Visions in Leisure and Business

Volume 16 | Number 4 Article 5

1998

A Convention Knowledge Based System: An Expert System Approach

Kuan-Chou Chen

A CONVENTION KNOWLEDGE BASED SYSTEM: AN EXPERT SYSTEM APPROACH

BY

DR. KUAN-CHOU CHEN, PRESIDENT

K.C.C. CONSULTING CO.
1200 HAPPY HOLLOW RD. #206
W. LAFAYETTE, INDIANA 47906

ABSTRACT

Expert systems are being used in a variety way. Convention management professionals have yet to embrace this computing trend. In order to gain access to this set of problem solving tools, a review of expert system fundamentals is in order. There are a range of methods for building expert system on micro- computers. Development tools such as expert system shells provide a knowledge engineering environment that facilitates system building by non-specialists. Shells offer a variety of knowledge representations and control strategies that suit various classes of problem solving tasks. System building fundamentals and knowledge representations are reviewed as a precursor to system development. Systems under development illustrate the use of the expert system approach for problem solving tasks that involve both shallow and deep reasoning.

INTRODUCTION

Computer-based information systems pervade almost every aspect of our lives. Their ability to help with problem solving and decision making makes them indispensable in business and management. Businesses continually face problems, which are undesirable situations. Decisions are called for when a course of action must be chosen from among many possible alternatives. Currently data base, spreadsheet, and statistical programs are the tools most widely utilized by tourism and hospitality industry professionals. The importance of these traditional applications will persist. The emerging area of knowledge based systems (KBS) however, will gain prominence where computational elegance is needed to provide real time advice for practical problem solving. Advances in both hardware and software have made the development of knowledge based system feasible for both large and small tourism and hospitality organizations. These "expert systems" strive to emulate the reasoning process of human experts to provide advice where it is scarce, or needed in many locations. Generally expert systems are designed to handle problem solving tasks including diagnosis, prediction, design, planning and monitoring (5).

Management information systems professionals have embraced this technology whereas tourism/convention professionals have not (5). To gain momentum in this arena, a survey of expert system concepts and approaches is warranted. This review intends to in introduce concepts of knowledge based computing and is not comprehensive. There is a plethora of publications available to assist the reader in the development process of expert systems. Many of these are not targeted to computer scientists and the reader is encouraged to consult them for a deeper presentation of expert systems.

BACKGROUND

In the late 1950s and early 1960s, computer scientists tried to build computers that would be able to perform intelligent tasks. The efforts at time were aimed at developing a general problem solver, a machine that would be able to mimic human thought processes, to solve any given problem that a human being can solve. These efforts failed because the programmers needed for the task would have to be unrealistically huge. Scientists realized that they had to concentrate on designing systems to solve much more specialized types of problems. The effort were then directed toward the design of programs to solve problems in specific areas by utilizing experts' knowledge and reasoning. These programs are referred to as expert system. Especially, hardware and software advances in the early 1970's created a means to approach the problem of making computers "think." The area of artificial intelligence emerged in universities and corporations as a discipline involving computer scientists, engineers, and psychologists. By the 1980's expert systems emerged to handle practical problem solving tasks.

The development of early expert systems took place in academic research centers. For example, DENDRAL, a program that identifies molecules from spectroscopic data, was developed at Standford University in 1965; MACSYMA, a solver of complex mathematical problems, was developed at MIT in 1969; MYCIN, which was developed at Standford University in 1973, was specifically designed to diagnose meningitis infections and recommend microbial therapy (3).

PROSPECTOR, a software package developed by SRI International, Inc. to target sites for molybdenum exploration based on geological data input, marked the beginning of the large-scale vending of commercial Artificial Intelligent (AI) applications, starting in 1980. Many other systems followed, developed by commercial designers for commercial applications. Expert systems now designed to help in various domain: medicine, engineering, financial analysis, insurance, and numerous other areas of business and industry (8).

Soon expert system applications began to cross disciplinary lines to approach problems classified as follows:

Prediction	Inferring likely consequences of given situations
Diagnosi	Inferring malfunctions from observable data
Design	Configuring objects under constraints
Planning	Designing actions
Monitoring	Comparing observations to expected outcomes
Debugging	Prescribing remedies for malfunctions
Repair	Executing plans to administer prescribed remedies
Instruction	Diagnosing, debugging, and repairing student behavior
Control	Governing overall system behavior (12)

Expert systems were initially developed to handle narrowly defined problems-solving tasks. Soon the knowledge bases of these programs were isolated from the inference mechanism resulting in generic system building tools called shells. Expert system shells enable the programmer to focus on developing knowledge structures without programming inference strategies. Thus expert systems can more readily apply to problem solving tasks within the realities of time and fiscal constraints.

COMPONENTS OF EXPERT SYSTEM

An expert system consists of a collection of integrated and related components, including a knowledge base, an inference engine, an explanation facility, a knowledge base acquisition facility, and a user interface. Figure 1 displays a typical expert system. In this figure, the user interacts with the user interface, which interacts with the inference engine. The inference engine interacts with the other expert system components. The inference engine is the control of search strategy that brings the knowledge base to life. These components must work together in providing expertise and guidance in the decision-making process.

The User Interface

Specialized user interface software is employed for designing, creating, updating, and using expert systems. The overall purpose of the user interface is to make the development and use of an expert system easier for users and decision makers. At one time, skilled computer personnel were needed to create and operate most expert systems; today, the user interface permits decision makers to develop and use their own expert systems. Because expert systems place more emphasis on directing user activities than do other types of systems, text-oriented user interfaces (using menus, forms and scripts) may be more common in expert systems than the graphical interfaces often used with decision support systems.

