TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER ONE
KNOWLEDGE BASED PROGRAMS AND EXPERT SYSTEMS

In the last decade, Artificial Intelligence has seen the emergence of several programs making substantial inroads into previously intractable domains. Operating in very diverse domains such as speech, vision, chemical structures, experiment design and medical diagnosis, these programs have in common a methodology which has come to be called knowledge based programming. All these programs rely on extensive domain knowledge represented in explicit, easily understood structures. Knowledge representation has thus become a central research topic.

This thesis examines the issue of knowledge representation in expert systems, one type of knowledge based program. Almost all expert systems rely on one of two representation formalisms: a rule-based production system or a hierarchical frame-based conceptual graph. The contention of the thesis in examining these formalisms is that the formalism and the particular representation [1] have a direct impact on power and meaning, essentially forming a meta- level of knowledge.

Consider the simple example of a table listing a store's inventory.

Item	Style	Size	Color	# ordered	# in stock
blouse	459	8	red	10	2
blouse	459	8	blue	10	3
blouse	459	8	eggplant	10	9

Fig. 1.1 A Store Inventory

The formalism of a table is an appropriate choice to organize knowledge in a domain where a group of objects, items of clothing, for example, all have identical properties and the interest lies in specifying the values of these properties. The tabular structure enhances the expression of these regularities by its groupings; all properties of a given item are available simply by reading across the appropriate row and the values of a given property are read down the columns. Contrasting the clarity provided with a paragraph of natural language containing the same information reveals the power that a well-chosen formalism confers.

The formalism, and its embodiment in some particular representation, also limit the type of knowledge and the type of relationships that can be expressed. A table would be an exceptionally bad choice to represent a family tree because it does not allow the relationships inherent in that domain to be expressed.

The formalism and representation of Fig. 1 interact to constrain the knowledge that can be expressed (supplier cannot be included), eliminate ambiguities (eggplant is a color not a vegetable), create expectations about plausible values (102 is probably an error as a value for size in column 3), and justify inferences based on regularities (eggplant is not a popular color for size 8 blouses of this style).

Because any existing knowledge representation formalism is highly constrained in comparison to the wide range of human intellectual activities not all domains can be represented. The successful use of a knowledge based approach requires that the domain has an underlying, recognizable structure which can be captured in the pre-defined configurations of a knowledge representation formalism. The representation should add clarity by providing pointers to the

1.Throughout the thesis the term formalism is used to refer to the abstract concept (any rule, any frame) and the term representation to refer to a particular instantiation which includes the choice of conceptual primitives, functions, etc.

meaning, function and significance of the knowledge. When the formalism and the representation match the domain, enormous gains in power and clarity are made; conversely inappropriate choices are restrictive and opaque. Recognizing appropriateness requires an analysis of the structures and patterns of the domain and an awareness of the implications of choices about the formalism and representation.

The remainder of chapter one examines the nature of knowledge based programs, focussing on the special needs of expert systems in particular, and defining important criteria for evaluating knowledge representation. Chapter two describes the two commonly used formalisms, rule-based production systems and frame-based conceptual graphs and outlines the theoretical arguments that they meet these special needs and criteria. Chapter three then evaluates several examples of these formalisms to determine the extent to which the theoretical claims are born out in practice. Chapter four continues by examining possible representations for the interpretation of personality inventories. Based on the studies and practical experience the criteria of chapter one are reexamined and additional guidelines for evaluation established.

1. KNOWLEDGE BASED PROGRAMMING

An example is the quickest way to illustrate the differences between a knowledge based and a traditional approach.

1.1. A Tic-Tac-Toe Example

Consider the problem of writing a program which wins or draws at tic-tac-toe.

The traditional approach, the exhaustive search of a game tree, is computationally possible. A complete tree consists of 9! or 362,880 nodes. Allowing for symmetries it is possible to search the tree for solutions.

A knowledge based approach is illustrated by the following example. In an informal survey, 40 undergraduates were presented with the problem: Suppose that your 8 year old nephew who knows how to play the game but doesn't win very often asks you to teach him how to win. What would you tell him? The people asked were either unable to respond or did so by producing a list of 6-8 rules which allow a competitor to, at worst, tie each game. The rules were of two forms. First, rules for specific times; e.g. If you play first, put the X in the corner. Second, rules for situations; e.g. If there is a threat block it. A knowledge based tic-tac-toe system requires definitions of the high level concepts such as threat, the rules, and an interpreter which matches the rules to the situation. No search is done.

1.2. Principles of Knowledge Based Programming

In characterizing knowledge based programming, it is important to realize that defining any approach which has emerged from practice normally requires accepting a description of commonalities and characteristics which do not form a sharply defined demarcation, but which form a continuum from the traditional approach to the knowledge based approach along which programs may be placed.

Two of the distinguishing principles of knowledge based programming emerge from the tic-tac-toe example. Additional characteristics of knowledge based programming will be examined later.

The first principle is that domain specific knowledge actively controls the process of the execution. If the issue is a search, for example, the domain knowledge directs the search, focusing the attention in the most useful direction and pruning alternatives to control combinatorial explosion. In the tic-tac-toe example the search is constrained to the point that there is only one possible path in any situation. Thus the first and most significant principle is that the program's success is directly dependent on the extent and quality of knowledge that it contains.

The second principle is that their dependence on knowledge has resulted in generally similar architectures to that seen in the tic-tac-toe example. Knowledge based programs normally have two components: a control structure or inference engine and a database of domain knowledge.

The control structure represents a strategy for problem solving, for example, a plan for ordering the goals and hypotheses. In many cases, it has proved sufficiently general that the inference engine may be removed and used in multiple applications. The inference engines of several expert systems have, in fact, been turned into tools for building new systems in just this way (Hayes-Roth, Waterman and Lenat, 1983).

The database contains extensive domain specific knowledge which forms a model of the domain. It represents the structure and organization of the objects in the domain, expresses the relationship between objects or properties of objects, defines the central concepts and states the heuristics for reasoning in the domain.

2. EXPERT SYSTEMS

Within the AI community the terms knowledge based program and expert system are used synonomously. Outside the AI community the term expert system may mean anything. There is, however, a use for the term which has some utility. A knowledge based program may be considered an expert system if it tackles a problem normally considered to require a human expert and to produce solutions that are acceptable at an expert level. Although the system may be at a research stage, the intent is to provide useful results. The nature of expert systems create specific demands on knowledge representation. Extensive explanation facilities are required and knowledge must both be expressed declaratively and in terms similar to those used by humans reasoning in the domain.

Perhaps the greatest problem that a computer expert faces is credibility. In the medical field, for example, Bayesian techniques have shown human levels of expertise, but the 'black box' nature of the solution has left doctors reluctant to use them (Shortliffe, 1976, 1981). As a result, a successful explanation facility has been a primary goal of all medical expert systems (Shortliffe, 1976), and has, in fact, been considered a defining characteristic of them (Hayes-Roth, Waterman and Lenat, 1983). Without a complete and accurate explanation of the steps in the line of reasoning, people are reluctant to apply the advice in a real situation. This is especially true in domains where human experts frequently differ. Because there is not a single clear-cut answer the reasoning must be evaluated for its correctness in the current case.

For systems which engage in an ongoing interactive dialogue with the user, a second requirement for credibility is that the order of questioning is sensitive to the current context of the consultation and that it appears as a logical sequence of steps in the line of reasoning. The facts must be combined and the questions asked in a credible sequence.

Explanation is important to an expert system for another reason. Clear justifications allow the user to alter or over-ride the advice if circumstances require. For example, if a drug dose is counter-indicated for reasons such as nausea, or hair loss and it is the drug most readily available in a critical situation, it still makes sense to administer it.

Expert systems require a knowledge base drawn from experts who are not usually familiar with computer programming. Practically, the task of obtaining and maintaining co-operation is almost impossible if the expert has to learn a programming language. Instead, control structures and knowledge representation need to be sufficiently clear at some level of abstraction (often through a well-designed interface or authoring system that translates the implementation) that the expert can understand the system's behavior and can formalize his own knowledge for use. The closer the representation can approach the patterns and vocabulary common in the domain the more easily the system may be developed and modified.

2.1. Characteristics of the Domains of Expert systems

Knowledge based programs are difficult to develop. There are few tested approaches which can be selected with confidence in any given situation. More importantly, once an approach has been determined, the knowledge must be painstakingly extracted from the expert and hand-crafted to develop the base. Given these restrictions it is important to identify those characteristics which would predict when a knowledge based approach is appropriate.

2.1.1. Principle of Rationality

Expert systems are useful in problems which are approached rationally as opposed to creatively; in which knowledge is communicated and represented through logical structures rather than through visual images or demonstration. Further, the domain must have some degree of structure and some agreement as to the conceptual primitives and as to the ways of combining them in inferences. Without these constraints successful representation is highly unlikely at this stage of research.

2.1.2. Data

Since traditional programs require exact input, knowledge based programs are useful in situations where the information presented to the system may have data missing or the data may be erroneous, extraneous, or contradictory. In a domain such as signal processing there is the difficulty of distinguishing signal from noise, but in many less obvious domains it may also be very difficult to determine what is relevant data. In a medical diagnosis for example, the patient may present inaccurate descriptions of symptoms or present symptoms which include those of irrelevant conditions such as a cold. Data may be missing due to imprecise measurements or to circumstances peculiar to a given case. Alternately, various aspects of the problem may require multiple, disparate sources of knowledge as specific situations arise during the solution.

2.1.3. Reasoning

The form of reasoning required also indicates the need for a knowledge based program. Traditional programs require a precise statement of method. In many domains, however, the approach is judgemental, relying on experiential knowledge of previous problems, and probabilistic. For example, processes in the domain may not be fully understood requiring reasoning about partial models. Reasoning may have to proceed based on assumptions, made, for example, to correct problems with the data, which may turn out to be incorrect. Experts use stereotypes derived from previous experience to recognize patterns and to suggest strategies and solutions. All of these approaches are tentative. Conclusions may need to be reconsidered and altered as the reasoning progresses.

2.1.4. Combinatorial Explosion

Other domains may present problems due to sheer size. The most typical example is the combinatorial problems inherent in search strategies. Domain knowledge can be used to radically constrain the search space as in the tic-tac-toe example. Another application is to the design process. Many very specific details are only important at a very low level. At higher levels reasoning can proceed more clearly and more quickly if these details can be generalized. Many examples can be found where humans use context and expectation to direct the focus of their attention, use their judgement to select the best alternative to explore first or use hierarchical organization to facilitate reasoning. Expert systems are useful whenever the principles governing these human activities can be structured in some form of heuristic knowledge.

2.1.5. Expanding Research

Typically, experts work in domains where knowledge is incomplete. as research continues, new knowledge emerges. New facts may have to be added to the domain or relationships redefined. Reasoning may alter as processes become better understood. Improved technology may affect the certainty with which judgements can be made. New attributes may be discovered. A system must be able to absorb the new knowledge without substantial re-writing.

2.2. The Demands of Expert Systems

From the preceding discussion of expert systems and the characteristics of their domains, it is possible to summarize three special needs.

2.2.1. Explanation

At the very least, the system must be able to tell the user what information it has about any given aspect of the domain and to provide the steps in the chain of reasoning which led to the conclusions. In addition, to the reasons for a good explanation outlined above, clear explanations help identify erroneous or missing information as the system is being developed and updated.

2.2.2. Ease of Modification

Expanding the knowledge base of an expert system is an on-going process beginning with the initial prototype and in most cases, never ending. It is, therefore, crucial that modifications be done easily. This requires, at least, that the representation be modular so that no side effects occur or that relationships are immediately clear and that the unit of knowledge to be modified is self-explanatory (e.g. its function in the line of reasoning is clear).

2.2.3. Capturing an Expert's Reasoning

The requirement to use the types of reasoning demonstrated by the expert is a practical one. The methods in use by the domain expert are the only ones currently known to work. An expert system must be able to generate the same conclusions from the same evidence. At issue is not whether the computer does it in exactly the same manner. However, unless the method is one whose steps are clear and acceptable to the expert, representing and verifying the knowledge is likely to be difficult.

3. STANDARDS FOR ASSESSING FORMALISMS

Given the crucial role of knowledge, representing it has become one of the central research issues an AI. In order to assess the extent to which representation systems meet these demands some general guidelines must be defined. As will be seen, these criteria are not necessarily compatible.

3.1. Adequacy of Expression

Adequacy of expression refers to the ability of the chosen representation to reflect all necessary distinctions within the given context and, of equal importance, not to force irrelevant distinctions which would only confuse the reasoning. For example, a program recommending the correct wax for cross-country skis must have the ability to distinguish more than a dozen types of snow, distinctions useless in almost any other context.

3.2. Explicitness of expression

Virtually every paper on expert systems uses the term 'explicit'. It is not at all obvious what the various authors mean. For the purposes of this thesis, the argument that the degree of explicitness is a criterion for evaluating a representation rests on the preceding discussion of the special needs of expert systems. Knowledge is explicit if it can be made available in a form and language that is understood by someone familiar with the domain, but not with programming. This excludes all programming languages. The degree of explicitness refers to the extent of the knowledge so available. Ideally, all the domain specific and general control knowledge should be explicit.

3.3. Naturalness of Expression

Not only should the relevant knowledge be represented explicitly, but also the representation chosen should reflect easily and naturally the structure of knowledge in a given domain. There is a subjective element to this, but consider some domain (Kunz, Kehler and Williams, 1981) where the objects and relationships can be represented by a concept hierarchy of 40 nodes each with 3 attributes (Fig. 1.2).

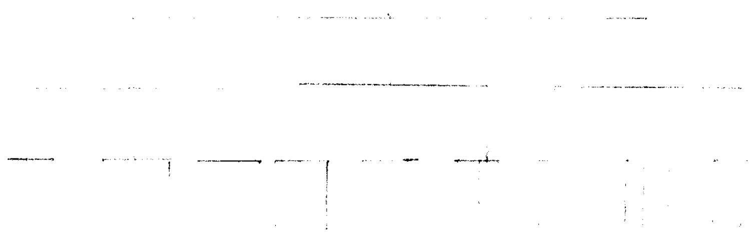

Fig. 1.2 A Hierarchy

Reasoning about the concepts is captured in 75 rules. This is directly and clearly represented by frames or a conceptual graph plus rules. To use rules alone requires 200-400 rules to represent the hierarchy depending on whether the interpreter has a knowledge of inheritance. The ensuing representation is opaque, difficult for a human to follow.

