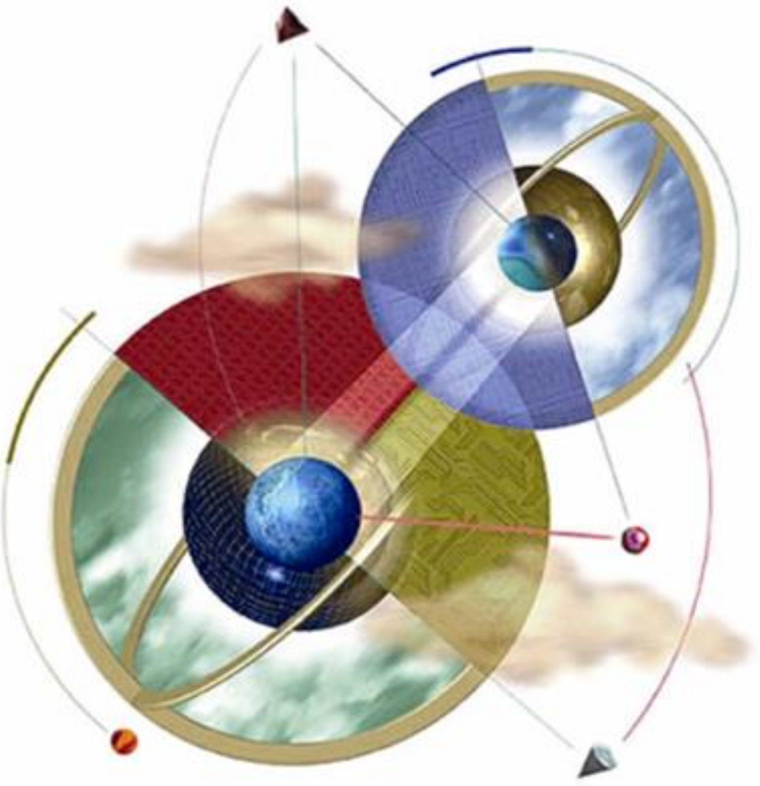


# Chapter 1: Introduction to Expert Systems

Expert Systems: Principles and Programming, Fourth Edition

# Objectives

- Learn the meaning of an expert system
- Understand the problem domain and knowledge domain
- Learn the advantages of an expert system
- Understand the stages in the development of an expert system
- Examine the general characteristics of an expert system

# Objectives

- Examine earlier expert systems which have given rise to today's knowledge-based systems
- Explore the applications of expert systems in use today
- Examine the structure of a rule-based expert system
- Learn the difference between procedural and nonprocedural paradigms
- What are the characteristics of artificial neural systems