Inference Engine

The overall purpose of the inference engine is to seek information and relationships from knowledge base and to provide answers, predictions, and suggestions the way a human expert would. In other words, the inference engine is the control scheme to manipulate the knowledge into producing conclusions. Essentially, it is the problem-solving strategy used to search the knowledge base. A backward chaining inference strategy is initiated by setting a goal for the engine to meet. For example if the GOAL "procedure found" is stated prior to the example knowledge base, a backward search is carried out by the inference engine, querying the user for input until the goal is reached. In many bodies of knowledge, facts cannot be represented with a simple "yes" or "no" queries. In other words, facts are not always strictly true or false in expert problem solving behavior. Thus certainty factors are often assigned to conditions of rules. Certainty factors are almost arbitrary values associated with facts. In the proceeding rule set the first condition of Rule 2 (IF numbers used to rank items are numerical equidistant) may be interpreted differently by a social scientist and a mathematician. The meaning of "equidistant" is variable. To handle this variance, a certainty factor could be assigned by the user to assign a degree of truth to the condition. The user could be asked by the system to rank his/her confidence in the truth of the condition on a scale of 0 to 100. The addition of certainty factors to rules adds a basis for "fuzzy" thinking often used by experts. Clearly, when a knowledge base has hundreds of rules, these certainty factors add a confounding element in the accurate representation of domain expertise.

The Knowledge Base

The knowledge base is the symbolic representation of expertise in a given area. In other words, a knowledge base must be developed for each unique application. The knowledge base is expertise gleaned from domain experts and configured into a formal knowledge representation structure. It often contains rules, facts, attributes, and rules of thumb (heuristics) that represent the proficiency of expert problem solving behavior. The symbolic representation of this expertise may be modeled as semantic networks, frames, production rules, or objects.

Semantic nets represent knowledge as a network of nodes (concepts) linked to each other by relationship describing arches. Frame representations are collections of concepts described by another collection of attributes called slots. Both semantic net and the frame representations are hierarchical structures with lower nodes inheriting attributes of higher level nodes. These methods are therefore especially suited for representing taxonomies in natural systems.

Production Rules

The production rule is conditional statement that links given conditions to actions or outcomes (1). A production rule is constructed using if-then statements. If certain conditions exist, then specific actions are taken or certain conclusions are reached. In an expert system for a tourism forecasting operation, for example, the rules could state that if certain data structure exist with a given time factor, data types, and measurement factor, then a specific forecast model will be made, including time series, cross-sectional, and structural model. For example, the following production rules determine an appropriate procedure for a convention site selection:

Rule 1

IF the criterion variable is scaled interval
OR the criterion variable is scaled ratio
AND there is one criterion variable
AND there is more than one predictor
    variable
AND the predictor variable is scaled
    nominally
THEN implement ANOVA
AND procedure found

This simple structure contains conditions that are possibly unclear. To make the knowledge base usable, these conditions may be represented in more rules. Condition (1) may be satisfied by firing Rule 2.

Rule 2

IF numbers used to rank items are
    numerically equidistant
AND the zero point and measurement
    ends are arbitrary
THEN the criterion variable is scaled
    interval

In order to satisfy the first condition of Rule 1, Rule 2 must first be met. Each condition in each rule may require assessment of additional rules for the rule to "fire." A rule fires when all conditions are met according to Boolean logic. Thus the simple production rule representation becomes an increasingly complex set of interdependent nodes. Typically a acknowledged base will contain from 50 to 500 rules.

Object Oriented Programming

A knowledge representation that is gaining prominence is object oriented programming.

Objects are entities that are descriptions of chunks of knowledge that contain data, attributes, values and procedures. Unlike conventional programming structures, objects contain both data and procedures making the approach suitable for knowledge based programming (9). A group of similar objects comprises a class, which is in turn a member of a metaclass. Class variables and methods are inherited to the superclass forming a latus knowledge structure. Object inheritance simplifies the definition of concepts and is illustrated in Figure 2. Object oriented programs are active taxonomies of domain knowledge where objects send messages between one another to perform the problem solving task. Restructuring our simple rule based representation into classes and objects results in the following (truncated) example.

Metaclass:	statistical methods
Class:	multivariate methods
Variables:	measurement scales
	distribution
	variance
	predictor variables
	criterion variables
Method:	perform procedure
Sub class:	ANOVA
variables:	criterion variables_ 1
	predictor variables_ 1+
	measurement scales criterion_
	interval+
	measurement scales
	predictor_ nominal
Method:	perform procedure ANOVA
	display results
Object:	interval measurement scale
Obj.variables:	zero and end_ arbitrary
	number ranks_ equidistant
Method:	ASK values
	SEND values

The subclass (ANOVA) inherits its variables from the class (multivariate methods) and are instantiated with values. The object (interval scale) contains variables that relate to other measurement scale objects such as (ratio scale).

EXPERT SYSTEMS DEVELOPMENT

Like other computer systems, expert systems require a systematic development approach for best result (Figure 3). This approach includes determining the requirements for the expert system, identifying one or more experts in the area or discipline under investigation, construction the components of the expert system, implementing the results, and maintaining and reviewing the complete system.

The question of the applicability of an expert system approach to a problem is similar to that of assessing experimental, survey, and qualitative research designs. The expert systems approach to problems embodies elements of each of these designs and can be theoretical or applied in nature. Theoretical research in expert systems involves the area of cognitive psychology where the focus is on discovering elements of expert cognition in problem solving. Proponents of this approach insist that until we understand human cognition, one cannot successfully develop valid expert systems. The applied approach is not directly concerned with cognitive processes; it is result-oriented and pursues developing working system by using existing technology.

Before the development of an expert system it must be determined if the approach is applicable to the selected problem. Waterman (12) has suggested that for expert system method to be applicable to a problem the criteria of suitability, justifiability, and appropriateness must be met. Clearly Waterman's criteria are valid for convention management problem solving tasks.

