Chapter 10

Improving Decision Making and Managing Knowledge

LEARNING TRACK 2: THE EXPERT SYSTEMS INFERENCE ENGINE

HOW EXPERT SYSTEMS WORK

Human knowledge must be modeled or represented in a way that a computer can process. The model of human knowledge used by expert systems is called the knowledge base. A standard programming construct is the IF–THEN construct, in which a condition is evaluated. If the condition is true, an action is taken. For instance,

IF INCOME > $45,000 (condition)

THEN PRINT NAME AND ADDRESS (action)

A series of these rules can be a knowledge base. A software program for an expert system may have 200 to 10,000 of these rules, which are much more interconnected and nested than in a traditional software program.

Could you represent the knowledge in the Encyclopedia Britannica this way? Probably not, because the rule base would be too large, and not all the knowledge in the encyclopedia can be represented in the form of IF–THEN rules. In general, expert systems can be efficiently used only in those situations in which the domain of knowledge is highly restricted (such as in granting credit) and involves no more than a few thousand rules.

The AI shell is the programming environment of an expert system. In the early years of expert systems, computer scientists used specialized artificial intelligence programming languages, such as LISP or Prolog, that could process lists of rules efficiently. Today, a growing number of expert systems use AI shells that are user-friendly development environments. AI shells can quickly generate user-interface screens, capture the knowledge base, and manage the strategies for searching the rule base.

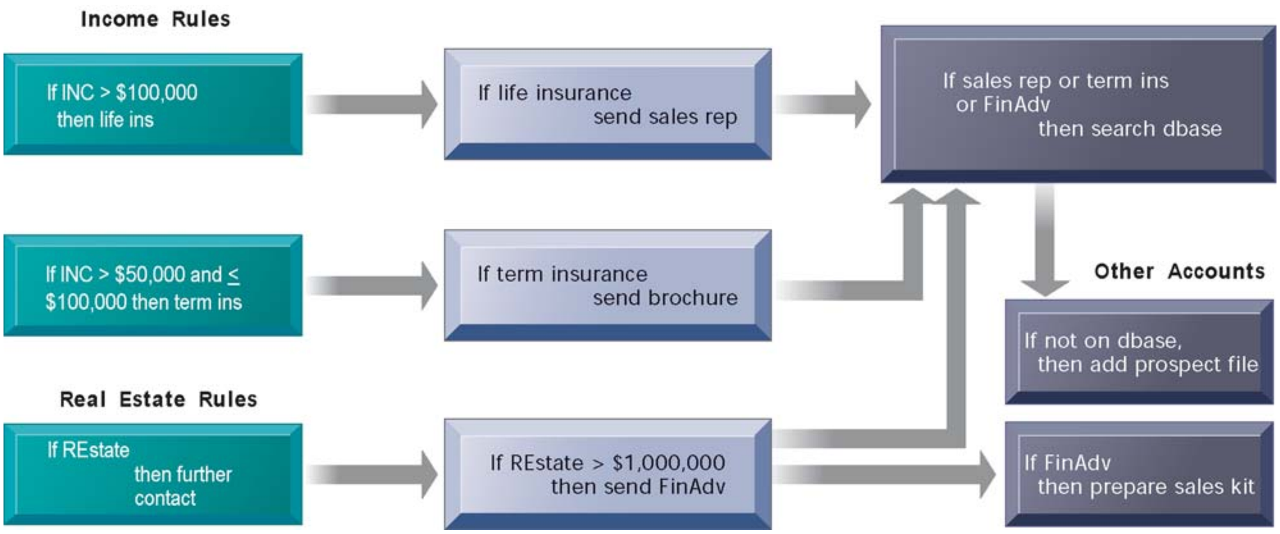
The strategy used to search through the rule base is called the inference engine. Two strategies are commonly used: forward chaining and backward chaining (see Figure 10-1).

In forward chaining the inference engine begins with the information entered by the user and searches the rule base to arrive at a conclusion. The strategy is to fire, or carry out, the action of the rule when a condition is true. In Figure 10-1, beginning on the left, if the user enters a client's name with income greater than $100,000, the engine will fire all rules in sequence from left to right. If the user then enters information indicating that the same client owns real estate, another pass of the rule base will occur and more rules will fire. Processing continues until no more rules can be fired.

In backward chaining the strategy for searching the rule base starts with a hypothesis and proceeds by asking the user questions about selected facts until the hypothesis is either confirmed or disproved. In our example, in Figure 10-1, ask the question, "Should we add this person to the prospect database?" Begin on the right of the diagram and work toward the left. You can see that the person should be added to the database if a sales representative is sent, term insurance is granted, or a financial adviser visits the client.

Developing an expert system requires input from one or more experts, who have a thorough command of the knowledge base, and one or more knowledge engineers, who can translate the knowledge (as described by the expert) into a set of rules. A knowledge engineer is similar to a traditional systems analyst but has special expertise in eliciting information and expertise from other professionals.

FIGURE 10-1 Inference engines in expert systems.



An inference engine works by searching through the rules and "firing" those rules that are triggered by facts gathered and entered by the user. Basically, a collection of rules is similar to a series of nested IF statements in a traditional software program; however, the magnitude of the statements and degree of nesting are much greater in an expert system.

An expert systems development project balances potential savings from the proposed system against the cost. The team members develop a prototype system to test assumptions about how to encode the knowledge of experts. Next, they develop a full-scale system, focusing mainly on the addition of a very large number of rules. The complexity of the entire system grows with the number of rules, so the comprehensibility of the system may be threatened. Generally, the system is pruned to achieve simplicity and power. The system is tested by a range of experts within the organization against the performance criteria established earlier. Once tested, the system is integrated into the data flow and work patterns of the organization.