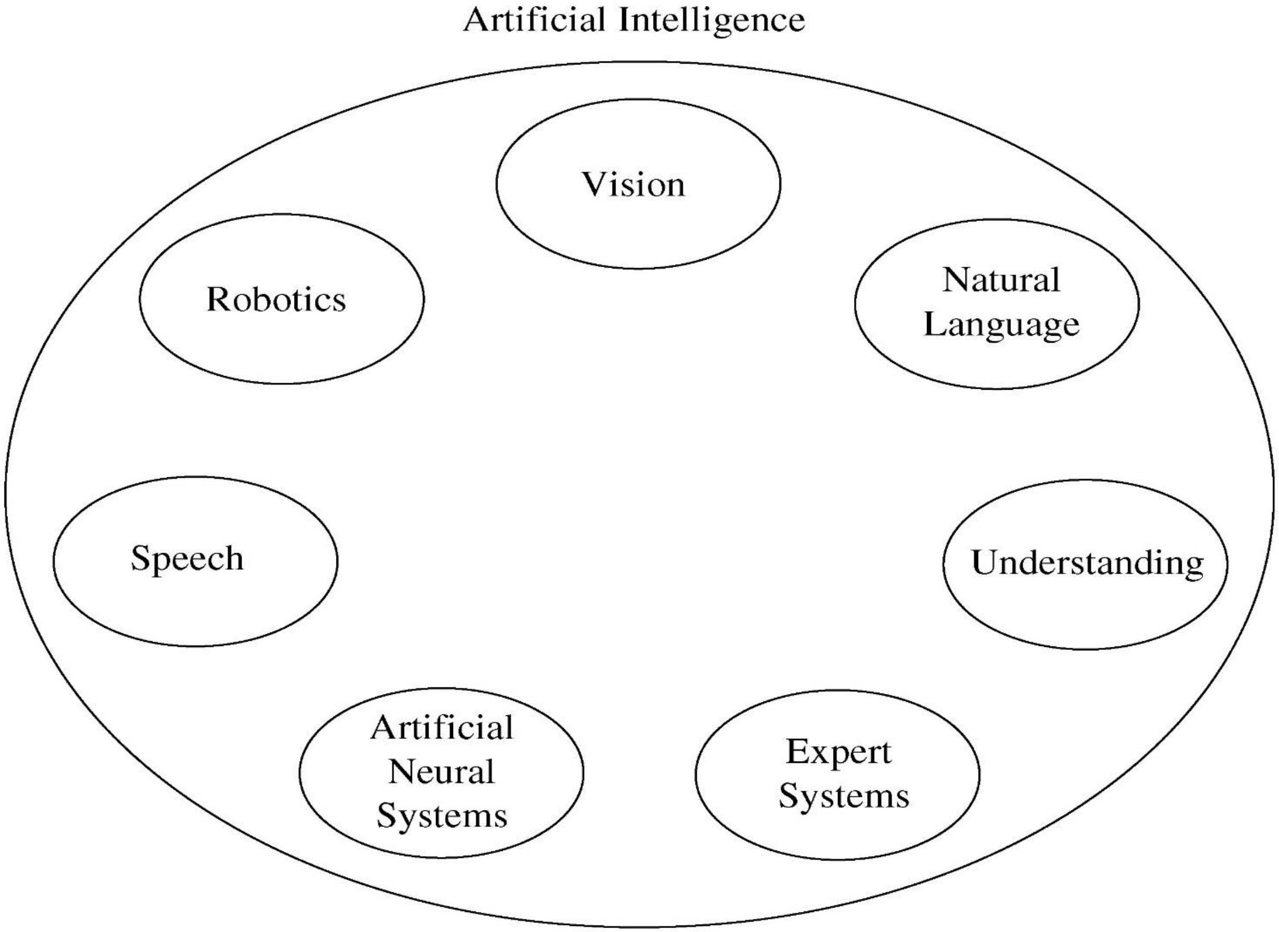
# What is an expert system?

- "An expert system is a computer system that emulates, or acts in all respects, with the decision-making capabilities of a human expert."
- "An intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solutions."