Many problems in convention management are suitable for applying an expert system solution. Of the seven characteristics in Figure 4, the third may be problematical. As in many other domains convention experts do not always articulate their actual cognitive processes. Often experts rely upon expected or trained problem solving methods when reporting them to the system builder. The methodology of knowledge engineering strives to uncover latent decision making heuristics and cross validate them with other reliable sources. The topic of knowledge engineering is too broad to be adequately detailed in this review so the reader is encouraged to consult other materials to fill this gap.

The five elements in 5 illustrate justification for expert system development. As an example, assume one envisioned building a system to aid convention managers in implementing trend measurement methodologies. The task solution would have high payoff by providing a consultant system to carry out methodologies for individual organizations. Also experts in this area are scarce in proportion to the vast array of convention providers. This expertise is also needed in many administrative locations. Although expert system development may be justified, it still may not be the appropriate approach to the problem.

The expert system approach is appropriate for problem solving if it fulfills certain intrinsic qualities (Figure 5) (12). The nature of the task must require symbol manipulation and heuristic solutions. Quite simply the combination of concepts and rules of thumb must form the basis of the problem solving tasks. For the problem to be appropriate it also must have practical value, with no easy solution. The manageability of size requirement mirrors any research endeavor. In expert system development however, this requirement has enhanced meaning. In the development process a problem that appears to be narrow may combinitorically explode due to discovered attributes in the problem solving behavior. Thus the scope of a system is routinely narrowed during the development process as new latent knowledge process are disclosed.

EXPERT SYSTEM DEVELOPMENT TOOLS

Tool selection

Theoretically, expert systems can be developed from any programming language. Since the introduction of computer systems, programming languages have become easier to use, more powerful, and increasing able to handle specialized requirements. In the early days of expert systems development, traditional high-level languages, including Pascal, FORTRANB, and COBOL, were used. (Figure 6). LISP was one of the first special languages developed and used for artificial intelligence (AI) applications. Prolog, a more recent language, was also developed for AI applications. Currently however there are packaged tools available for knowledge engineering called "shells." Shells offer a variety of knowledge representations and problem solving control strategies. Other components of shells include various types of user interfaces, explanation facilities, ability to access other programs, and certainty factors. The most important element in tool selection is the matching of domain characteristics to a particular knowledge representation. Here the system developer should have a firm grasp of the range of representations and a conceptual model of domain fundamentals. The knowledge engineering environment of the tool should be able to provide explanations of queries. The shell also should embody a means to process fuzzy knowledge or handle degrees of certainty in answering individual queries to the user.

System building

There are a variety of methodologies for developing expert systems. Weilinga and Bredeweg (1989), classify these methodologies into those that involve rapid prototyping, software engineering, or life cycle models. The life cycle modeling approach that is most broadly recognized is that of Hayes-Roth et al. (6). This approach is illustrated in Figure 7.

The identification phase can be summarized by the steps of identifying the participants, the problem, the resources, and the goals of the system. The conceptualization phase is where the modeling process will begin. The concepts discovered in the identification phase will be refined and embellished to provide a means to diagram the tasks with relationships made explicit. This step may involve domain experts, written materials, and other reliable sources of knowledge.

Generally, the formalization phase involves imposing the conceptualizations and relationships discovered in the conceptualization phase onto the specific knowledge representation and control structure provided by the development tool shell. Specifically this step involves determining the hypothesis space including developing specific hypothesis for the problem solving task, and determining the granularity of concepts and structure. Granularity refers to the size or level of detail of elements to form meaningful aggregates or "chunks" of knowledge. Another element in the formalization process include determining the underlying behavioral model that will impose logic upon concepts and relationships. Also one must determine a means to deal with uncertainty in the model, and identify hard and soft data. Hard data includes reliable prima facial elements, whereas soft data refers to less reliable, nebulous concepts.

The implementation phase is the actual programming of the system in the programming environment. Here a prototype of the system is developed based on information gained from the previous phases.

Testing of the prototype consists of consulting the system to discover weaknesses in its problem-solving behavior. This not only includes testing the accurateness of diagnostics, but also includes reviewing the representatives and clarity of queries to the user to evaluate if questions are answered in the intended way. The testing process again involves domain experts to aid in evaluating the validity of conclusions drawn from specific case elements. Testing leads to revision of the system to improve its performance.

It must be emphasized that each phase in constructing the expert system creates a feedback loop to earlier phases to refine the model. This evolutionary process is essential in maintaining proper focus and direction in approaching the problem solving task.

APPLICATIONS OF EXPERT SYSTEMS

Expert system methods have found wide application in the area of business management with a focus upon agricultural and forestry related problems. For exhaustive listing of applications in convention management see Davis and Clark (4) and Rauscher and Hacker (9).

The application of expert system for convention management problems is limited. The following applications developed by the author illustrate both easy and difficult problem solving tasks.

RANGER is a prototype expert system developed for the U.S.D.A. Forest Service intended to aid in marketing efforts by giving expert site selection advice and providing a means of monitoring client characteristics.

In the Forest Service setting, tourism and recreational resources are both extensive and diverse. Meeting the specific mix of customer needs to improve satisfaction is problematical because the expertise needed to direct clients to sites is needed in many locations and is scarce. Forest and district level personnel have this expertise yet their skills are focused upon operational elements of resource management. Also the peak tourism season coincides with the fire season so experts are often unavailable for consultation. The Recreational Opportunity Guides (ROG) inventories are extensive yet inefficient or impossible to use by the public. Most recreation inquiries are handled by receptionists who cannot efficiently sort through ROG's to arrive at an optimal site to meet the individual's needs.

RANGER is designed to be operated by untrained staff members or the public to match client needs to tourism and recreation resources. The problem solving task is diagnostic in nature. The consultation session consists of a series of queries that refine the profile of the user to reflect scenic preferences, facility needs, desired activities, and specific needs such as target fish species. The system subsequently searches the knowledge base (partially based on the ROG catalog) to match client needs and recommend a site.