The following chapters will apply these criteria to some existing systems.

CHAPTER TWO
THEORETICAL BASIS OF KNOWLEDGE REPRESENTATION FORMALISMS

The literature now contains research on scores of expert systems at various stages of completion, all purporting to deliver expertise in some domain. Despite the diversity of domains, however, to date, most developers have chosen variations of one of two of the available approaches to knowledge representation: a rule-based production system or a hierarchical frame-based conceptual graph. Both have shown substantial power. Both have also received substantial theoretical development both for knowledge representation in general and expert systems in particular.

This chapter examines these two formalisms. The characteristics of the formalism will be defined, an example of its use in an expert system given, and then the theoretical arguments supporting its use will be considered.

1. PRODUCTION SYSTEMS

A production system consists of three parts: a global database or working memory, a set of production rules and a control system (Nilsson, 1980; Waterman and Hayes-Roth, 1978). The global database may have any structure and holds both permanent and dynamic information. Production rules are implication statements of the form A->B where A may be arbitrarily complex and B is an action which alters the database. In a pure production system the rules are self-contained; no rule references or calls another. The control system reads the global database, selects a rule whose RHS matches a condition or set of conditions in the database and carries out the action specified by the LHS which alters the database. If more than one rule is eligible at any one time then either some conflict resolution strategy is applied to select the rule which will fire or all eligible rules are fired. The process repeats until some pre-determined goal configuration appears in the database.

Two major problem solving strategies lend themselves to production systems: goal-directed and data-driven. A goal-directed system, as its name implies, begins with the goal which is to be achieved. The control system determines the appropriate rules with the goal appearing on the LHS and attempts to establish the right-hand side thus setting up new goals. For example, given the production system of Fig.2.1, the goal is to establish the truth or falsity of C.

```
Database     (D E)

Rules   1. A and B -> C
        2. D -> A
        3. A -> B
```

Fig.2.1 A Production System

Rule 1 is first selected which sets up the two new goals, A and B. Rule 2 is invoked to establish A and Rule 3 to establish B .

A data-driven system selects a rule(s) matching the current state of the database, i.e. the data known at a given time, and executes that rule. Thus a data-driven system could first select Rule 2 which alters the database to (D E A) . Executing Rule 3 would make the database (D E A B). Rule 1 would then assert C.

1.1. Production Systems with Expertise: An Example

Two of the most successful expert systems, MYCIN [2] and R1, are production systems. R1

2.MYCIN is discussed in detail later.

has been in use for four years configuring Vax and PDP computers for DEC (Bachant and McDermott, 1984; McDermott, 1980; McDermott and Steele, 1981). It was built directly in the OPS production system (Forgy, 1979) with the addition of a database which describes the components and their properties necessary for the configuration task (as attribute/value pairs). The working memory contains a record of the steps of the configuration already completed. The rules are structured to recognize the current state of the task and extend the configuration one step further.

```
If: The current context is assigning devices to
        unibus modules
    And there is an unassigned dual port disk drive
    And the type of controller it requires is known
    And there are two such controllers neither of
        which has any devices assigned to it
    And the number of devices that these
        controllers can support is known

Then:  Assign the disk drive to each of the
        controllers and note that the two
        controllers have been associated and that
        each supports one device.
```

Fig.2.2 An R1 Rule

Virtually no backtracking is necessary. Actual experience has shown that the system is highly successful and easily extended.

1.2. Production Systems with Expertise: the Theory

Production systems have been in widespread use since the 1960's and in use as a basis for expert systems for the last decade. Throughout this time the theoretical approaches have been quite constant (Davis and Buchanan, 1977; Davis and King, 1977; Davis and Lenat, 1982; Waterman and Hayes-Roth, 1978). Discussions of production systems for expert behavior have focussed almost exclusively on the power of the set of production rules as knowledge representation. The other two components of a production system, the control structure or inference engine and the global database, have received little detailed attention. The inference engine is assumed to be a simple, general recognizer, easily divorced from the domain knowledge. Thus its impact on problem solving in the domain can be clearly stated by a few sentences stating its recognition and search strategy. It is frequently noted that the global database contains information about objects of the domain, but the impact of this information on the problem solving is rarely detailed. The impression created is that the information is necessary, perhaps for the implementation, but not very significant. Later examination will show these to be serious omissions, but at this point the theory will be stated as it is in the literature.

Rules are clearly not suitable for all domains. The knowledge in the domain must be decomposable into many independent pieces each small enough to be expressed in a single rule, with limited interaction between conceptual primitives. If too many factors are interrelated the RHS quickly becomes cumbersome. As well, the truth of each of the clauses in the premise must be able to be established without affecting the truth or falsity of the others.

For the appropriate domains rules have the following advantages.

[1] Capturing an Expert's Reasoning.

While inference rules do not necessarily model expert thought, the line of reasoning established by the chaining of rules is easily understood and widely accepted. Few people would argue that if A is known to be true and the rules A->B, B->C, C->D, express true relationships that D must also be true. Further, rules are useful in domains where associations are probabilistic or uncertain because the strength of association can be given a weight which then may be propagated according to an appropriate system such as probability or fuzzy logic. A modification to the classic production system is to allow rules which conclude not about the database but about the strategy or other details of the systems. Meta-rules order rules, select strategies, exclude rules and provide a more directed line of reasoning.

[2] Making Knowledge Explicit

Because the rules are stated by human experts in the vocabulary of their domain the knowledge is easily understood by any one familiar with that domain. Also as the example from R1 indicates, each rule states the context in which it applies in its premises. By stating the context, it is guaranteed that rules will not be applied to inappropriate situations. Each rule can then be understood in isolation. Stating the context also allows rules to be used to reason about processes changing over time provided that the process can be decomposed into discrete states and to reason about levels of abstraction in a hierarchy.

[3] Ease of Modification

Each rule expresses an independent chunk of domain knowledge. Because it is independent and because its context is explicit it is a straightforward task to modify old rules or add new ones.

[4] Explanation

Rules, for the reasons outlined above, provide a very powerful and easily accessed explanation facility. Rules are stated in terms appropriate to the domain, they represent the level of detail that a human expert thought appropriate and they are self-contained. A trace of the rule executions should provide an intuitively satisfying explanation.

According to the theoretical arguments, then, rules do provide significant advantages for naturalness, explicitness, explanation and modifiability. More recently, however, researchers (e.g. Aikens,1983) have attacked the inadequacies of rules and suggested frames as an alternative.

2. FRAME BASED CONCEPTUAL GRAPHS

The second major formalism used for knowledge representation in expert systems is the frame-based conceptual graph. A conceptual graph is a graph in which nodes stand for concepts, entities or objects depending on the domain while labelled arcs represent relationships between the nodes (Sowa,1984). Fig. 2.3 gives a simple example.

TABLE-------composed-of--------WOOD

Fig.2.3 A Simple Graph

Nodes themselves may be represented in a variety of ways. In expert systems the most common is the frame. A frame is essentially a structure bringing together knowledge about an object or concept. The frame is named to identify the object or concept it represents. Associated with each frame is a series of slots which identify certain properties of the object considered to be of interest. Each slot is capable of holding a value which is the value of that property for that object. A sample for TABLE might be

TABLE

A-KIND-OF	FURNITURE
FOUND-IN	ANY-ROOM
PARTS	TOP, LEGS
COMPOSED-OF	WOOD

Fig 2.4 A Simple Frame

The value of a slot may be any structure from a single symbol to a complex data type including another frame or a procedure which is triggered as the frame is evaluated. As will be seen, the value of a slot may be a set of rules resulting in a very powerful mixed representation. Fig.2.4 is a simple example. When a frame is part of a conceptual graph the convention is to have the relationship given as the name of the slot and the name of the frame to which it is linked as given as its value. Continuing the example above, the network could contain frames for coffeetable and nightstand whose A-KIND-OF value would be TABLE. These frames could inherit values and procedures defined for TABLE or FURNITURE thus defining a class of properties sharing certain values. The individual items of a particular structure are represented by the instantiation of the generic frames with the appropriate values.

It is important to note that a frame is a highly unconstrained structure. That is, the structure itself provides very little information about what it represents. Fig.2.5 illustrates an abstract expression of a pair of frames.

```
A           C
B    C       F    G
D    E       H    K
```

Fig. 2.5 An Abstract Frame Structure

Frame A is somehow connected to frame C, beyond that nothing can be stated. The rules state a specific relationship. The expressiveness of frames and conceptual graphs comes from the names chosen for the links which define the relationships i.e. it depends totally on a given representation. Contrast this dependence with rules. Since only the RHS of a rule determines when it fires, it always states its immediate context. If any claims about the advantages of so stating the context are true, they must be true for every production system. This is simply not the case for frames.

2.1. Frame-based Conceptual Graphs: An example

PATREC (Mittel, Chandraskaran, Sticklen 1984) is an expert system which acquires and organizes patient data and which uses inferential knowledge to answer questions based on this data for diagnostic reasoning. Domain knowledge is represented as a hierarchy of frames in a conceptual model, patient case data is represented as instances of those frames which are appropriate stored in a patient model. The frames are hierarchical with the top frame being medata, the next varieties of medical data, etc. For example, history, physical symptoms and drugs are types of medata; pain and fever are types of physical symptoms; anaesthetic and analgesic are types of drugs; halothane is a type of anesthetic.

A Conceptual frame

```
(SURGERY
  (LOCATION (ORGAN))
  (PERFORMED? (Default value: No)
       (IF-NEEDED (1. If no surgery
              in an enclosing organ then infer No)
                (2.  If surgery in a component,
              infer Yes)
                (3.  If anaesthetic given, infer
              surgery location unknown))))
```

A Patient-Model frame

```
(DR001
  (TYPE (HALOTHANE))
  (GIVEN? (YES))
  (PATIENT_EPISODE(AT ADMISSION)))
```

Fig. 2.6 PATREC Frames

Notice the use of default values given as answers if no other evidence is available. The IF-NEEDED slots store procedures which can be used to generate answers. Rules 1 and 2 rely on the hierarchical organization of the anatomical knowledge and the inheritance of attributes.

2.2. Frame-based Conceptual Graphs: the Theory

Based in part on the distinction made in Section 2.0, arguments about the merits of a frame-based conceptual graph for knowledge representation focus primarily on the potential of frames.

Frame theory (Aikens, 1984; Kunz, Kehler and Williams, 1984; Smith and Clayton, 1980) represents a significantly different attitude to the issue of knowledge representation. In contrast to rule theorists who stress modularity, frame theorists focus on the importance of aggregating information and representing connections in an obvious and clearly stated manner. All of the following advantages stem from this premise.

[1] Naturalness of Expression

For example, the organization of information in many domains is inherently hierarchical. Many domains have hierarchical taxonomies. Real world objects such as cars or factories are described as nested systems and subsystems. Certain problem solving strategies, successive approximation or establish and refine, for example, work down through levels of abstraction. The links of the conceptual graph allow these relationships to be expressed directly and naturally. By defining the links appropriately, complex relationships can be defined.

[2] Capturing Human Reasoning

Again by defining the links appropriately and by the judicious use of procedural attachments, a variety of types of reasoning can be expressed. Reasoning often proceeds based on some internal model. For example, a doctor may have a model of the progress of a disease and determine how advanced a disease is by matching the state of the patient to some state of the model. Alternatively, he may have several diseases and their typical symptoms in mind and match the present case to these prototypes to determine the closest match. By gathering all the relevant information, frames can express these models and prototypes. An important step in an expert's reasoning is often the recognition of incomplete or erroneous data. When data is grouped in frames it is possible to recognize when data is missing. Also frames can represent plausible values and incorrect data can be identified. Since problems with the input data are common to expert systems this is a significant advantage.

[3] Ease of Modification

Most frame theorists simply reject the idea that rules are in fact modular. Instead, they claim that although frames may appear more complicated to modify, the fact that the relationships are explicit actually makes the task simpler.

CHAPTER THREE
AN EXAMINATION OF EXISTING SYSTEMS

1. METHODOLOGY

As was established in Chapter one, assessing the success or failure of knowledge representation cannot rest solely on the expert level of performance of the system. Performance is necessary, but insufficient. Unfortunately, since the initial theoretical development preceded the development of the expert systems, it was convenient for researchers to present their systems as examples of a stated theory rather than presenting a highly detailed analysis of the reasoning or knowledge representation. Discussions have proceeded at such a high level of abstraction that to evaluate the achievements of a given research project is exceedingly difficult. For example, assessments of naturalness of expression frequently have this flavour: this research uses rules to represent knowledge; rules form a natural means of expression; here are two very natural sounding examples. The examples are impressive, but there is no way to determine if they are typical of the level achieved throughout the system. This is a serious issue because the claims made for the various formalisms are such that they can only be established by their successful application in functioning systems. Knowledge representation can only be assessed, first, if an acceptable definition of a given criteria can be achieved, and, second, if this definition results in specific guidelines and third if a detailed analysis of the representation has been made on these guidelines. Rarely are these standards met. However the literature does contain some detailed studies which allow the theoretical strengths of the formalisms to actually be assessed in practice. Since several of these studies deal with MYCIN or MYCIN-based systems it will be discussed first.

2. MYCIN

MYCIN (Davis, 1977, 1981; Shortliffe, 1976, 1981), is a medical program which deals with the diagnosis and treatment of bacteremia (bacteria in the blood) and meningitis (bacteria in the cerobrospinal fluid). Undoubtedly one of the most successful expert systems developed, it is presented in the literature as a production system and its success is identified with the advantages presented by the rule theorists. The system will initially be described as it is normally presented in the literature.