Professor Edward Feigenbaum

Stanford University

# Fig 1.1 Areas of Artificial Intelligence



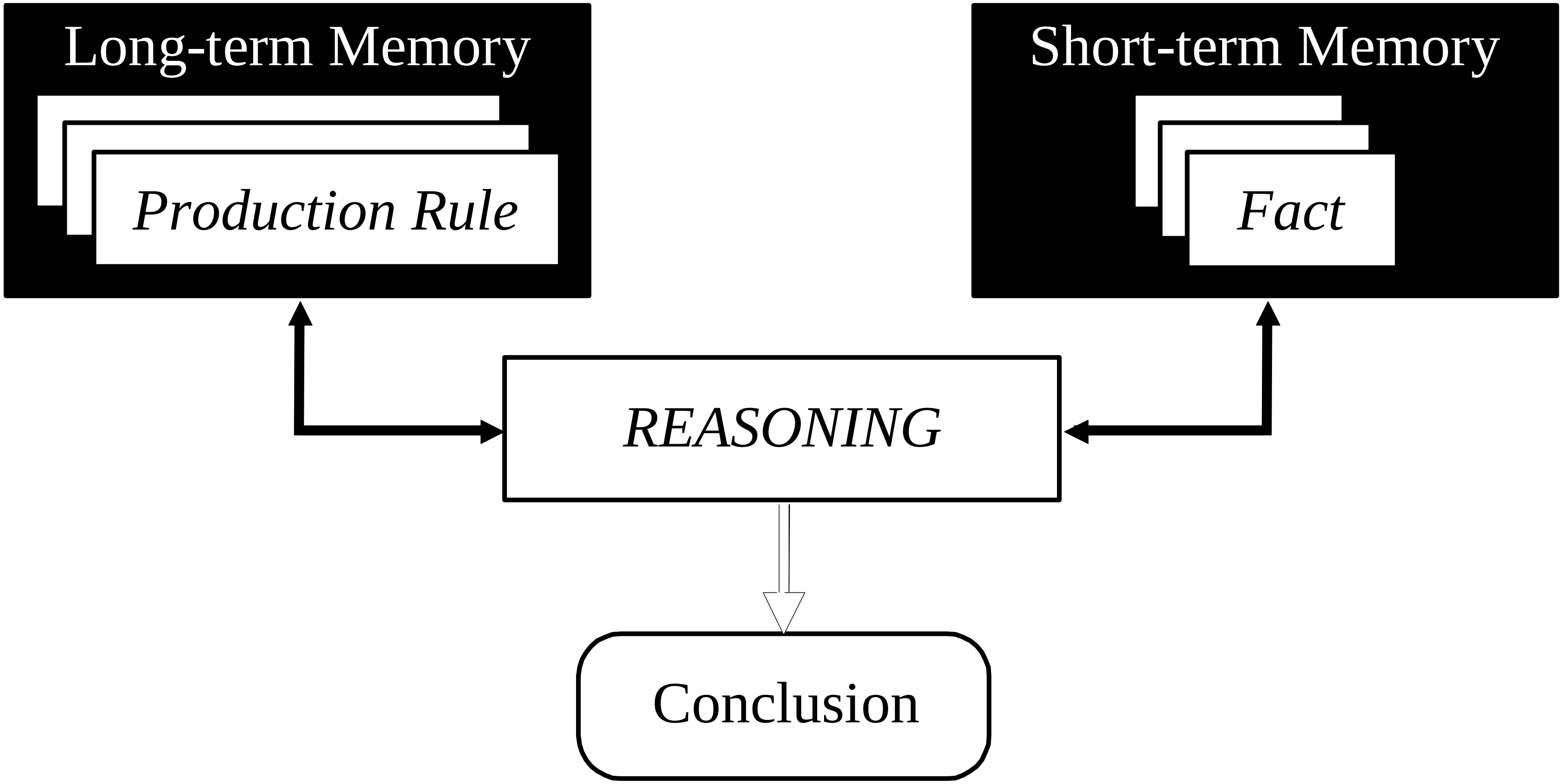
# Expert system technology may include:

- Special expert system languages – CLIPS
- Programs
- Hardware designed to facilitate the implementation of those systems

# Structure of a rule-based expert system

- In the early seventies, Newell and Simon from Carnegie-Mellon University proposed a production system model, the foundation of the modern rule-based expert systems.
- The production model is based on the idea that humans solve problems by applying their knowledge (expressed as production rules) to a given problem represented by problem-specific information.
- The production rules are stored in the long-term memory and the problem-specific information or facts in the short-term memory.