The knowledge base is not entirely passive to client needs. The system can be modified to act in a management mode to place users in sites that meet management objectives. For example, certain sites can be "marketed" while others may be "demarketed." Thus use can be spatially concentrated or dispersed based on management objectives. User profiles also may be used to shift use type. For example, many wilderness users may be more satisfied with semiprimitive resources. RANGER may be modified to identify "fence sitters" and direct them into under utilized sites.

The record of customer profiles provides a market research data base that can be evaluated to assist marketing and planning decision making. Profiles are written into a database which then may be assessed with conventional research methodologies.

In expert system terminology RANGER does not handle "hard" diagnostic tasks that require deep reasoning. Another system under development by the author attempts to handle the deep reasoning of legal decision making in recreation negligence.

Legal rules, precedents, and cases provide a framework for handling the "easy" diagnostic problems. The rules are structured in a hierarchical network to determine the charactenstics of the defendant, plaintiff and conditions surrounding the injury. The litigants subsequently become legal "objects" with attributes and values associated with them.

Diagnosing negligence is a difficult task that is based on deeper level of legal reasoning. This difficulty arises from the nebulous nature of legal terminology. Terms such as "reasonable" are considered open texture predicates because their meaning is a function of case context. In TOTO open texture predicates gain meaning in the context of the case by a process of gradual refinement. Theoretical legal tests are implemented with confidence factors to focus the operational elements of the predicate into a meaningful concept. Thus far the method of bringing meaning to open texture predicates is untested.

CONCLUSION

This review of expert systems introduces concepts and terminology that may be unfamiliar to many convention professionals. We must recall our first contact with computers and subsequent confusion and frustration this experience often created. With this perspective one can see that the development of expert systems can be challenging and ultimately rewarding if one has knowledge of basic principles.

It is impossible adequately review an entire branch of any discipline in a paper of this length. The basic principles presented here intend to provide a basis for approaching problems with this emerging set of methodologies. Expert system development methods are maturing and applications have been developed for both theoretical and applied problems. There is great promise for the application of expert system techniques to existing problems in convention management. Tourism and hospitality professionals are encouraged to add this approach to their collection of problem solving tools.

REFERENCES

1. E. M. Awad, Building Expert Systems: Principles, Procedures, and Applications, West Publishing Company, Minneapolis/St. Paul, 1996.

2. R. A. Benfer, Edward E. Brent, J., and L. Furbee, Expert Systems, SAGE Publications, New Park, 1991.

3. B. G. Buchanan and E. H. Shortliffe, Rule-based Expert Systems: The Experiments of the Standford Heuristic Programming Project, Addison Wesley, Reading, 1984.

4. J. R. Davis and J. L. Clark, A Selective Bibliography of Expert Systems in Natural Resource Management, AI Applications in Natural Resource Management, Vol. 3(3), pp. 1-18, 1989.

5. P. K. Forsberg, A Recreation Liability Knowledge Based System., Michigan State University, E. Lansing, Michigan, 1990.

6. F. Hayes-Roth, D. A. Waterman, and D. Lenat, Building Expert Eystem. Addison Wesley, Reading, 1983.

7. M. P. Martin, Analysis and Design of Business Information Systems. Prentice Hall, Englewood Cliffs, 1991.

8. E. Oz, Management Information Systems, Course Technology, Cambridge, 1998.

9. H. M. Rauscher and R. Hacker, Overview of AI Applications in Natural Resource Management, Paper presented at the Artificial intelligence and growth models for forest management decisions, Vienna, Austria, 1989.

10. R. M. Stair and G. W. Reynolds, Principles of Information Systems: A Managerial Approach, Course Technology, Cambridge, 1998.

11. R. B. Trelease, Connectionism Cybernetics and the Cerebellum, AI Expert, pp. 30-36, 1988.

12. D. A. Waterman, A Guide to Expert Systems, Addison Wesley, Reading, 1986.

13. B. J. Weilinga, B. Bredeweg, and J. A. Breuker, Knowledge Acquisition for Expert System, Springer-Verlag, Berlin, 1989.