Since this is a production system the domain knowledge is represented in part by a set of production rules such as shown in Fig. 3.1.

```
If:    (1) The stain of the organism is
               gramneg, and
       (2) the morphology of the organism is
               rod, and
       (3) the patient is a compromised host

Then:        There is suggestive evidence (.6) that the
       identity of the organism is pseudomonas.
```

Fig. 3.1 A MYCIN Rule

Additional information is stored in the global database in a series of tables and in lists of properties associated with the clinical parameters. Typical of the discussions of production systems, however, the papers stress only the contributions of the rules to the problem solving task.

The rules direct the problem solving task by establishing the successive goals that the system will pursue. MYCIN is a backward reasoning goal-directed system.

If: 1) Gather information about the cultures
taken from the patient and therapy he
is receiving,
2) Determine if the organism growing on
cultures require therapy
3) Consider circumstantial evidence for
additional organisms that therapy should cover

Then: Determine the best therapy recommendation.

Fig.3.2 MYCIN's Task Rule

Beginning with the goal of determining therapy (Fig.3.2), MYCIN pursues the successive subgoals of determining if there is an infection, identifying the organism and recommending drugs. Since this is a domain in which its recommendations could have serious consequences, the conservative strategy of pursuing every relevant rule is used. The result is an exhaustive depth-first search. One variation must be mentioned. If, for example, the clause from the premise that is to be established is "If the organism is e.coli" the goal set up is "Establish the identity of the organism". In pursuing goals MYCIN attempts to establish the more general goal. Goals may be established by internal reasoning or asking the doctor for data.

Since this is a highly probabilistic area, MYCIN reasons about uncertainty by the use of certainty factors. Each rule has a certainty factor associated with its conclusion which measures the physician's belief that the premises establish the conclusion. The conditions of the premise in turn have weights which reflect the certainty with which their truth or falsity has been established. The conclusions are asserted with a certainty obtained by multiplying the CF of the rule by the calculated weights of the premises. -1 indicates absolute disbelief and 1 absolute belief.

All of Shortliffe's discussions of MYCIN have stressed that the crucial need for an explanation facility (e.g. Shortliffe, 1981) was a primary design criterion. The solution was to provide an explanation facility that can handle several types of inquiry by displaying the appropriate rule. For example, during the consultation the user may respond "why?" to a question and see the rule that the system is attempting to establish. Successive queries will reveal the chain of rules which represent the line of reasoning. Similarly, after a consultation the user can trace the rules used in establishing the result. Davis's work (1981) has established that this explanation facility is sufficiently powerful to form the basis of an interactive authoring tool and a debugger.

Discussions of MYCIN in the literature are based on the understanding sketched above.

3. EXPRESSIVE ADEQUACY

The most fundamental requirement of a knowledge representation is expressive adequacy, the ability of the system to represent the knowledge necessary to the domain. If the representation is not expressively adequate it obviously can't be natural or explicit.

Cendrowska and Bramer (1984), in an unusual and hopefully precedent setting act, used the published accounts to develop RMYCIN, a reconstruction of MYCIN, with the intent of determining if discrepancies exist between its functioning and the theoretical accounts. In doing so they revealed the extent to which relevant knowledge could be expressed by the rules.

The production system rule set would possess expressive adequacy if all knowledge relevant to the problem solving task could be expressed within it. This proved not to be the case with MYCIN. First, rules, as encoded in MYCIN, proved to be inadequate for the drug therapy selection which was coded directly in Lisp. Early discussions of the system by Shortliffe suggested that this should be altered but no published accounts have shown it was done. It is therefore not clear whether it can be done in any appropriate way. Second, the global database also contains lists of the sterile and non-sterile sites, a table of staining characteristics, morphology and aerobicity of each organism known to MYCIN and a table of normal flora associated with each site known to MYCIN. Clearly these represent important segments of the domain knowledge. In addition, types of objects in the domain called contexts, patient, culture, organism, have lists of properties called parameters. Cendrowska and Bramer discuss in some detail these context types and clinical parameters normally ignored in discussions of MYCIN as a production rule system. These prove to have a substantial impact on the line of reasoning. The parameters enable the program to invoke the correct rules, ask questions in the appropriate context and specify plausible values. One attribute specifies that the system iterate until all contexts of this type are instantiated. Also associated with the context types are lists of questions to be asked, plus a list of clinical parameters to be established, as soon as a context of that type is established. Instantiating a current organism context, for example, causes the system immediately to trace rules to establish the identity of the organism. Because the information gathered in this manner influences the order in which rules will be traced for the rest of the consultation, the impact can be quite widespread and obscure.

One of the most important consequences of the situation described above is that none of this knowledge is explicit. The explanation facility, reflecting the orientation of the production rule theory, is concerned exclusively with the rules as sufficient explanation for developing the line of reasoning and drawing conclusions. Therefore, knowledge not in the rules cannot be made explicit. There is no mechanism to display the tables which are used to determine certain facts during the consultation, for example.

The work on RMYCIN has established that considerable significant knowledge is not expressed in the production system representation. It is not only the match of conditions to the current state of working memory which cause a rule to fire. Nor is information put in the working memory solely as the result of matching a rule's LHS. Nor is there any way to explain how the attributes of contexts and parameters affect the problem solving.

4. EXPLANATION

One of the major strengths of MYCIN, as it is reported in the literature, is its powerful explanation facility. Like all production systems its explanations are based on displaying the relevant rules as justification of its reasoning. Clancy (1981) set out to use MYCIN's knowledge base to develop a tutorial program for medical students. Given the existence of these facilities he assumed his task would be straightforward. Instead he found that the relationships within the system were implicit and complex.

What he found was that the expert's understanding of the domain knowledge, his problem solving strategy and the function of the rules was implicit and often difficult to ferret out. The rules were neither modular nor self-explanatory. Developing the new explanation facilities required many hours with the rule authors articulating the implicit knowledge.

An analysis of the following MYCIN rule illustrates his major criticisms.

If: 1)The infection which requires therapy is meningitis
2)Only circumstantial evidence is available for this case
3)The type of meningitis is bacterial,
4)The age of the patient is greater than 17 years, and
5)The patient is an alcoholic,

Then: There is evidence that the organisms which might be causing the infection are diplococcus-pneumoniae (.3) or e.coli (.2).

Fig. 3.3 An Alcoholism Rule

First, the clauses of the rule are not logically independent. Viewed procedurally, the clause order represents a strategy of top-down refinement: is there an infection? is it meningitis? is it bacterial? is it diplococcus-pneumoniae? Given MYCIN's depth-first search strategy the search is essentially determined by the order of the clauses. Another relationship between the clauses is illustrated by clauses 4 and 5. Clause 4's function is to prevent asking the user about alcoholism (with the resulting lack of credibility) when the patient is a child. Other examples of screening clauses express set, subset relationships.

Second, the logical connection between premise and conclusion is often not clear because the rules express a variety of relationships. MYCIN uses the implication links to express class properties, social and domain facts, cost/benefit judgements plus causal relationships of several sorts. These causal relationships may be empirical association (a correlation for which the process is not well understood), a complication (the direction of causality is known but the conditions of the process are not understood) and mechanistic (the process is well-modeled). The nature of the connection can be important to understanding the significance of a piece of information. It may not be clear, for example, whether or not the relationship is mechanistic or correlational, but situations are much more likely to alter correlational data over time than mechanistic.

Third, patterns existing across rules affect the meaning of individual rules. This rules seems to suggest that alcoholism contributes a certainty factor of .3 to a diagnosis of diplococcus-pneumoniae. However, Clancy found that another rule states the same relationship with a lower certainty factor for situations with hard evidence. The certainty factor thus doesn't reflect the likelihood of alcoholism indicating the disease, but rather the importance that alcoholism should be given in the presence or absence of other factors. Forty other rule pairs reflect this distinction between hard and soft evidence.

4.1. Meaning of Explanation

Clancy's work seeks ways to make the strategies and concepts underlying MYCIN's rules explicit and thus explainable. In so doing he attempts to define the standards by which the explanation facility of an expert system should be evaluated. Explanation is making clear identities and relationships pertinent to the line of reasoning in this domain: how a request for data is related to a goal; how one goal leads to another; how a goal is achieved. He is concerned with two types of explanation. The first is to explain a rule by justifying the association of premise and conclusion. The second is to explain a strategy, the plan for problem solving being followed, the plan by which goals and hypotheses are ordered in the solution space. In both these cases, he defines a good explanation as one which relates the material to a generalization which is within the user's experience.

Given this position, the explanation process is not the same for all rules. Rules which state definitions or identify properties of classes cannot be explained beyond stating them. Rules which state social or world facts are either self-evident or their explanation lies outside the domain; for example, if the patient is a male, he is not breast-feeding or pregnant. The causal rules should relate to a general model of the disease process. For bacterial infections the explanation would connect the rules to one of the following stages: the entry of organism into the body; passage to site of infection; the reproduction of organism ; the causing of observable symptoms. This then is the explanation NEOMYCIN would give for the alcoholism rule of Fig.3.3.

> The fact that the patient is an alcoholic
> allows access of organisms from the throat
> and mouth to lungs (by reaspiration
> of secretions).
>
> The fact that the patient is an alcoholic
> means that the patient is a compromised host
> and so susceptible to infection.

Determining the level of explanation is a subjective matter. The alcoholism example leaves unexpressed that diplococcus is normally found in the mouth and proceeds from the lungs to the meninges by the blood. These are steps in the disease process it is assumed need not be explained.

The explanations generated by NEO-MYCIN are quite removed from the actual rules used to drive the reasoning. The explanation facility can recognize such factors as screening clauses and the clauses which did not represent medical information pertinent to the current stage of the consultation were removed. In the vast majority of cases when context and screening clauses are removed only one clause remains, relating to one of the stages in the disease process. If more than one is left, then one of three types of relationships exist between the clauses: first, a confirmed diagnosis explains a symptom in the current case; second, two symptoms together are different than one alone; three, weak circumstantial evidence is made irrelevant by stronger evidence. These relationships can then be recognized by the explanation facility and expressed explicitly. Additional information was also added to the system to associate the data parameters of the existing rules with the correct stages.

Clancy further believes that the appropriate level of generalization for control knowledge is domain independent heuristics. He states these as meta-rules. Some of these rules are very general, group and refine the hypotheses; others less so, consider common causes before uncommon causes.

Note that Clancy has not altered the rule-based structure of MYCIN. He has added additional properties to the parameters, added a group of meta-rules, and written a powerful program to translate the rules into statements which make the underlying concepts explicit. His claim is that rules with the loosely associational links are the appropriate representation for this domain because processes are poorly understood. The rules are concise, empiric and efficient. Relating information of the current case to a process is more a check on the reasoning than a practical way to drive it. Because the underlying structure remains the same, the problems identified in examining RMYCIN remain. The program which addresses these concerns directly is CENTAUR.

5. CENTAUR

CENTAUR (Aikens, 1983) represents a very significant process in research; the re-working of the same problem in another form to compare the results. Initially, EMYCIN was used to build PUFF, a system for the interpretation of pulmonary function tests. The system was judged successful in comparison with human experts, but certain problems arose representing prototypical patterns, adding or modifying rules, altering the order in which knowledge is requested, and providing explanation. Aikens argues that many of these problems arose because the rule formalism chosen was not a natural fit for a domain which is a natural hierarchy. Once the hierarchy is represented directly the expert's reasoning can be captured more accurately and modifications and explanations are more straightforward.

For these reasons, Aikens has chosen to represent the PUFF knowledge base in frames. Prototypes represent the disease types in the domain. (Fig.3.4) Each prototype has associated with it a series of frames called components.

```
Obstructive Airways Disease Prototype

Author: Aikens
Date:
Source: Fallat
Pointers: (degree MILD-OAD) (degree MODERATE-OAD)...
          (subtype ASTHMA)  (subtype EMPHYSEMA).....
Hypothesis:  There is Obstructive Airways Disease.
If-Confirmed:  Deduce the degree of OAD
               Deduce the subtype of OAD
Action: Deduce any findings associated with OAD
        Print the findings associated with OAD
Fact-Residual Rules: Rule 157, Rule 158.......
Refinement Rules: Rule 036, Rule 038, Rule 039.....
Summary Rules:  Rule 053, Rule 054......

Components:

Total Lung Capacity          Plausible Values: >100
                        Importance Measure: 4
Reversibility               Inference Rules: Rule 019,
                            Rule 020....
                        Importance Measure: 0
```

Fig. 3.4 CENTAUR Frames

The frames form a hierarchy as indicated by the Pointers slot. Three other prototypes are the Consultation Prototype which controls the stages of the consultation through its control slots, the Review prototype which summarizes the consultation and generates the report and the Pulmonary-Function prototype which represents information common to all prototypes.

Several advantages arise from this representation. First, the control is sensitive to the context because it is attached to the prototypes. Consider for example the If-confirmed and Action slots. Through these the actions appropriate to a limited context can be specified. The rules are also tied to prototypes and are therefore more sensitive to context. In PUFF if the system was considering the reversibility of a disease, all rules concluding about reversibility were considered; here, because the rules are attached to components and hence to specific diseases only those relevant to that disease are considered. Second, many of the complexities can be removed from rules. Making the context specific in the frames means that the context need not be given in the rules. Slots allow default values to be set. Therefore clauses in the rules are more likely to be concerned with inferring a piece of medical knowledge. Third, the functions of rules should be clearer because they are attached to slots which identify their function. Fourth, many of the values which were represented in the clinical parameter list are now explicit.

While much of the static knowledge has been made explicit, the control structure is necessarily more complex. PUFF requests the test results which trigger certain prototypes. The filling in of the slots of the prototypes invoke other prototypes or components. These invocations result in weighted tasks being put on an agenda. Tasks can also be placed on the agenda by other tasks or by the control slots of prototypes.

The representation does not cure all ills. The underlying strategies are still unstated. The approach is match and refine, but this is not available as an explanation. The control structure is considerably more complex than in a production system and understanding of it relies on the interpretation of the phrases in the control slots. Not all of these are very informative. What action will "Deduce any findings associated with OAD" produce? If the user can't guess then this is not explanatory. Further control is buried in unlikely slots. Plausible values, for example, are really situation/action pairs where a value may trigger an action which makes a conclusion, prints a statement, triggers a proto-type or does nothing.

