to individual needs, styles and interests. This also considers methods appropriate to the current tutorial task.
training system roles (sec 4.1) - in particular, Demonstrators and Exercisers in the context of complex problem solving, and general Examiner capabilities.

2. Where might one find expertise to support the development of expert training systems?

We concentrate just on U.K. resources. For expert systems:

Univ. of Edinburgh, Dept. of Artificial Intelligence (Howe, Bundy, O'Keefe, Byrd, Fisher)
Univ. of Edinburgh, Machine Intelligence Research Unit (Michie, Reiter, Barth)
Univ. of Sussex, Cognitive Studies Program (Sloman)
Univ. of Leeds (Sleeman)
Univ. of Aston (Chisholm)

For educational systems:

Univ. of Edinburgh, Dept. of Artificial Intelligence (Howe, Ross, Pain)
Univ. of Aberdeen (duBoulay)
Open University (O'Shea, Sharples)
Univ. of Leeds (Sleeman)
Medical Research Council, Applied Psychology Unit, Cambridge (Young)

3. <u>How successful are current expert training systems? (3.2, 3.4)</u>

We examined three systems:

<u>SOPHIE</u> – supported students exercising trouble-shooting skills while debugging a faulty electronic circuit. It was an effective Exerciser system (with an underlying expert simulation program), but was dependent upon the students knowing what they were doing. It could answer correct/incorrect or data value questions and evaluate the student's hypotheses.

<u>EXAMINER</u> – an Examiner type system. It criticized student solutions to medical diagnosis problems. It gave effective factual criticism, but was not capable of diagnosing student errors.

<u>GUIDON</u> – a Tutor type system. It attempted to teach medical diagnosis skills by helping the students solve selected cases. In this "state-of-the-art" system, considerable attention was paid to tutorial strategy and dialogue maintenance techniques. Another important feature was the student modelling and GUIDON's efforts 'o deduce the students reasoning. Its tutorial limitations followed trom: inadequate motivation of topics, overly verbose presentations, and the unnatural reasoning methods used by the underlying expert system.

In essence, these systems have demonstrated that the expert training system concept is feasible, and have begun to consider their functions and construction. Overall, we feel that the research has been promising, especially the GUIDON project, but much remains to be done.

4. <u>What is the state-of-the-art in Knowledge Engineering? (2.8, 5.5)</u>

The bottom line is that there are expert systems whose performance is equivalent to the best human experts. In some cases, they are faster or more accurate. Unfortunately, a large measure of this success has come from the basic powers of the computer (speed and accurate bookkeeping) rather than any extraordinary reasoning capabilities (though current methods are considerably more powerful than previous techniques).

There has been success with representing the factual and causal knowledge associated with the subject domains, including methods of representing uncertainty. Further, researchers recognize that there are often multiple views of the same problem, with corresponding multiple sets of expertise. Research on reasoning methods is the least satisfactory. Most methods are unlike human reasoning and, as such, do not contribute to well understood solutions, effective training or the best of frameworks for representing human expertise. There has also not been sufficient research on the strategic aspects of problem solving such as how to select techniques to apply or which factors ought to be considered first.

Research continues on many topics, of which explanation generation and domain-independent expert system frameworks are two of interest to us. The explanation generation research focuses upon explaining what the expert systems knows and how it solved problems. The domain-independent expert system research considers how to construct expert systems from pre-packaged frameworks with adding only domain-specific knowledge.

Most expert systems execute considerably faster than a human expert, and, though computer time is more expensive, the final cost is cheaper. The current development of an expert system is on the order of the time needed to train a human expert, though this is a one-time cost. Future domain-independent expert systems should reduce this cost to 1-2 man years development per expert system.

5. <u>What is the state-of-the-art in Intelligent Teaching Systems? (3.2, 3.3, 3.4)</u>

All the systems discussed in sections 3.3 and 3.4 are limited experiments. Each focuses upon only a few issues: more often an answer to how to build an intelligent teaching system. In a sense, almost every experiment has been a failure from a theoretical viewpoint: unlike the research in expert systems, the methods and techniques developed have not been generalized to a range of problems.

6. <u>Has there been research on teaching procedural knowledge? (3.2, 3.3, 3.4)</u>

At the same time, there has been research on the transfer of reasoning abilities to the students (as well as factual knowledge). This has required a system which is capable of doing the reasoning, observing that reasoning, explaining it to the student and evaluating the student's reasoning. (This research is not conclusive. Problems include: whether the style of reasoning being taught is natural for humans, and whether the student actually learns it.)