Figure 1

Components of an Expert System

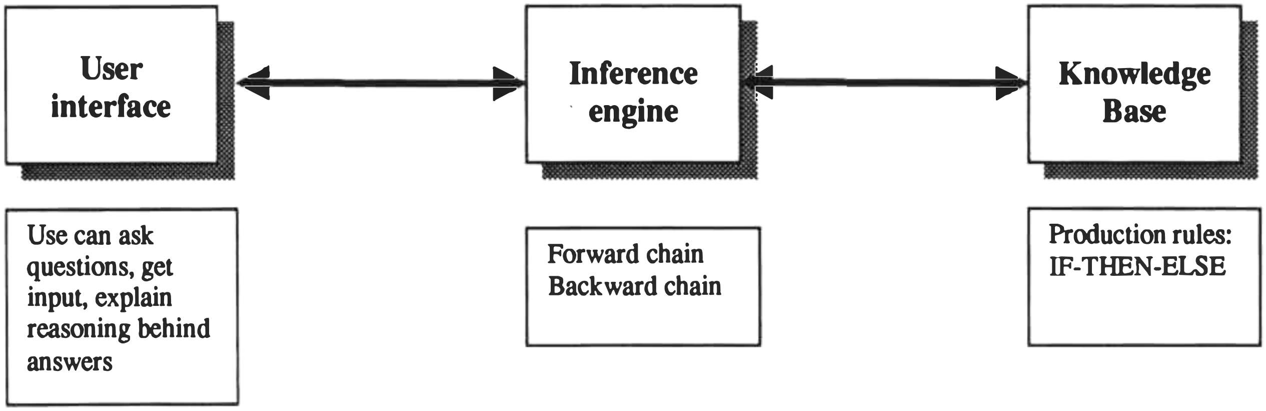


Figure 2

Example of Object Inheritance

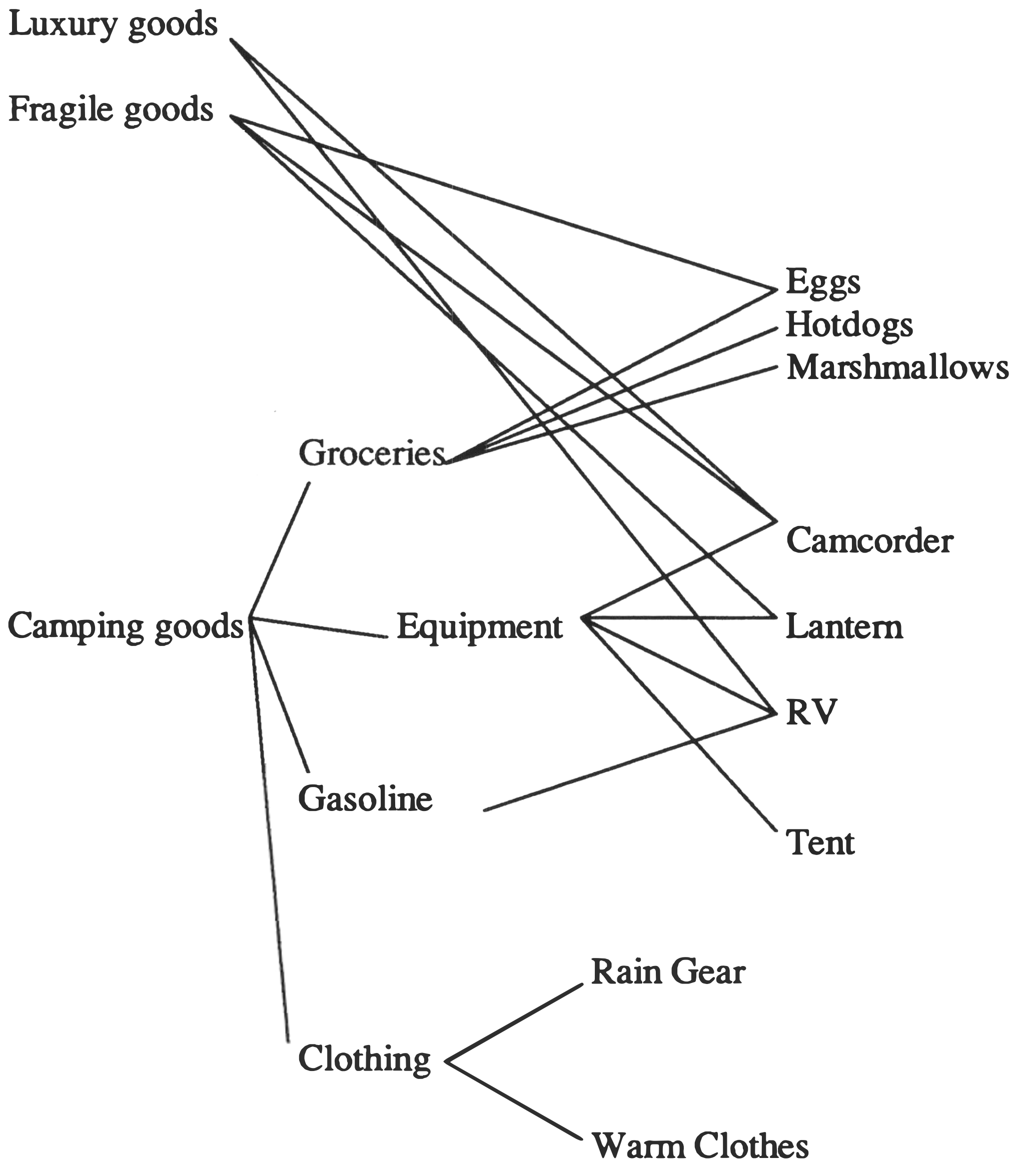


Figure 3

Expert Systems Development Process

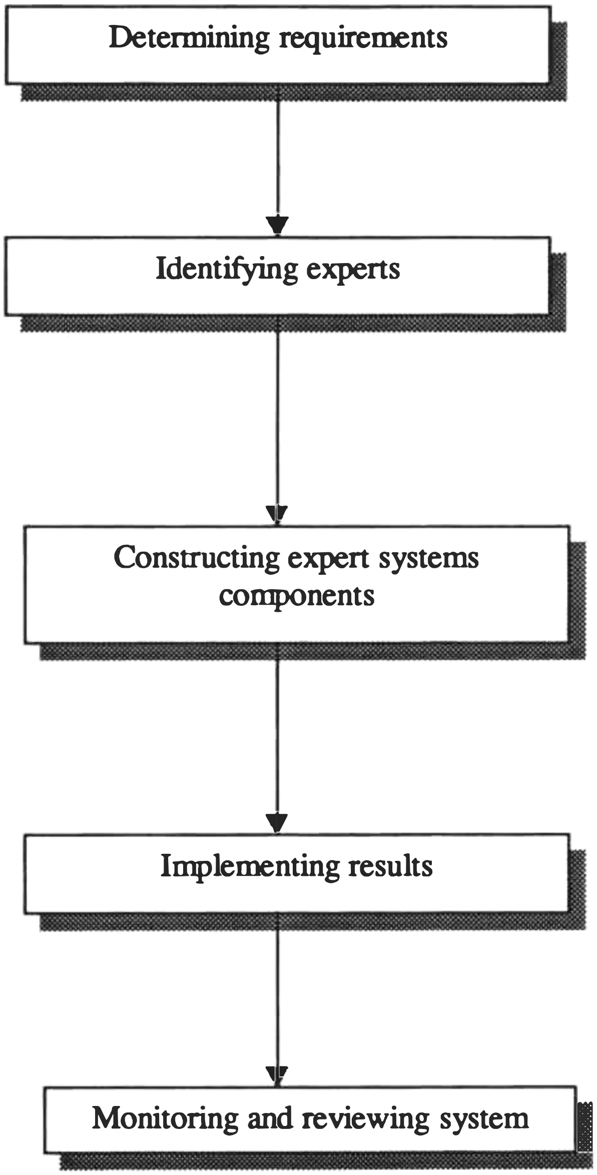

Determining requirements
Identifying experts
Constructing expert systems components
Implementing results
Monitoring and reviewing system