# Production system model



Long-term Memory

*Production Rule*

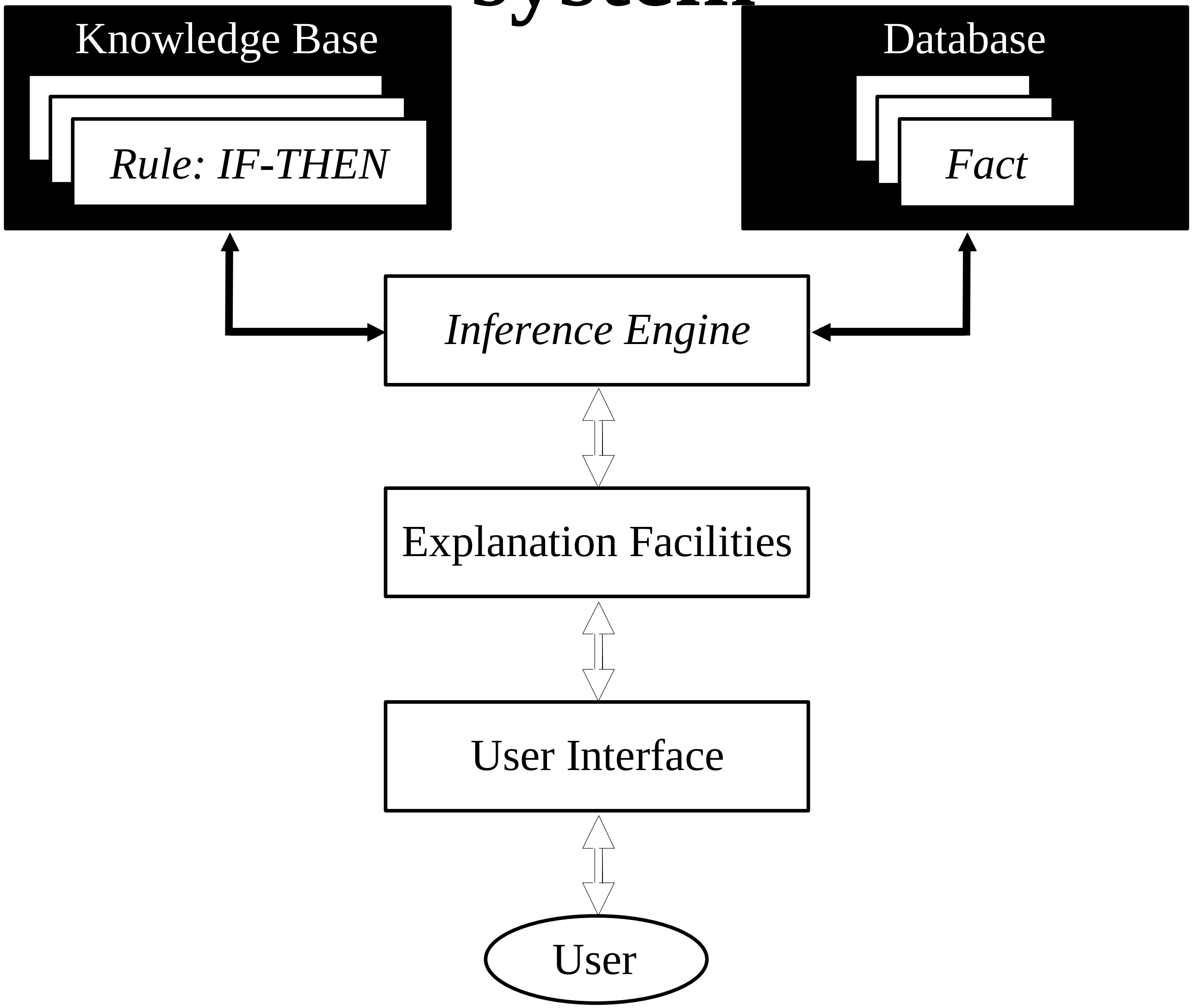
Short-term Memory

*Fact*

*REASONING*

Conclusion

# Basic structure of a rule-based expert system



- **The knowledge base** contains the domain knowledge useful for problem solving. In a rule-based expert system, the knowledge is represented as a set of rules. Each rule specifies a relation, recommendation, directive, strategy or heuristic and has the IF (condition) THEN (action) structure. When the condition part of a rule is satisfied, the rule is said to fire and the action part is executed.
- **The database** includes a set of facts used to match against the IF (condition) parts of rules stored in the knowledge base.