7. <u>What other research is of importance to expert system based training aids? (3.2)</u>

Several of the systems built during the 1970's encourage active participation on the part of the student. They have allowed greater flexibility in tutorial strategies while requiring more complicated models of what the student understands, and more involved techniques for evaluating that understanding.

A further set of developments have centered around the type of interface between the student and machine tutor. Current research is concentrating on mixed-initiative dialogues (ie., both parties can select new topics) in a natural language medium. With these innovations, there is then the problem of how the machine tutor manages to keep the session focussed to meet its tutorial goals.

8. <u>How much preparation would an instructor or a student need prior to use of a training system? (6.1, 6.2)</u>

Both the student and the instructor would need a period of classroom and preliminary computer use instruction. For the student, this pre-training would cover:

mechanics of system usage
use of input interface
domain model
domain knowledge structure
reasoning structure/problem solving method

In addition, he/she would be expected to have learned the basic domain knowledge previously. The pre-training (exclusive of domain knowledge) is estimated to require 40 hours.

The instructor will also need an additional training period. This is to cover the actual case problems to be solved by the students, recognizing and repairing system failures, and analyzing logging information. The case problem experience will probably require 100 hours on-line practise, with the other items an additional 10 hours.

9. What roles can expert training systems be expected to fulfill? (4.1)

We have identified five potential roles for an expert training system. They are:

Adviser - provides expert level advice to help user solve the problems
Demonstrator - demonstrates the solution of selected case problems
Examiner - critiques a student's solution to a case problem
Exerciser - provides usage of specific problem solving skills, to either add new skills or sharpen up existing ones
Tutor - has full range of skills of a human tutor: information presentation, problem solving, student comprehension and examination, and diagnosis and correction

8. References

[Aik77] Aikins,J.,"The Use Of Models In A Rule-Based Consultation System", Proc. 5th IJCAI, pg 788, 1977

[Aik79] Aikins,J.S., "Prototypes And Production Rules: An Approach To Knowledge Representation For Hypothesis Formation", Proc. 6th IJCAI, pp 1-3, 1979

[Bal80] Balzer,R., Erman,L., London,P., Williams,C., "Hearsay III - A Domain-Independent Framework For Expert Systems", Proc. 1st NCAI, 1980

[Bar76] Barr,A., Beard,M., Atkinson,R.C., "The Computer As A Tutorial Laboratory: The Stanford BIP Project", IJMMS Vol 8, pp567-596, 1976

[Bar79] Barstow,D.R., "Knowledge Engineering In Nuclear Physics", Proc. 6th IJCAI, pp34-36, 1979

[Ben79] Bennett,J.S., Engelmore,R.S., "SACON: A Knowledge-Based Consultant For Structural Analysis", Proc. 6th IJCAI, pp47-49, 1979

[Bob77] Bobrow,D.G., Kaplan,R.M., Kay,M., Norman,D.A., ,Thompson,H., Winograd,T., "GUS, A Frame-Driven Dialog System", Artificial Intelligence, Vol. 8, pp155-173, 1977

[Bol80] Boley,H., "The Preliminary Survey Of Artificial Intelligence Machines", ACM SIGART Newsletter, #72, pp21-28, 1980

[Bro75] Brown,J.S., Burton,R.R., Bell,A.G., "SOPHIE: A Step Toward Creating A Reactive Learning Environment", IJMMS Vol 7, pp675-696, 1975

[Bro77] Brown, J.S., "Uses of Artificial Intelligence and Advanced Computer Technology in Education", in Seidel and Rubin (Eds), Computers and Communications: Implications for Education", New York, Academic Press, 1977.

[Bro78a] Brown,J.S., Burton,R.R., "A Paradigmatic Example Of An Artificially Intelligent Instructional System", IJMMS Vol 10, pp323-339, 1978

[Bro78b] Brown,J.S., Burton,R.R., "Diagnostic Models Of Procedural Bugs In Basic Mathematical Skills", Cognitive Science, Vol 2, pp155-192, 1978

[Buc78] Buchanan, B.G., Feigenbaum, E.A., "DENDRAL And Meta-DENDRAL: Their Applications Dimension", Artificial Intelligence, Vol 11, 1978, pp5-24.