Figure 4

Characteristic Necessary for Expert System Development

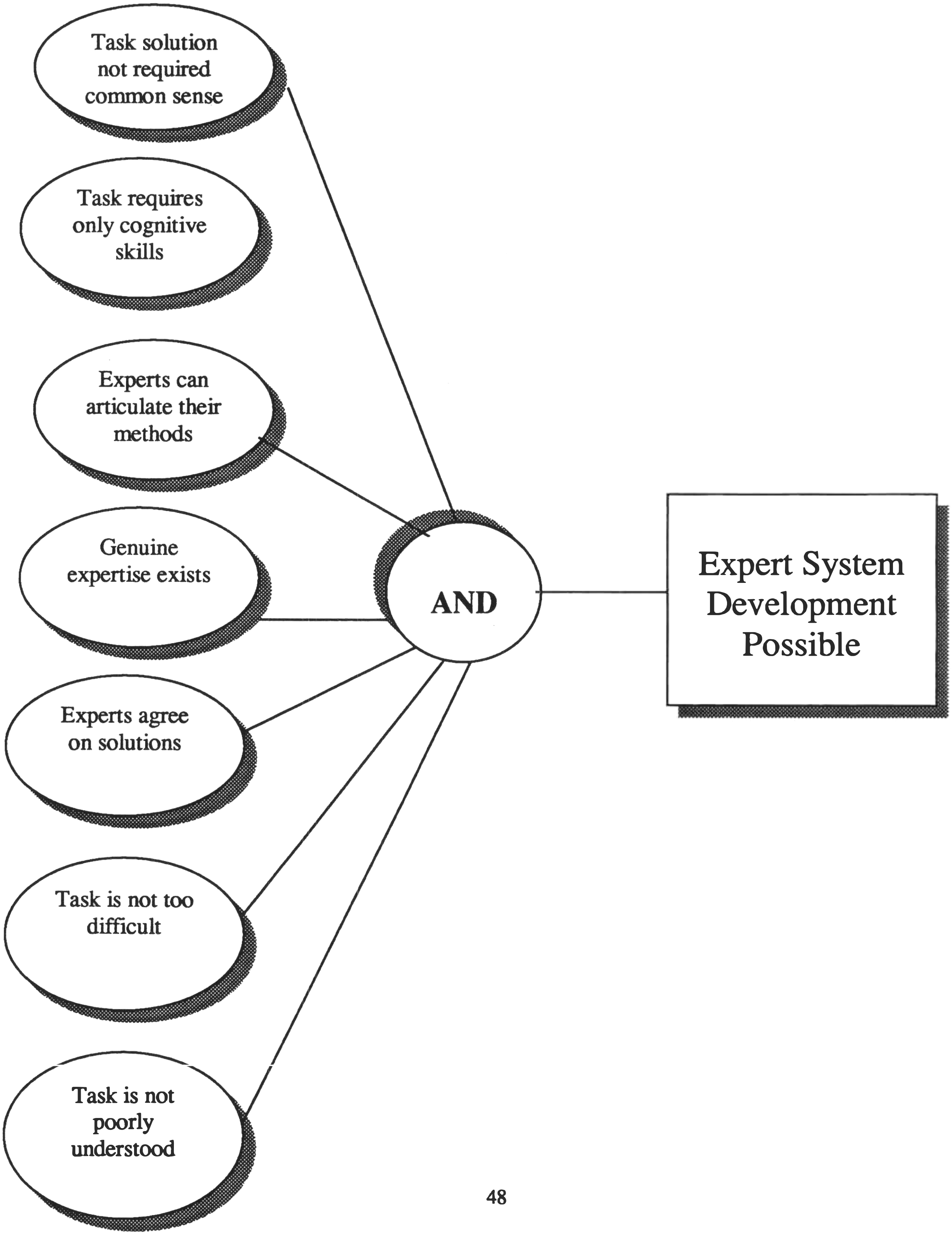

Task solution not required common sense
Task requires only cognitive skills
Experts can articulate their methods
Genuine expertise exists
Experts agree on solutions
Task is not too difficult
Task is not poorly understood
AND
Expert System Development Possible