The approach that CENTAUR follows is hypothesis and match. Representations of classes of hypotheses are matched against the actual data of the case. For this approach to be useful a domain must have a natural hierarchy of prototypes which can be defined. Also, unless some standard set of input data is presented to be matched the systems degenerates to exhaustive search.

6. MODIFYING KNOWLEDGE

All of the preceding studies have had substantial implications for the ease with which an expert could develop and modify knowledge in the system. An alternative approach, still very much at the research stage, is to have the system derive its own knowledge. At least one study (Michalski and Chilausky, 1980) has shown that in simple, well-defined domains the rules derived from examples supplied by experts were superior to those supplied by expert's directly. Examining a system which can modify its own knowledge highlights some of the knowledge representation factors which would affect modification.

AM (Lenat 1980) was written not to provide consultation but to make scientific discoveries by adding to its own knowledge. AM began with a limited amount of set theory expressed as the instantiated slots or facets of frames. Each frame represented a concept. Given a large body of heuristics, AM was to 'discover' interesting new math concepts and to learn new knowledge about existing concepts. Fig. 3.5 illustrates a sample frame representing a concept discovered by AM. Frames are organized hierarchically by specialization and generalization which allows heuristics to be inherited. AM proceeds to fill in a facet of some concept (instantiate a slot in a frame) or to create a new concept by picking up the top job on the agenda and executing it. Typical jobs might be "Fill-in examples of primes" or "Fill in new algorithms for set union". Jobs are proposed by heuristic rules which rate the interestingness of concepts

```
NAME: Prime Numbers
DEFINITIONS:
  ORIGINS: Number-of-divisors-of(x)=2
  PREDICATE-CALCULUS: Prime(x) iff
    (For-all z)(z/x implies z=1 XOR z=x
  ITERATIVE: (for x>1): For i from 2 to x-1, not (i/x)
EXAMPLES:2,3,5,7,11,13,17
  BOUNDARY: 2,3
  BOUNDARY-FAILURES: 0,1
  FAILURES: 12
GENERALIZATIONS: Numbers, Numbers with an even number of divisors,
    Numbers with a prime number of divisors
SPECIALIZATIONS: Odd primes, Prime pairs, Prime uniquely-addables
      .
      .
      .
```

FIG. 3.5 AM's Prime Numbers Frame

and determine what to explore next. For example, if very few examples exist it might be interesting to explore a more general concept. If a function is interesting and its inverse exists then investigate the inverse. Heuristics also suggested how to define new concepts. Heuristics are attached to a facet of the most general concept to which they apply and are inherited by concepts, including those actually created by AM, which are specializations of it.

The following example illustrates how AM's heuristics are used to modify its existing knowledge. One of the slots of each concept stores a recursive algorithm to compute the

```
FAST-ALG (lambda(x y)(equal x y))
RECUR-ALG (lambda(x y)
      (cond((or(atom x)(atom y))(eq x y))
           (t(and
              (list-equal(car x)(car y))
              (list-equal(cdr x)(cdr y))))))
```

Fig. 3.6 List Equal

function. Fig. 3.6 gives the function for list-equal. In the course of its runs AM applied its heuristic which states that to generalize a function with two conjoined recursive calls, remove one of the recursive calls or replace the AND with an OR. Removing the recursion on the CAR produced a function which returned true if both lists are of the same length, the crucial step in establishing the concept of natural numbers.

Lenat himself thought, at the time that AM was written, that it had two lessons. One, that simple production architecture could not support a program designed for scientific discovery. He mentions, for example, the need for multiple data structures instead of uniform memories, complex access functions instead of read/write primitives and the need for data structures to reference each other as in the General/Specialization slots. Two, that to continue to be successful, AM needed to discover new heuristic rules to reason about the new concepts created. As AM moved further from the original concepts and their heuristics its performance degenerated. The heuristics were still valid, but too general or too weak. For example, while considering the addition of prime numbers, AM had only heuristics about set union. This second lesson proved to be very important because it led Lenat to study the nature of heuristics (Lenat, 1980) and to work on

EURISKO, a new system applying the same methods to the discovery of heuristics.

EURISKO's initial failure led to a detailed study of the representation and a more fundamental analysis of AM. Lenat claims that it is "the density of worthwhile math concepts as represented in LISP that is the crucial factor." EURISKO needed a new representation which matched the domain as well as LISP fit the domain of mathematics.

The major problem was one of expressive adequacy. AM's heuristics were long pieces of code (many as long as two pages of Lisp) containing many details, such as updating local variables, irrelevant to the meaning of the heuristics as a piece of mathematical knowledge. It was too fine grained, containing syntactic distinctions not meaningful at the level of knowledge of the domain. The solution, evolved over four years of work, was a new representation language. Fig. 3.7 gives the heuristic, its original Lisp representation and the new EURISKO.

Lisp function

```
IF: (AND(EQ Cur-action 'Find)
        (EQ Cur-Slot 'Examples)
        (MEMBER 'Collection(IsA Cur-Concept)
        (SETQ f(SOME(Examples 'Function)
              '(LAMBDA(g)
                (DoesIntersect(Range g)
                (Generalizations Cur-Concept))))))
THEN: (SUBSET(MAPCAR(Examples(Domain f))
             '(LAMBDA(x)
              (APPLY*(Alg f)x))
             '(LAMBDA(e)
              (APPLY*(Defn Cur-Concept)e))))
```

EURISKO Form

```
IfCurAction:      Find
IfCurSlot:  Examples
IfCurConcept:     a Collection
IfForSOme:        a Function f
IfIntersects:     (Generalizations CurConcept)(Range f)
ThenMapAlong:  (Examples(Domain f))
ThenApplyToEach:(Alg f)
CollectingIfTrue:(Defn Cur-Concept)
```

Fig. 3.7 A EURISKO Frame

The new structure represents a functional decomposition along the dimensions of slots and values where the distinctions are now at the appropriate level to be semantically meaningful.

CHAPTER FOUR
AN EXPERIMENT IN KNOWLEDGE REPRESENTATION

In the light of some of the discrepancies between the theoretical claims and the practical experiences reported in the literature, it seemed useful to design a series of small prototypes investigating these issues. The approach was to select a real domain so that the problems arising would be legitimate. Because simplicity and uniformity are highly desirable as a basis for explanation and modifiability the pure production system approach was chosen as a starting point. Each of the prototypes tackled a more complex problem within the domain.

1. THE DOMAIN

The domain chosen was the interpretation of psychological testing using pencil and paper personality inventories. The domain is a suitable one for an expert system for two reasons. First, there is a basic agreement on the procedure for meaningful interpretation. Results are standardized over thousands of cases. Secondly, insight and experience create a marked difference between the interpretations of a novice and skilled practitioner. An expert system capturing the knowledge gained by the skilled interpreter would far surpass existing mechanical methods.

The application chosen was that with which the expert works daily, the use of these tests in the context of police work. He is a psychologist with the Psychological Services Branch of the Calgary Police Force. He and another psychologist provide a wide-range of services including individual and private therapy, formal psychological assessment for treatment and assessments for recruit selection. In all cases, the samples are actual cases from the files.

The most frequent use is in the process of selecting recruits. Recruits are first given competency and physical tests and then sent to the Psychological Services branch for assessment. Two tests are given, the Jackson Personality Inventory (Jackson, 1976) and the Minnesota Multiphasic Personality Inventory (Hathaway and McKinley, 1966). Each test consists of a number of true/false statements. The raw responses are grouped on a variety of dimensions. In very general terms, the process of interpreting theses scores is to first determine if the test is valid by examining the validity scales built into each and to see if any obvious inconsistencies exist. The scores on the various dimensions are compared with those obtained by control groups of individuals assessed as normal. The most accurate interpretation is to use norms appropriate to the context and population. Norms vary according to age, sex, and ethnic group, but also by occupation. In assessment for employment police norms would be used, for general therapeutic assessment the appropriate general population norms would be selected.

The JPI is a relatively uncomplicated test designed to assess normal behavior. Its scales measure anxiety, breadth-of-interest, complexity (the ability to see more than simple solutions), energy level, innovation, interpersonal affect (the degree to which emotional feelings can be expressed), organization, responsibility, risk taking, self esteem, social adroitness (skillfulness in social situations), social participation, tolerance, value orthodoxy (conservative social values), and infrequency (a validity scale which measures lying).

The MMPI is far more complex. It has three validity measures which influence the interpretation of other scores and a much wider range of application since its norms have been developed to assess a variety of psychopathological conditions. Fig. 4.1 gives its scales and a brief indication of their meaning.

L: lie	-general, deliberate but unsophisticated evasion
F: frequency	-maladjusted thoughts and beliefs
K: defensiveness	- like L but a more sophisticated measure of refusal to admit any inadequacy
Hs: hypochondriasis	-conversion of psychological problems to physical ones
D: depression	

--

Hy: hysteria -friendly, enthusiastic, suggestible
Pd: psychopathic deviate -impulsivity, low frustration level, poor
social adjustment
Mf: masculinity/femininity
Pa: paranoia
Pt: psychasthenia -anxiety, self-doubt, guilt, worry
Sc: schizophrenia -social alienation, peculiarities of perception
Ma; hypomania - expansiveness, activity level, excitement
Si: social introversion -comfort in interpersonal relations
The police psychologists use two other scales
Con: consistency -a validity check
Tma: Taylor Manifest Anxiety

Fig. 4.1 MMPI Scales

2. PROTOTYPE I

The first experiment was the interpretation of the JPI, a simplified application which is not used on its own by the police force, but is used alone in other settings. A pure production system was used. Both prototypes were written in Prolog. The rules were of the form

```
if <list of scale elevations> <context>
then generate statement and assert level of
    recommendation.
e.g.
If anxiety is normal and risk taking high
then print "likely to take dangerous chances
in police situations"; recommend 2
```

An interpreter took the calculated scales [2] as input. It first selected the appropriate norms by using rules which selected the correct table based on sex and context (police employment) and simple arithmetic calculations were used to generate a list giving the range of each scale, normal, high, extremely high, low, extremely low. This list became the working memory. The rules for determining consistency and validity were then applied. If the test was valid the rules for interpretation were tested. In order to make the report produced more logical all rules concerning any one scale were fired first. A record of the recommendations is kept and after the rules are examined the text for the recommendation is printed out.

The code for the interpreter, a sample report and the rules used to generate it, are given in Appendix I.

Five cases were run through the system successfully. An additional five were created and tested on paper. Cases used from 1 to 6 rules to generate the report. The five cases implemented required 17 rules in total. Within the cases tested, there was no overlap of rules. However, since there is a limited number of possible JPI scale combinations, and, since a context such as police employment is not attractive to all types of people, a high degree of overlap would be likely when more cases were entered. This assumption was supported by the psychologist's general experience and the results of Prototype II.

The expert's assessment was that this simple system worked amazingly well. He felt that he could express any interpretation needed. In the light of his assessment and because certain of the issues raised by the literature (discussed in the following section) had been encountered, no further work was done on Prototype I.

[2]. A useful system would of course begin with the raw scores but the cases from the files had already been scaled and since this is a very mechanical procedure it was deemed irrelevant.

2.1. An Evaluation of the Knowledge Representation

First, even within this simple context it was impossible to follow a pure production architecture which would decree that any rule could fire at any time. Instead a partitioned rule set was used. One set of rules determined the appropriate norms, another validity, another the interpretation, and a fourth the recommendations. These must be applied in order but within each set any rule is eligible. This was a minor modification, but it is suggestive of the limitations of pure production systems.

Since rules related patterns of scales directly to textual interpretation, displaying the rules provided a certain level of explanation. The user could request the rules generating a piece of text, for example. The explanation, however, relied on the user being familiar with the dimensions measured by each scale (not a substantial problem since the names of the scales are quite self-explanatory). No explanation of the connection between scale and interpretation is possible if the text does not seem obvious. An interface can be written which explains the strategy and the structure of the knowledge base. This reinforces the conclusions derived from NEO-MYCIN.

Since one of the crucial problems for expert systems was identified as credibility, one problem that arose in the validity stage was quite serious. The JPI has a built in validity scale which is easily checked but the psychologist also detects inconsistencies by looking at the overall profile. Someone is unlikely to be anxious, conservative, and a high risk-taker. This was handled by rules which stated the inconsistencies individually. It is important that inconsistent or invalid scores are not used to generate reports normally. While the expert thought that he could list all the inconsistent possibilities, the fact that the system had no understanding of the text it was generating meant that it could produce nonsense. An interpretation based on perceived anxiety and conservativism, for example, could caution that the person would be hesitant and avoid acting when needed, the high risk taking would result in a caution that the person could act too quickly and foolishly. This illustrates clearly one of the problems of existing rule-based knowledge representations in expert systems. The systems cannot understand and reason about the justifications on which the loosely associational rules are built.

Developing this first prototype was an object lesson in the difficulty in meeting the basic criteria of explicitness. Despite an awareness of the importance of explicitness, several important aspects of knowledge e.g. the determination of the norms, and the entire control strategy ended up in the procedural code. Given this problem, it is even more crucial that criteria for the evaluation of knowledge representations be clarified and the specific questions to be asked of the representation be spelled out for checks during implementation.

3. PROTOTYPE II

The next step was to add the MMPI to the simple scheme outlined above. This then duplicated the exact situation used in the office. Again only recruit selection was implemented. At first the basic structure was used but modifications needed to be introduced. Unlike prototype I, the scales were no longer indicative of the same interpretations in all situations. Instead, because the tests now allowed for more subtlety of interpretation, the significance altered in different combinations. For example, a high breadth-of-interest in a basically normal profile simply indicates some one who is curious and perhaps involved in many pursuits. In the presence of profiles showing certain problems it likely means some one who is too scattered to carry out activities. The process now is to identify a dominant note and to interpret other scales in this context. Rules now assert main and secondary features. Additional rules generate text from the descriptions found in the working memory.