[Buc81] Buchanan, B.G., "Research On Expert Systems", Stanford Heuristic Programming Project memo HPP-81-1, 1981

[Bun69] Bunderson, C.V., "Projections For A University-Based CAI Activity", Proc. Amer. Educ. Research Assoc. Conf., Los Angeles, 1969

[Bun70] Bunderson, C.V., "The Computer And Instructional Design", in Computer-Assisted Instruction, Testing and Guidance, Ed. Holtzman, Harper and Row, 1970

[Bun71] Bunderson, C.V., The Design And Development Of Quality CAI Programs", in Romano and Rossi (eds), Proc. of Conf. on Computers in Education, Adriatica Editrice, Buri, 1971

[Bun79a] Bundy,A.,Byrd,L.,Luger,G.,Mellish,C.,Palmer,M., "Solving Mechanics Problems Using Meta-Level Inference", in Michie (ed), Expert Systems In The Micro-Electronic Age, 1979

[Bun81] Bundy, A., Welham, B., "Using Meta-Level Inference For Selective Application Of Multiple Rewrite Rule Sets In Algebraic Manipulation", Artificial Intelligence, Vol 16, #2, May 1981, pp189-211

[Bur79a] Burton,R.R., Brown,J.S., "An Investigation Of Computer Coaching For Informal Learning Activities", IJMMS, Vol 11, pp5-24, 1979

[Bur79b] Burton, R.R., Brown,J.S., "Towards a Natural-Language Capability For Computer-Assisted Instruction", in Procedures For Instructional Systems Development, ed O'Neil, Chapter 10, 1979

[Car77] Carr,B., Goldstein,I.P., "Overlays: A Theory Of Modelling For Computer-Aided Instruction", MIT AI memo #406, 1977

[Car79] Carhart,R., "CONGEN: An Expert System Aiding The Structural Chemist", in Expert Systems In The Micro-Electronic Age, Michie (ed), 1979

[Cha79] Chandrasekaran, B., Gomez,F., Mittal,S., Smith,J., "An Approach To Medical Diagnosis Based On Conceptual Structures", Proc. 6th IJCAI, pp134-142, 1979

[Chi79] Chisholm,I.H., Sleeman,D.H., "An Aide For Theory Formation", in Expert Systems In The Micro-Electronic Age, Michie (ed), 1979

[Cla77] Clancey,W.J., "An Antibiotic Therapy Selector Which Provides Explanations", Proc. 5th IJCAI, pg 858, 1977

[Cla79a] Clancey,W.J., "Tutoring Rules For Guiding A Case Method Dialog", IJMMS, Vol 11, pp25-44, 1979

[Cla79b] Clancey,W.J.,"Dialogue Management For Rule-Based Tutorials", Proc. 6th IJCAI,pp 155-161, 1979

[Cla81a] Clancey,W.J., Letsinger,R., "NEOMYCIN: Reconfiguring A Rule-Based Expert System For Application To Teaching", Stanford Heuristic Programming Project HPP-81-2, 1981

[Cla81b] Clancey, W.J., Bennett, J.S., Cohen, P.R., "Applications-oriented AI Research: Education", in Barr and Feigenbaum (eds), Handbook of Artificial Intelligence, to appear

[Dav75] Davis,R., Buchanan,B., Shortliffe,E., "Production Rules As A Representation For A Knowledge-Based Consultation System", Artificial Intelligence, Vol 8, 1977, pp15-45

[Dav77a] Davis,R., King,J., "An Overview Of Production Systems", in Machine Intelligence 8, Elcock & Michie (eds), John Wiley, 1977

[Dav77b] Davis,R., Buchanan,B., "Meta-Level Knowledge: Overview And Applications", Proc. 5th IJCAI, 1977

[Dud79] Duda,R., Gaschnig,J., Hart,P.,"Model Design In The PROSPECTOR Consultant System For Mineral Exploration", in Expert Systems In The Micro-Electronic Age, Michie (ed), 1979

[Eng79] Engelmore,R., Terry,A., "Structure And Function Of The CRYSALIS System", Proc. 6th IJCAI, pp250-256, 1979

[Fag79] Fagan,L.M., Kunz,J.C., Feigenbaum,E.A., Osborn,J.J., "Representation Of Dynamic Clinical Knowledge: Measurement Interpretation In The Intensive Care Unit", Proc. 6th IJCAI, pp260-262, 1979

[Fei79] Feigenbaum,E., "Themes And Case Studies Of Knowledge Engineering", in Expert Systems In The Micro-Electronic Age, Michie (ed),1979

[Fri79] Friedland,P., "Knowledge-Based Experiment Design In Molecular Genetics", Proc. 6th IJCAI, pp285-287, 1979

[Gab80] Gable,A., Page,C.V., "The Use Of Artificial Intelligence Techniques In Computer-Assisted Instruction: An Overview", IJMMS, Vol 12, pp259-282, 1980

[Gas80] Gaschnig,J., "An Application Of The PROSPECTOR System To DOE's National Uranium Resource Evaluation", Proc. 1st NCAI Conf.,1980