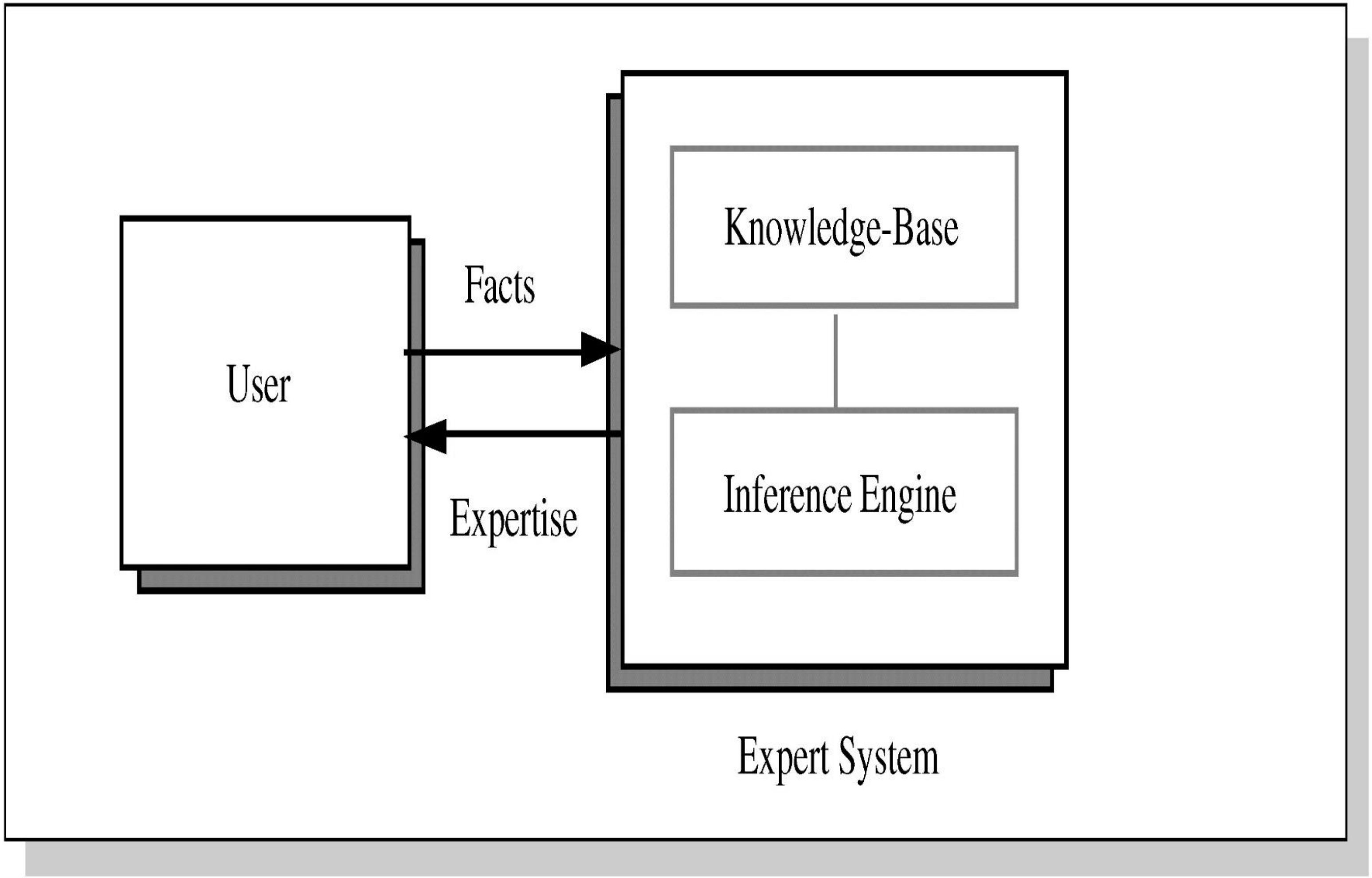
- **The inference engine** carries out the reasoning whereby the expert system reaches a solution. It links the rules given in the knowledge base with the facts provided in the database.
- The **explanation facilities** enable the user to ask the expert system ***how*** a particular conclusion is reached and ***why*** a specific fact is needed. An expert system must be able to explain its reasoning and justify its advice, analysis or conclusion.
- The **user interface** is the means of communication between a user seeking a solution to the problem and an expert system.

# Expert System Main Components

- Knowledge base – obtainable from books, magazines, knowledgeable persons, etc.