The complete code, rulebase and sample reports are given in Appendix II. Twenty cases were implemented. They required 34 rules covering 8 main features and 15 secondary features. The remaining rules covered situations true in all situations (i.e. not dependent on a particular main or secondary feature. Within this number some overlap of main features was beginning to appear. Nineteen of the cases were judged successful based on a comparison of the reports produced by the program and those in the files. In the unsuccessful case, the test results appeared contradictory. The system produced no output because it could find no match in its knowledge base. The expert's reasoning process involved examining the individual test items and reasoning from basic principles of personality theory. The need to express domain principles as opposed to associations was again evident.

3.1. An Evaluation of the Knowledge Representation

Several problems emerged when a more complex domain was represented in this simple architecture. Those most relevant to the preceding chapters are discussed below.

First, the representation lacks explicitness as revealed by the difficulty in providing explanation facilities. The explanation facility now suffers in that the rules are not self-explanatory. To make the explanation clear the significance of the scores would have to be expressed. This is complex because it too is situation dependent. The therapist uses highly condensed associational rules in almost all cases. When a profile is contradictory he utilizes a personality model to reason about the connections. The likely explanation is that all interpretations rest on the underlying model. Frequently seen profiles are compiled on the basis of experience. The explanation facility would have to expand these underlying principles.

Second, the expert did not find the rules a natural way to express his knowledge. The rules become incredibly cumbersome when expressed in this simple syntax. Consider the problem of asserting a minor variation from normal. Everyone of the scales must be listed to prevent misapplication. The most natural way, for example, to express a minor variation in an otherwise normal profile would be

"If breadth-of-interest is high and all other scores are
normal...",

however, the only form possible is

"If breadth-of-interest is high, anxiety is normal, complexity is normal, energy level is normal, innovation is normal, interpersonal affect is normal, organization is normal, responsibility is normal, risk taking is normal, self-esteem is normal, social adroitness is normal, social participation is normal, tolerance is normal, value orthodoxy is normal, infrequency is normal, L is normal, F is normal, K is normal, Hs is normal, D is normal, Hy is normal, Pd is normal, Mf is normal, Pa is normal, Pt is normal, Sc is normal, Ma is normal, Si is normal, Con is normal and Tma is normal...".

Another area that the expert found annoying was that multiple rules existed to establish a main feature. He felt that since the knowledge was dispersed it was hard to evaluate for completeness. This was not considered too serious since it could be handled by the explanation interface.

The most significant problem was the lack of expressive adequacy. Two significant activities could not be represented at all. First, in certain situations the psychologist wished to consult individual test items in order to refine an interpretation or resolve an inconsistency. Second, the classification of very high, high, normal, low and very low is inadequate. Within the cases tested it was necessary to distinguish gradations within high scores. A high M/f is dominant only if it is the highest elevation. It is likely that other cases of this kind would arise.

4. PROTOTYPE III

Given the inadequacies of the implementation of Prototype II, it was clear that a more complex representation was required. No implementation was done but some speculation about an improved representation can be made. Two approaches were considered.

The first was to remain with the production system formalism and adopt the MYCIN style of rules. Adding functions increases the expressive power substantially, easily removing the cumbersomeness of the simple pattern recognition. Allowing comparison of the scores within a category, e.g. M/f is the highest, involves altering the data structure in which the evaluated scores are kept to include the number of standard deviations from the norm. An interface which indexed the rules to some standard e.g.main feature would remove some of the expert's concern

--

with the lack of organization in the rule base.

Two problems could not be handled so easily. First, providing the ability to examine the responses to individual questions requires completely redesigning the data and control structures. The new control structure would not have the same simple progression from scores to dominant feature to secondary features because it would return to the test in the final stages of refinement. In the existing system the flow of control is obvious from the rules themselves. More significantly, the problems of explanation are untouched. This seemed especially crucial because assessment for employment is very seldom done by psychologists. With existing mechanical methods people must accept computerized assessment as a black box. Since the expert's rules always showed the same highly abbreviated quality, the solution seemed to lie in creating a static structure which might make the underlying relationships clearer.

Frames seemed to offer the best approach. An examination of the tests (e.g. American Psychological Association, 1980) showed that the typical presentation was by diagnostic categories, essentially prototypes of personality patterns. Since the interpretation already first identified main features this seemed an excellent approach. The psychologist seemed far more comfortable with a frame structure possibly because of its familiarity to psychological literature.

The design creates a prototype for each main feature. In the context of assessment for employment these are informal personality profiles derived from experience with the immediate context. Police work, for example, attracts, but is very cautious about accepting, the John Wayne type. The expert thought that there might be about two dozen profiles varying in their degree of specificity to police work. General assessments or assessment in areas where informal profiles have not been identified could use standard diagnostic categories.

Until a substantial effort at implementation is made no firm conclusions could be made about the number of prototypes, the depth of the hierarchy, or the slots. A rough outline can be given.

One of the main features identified was depression, a sufficiently common condition that it would appear in any system. A main prototype would have an initial slot gathering all the ways that its existence would initially be established. Usually one of a number of patterns is sufficient. In the case of a disfunctional disorder such as depression sub-frames would represent types or degrees. Rules would still play a significant role in driving the strategy of interpretation, the frame would simply provide a clearer structure. The report of Appendix III Sample 1 would be generated by the depression frame by the rules attached to the slot labelled Coping Strategies.

The frames representation should provide a structure flexible enough to expand with the additional information which needs to be considered. Sub-types are frequently established by examining the original responses and by interviews. For example, a frequent concern in cases of depression and others is the determination that a condition is reactive or chronic. A reactive depression is a response to a serious life event. In treatment a quite different approach may be used. In employment situations, even if the person is not immediately suitable it might be good to encourage them to reapply in six months when a reactive depression is normally decreasing in intensity.

--

CHAPTER FIVE
PRINCIPLES FOR DESIGNING KNOWLEDGE REPRESENTATIONS

Assessing a knowledge representation or formalism requires a clear sense of the criteria and a detailed examination of the actual representation implemented. Examining the existing systems and experimenting with knowledge representation has raised a number of issues which suggest that the definitions and theory presented in Chapters one and two are overly simplistic. Also, the domains examined are drawn from the realm of diagnostic expert systems which have been most successful, in part because their knowledge lends itself to the existing formalisms. As expert systems attempt other types of domains, issues now being dealt with in other types of knowledge based programs will become relevant; they will, therefore, be used to inform this discussion.

1. EXPRESSIVE ADEQUACY

Expressive adequacy was initially defined as the ability of the chosen representation to reflect all the necessary distinctions within the given context and, of equal importance, not to force irrelevant distinctions. Although this is the most fundamental standard of expressive adequacy, as the discussions of chapter three indicated, systems do not meet it and, yet, even more subtle guidelines must be met if a representation is to be successful in complex domains.

1.1. Syntax and Semantics

One guideline to the adequacy of any representation is that syntax should reflect the semantics. If two pieces of knowledge are different in some significant way this should not be disguised by syntactic similarity. Otherwise it is impossible to determine the significance of any particular statement in isolation. In expert systems this is crucial for two reasons. First, an expert developing and modifying the system needs this help in order to determine how the knowledge he wishes to add or modify functions in the line of reasoning in order to avoid side effects which may cause an unexpected deterioration in performance. If the function is not self-evident and the expert is forced to trace its role through several executions he can easily miss something and will almost certainly be unwilling to commit the time needed. Second, the inference engine relying on the recognition of the formal patterns of syntax could misapply the information. Explanation facilities, also relying on syntax, must be able to determine function in order to provide an explanation.

Production systems with the uniform syntax of rules face serious problems. Both Clancy and Aikens commented at length on the confusion arising from the multiple meanings of the implication arrow in MYCIN. This not the problem of a bad representation, but is inherent in the formalism itself. Consider the following which are only a few of the possible meanings a human can ascribe to implication.

physical causation:
    If it is raining
    then the ground is wet.
associational:
    If the ground is wet
    then it has probably been raining.
situation/action:
    If it is raining
    then open an umbrella.

Unless the person reading these clauses already understands the connection between the premises and conclusion there is no way to differentiate these relationships. Each relationship would play different role in a line of reasoning, would inspire different levels of confidence, and would provoke different actions. Instead, if the principle of syntax supporting semantics is to be maintained, rules must be used for exactly one type of relationship. This is possible only in very simple domains such as the interpretation of the JPI. If more than one type of relationship exists all others must find new representations. This has been done in a minor way in MYCIN, for example, through tables and the attribute lists of clinical parameters and contexts. Radically increasing the knowledge in these or other representations obviously moves the system a long way from production rule theory with the resulting loss of simplicity and uniformity. Perhaps more seriously, the consequent explosion of informal data structures makes modification and explanation exceedingly difficult.

Here, frames appear to have a distinct advantage. Because the formalism itself provides no restrictions as to meaning the representation is looked to for guidance and the links between frames can be labeled. The name of the slot then indicates the nature of the relationship. Examples of this are prevalent in any frame system. CENTAUR (Aikens, 1984) links laboratory tests with the disease prototype they are instrumental in establishing through the Components slots and AM (Lenat, 1978) establishes an inheritance hierarchy through the generalization/specialization slots.

In practice, however, frame systems are as guilty of non-discrimination as rules. Representations name slots identically but use them to represent different relationships. Frame representations in expert systems have not received the detailed analysis given to rules. They are, however, widely used in natural language systems and have been scrutinized carefully. Much of this discussion is relevant to expert systems. The link which has received the most attention in natural language representations is the IS-A link. Since many of the theoretical arguments for a frames representation rest on their ability to represent inheritance hierarchies based on the IS-A link, these discussions are very relevant.

Consider the following examples.

Clyde is an elephant.
Clyde is a father.
Clyde is a circus performer.
Clyde is male.

Presumably the first and last are permanent and unalterable, the second is a characteristic which can only appear over time but then in some sense is permanent, the third could change at the whim of a circus owner. Like the varying kinds of implications these IS-A connections would play a different role in the line of reasoning and would inspire different levels of confidence. Further, since the IS-A connections represent different types of relations and are the links in the inheritance hierarchy, the inheritance mechanism must have implicit knowledge to differentiate chains such as: Clyde IS-A elephant; elephant IS-A mammal; Clyde IS-A mammal and Clyde IS-A elephant; elephant IS-A species; Clyde IS-A species. Brachman (1983), in fact, distinguished over 30 possible interpretations of an IS-A link.

Nor are all the problems of representing hierarchies solved even by the precise definition of links. The concepts of a domain may not fit a simple hierarchy. The PATREC system (Sanjay, Chandrasekaran and Stickles, 1984), for example, had difficulties with a hierarchy of drugs; the drug frame had two sub-types anaesthetic or analgesic. Halodane is both. Some frame based representation languages have introduced the idea of perspective (e.g. Bobrow and Winograd, 1977). By defining a point of view, different virtual hierarchies can be created from the same basic frame structure. Nothing approaching this complexity has been tried in an expert system.

Two quite different approaches could be taken. First, in a given representation it may be possible to simply have two frames for halodane or two links without creating problems to the inferencing. As the system grows, however, the ramifications of this type of patchwork solution could be quite unpredictable. Second, a complex representational language could be used. Given the need for modifications and explanation a very sophisticated interface would be required. The problems for explanation and modification are substantial.

1.2. Types and Degrees of Certainty

Humans rate the certainty by which they know something on at least two different scales. One is the actual probability of the fact being true. For example, "if the lawn is wet it has been raining" could have a lower degree of probability since it is just as likely to have been watered. Another is the source of information. The certainty that it has been raining is 100% to someone who got drenched, less to someone who heard it third hand. The latter issue has been dealt with in a minimal way by recording rule authors. A meta-rule can prefer rules written by those with greater competence. Given the characteristics of the domains of expert systems, the ability to adequately express levels of certainty will be crucial to continuing success.

One type of certainty depends on differentiating intension and extension. Intensional statements are true by definition. Reasoning about intension follows formal logic. Extensional statements are matters of fact, alterable over time, whose truth or falsity is established by observation. In knowledge bases it is not always clear which is which. A statement like all students who first enrolled in 1984 have a student number beginning 84 could be coincidence (extensional) or a fixed principle of the numbering scheme (intensional).

Clearly reasoning about statements requires knowing which is which. Weights offer one method. In MYCIN weights of 1.0 are theoretically reserved for intensional statements. Others are extensional and the certainty factors indicate, among other things, how frequently two observed events occur together. In practice, conclusions derived by the system during a consultation can have weights of 1.0 so the distinction is blurred.

The issue with frames is more complex because it touches on several other issues. Consider the problem of inheritance. If properties A, B, and C are, by definition, true of concept X and an IS-A link asserts that concept Y IS-A X, then Y must possess A, B, and C. These properties cannot be cancelled or modified. If property D of X is extensional, concept Y may be assumed to possess it but only actual verification will make it a certainty. In expert systems there can be three serious consequences of inheritance through defaults based on extensional properties. First, in many systems, CENTAUR for example, instantiating a frame asserts its existence. If a property of frame A is inherited by default through one or more steps to frame C it fills in a slot. Several other slots may have been filled in by defaults. When frame C is instantiated it fills in a slot of the frame above it and contributes to the assertion of its existence. There exists the possibility of making assertions on very little actual evidence. Second, when these values are inherited, changes in weights may result. If later evidence proves the defaults were incorrect the effects on the propagation of certainty cannot be easily reversed. Third, if extensional properties can be cancelled and only extensional links exist between two frames in the hierarchy, then certain assertions cannot be justified. Just because Clyde has all the properties of an elephant does not mean that he is an elephant. He could be a giraffe with all the giraffe properties cancelled and elephant properties asserted. Class membership cannot be asserted on cancellable properties (Brachman, 1984).

Given that expert systems have been relatively constrained domains these may not be problems. Certainly no expert system research has addressed them directly.