[Gen77] Gentner,D.R., "The FLOW Tutor: A Schema-Based Tutorial System", Proc. 5th IJCAI, pg 787, 1977

[Gen79] Genesereth,M.R., "The Role Of Plans In Automated Consultation", Proc. 6th IJCAI, pp 311-319, 1979

[Gol77] Goldstein, I.P., Grimson, E., "Annotated Production Systems: A Model For Skill Acquisition", Proc. 5th IJCAI, pp311-317, 1977

[Gol78] Goldstein,I., "Developing A Computational Representation For Problem Solving", MIT AI memo 495, 1973

[Gre79] Green,C., Gabriel,R.P., Kant,E., Kedzierski,B.I., McCune,B.P., Phillips,J.V., Tappel,S.T., Westfold,S.J., "Results In Knowledge Based Program Synthesis", Proc. 6th IJCAI, pp342-344, 1979

[Gre80] Greiner,R., Lenat,D., "A Representation Language Language", Proc. 1st NCAI, 1980, pp165-169

[Gri80] Grinberg,M., "A Knowledge Based Design System For Digital Electronics", Proc. 1st NCAI Conf., 1980

[Har73] Hartley, J.R., "The Design and Evaluation of an Adaptive Teaching System", IJMMS, Vol 5, 1973

[How73] Howe, J.A.M., "Individualizing Computer-Assisted Instruction", in Elithorn and Jones (eds), Artificial & Human Thinking, pp94-101, 1973

[How78] Howe, J.A.M. "Artificial Intelligence and Computer-Assisted Learning: Ten Years On", Programmed Learning and Educational Technology, Vol 15, #2, pp114-125, 1978

[HPP80] Heuristic Programming Project, Heuristic Programming Project 1980", Computer Science Dept., Stanford University, 1980

[Kan77] Kant,E., "The Selection Of Efficient Implementations For A High Level Language", Proc. Symp. on Artificial Intelligence and Programming Languages, ACM SIGART Newsletter, No. 64, 1977

[Kof76] Koffman,E.B., Perry,J.M., "A Model For Generative CAI And Concept Selection", IJMMS Vol 8, pp397-410, 1976

[Len77] Lenat,D.B.,"The Ubiquity Of Discovery", Proc. 5th IJCAI, 1977, pp1093-1105

[Les77] Lesser,V.R., Erman,L.D., "A Retrospective View Of The HEARSAY-II Architecture", Proc. 5th

IJCAI, pp790-800, 1977

[Les80] Lesser,V., Reed,S., Paulin,J., **"Quantifying And Simulating The Behavior Of Knowledge-Based Interpretation Systems"**, Proc.1st NCAI Conf., 1980

[Lig73] Lighthill, J., **"Part 1: Artificial Intelligence: A General Survey"**, in Artificial Intelligence: A Paper Symposium, Science Research Council Report, pp1-21, 1973

[Lov80] Lovell,K., **"Intelligent Teaching Systems And The Teaching Of Mathematics"**, AISB Quarterly, Issue 38, 1980

[McC80a] McCarty,L., **"Some Requirements For A Computer-Based Legal Consultant"**, Proc. 1st NCAI Conf., 1980

[McC80b] McCune,B., **"Incremental, Informal Program Acquisition"**,Proc. 1st NCAI Conf., 1980

[McD80] McDermott,J., **"R1: An Expert In The Computer Systems Domain"**, Proc. 1st NCAI Conf., 1980

[Mos71] Moses,J., **"Symbolic Integration: The Stormy Decade"**, Comm. of the ACM, Vol 14, #8, pp548-560, 1971

[Nii78] Nii, H.P., Feigenbaum, E.A., **"Rule-Based Understanding Of Signals"**, in Waterman and Hayes-Roth (eds), <u>Pattern Directed Inference Systems</u>, pp483-501, 1978

[Nii79] Nii, H.P., Aiello, N., **"AGE (Attempt to Generalize): A Knowledge-Based Program For Building Knowledge-Based Programs"**, Proc. 6th IJCAI, pp645-655, 1979

[Ole77] Oleson,C.E., **"EXAMINER: A System Using Contextual Knowledge For Analysis Of Diagnostic Behavior"**, Proc. 5th IJCAI, pp 814-818, 1977

[OSh79] O'Shea,T., **"Rule Based Computer Tutors"**, in <u>Expert Systems In The Micro-Electronic Age</u>, Michie (ed), 1979

[Pop77] Pople,H.E., **"The Formation Of Composite Hypotheses In Diagnostic Problem Solving - An Exercise In Synthetic Reasoning"**, Proc. 5th IJCAI, pp1030-1037, 1977