- Inference engine – draws conclusions from the knowledge base

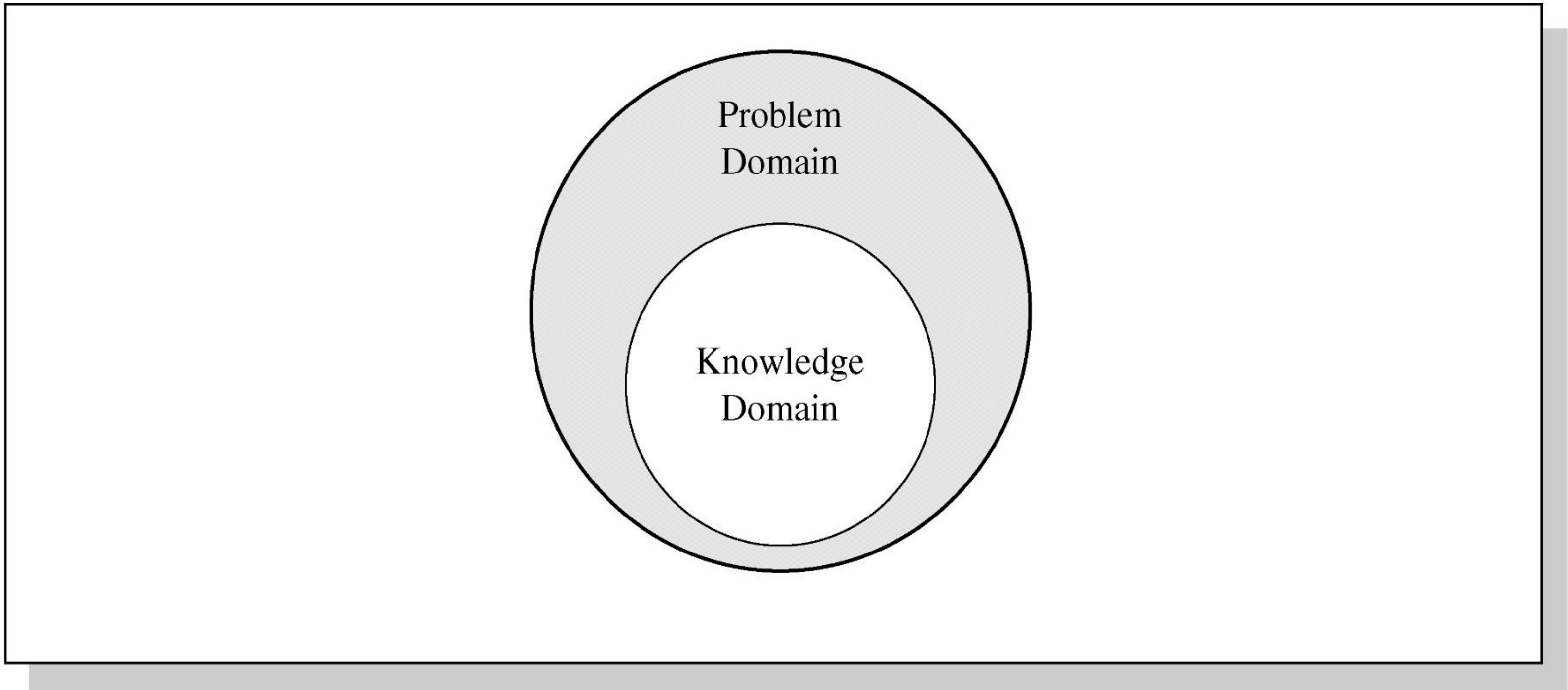
# Figure 1.2 Basic Functions of Expert Systems



# Problem Domain vs. Knowledge Domain

- An expert’s knowledge is specific to one problem domain – medicine, finance, science, engineering, etc.
- The expert’s knowledge about solving specific problems is called the knowledge domain.
- The problem domain is always a superset of the knowledge domain.