Figure 5

Factors Justifying Expert System Development

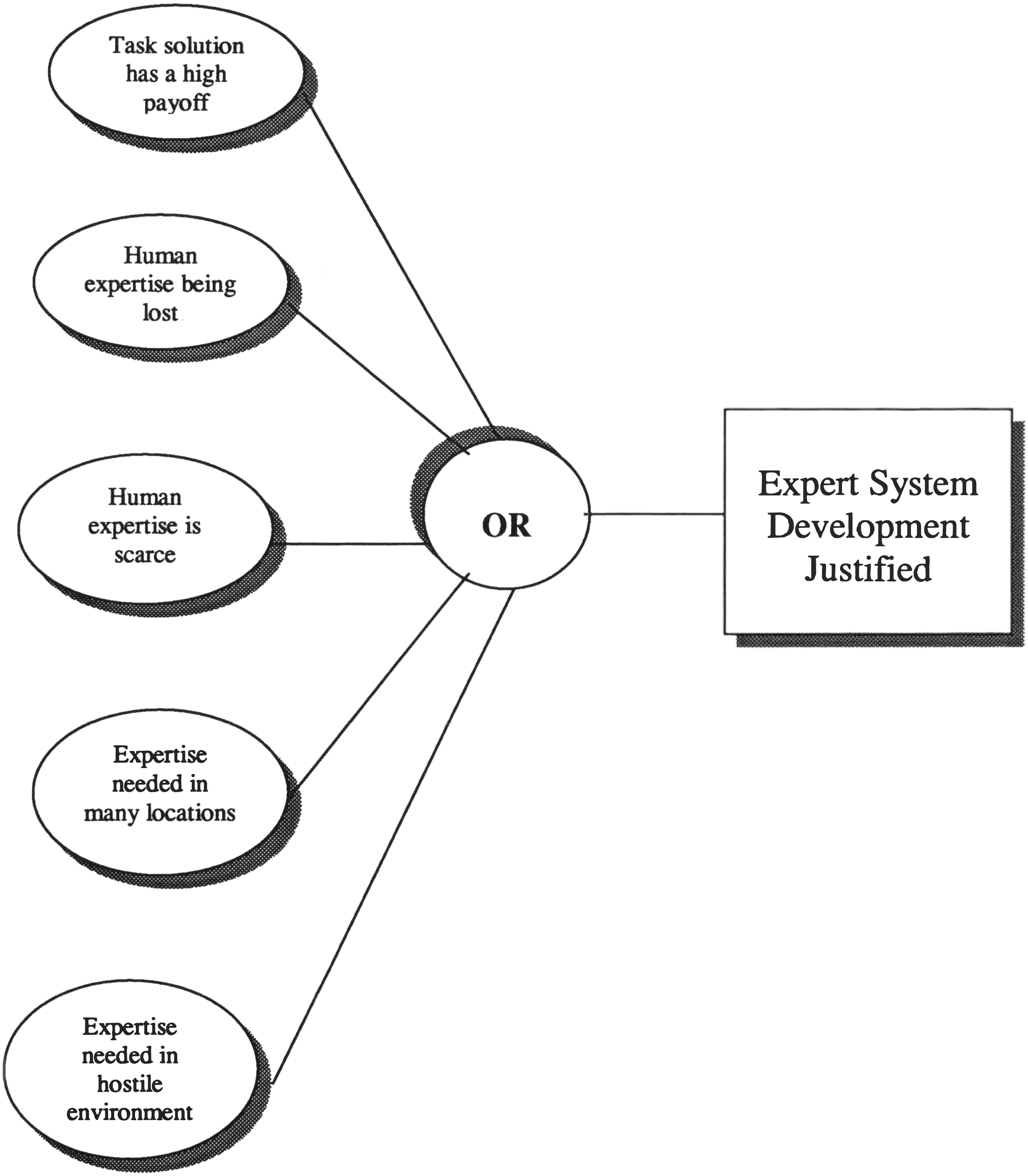

Task solution has a high payoff
Human expertise being lost
Human expertise is scarce
Expertise needed in many locations
Expertise needed in hostile environment
OR
Expert System Development Justified