[Rae80] Rae,R., **"Visit To United States"**, Dept. of Artificial Intelligence, University of Edinburgh, DAI Occasional Paper 16, 1980

[Rei80] Reiter, J., **"AL/X: An Expert System Using Plausible Inference"**, Intelligent Terminals Ltd. report, Machine Intelligence Research Unit, University of Edinburgh, 1980

[Ryc80] Rychener,M., **"Approaches To Knowledge Acquisition: The Instructable Production System Project"**, Proc. 1st NCAI Conf, 1980

[Sel77] Self, J.A., **"Artificial Intelligence Techniques In Computer Assisted Instruction"**, Australian Computer Journal, Vol 9, #3, pp118-127, 1977

[Sho76] Shortliffe,E., **"Computer-Based Medical Consultations: MYCIN"**, American Elsevier, New York, 1976

[Sle77] Sleeman,D.H., **"A System Which Allows Students To Explore Algorithm"**, Proc. 5th IJCAI, pp780-786, 1977

[Sle79] Sleeman,D.H., Hendley,R.J., **"ACE: A System Which Analyzes Complex Explanations"**, IJMMS, Vol 11, pp125-144, 1979

[Sle81] Sleeman, D., **"Assessing Aspects Of Competence In Basic Algebra"**, in Sleeman and Brown (eds), <u>Intelligent Tutoring Systems</u>, Academic Press, forthcoming

[Smi80] Smith,D.E., Clayton,J.E., **"A Frame-Based Production System Architecture"**, Proc. 1st NCAI Conf., 1980, pp154-156

[Szo78] Szolovits, P., Pauker, S.G., **"Categorical and Probabilistic Reasoning in Medical Diagnosis"**, Artificial Intelligence, Vol 11, pp115-144, 1978

[Ste79] Steels, L., **"Reasoning Modeled As A Society Of Communicating Experts"**, MIT AI report TR-542, 1979

[Sus77] Sussman,G.J., **"Electrical Design - A Problem For Artificial Intelligence Research"**, Proc. 5th IJCAI, pp894-900, 1977

[Sus80] Sussman,G., Holloway,J., Knight,T., **"Computer Aided Evolutionary Design For Digital Integrated Systems"**, Proc. 1980 AISB Conf, 1980

[Swa77] Swartout,W.R., **"A Digitalis Therapy Advisor With Explanations"**, Proc. 5th IJCAI, pp819-825, 1977

[Tri77] Trigoboff, M., Kulikowski, C.A., **"IRIS: A System For The Propagation Of Inferences In A Semantic Net"**, Proc. 5th IJCAI, pp274-280, 1977

[van78] vanMelle,W., **"MYCIN: A Knowledge-Based Consultation Program For Infectious Disease Diagnosis"**, IJMMS Vol 10, pp313-322, 1978

[van79] vanMelle,W., **"A Domain Independent Production-Rule System For Consultation Programs"**, Proc. 6th IJCAI, pp923-925, 1979

[War81] Warren, D.H.D., Pereira, F.C.N., "An Efficient Easily Adaptable System For Interpreting Natural Language Queries", Edinburgh Dept. of AI Research Paper #155, 1981

[Wat79] Waterman,D.A., "User-Orienting Systems For Capturing Expertise: A Rule Based Approach", in Expert Systems In The Micro-Electronic Age, Michie (ed), 1979

[Wat80] Waterman,D.A., Peterson,M., "Rule-Based Models Of Legal Expertise", Proc 1st NCAI Conf, 1980

[Wei77] Weiss,S.M., Kulikowski,C.A., Safir,A., "A Model-Based Consultation System For The Long-Term Management Of Glaucoma", Proc. 5th IJCAI, pp826-832, 1977

[Wei79] Weiss,S.M., Kulikowski,C.A., "EXPERT: A System For Developing Consultation Models", Proc. 6th IJCAI, pp942-947, 1979

[Wip78] Wipke W.T., Ouchi, G.I., Krishnan, S., "Simulation and Evaluation of Chemical Synthesis - SECS: An Application of Artificial Intelligence Techniques", Artificial Intelligence, Vol 11, pp173-193, 1978

Abbreviations used:

NCAI	National Conference on Artificial Intelligence (sponsored by American Association for Artificial Intelligence)
ACM	Association For Computing Machinery
AISB	Artificial Intelligence and Simulation of Behaviour
IJCAI	International Joint Conference on Artificial Intelligence
IJMMS	International Journal of Man-Machine Studies
MIT	Massachusetts Institute of Technology