The only method now used for expressing the degree of certainty is weights. Many studies exist discussing the mathematical properties of determining uncertainty. The concern here is with what certainty factors seem to mean. They are a way of enabling a computer to understand phrases like "probably", "some of the time", "may be" and combining them. Like rules, however, they may express a variety of relationships. The simplest may be the statistical probability that two events, e.g. a symptom and a disease will occur together. They may express the reliability of a means of measurement. Consider the following possible interpretations of A->B with a certainty factor of .7 (assume 1.0 to be absolute certainty). A occurs in 70% of the cases of B. A always occurs with B, but it also occurs in other cases. A occurs only and always with B, but the means of measuring it returns a significant number of false positives. The next step in the line of reasoning should be different in each case, but only an examination of the entire rule base is likely to make clear which is the correct approach. Even more confusing to someone trying to modify the rules, each certainty factor probably measures some combination of factors. In certain cases these can be split off. PUFF had two rules connecting a lab finding with the presence of a disease with different certainty factors depending on the means of obtaining the lab data. CENTAUR's frames allow two measures, one relating the finding to the disease, one expressing the reliability of the testing device. This is certainly clearer, but any proliferation becomes complex to handle.

Systems relying on certainty factors have claimed that the weights given to inferences are not affected by extraneous issues such as the order of execution. At least one example has been published to establish that for a given rule type in a given set this is not so (Cendrowska and Bramer, 1984). Others should be investigated in actual practice since these seem not to be based on mathematical but intuitive approaches. In MYCIN when one of the possibilities reaches 1.0 the rest are deleted, but none of the literature has established that their values cannot exceed one. Perhaps another possibility might not only reach 1.0 but also exceed the value of the first one on succeeding cycles.

1.3. Types of Expert Reasoning

The nature of expert systems requires that they deal with incomplete knowledge in two ways. First, since their domains are complex the knowledge in the system may not represent all of what is known and further the existing knowledge may be incomplete. Defaults, for all their problems, are one approach, but this is an area of very few solutions. At this stage probably the best that can be done is indicate a few of the problems.

First, certain values in certain domains may need to be marked clearly as unknown rather than inheriting defaults or being treated as false. It is not clear how this might affect other parts of the system. MYCIN assumes that a value above the arbitrarily chosen value +.2 is true and below -.2 is false, however, in reasoning, anything below +.2 is treated as false not unknown.

Second, humans have a capability to assert that if it were true I would know it. Assumptions of this kind are important in expert systems. A therapist reading an intake report assumes that if the patient had appeared in an elephant suit the report would have said so. More realistically, because appearance can be a contributing factor to diagnosis such as the degree of depression, an unusually dirty and unkempt appearance would be reported. Another form of this capacity is to assert that because I know about this area and X is an important fact, if it were true I would know it.

Third, systems cannot evaluate the gaps in their knowledge. Systems may infer far too much. For example, if three types of rock are listed as subtypes it may be assumed incorrectly that only three types exist. Or if a frame exists it may be assumed that the concept it represents exists. Unlike human experts, who can first determine if a problem is within their expertise and sometimes recognize during the solution when they have exceeded their competence, a computer expert will return whatever it generates.

2. EXPLICITNESS

The degree of explicitness was initially defined as the degree to which knowledge could be expressed in a representation directly understandable by some one familiar with the domain but not with programming. As a first approximation it was taken to mean that knowledge was represented in the language of the formalism and not in code. Closer examination shows that this is simplistic. First, the connection between the implementation code and the representation are more complex and must be clarified.

One issue that it raises is the relationship of the abstract knowledge representation seen by the expert and user to the actual programming implementation. Many techniques exist for more efficient code, for example, search and sort algorithms. The programmer will want the most efficient code possible which may mean that a program does not run exactly as described. For example, instead of testing each rule and clause each cycle OPS4 (Forgy, 1977) precompiles the rules into a tree and changing certainties are automatically propagated up the tree. Because the line of reasoning develops as if every rule were tested the actual details of the implementation are of no concern in assessing the formalism. Contrast this with an example from MYCIN. Assume that the data base contains the following rules:

1. A->B
2. B->C
3. C->A

The interpreter detects the circularity and aborts the trace of rule 1 thus not asserting B. This is completely unpredicted by the production system formalism or by examining rule 1, a condition which directly contravenes the modularity of rules, an essential characteristic of production systems.

Our concern, then, is when the implementation adds to or alters the details of knowledge or reasoning as expressed at the level of the formal structure.

Second, for judging whether or not a single item of knowledge is explicit, distinguishing between the representation and the code is sufficient. However, not only individual items of knowledge but also the relationships between objects and the strategy for problem solving must be explicit. Clancy's work on the epistemology of rules revealed that the representation itself failed to make the relationships clear between the premises of clauses and relied on the ordering of clauses to control the search strategy implicitly. When the representation itself is examined, all of the issues presented under expressive adequacy are seen to be equally relevant to considerations of explicitness.

2.1. Making Relationships and Strategies Explicit

Production systems and frames present different problems in making relationships and strategies explicit. Section 5.1. already outlined some of the care that needs to be taken to limit the meaning of implication. If equal care is taken to limit the rule premises to a single relationship then an impossible situation arises. Consider the following MYCIN rule which begins "if no CBC is available and no WBC is available..". White blood count is a part of a complete blood count. The relationship is that of subset; the purpose is to prevent the more costly search for WBC being done if the CBC was not done. There is no way in the context of the rule to include the subset relationship or to indicate its function. Separating the two clauses into a new rule "If no CBC then no WBC" doesn't make the relationship clearer and adds another meaning to the implication sign. Frames apparently make the relationships clearer since they explicitly represent objects, attributes and relationships, but, in fact, for the reasons given above, they usually fail in practice. A common situation occurs in AM. The system is heavily dependent on the inheritance of values. Some values are inherited downwards, some upwards and some not at all. The decisions are implicit in the inheritance code.

Both frames and production systems have problems in making problem solving strategies explicit. Here it is production systems which have the advantage. In a production system rule order affects the line of reasoning and the the strategy remains embedded in the inference engine. Normally, however, this strategy is uniform and simple. It is therefore relatively easy and straightforward to understand. Frames, on the other hand, need have no uniform reasoning but can proceed by procedural attachments.

2.2. Degree of Explicitness Required

An explicit representation is mainly crucial for explanation and modifiability and it was for these reasons that explicitness was made one of the three criteria. Explicitness is not required for good performance in many domains. Clancy, in fact, claimed that the demands of explanation and problem solving were not compatible in one representation. He felt that the correct approach to explanation was to relate each rule to a step in the progress of the disease. This could not be the approach used to drive the reasoning because processes are not really understood and are used as justification after the fact. My work on personality inventories confirmed Clancy's conclusions in another domain. The expert worked in a highly abbreviated short-hand, interpretations were based on associations of scores in profiles. There was no conscious application of any personality model until a difficult profile, usually one appearing inconsistent, was encountered. Then he returned to basics about personality. His experience allowed a very efficient problem solving strategy based on compiled knowledge. He understood his short-hand clearly and was able to modify and expand the system easily. However, these connections were not at all apparent to others.

2.3. Using Different Representations

Clancy's (1981) solution to explaining MYCIN was to create a complex interface. Any expert system is going to require some form of interface to translate between the internal representation and a form that the user can understand. The great strength of rules is that the translation is a very simple matter of expanding internal representations, adding a few words and rearranging the order to seem natural. The more complex the interface becomes the greater the potential for trouble. Relying on the interface requires psychological confidence. When debugging, for example, is it the translation program or the interpreter that is at fault. This form of complex system can also not function as an authoring tool. While the NEO-MYCIN interface can remove irrelevant clauses in order to generate the most pertinent explanation, it could not take a new rule and add the necessary clauses.

3. NATURALNESS

Although all three criteria have a degree of subjectivity, naturalness is by far the most subjective. It can only be defined in terms such as ease of use or comfort. It is most apparent when it is lacking because the representation seems constantly in the way. Assessing the degree to which the representation is or is not intrusive can really only be done by anecdotal report. Davis, for example, reported that experts find expressing their knowledge in rules awkward; "they tend to think of a sequence of operations in procedural terms and find flowcharts a convenient medium of expression." Iterative operations seem to give particular difficulty since they are not used to breaking things into such fine steps.

Further, it is important to realize that true naturalness is unobtainable. The most natural form of expression for any expert would obviously be that which he already uses in discussing his domain, natural language plus in most cases diagrams and mathematical or other symbolic expression. Given the state of natural language processing this is clearly impossible. The computer requires some highly constrained and uniform language.

Anyone reading examples of rules from the research is struck by their seeming ease and naturalness. This is deceptive. Underlying both rules and frames is the very substantial task of defining the vocabulary and functions of the domain. In MYCIN a rule is always of the form

<pred func> <object> <attribute> <value>

with (at one stage) 24 predicate functions such as "same", "known", "definite"; 80 attributes such as "site", "identity", "sensitivity"; and 11 objects such as "organism", "culture", "drug". Each rule must be written in this limited vocabulary.

Even the degree of naturalness it is possible to attain may well be at odds with the other requirements of the system. The discussion of expressive adequacy above made the argument for much greater rigor. Other claims, related to this, have demanded explicitness. One of the characteristics of expert's thinking is often that they have unconsciously condensed much of their knowledge. Steps may be omitted or terminology become a kind of shorthand. It is probably quite natural to use -> in a multitude of ways, and it is unnatural for the expert to spell out the steps. Clancy (1981) said of his work "The framework of knowledge types and purposes that we have described would constitute a 'typed' rule language that could make it easier for an expert to organize his thoughts. On the other hand, we must realize that this meta-level analysis may impose an extra burden by turning the expert into a taxonomist of his own knowledge - a task that may require considerable assistance."

From this discussion and the research examined, it is clear that naturalness while clearly highly desirable to enhance acceptance and ease of development and modification, is far too elusive a quality to define. Many of its aspects, in fact, are likely to emerge as questions of personal taste. Only two firm conclusions can be made. First, if there is a physical structure which is significant to the domain, then the representation should try to model it. In an interesting comparative study (Hayes-Roth, Waterman and Lenat, 1983), several representation languages were used on the same problem, searching a tree structured drainage ditch system. The results were more varied than might be expected, ranging from easy and straightforward to completely impossible. Second, if there is a way in which concepts are conventionally ordered by practitioners in the domain these should be maintained. Bennet and Hollander (1981) detailed the problems arising from asking the experts to substantially alter their modes of thinking. In my work, the psychologist preferred frames largely because they matched the diagnostic categories found in his domain.

--

CHAPTER SIX
CONCLUSION

Expert systems, a field of study still in its infancy, has produced some impressive performances. Having accomplished the first stage by showing that it is possible to represent expertise in certain domains the next step is to develop general guidelines for evaluating them. Chapter one discussed the special characteristics of expert systems and the demands that these place on a knowledge representation. Based on these it was established that performance is not a sufficient measure. In order for an expert system to be accepted, there are constraints on how the system may achieve that performance. To achieve credibility the system has to be self-explanatory; it has to capture the types of reasoning normally used in the domain; and it has to be easily modified to accept new developments in the field. In order to meet these goals, the knowledge representation has to meet three criteria: expressive adequacy, naturalness, and explicitness. Initial definitions of these terms were given. Chapter two summarized the theoretical literature supporting the appropriateness of the two most commonly used formalisms: rule-based production systems and frame-based conceptual graphs. Examining the knowledge representations in successful expert systems in Chapter three established that the theory was simplistic and that existing expert systems were inadequate in expressive adequacy, explicitness, and/or naturalness. A better understanding of these criteria was sought by examining alternative representations within the domain of psychological test interpretation. The initial definitions for expressive adequacy, explicitness and naturalness were found to be inadequate for assessing either a knowledge representation or formalism. Based on the study of the existing systems and on knowledge representation research from other areas of AI, new principles for evaluating these qualities were proposed.

Expressive adequacy was defined as the ability to express all the distinctions necessary to the domain in the formalism and not force irrelevant distinctions. The basic principle of evaluation is that the knowledge be expressed in the representation language without having to resort to procedural code. This is not a sufficient measure. In addition, the syntax and semantics must be congruent; the varying degrees of certainty, including the grounds for certainty, must be expressible; and all types of reasoning used by experts in the domain must be represented.

Explicitness has usually been treated in the literature as an automatic consequence of expressive adequacy. If knowledge is expressed in the representation language, it is explicit. Such is not the case. First, explicitness must be examined in the context of function; the degree of explicitness required for reasoning differs from that required for explanation. Second, the underlying relationships of the domain, including those assumptions of which the expert is no longer conscious, must be stated. Third, since the representation can be made more explicit by a clever interface, the distance between the explicit representation and the representation used to drive the reasoning must be examined.

Naturalness is such a highly subjective quality that the only possible assessment is the anecdotal report of experts and users. Two principles will aid in creating naturalness; first, minimizing the constraints created by the demands of the implementation and of explanation and authoring facilities; and, second, selecting representations compatible with those chosen in the literature of the domain.

Expert systems research is in a stage of transition. The feasibility of expert systems has been clearly established. Among the challenges now are the tasks of bringing more systems into full use in real world applications, expanding successful applications to more complex domains, and discovering general tools and principles to facilitate system development. Hopefully, clarifying the criteria for assessing knowledge representation will contribute to this latter task.

--

BIBLIOGRAPHY

Aikins, J.S. (1983) "Prototypical Knowledge for Expert Systems" *Artificial Intelligence, 20,* 163-210.

American Psychological Association (1980) *Diagnostic and Statistical Manual of Mental Disorders.* APA, Washington, D.C..

Bachant, J. ar McDermott, J. (1984) "R1 Revisited: Four Years in the Trenches" *Artificial Intelligenc Magazine,* Fall.

Bennett, J.S. and Hollander, C.R. (1981) "DART: An Expert System for Computer Fault Diagnosis" *Proceedings of the International Joint Conference on Artificial Intelligence,* 843-845.

Brachman, R.J. (1983) "What IS-A is and isn't: An Analysis of Taxonomic Links in a Semantic Network" *IEEE Computer,* 30-36, October.

Cendrowska, J. and Bramer, M.A. (1984) "A Rational Reconstructon of the MYCIN Consultation System" *International Journal of Man-Machine Studies, 20.*

Chandrasekaran, B. and Mittal, S. (1983) "Deep versus Compiled Approaches to Diagnostic Problem Solving" *International Journal of Man-Machine Studies, 19,* 425-436.

Clancy, W. (1981) *Epistemology of a Rule-Based Expert System: a Framework for Explanation.* Report No. STAN-CS-81-896 University of Stanford, Nov.