Figure 6

Software Evolution for Expert Systems Development

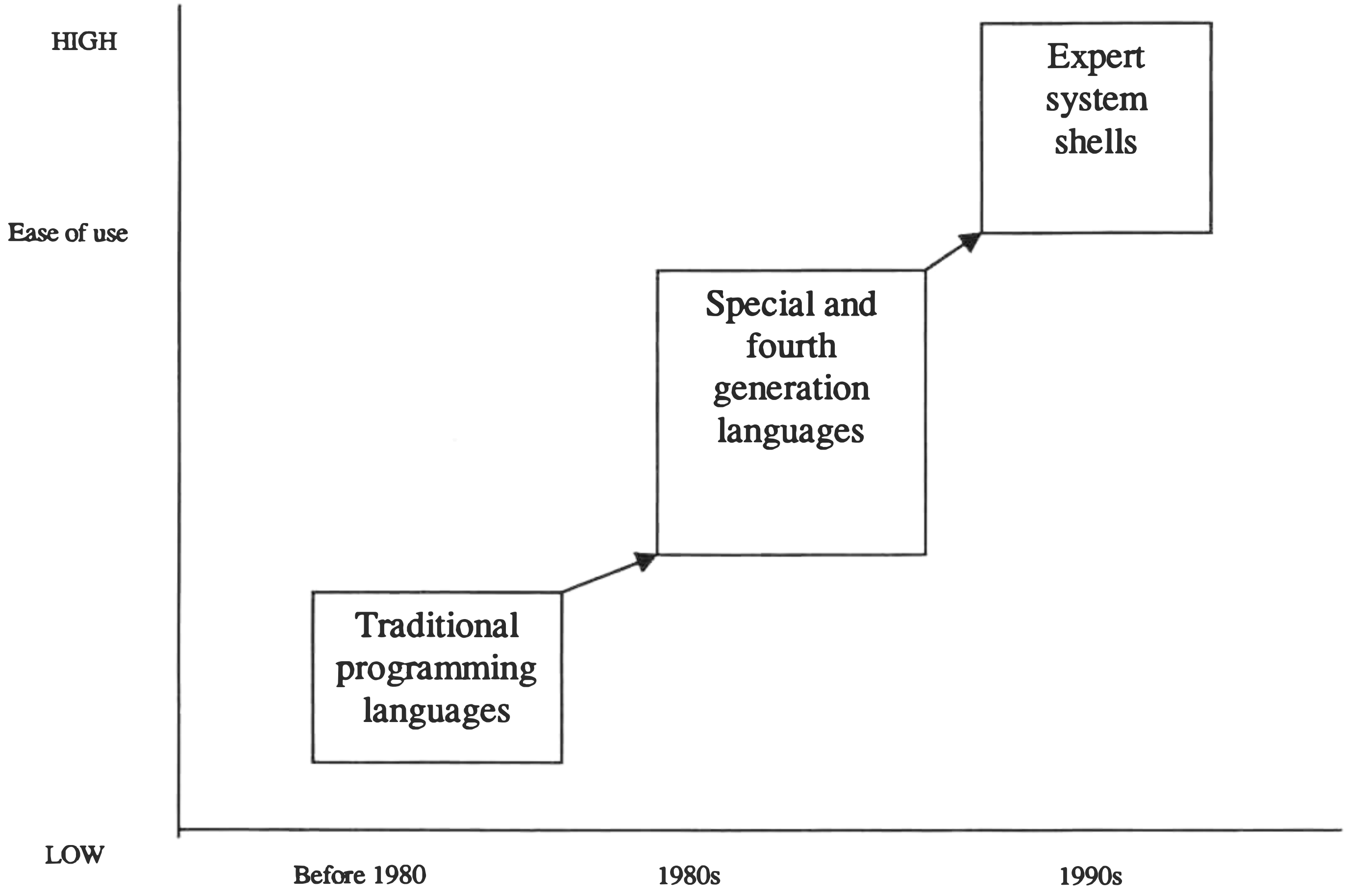
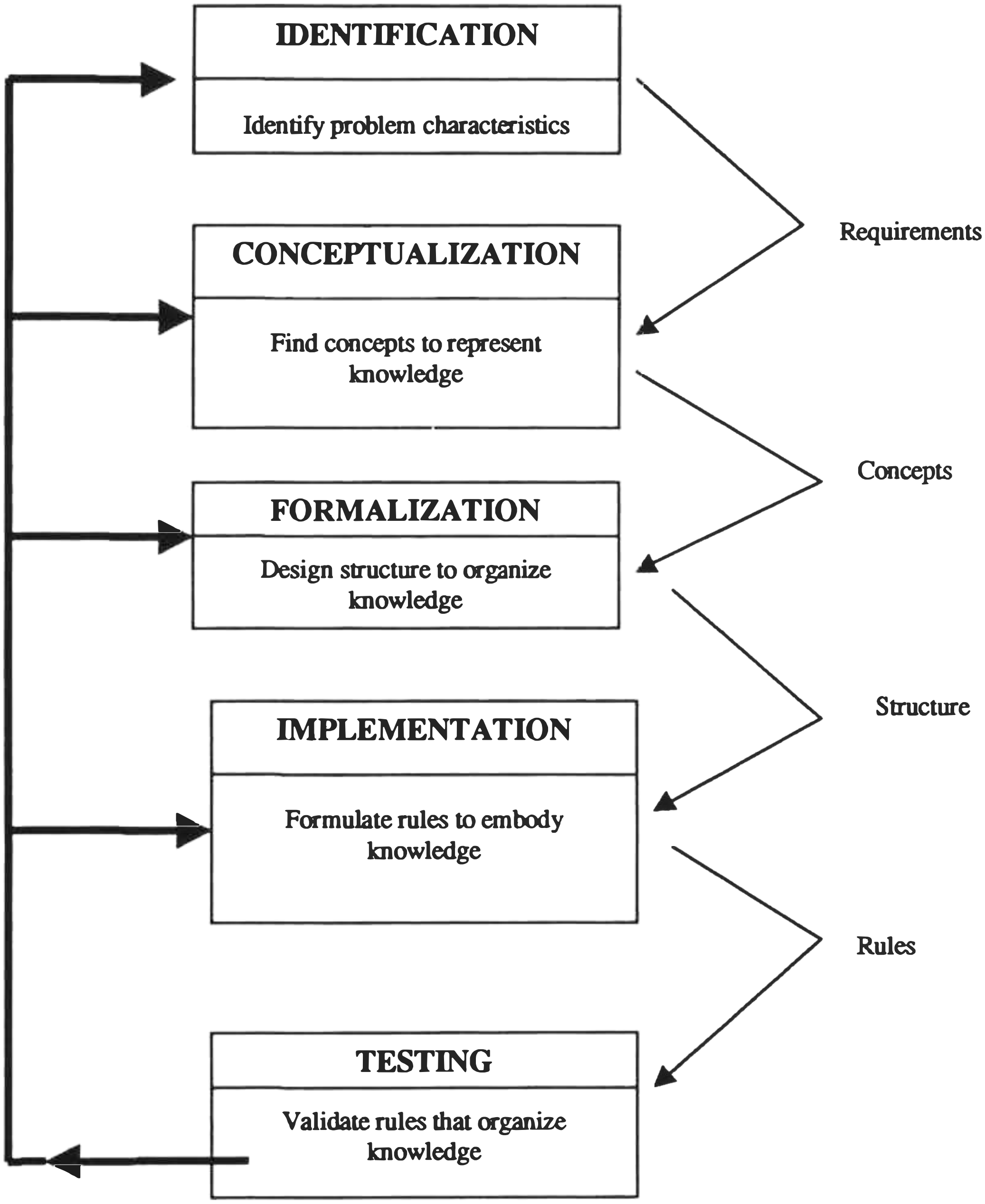

HIGH
Ease of use
LOW
Expert system shells
Special and fourth generation languages
Traditional programming languages
Before 1980
1980s
1990s

Figure 7

The Hays-Roth Life Cycle Model for Expert Systems Development


IDENTIFICATION
Identify problem characteristics
CONCEPTUALIZATION
Find concepts to represent knowledge
FORMALIZATION
Design structure to organize knowledge
IMPLEMENTATION
Formulate rules to embody knowledge
TESTING
Validate rules that organize knowledge
Requirements
Concepts
Structure
Rules