Clancy, W. and Letsinger, R. (1981) "NEO-MYCIN: Reconfiguring a Rule-Based Expert System for Application to Teaching" *Proceedings of the International Joint Conference on Artificial Intelligence,* 829-836.

Davis, R. and Buchanan, B. (1977) "Production Rules for a Knowledge-Based Consultation Program" *Artificial Intelligence, 8,* 15-51.

Davis, R. and King, J. (1977) "An Overview of Production Systems" *Machine Intelligence, 8,* 300-331.

Davis, R. (1981) "Interactive Transfer of Expertise: Acquisition of New Inference Rules" in *Readings in Artificial Intelligence,* edited by Webber and Nilsson, pp 409-428.

Davis, R. and Lenat, D.B. (1982) *Knowledge Based Systems in Artificial Intelligence.* McGraw-Hill Inc., New York.

Duda, R. (1981) "Subjective Baysian Methods for Rule Based Inference Systems" in *Readings in Artificial Intelligence,* edited by Webber and Nilsson.

Findler, N.V. (1979) *Associative Networks: Representation and Use of Knowledge by Computers.* Academic Press, New York.

Forgy, R. (1979) *OPS4: User's Manual.* Department of Computer Science, Carneige Mellon University.

Genesereth, M.R. (1982) "Metaphors and Models" *Proceedings of the Conference of the American Association for Artificial Intelligence* , 208-211.

Hasling, D.W., Clancy, W.J. , and Rennels, G. (1983) "Strategic Explanations for a Diagnostic Consultation System" *International Journal of Man-Machine Studies, 19.*

Hathaway, S.R. and McKinley, J.C. (1966) *Minnisota Multiphasic Personality Inventory.* The Psychological Corporation, New York.

Hayes, P.J., Hayes-Roth, F., Waterman, D.A. , and Lenat, D.B. (1983) "Building Expert Systems" in *Readings in Artificial Intelligence,* edited by Webber and Nilsson, pp 451-458 . Addison-wesley Publishing Company Inc., London.

Jackson, D.N. (1976) *Jackson Personality Inventory.* Research Psychologists Press, New York.

Kunz, J.C., Kehler, T.P., and Wiliams, M.D. (1984) "Applications Development Using a Hybrid AI Development System" *Artificial Intelligence Magazine,* Fall.

Lachar, D. (1974) *The MMPI: Clinical Assessment and Automated Interpretation.* Western Psychological Services.

Lenat, D.B. and Harris, G. (1978) "Designing a Rule system that Searches for Scientific Discoveries" in *Pattern-Directed Inference Systems,* edited by Waterman and Hayes-Roth, pp 25-51.

Lenat, D. (1982) "Heuretics: Theoretical and Experimental Study of Heuristic Rules" *Proceedings of the American Association of Artificial Intelligence,* 159-163.

Lenat, D. and Brown, J.S. (1984) "Why AM and EURISKO Appear to Work" *Artificial Intelligence, 23,* 269-294.

McDermott, J. (1980) "R1: An Expert in the Computer Systems Domain" *Proceedings of the Conference of the American Association for Artificial Intelligence,* 269-271.

McDermott, J. and Steele, B. (1981) "Extending a Knowledge Based System to Deal with Ad Hoc Constraints" *Proceedings of the International Joint Conference on Artificial Intelligence,* 824-828.

Michalski, R. and Chilausky, R. (1980) "Knowledge Acquisition by Encoding Expert Rules versus Computer Induction from examples: a case study involving soybean pathology" *International Journal of Man-Machine Studies, 12,* 63-87.

Michie, D. and Elcock, E.W. (1977) *Machine Intelligence 8.* Wiley and Sons, New York.

Mittel, S., Chandrasekaran, B., and Stickles, J., and Newall, A. (1982) "The Knowledge Level" *Artificial Intelligence, 18,* 87-127, September.

Nilsson, N.J. (1980) *Principles of Artificial Intelligence.* Tioga Publishing Company, Palo Alto.

Pasik, A. and Schor, M. (1984) "Table Driven Rules in Expert Systems" *Sigart Newsletter* (87) 31-33, January.

Ritchie, G.D. and Hanna, F.K. (1984) "AM: A Case Study in AI Methodology" *Artificial Intelligence, 23,* 249-268.

Shortliffe, E.H. (1976) *Computer Based Medical Consultation Systems:MYCIN.* Elsevier, New York.

Shortliffe, E.H. (1981) "Consultation Systems for Physicians: the Role of AI Techniques" in *Readings in Artificial Intelligence,* edited by Webber Nilsson, pp 322-333.

Smith, D.E. and Clayton, J.E. (1980) "A Frame Based Production System Architecture" *Proceedings of the Conference of the American Association for Artificial Intelligence,* 154-156.

Sowa, J. (1984) *Conceptual Structures.* Addison Wesley Publishing Company, Inc., New York.

Stefik, M., Aikins, J., Balzer, R., Benoit, J., Birnbaum, L., Hayes-Roth, F., and Sacerdoti, E. (1982) "Organizaton of Expert Systems: a Tutorial" *Artificial Intelligence, 18,* 135-172.

Weiss, S.M., Kulikowski, C.A., Amarel, S., and Safer, A. (1978) "A Model Based Method for Computer-Aided Medical Decision Making" *Artificial Intelligence, 10,* 145-171.

APPENDIX I

Appendix I gives the code used in prototype I's interpreter. Because the interface was identical to the second prototype, it is given with that complete program. The section also contains one set of rules and the report that they generated.

THE FIRST OPERATORS

These occur immediately after the interface operators which gather information and set up the report file. The interface operators are given at the end of appendix II since they are common to both programs.

Scale provides the list of scales for the test and is used to contrl the clauses which take the scores and assign them to ranges.

```
scale([anxiety,breadth_of_interest,complexity,conformity,
energy_level,innovation,
interpersonal_affect,organization,responsibility,risk_taking,
self_esteem,
social_adroitness,social_participation,tolerance,value_orthodoxy,
infrequency]).
```

JPI is the main control operator. It first creates two lists: scales with high or low scores and scales with normal scores. It them invokes the interpreter i. The assertz clauses are ued to provide a record for an explanation facility which is not included because only preliminary work was done.

```
jpi(File,Name,Context,Sex,Scores):- assertz(score(Name,Scores)),
        assertz(context(Name,Sex,Context)),
        scale(Scales),
        build_list(Context,Sex,Scores,Scales,Deviant,Normal),
        assertz(deviant(Name,Deviant)),
        i1(File,Name,Deviant,Normal),
        recommend(File,Name).




build_list(Context,Sex,[],[],[],[]).

build_list(Context,Sex,[S|Ts],[X|Tx],[X,D|Td],Normal):-
         norm(Context,Sex,Value),
        v(Value,X,Low,High,Dev),
        dev(S,Low,High,Dev,D),!,
        build_list(Context,Sex,Ts,Tx,Td,Normal).

build_list(Context,Sex,[_|Ts],[X|Tx],Deviant,[X,normal|Tn]):-
        build_list(Context,Sex,Ts,Tx,Deviant,Tn).
```

```
norm(police_job,male,1).
norm(police_job,female,2).


dev(S,Low,High,Dev,low):-S<Low, S>Low-Dev.
dev(S,Low,High,Dev,extemelow):-S<Low-Dev.
dev(S,Low,High,Dev,high):-S>High,S<High+Dev.
dev(S,Low,High,Dev,extremehigh):-S>High+Dev.
```

i1 and i are the actual interpreters invoked by jpi. i1 determines that if all scores are normal no further work need be done. i is invoked when there are deviant scores. It processes the list of deviant scores recursively. If the first deviant scale is a member of the premise list of rule, it checks to see if the remaining conditions of the premise are satisfied. If they are it writes the appropriate text to the report file and calls itself recursively with the rest of the deviant list.

```
i1(File,Name,[],Normal):-prtstr("All results are within
        normal limits.",1),
          assertz(r(Name,0,0,0)).
i1(File,Name,Deviant,Normal):-i(File,Name,Deviant,Normal).


i(File,Name,[],Normal).
i(File,Name,[X,Y|T],Normal):-
        clause(if(Conditions,Recommend,Rulenum),B),
          member(X,Y,Conditions),
          satisfy(Conditions,[X,Y|T],Normal),
          num(Entry),
          assertz(r(Name,Recommend,Rulenum,Entry)),
          tell(File),
          write(Entry),
          call(B),
          tell(user),
          write(Entry),
          call(B),
          fail.
i(File,Name,[X,Y|T],Normal):-i(File,Name,T,Normal).



satisfy([],Deviant,Normal).
satisfy([X,Y|T],Deviant,Normal):-
          (member(X,Y,Deviant);member(X,Y,Normal)),
          satisfy(T,Deviant,Normal).



member(X,Y,[X,Y|T]).
member(X,Y,[_,_|T]):-member(X,Y,T).
```

--

RECOMMEND finds the highest level recommendation created by the interpreter and prints the appropriate message.

```
recommend(File,Name):-tell(File),nl,nl,nl,
          tell(user),nl,nl,nl,
          filewrite(File,"Recommendation: ",2),
          find_recommend(File,Name).

find_recommend(File,Name):-clause(r(Name,3,_,_),B),!,
          filewrite(File,"Recommend rejection.",1).
find_recommend(File,Name):-clause(r(Name,2,_,_),B),!,
          filewrite(File,"The profile raises serious
                           concerns which could indicate
rejection.  These concerns should be pursued carefully
at the interview.",1).
find_recommend(File,Name):-clause(r(Name,1,_,_),B),!,
          filewrite(Name,"Results are basically within
         normal ranges,but the profile
raises some concerns which should be assessed at the
        interview.",1).
find_recommend(File,Name):-
          filewrite(File,"Recommend hiring.",1).


num(Entry):-e(X),
          retract(e(X)),
          Entry is X+1,
          asserta(e(Entry)).
e(0).
```

Following are the rules used to generate one report. The report that they generate is typical of Protoytype I and is given on the next page.

```
if([risk_taking,high],2,1):-
prtstr("  - impulsive and willing to take chances.",1).

if([interpersonal_affect,low],2,2):-
prtstr("  - unlikely to show feelings.
People like this tend to be ungiving in relationships,
superficial,and defensive.",1).

if([anxiety,normal,risk_taking,high,self_esteem,normal],2,3):-
prtstr("  -likely have trouble being cautious in appropriate
situations.",1).

if([anxiety,normal,risk_taking,high,self_esteem,normal,
          interpersonal_affect,low],2,4):-
       prtstr(" - difficult to manage because he doesn't
 respond openly.",1).
```

Sample 1. A JPI based report

PSYCHOLOGICAL TESTING

Assessment is based on:
the Jackson Personality Inventory

Assessment is for police_job

Name:
Sex: male
Profile:
-impulsive and unwilling to take chances
-unlikely to show feelings. People like this
tend to be ungiving in relationships, superficial
and defensive
-likely to have trouble being cautious in apprpriate
situations
-difficult to manage because he doesn't respond openly

Recommendation:

The profile raises serious concerns which could indicate
rejection. These concerns should be pursued carefully
at the interview.

--

APPENDIX II

The structure of Prototype II is essentailly similar to I. The interface operators and those which are very similar to Prototype I are given at the end of the code section.

The interpretation was done by i which searched the rulebase once firing any rule which matched its deviant or normal list. It assumed that the rulebase was maintained in partitions with all main feature rules coming first. It is invoked by the interpret clause which is given after and is essentially similar to the jpi clause of prototype II. Thus it follows the clauses which gather information, format the report and translate the raw scores into lists of deviant and normal ranges.

```
i(File,Name,Deviant,Normal):-
        clause(if(Conditions,Rulenum),B),
        satisfy(Conditions,Deviant,Normal),
        num(Entry),
        write(Entry),
        call(B),
        fail.
i(File,Name,[X,Y|T],Normal).
```

```
satisfy([],Deviant,Normal).
satisfy([X,main|T],Deviant,Normal):-
        main(X),satisfy(T,Deviant,Normal).
satisfy([X,secondary|T],Deviant,Normal):-secondary(X),
         satisfy(T,Deviant,Normal).
satisfy([X,Y|T],Deviant,Normal):-
        (member(X,Y,Deviant);member(X,Y,Normal)),
        satisfy(T,Deviant,Normal).
```

```
member(X,Y,[X,Y|T]).
member(X,Y,[_,_|T]):-member(X,Y,T).
```

The following cluases are essentially the same as Prototype I.

```
scale_jpi([anxiety,breadth_of_interest,complexity,
conformity,energy_level,innovation,
interpersonal_affect,organization,responsibility,
risk_taking,self_esteem,
social_adroitness,social_participation,tolerance,
value_orthodoxy,infrequency]).

scale_mmpi([l,f,k,hs,d,hy,pd,mf,pa,pt,sc,ma,si,con,tma]).
```

```
interpret(File,Name,Context,Sex,Score_jpi,Score_mmpi):-
        combine(Score_jpi,Score_mmpi,Scores),
        scale_jpi(Sj),
        scale_mmpi(Sm),
        combine(Sj,Sm,Scales),
        build_list(Context,Sex,Scores,Scales,Deviant,Normal),
        nl,nl,nl,
        prtstr("Casenumber:  ",0),
        write(Name),nl,
        i1(File,Name,Deviant,Normal),
        apply_to_context(File,Name,Deviant,Normal,Context),
        recommend(File,Name),
        prtlist(Deviant),
        clear,
        explain(File,Name,Context,Sex,Scores,Deviant,Normal).



build_list(Context,Sex,[],[],[],[]).

build_list(Context,Sex,[S|Ts],[X|Tx],[X,D|Td],Normal):-
         norm(Context,Sex,Value),
        v(Value,X,Low,High,Dev),
        dev(S,Low,High,Dev,D),!,
        build_list(Context,Sex,Ts,Tx,Td,Normal).

build_list(Context,Sex,[_|Ts],[X|Tx],Deviant,[X,normal|Tn]):-
        build_list(Context,Sex,Ts,Tx,Deviant,Tn).

combine([],L,L).
combine([X|L1],L2,[X|L3]):-combine(L1,L2,L3).

norm(police_job,male,1).
norm(police_job,female,2).



dev(S,Low,High,Dev,low):-S<Low, S>=Low-Dev.
dev(S,Low,High,Dev,extemelow):-S<Low-Dev.
dev(S,Low,High,Dev,high):-S>High,S=<High+Dev.
dev(S,Low,High,Dev,extremehigh):-S>High+Dev.

i1(File,Name,[],Normal):-
        prtstr("All results are within normal limits.",1),
        assertz(r(Name,0,0,0)).
i1(File,Name,Deviant,Normal):-i(File,Name,Deviant,Normal).


apply_to_context(File,Name,Deviant,Normal,Context):-
        tell(File),nl,nl,nl,
        tell(user),nl,nl,nl,
        filewrite(File,"Recommendation: ",2),
        clause(if(Conditions,Context,Recommend,Rulenum),B),
        satisfy(Conditions,Deviant,Normal),
```

--

```
        assertz(r(Name,Recommend,Rulenum)),
        call(B),
        fail.
apply_to_context(_,_,_,_,_).

recommend(File,Name):-
        find_recommend(File,Name).

find_recommend(File,Name):-clause(r(Name,3,_),B),!,
        filewrite(File,"Recommend rejection.",1).
find_recommend(File,Name):-clause(r(Name,2,_),B),!,
        filewrite(File,"The profile raises serious concerns
         which could indicate
rejection.  These concerns should be pursued carefully
at the interview.",1).
find_recommend(File,Name):-clause(r(Name,1,_),B),!,
        filewrite(Name,"Results are basically within normal
 ranges,but the profile
raises some concerns which should be assessed at the interview.",1).
find_recommend(File,Name):-
        filewrite(File,"Recommend hiring.",1).


num(Entry):-e(X),
        retract(e(X)),
        Entry is X+1,
        asserta(e(Entry)).
e(0).

main_feature(X,Y):-assertz(main(X)),prtstr(Y,1).
secondary_feature(X,Y):-assertz(secondary(X)),prtstr(Y,1).


explain(_,_,_,_,_,_).
```

Protoytpe II was set up to handle multiple cases.

```
clear:-retract(e(X)),
        asserta(e(1)),
        retractall(main(Y)),
        retractall(secondary(Z)).
retractall(X):-retract(X),fail.
retractall(X):-retract((X:-Y)),fail.
retractall(_).

prtlist([]).
prtlist([X,Y|T]):-
        write(X), prtstr("    ",0),write(Y),nl,prtlist(T).
```

The following oprators handled all the interface for both the Prototypes. The are simply a sequence of read / writes which gather the information about a particular case and set up the format for the report to be generated.

```
pti:-header(F,C),case(F,C).

header(F,C):-prtstr("File for output:   ",0),
        read(F),
        nl,
        prtstr("Reason for assessment:  ",0),
        read(C),
        nl,
        tell(F),
        nl,nl,
        prtstr("            PSYCHOLOGICAL  TESTING",2),
        nl,
        prtstr("Assessment is based on:
     the Jackson Personality Inventory
     the Minnisota Multiphasic Personality Inventory.",1),
        prtstr("Assessment is for ",0),
        write(C),
        nl,nl,nl,
        tell(user).


case(F,C):-info(Name,F,Sex,Sj,Sm),
        filewrite(F,"Profile:",2),
        interpret(F,Name,C,Sex,Sj,Sm).


info(Name,F,Sex,Sj,Sm):-nl,
        prtstr("Last name: ",0),
        read(Name),
        nl,
        tell(F),
        prtstr("Name: ",0),write(Name), prtstr(", ",0),
        tell(user),
        prtstr("First name: ",0),
        read(Fname),
        nl,
        tell(F),
        write(Fname),
        nl,
        tell(user),
        prtstr("Sex:  ",0),
        read(Sex),
        nl,
        tell(F),
        prtstr("Sex: ",0),write(Sex),nl,
        tell(user),
        prtstr("Enter JPI scores.",1),
        read(Sj),
        prtstr("Enter MMPI scores.",1),
        read(Sm).



filewrite(F,Str,N):-tell(F),prtstr(Str,N),
        tell(user),prtstr(Str,N).

prtstr([],0).
prtstr([],1):-nl.
prtstr([],2):-nl,nl.
prtstr([H|T],N):-put(H).prtstr(T,N).
```

--

```
/* THE RULE BASE                    */



if([anxiety,high,self_esteem,low,energy_level,low,
            d,extremehigh],1):-
        main_feature(depression,
        "  -the main feature is depression.").


if([energy_level,low,responsibility,high,
            self_esteem,high],7):-
        main_feature(burned_out,
          "  -takes life too seriously. Worn out.").

if([mf,high],9):-
        main_feature(intellectual,
    "  -highly verbal,intellectualizer.").

if([pd,high,sc,high],17):-
        main_feature(par_schiz,
      "  - paranoid schizaphrenic.").

if([sc,low],19):-main_feature(practical,"  -practical.").

if([pa,extremelow],30):-
        main_feature(paranoid," -very suspicious of everyone.").

if([complexity,high,anxiety,normal,energy_level,normal,
  interpersonal_affect,
  normal,organizaton,normal,responsibility,normal,
  self_esteem,normal,
  social_adroitness,normal,social_participation,normal,
    tolerance,normal,
  k,normal,hs,normal,d,normal,hy,normal,pd,normal,mf,
     normal,pa,normal,
  pt,normal,sc,normal,ma,normal,si,normal,con,normal,
     tma,normal],33):-
        main_feature(normal,
        "  -esentially normal profile with above average
        ability to be flexible and not simplistic.").

if([anxiety,high,tma,high],34):-
        main_feature(anxiety,"  -anxious.").

if([anxious,main,conformity,high],35):-
        secondary_feature(fitting_in,
   "  -handles anxiety by conforming.").

if([energy_level,high,breadth_of_interest,high,complexity,high,
   value_orthodoxy,low],36):-
        secondary_feature(liberal,
        "  -interested in a wide variety of subjects
        enthusiastic,flexible.").
```

```
if([d,high,hy,high,pt,high,k,high],37):-
        main_feature(neurotic,
             "  -neuotic,depressed,defensive,worried.").

if([neurotic,main,social-participation,high]):-
        secondary_feature(fitting_in,
        " -covers depression by becoming involved with people.").

if([paranoid,main,social_adroitness,low],31):-
        secondary_feature(antagonistic,
        "  _handles people in inappropriate ways which
             creates hostility
        and aggravates the sense of paranoia.").

if([intellectual,main,anxiety,high,pt,high],10):-
        secondary_feature(perfectionist,
        "  -high,perfectionist demands create anxiety,
            self-dissatisfaction.").

if([intellectual,main,anxiety,high,conformity,high,
      social_participation,high],
  11):-secondary_feature(fitting_in,
        "  -relies on others for emotional support,conforms
              to perceived peer norms.").

if([fittin_in,secondary,energy_level,low],13):-
        secondary_feature(tired,
        "  -working so hard to avoid fear,tired and immobilized.").

if([practical,main,hs,low,hy,low],24):-
        secondary_feature(unpsych,"  -blunt,unpsychological.").

if([social_adroitness,high,responsibility,normal,k,normal],25):-
      secondary_feature
      (manipulative,"  -minor degree of manipulativeness.").

if([social_adroitness,high,responsibility,low],26):-
        secondary_feature(manipulative,"  -very manipulative.").

if([practical,main,interpersonal_affect,low],20):-
        secondary_feature(machine,"  -machine-like.").

if([tolerance,extremelow,pd,high],40):-
        secondary_feature(intolerant," - intolerant.").

if([depression,main,interpersonal_affect,low,si,high],2):-
        secondary_feature(avoidance_people,
        "  -using avoidance to cope with the depression
         by withdrawing
         from people.").

if([depression,main,breadth_of_interest,low,organization,
         low,responsibility,
        low],3):-
        secondary_feature(avoidance_life,
        "  -using avoidance to cope with the depression by
         avoiding the
        demands of situations and events.").

if([depression,main,organization,low,responsibility,low],4):-
        secondary_feature(disorganized,
        "  -unwilling to make plans and accept responsibility.").

if([intolerant,_],police_job,2,21):-
        prtstr("  intolerance may provoke incidents.",1).


if([neurotic,main],police_job,3,38):-
        prtstr("  obsessed with own problems,ineffective.",1).

if([paranoid,main,antagonistic,secondary],police_job,3,32):-
        prtstr("  suspicion and inability to handle people
       will constantly
        create bad incidents.",1).

if([practical,main,energy_level,high],police_job,1,22):-
        prtstr("  hard working",1).

if([par_schiz,main], police_job,3,18):-
        prtstr("out of touch with reality.",1).


if([avoidance_people,_],police_job,2,5):-
      prtstr("  -avoidance will make the person unwilling to
       participate",1).

if([fitting_in,_],police_job,2,15):-
        prtstr("  likely to make decisions to be popular rather than
         to be right.",1).

if([perfectionist,_],police_job,1,16):-
      prstr("  worry about perfectionism may interfere with
              performance.",1).

if([burned_out,main],police_job,1,8):-
        prtstr("  determine if reactive.",1).

if([disorganized,_],police_job,2,6):-
        prtstr("  lack of organized thought and inability to follow
         through plans   will make learning a new job difficult.",1).
```

--

```
/* Clauses used to evaluate raw scores.    */

/* v(category,scale,low range of normal,high range,
standard deviation.  */

/*  1=male police officers  */

v(1,anxiety,4,12,5).
v(1,breadth_of_interest,6,14,4).
v(1,complexity,6,10,3).
v(1,conformity,4,11,4).
v(1,energy_level,10,16,4).
v(1,innovation,7,15,3).
v(1,interpersonal_affect,6,13,4).
v(1,organization,10,16,4).
v(1,responsibility,12,17,3).
v(1,risk_taking,3,10,4).
v(1,self_esteem,11,18,4).
v(1,social_adroitness,6,12,4).
v(1,social_participation,7,14,4).
v(1,tolerance,9,15,4).
v(1,value_orthodoxy,9,15,4).
v(1,infrequency,0,1,1).


/*mmpi scales*/

v(1,l,3,7,3).
v(1,f,1,4,2).
v(1,k,14,22,4).
v(1,hs,8,13,2).
v(1,d,14,18,3).
v(1,hy,16,22,3).
v(1,pd,18,24,4).
v(1,mf,19,26,4).
v(1,pa,6,11,2).
v(1,pt,19,27,4).
v(1,sc,19,27,4).
v(1,ma,16,23,3).
v(1,si,12,24,5).
v(1,con,1,2,1).
v(1,tma,6,15,5).



/*    2=female polic officers    */

v(2,l,3,6,2).
v(2,f,1,3,2).
v(2,k,15,22,4).
v(2,hs,8,13,3).
v(2,d,15,19,2).
v(2,hy,16,23,4).
v(2,pd,19,24,3).
```

```
v(2,mf,29,42,11).
v(2,pa,7,11,2).
v(2,pt,22,28,3).
v(2,sc,21,26,3).
v(2,ma.18,25,4).
v(2,si,12,23,5).
v(2,con.1,2,1).
v(2,tma,7,15,3).
```

--

APPENDIX III

Sample 1. Main Feature is Depression.

PSYCHOLOGICAL TESTING

Assessment is based on:

the Jackson Personality Inventory

the Minnesota Multiphasic Personality Inventory

Assessment is for police_job

Name: _940, _1015

Sex: male

Profile:

1 -the main feature is depression.

2 -using avoidance to cope with the depression

by withdrawing from people.

3 -using avoidance to cope with the depression

by avoiding the demands of situations and events.

Recommendation:

-avoidance will make the person unwilling to participate.

-lack of organized thought and inability to follow

through plans will make learning a new job difficult.

The profile raises serious concerns which could indicate

rejection. These concerns should be pursued carefully

at the interview.

Sample 2. Main Feature is an Intellectualizer with perfectionist tendencies.

PSYCHOLOGICAL TESTING

Assessment is based on:

the Jackson Personality Inventory

the Minnesota Multiphasic Personality Inventory

Assessment is for police_job

Name:

Sex: male

Profile:

1 -highly verbal, intellectualizer

2 -perfectionist demands create anxiety, self-dissatisfaction

Recommendation:

-worry about perfectionism may interfere with performance

The profile raises serious concerns which could indicate

rejection. These concerns should be pursued carefully

at the interview.

Sample 3. An intellectual with high anxiety and conformity

PSYCHOLOGICAL TESTING

Assessment is based on:

the Jackson Personality Inventory

the Minnesota Multiphasic Personality Inventory

Assessment is for police_job

Name:

Sex: male

Profile:

1 -highly verbal, intellectualizer

2 -relies on others for emotional support, conforms to
perceived peer norms

Recommendation:

-may make decisions to be popular rather than to
be right

Results are basically within normal ranges but the profile
raises some concerns which should be assessed at the interview.

--

4. A Normal Profile.

PSYCHOLOGICAL TESTING

Assessment is based on:

the Jackson Personality Inventory

the Minnesota Multiphasic Personality Inventory

Assessment is for police_job

Name:

Sex: male

Profile:

1 -the profile is essentially normal with above average

ability to be flexible and not simplistic

Recommendation:

Recommend hiring.

--

5. A Serious Disorder

PSYCHOLOGICAL TESTING

Assessment is based on:

the Jackson Personality Inventory

the Minnesota Multiphasic Personality Inventory

Assessment is for police_job

Name:

Sex: male

Profile:

1 -very suspicious of everyone

2 -handles people in inappropriate ways which creates
hostility and aggravates the sense of paranoia

Recommendation:

-suspicion and inability to handle people will create
bad incidents

Recommend rejection.

ACKNOWLEDGEMENTS

Research does not occur in a vacuum. Many friends and colleagues provided stimulating discussions, constructive criticism and endless support. Special mention must be made of John Cleary for his patience, support and insight into the issues of knowledge representation; of the Police Psychological Services for allowing their procedures and files to be used; of Dr. Ed Kramer who provided his expertise as a psychologist with patience and skill and also his support and encouragement; and Mireille who finally decided to sleep through the night.