# Figure 1.3 Problem and Knowledge Domain Relationship



# Advantages of Expert Systems

- Increased availability
- Reduced cost
- Reduced danger
- Performance
- Multiple expertise
- Increased reliability

# Advantages Continued

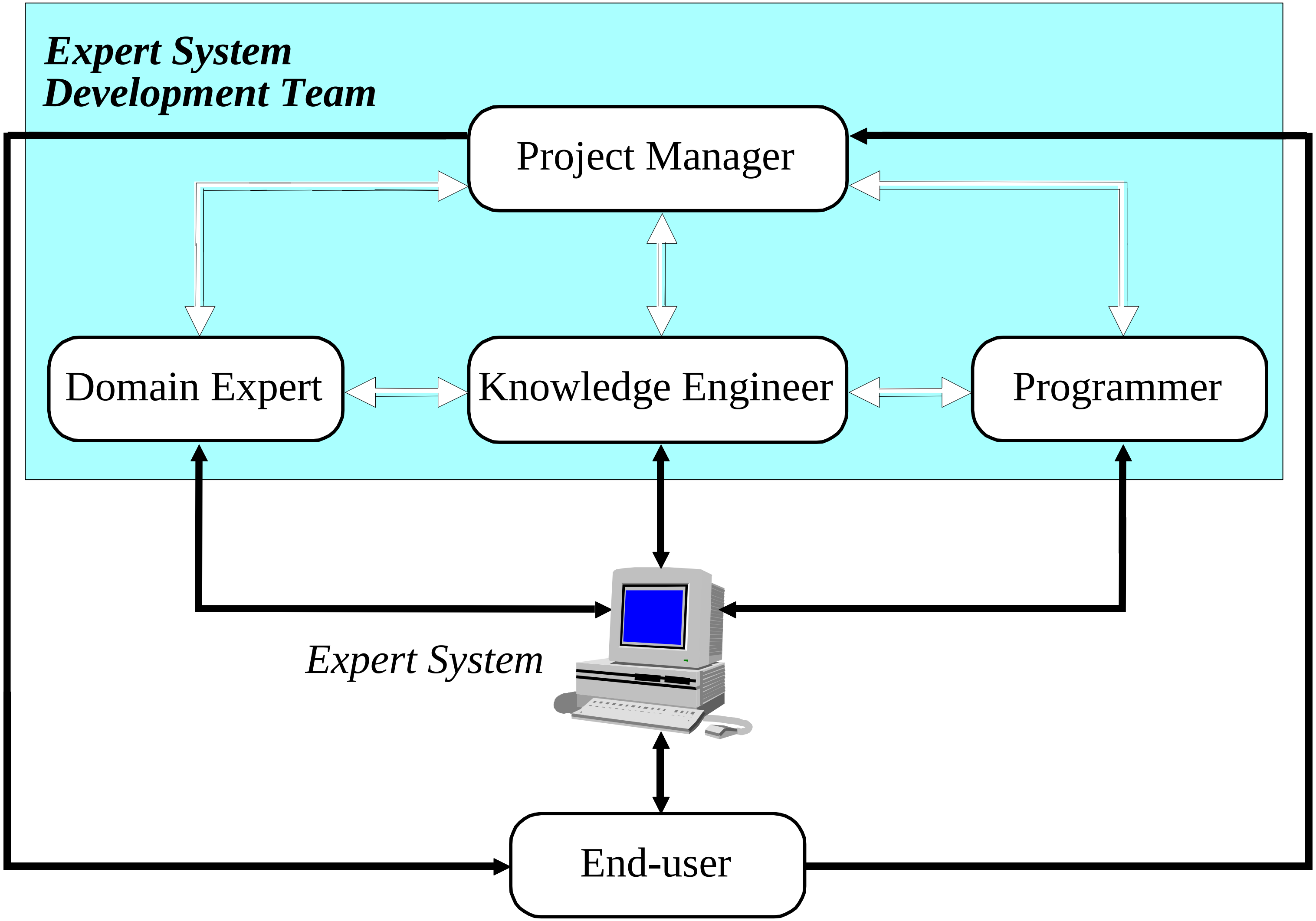
- Explanation
- Fast response
- Steady, unemotional, and complete responses at all times
- Intelligent tutor
- Intelligent database

# Representing the Knowledge

The knowledge of an expert system can be represented in a number of ways, including IF-THEN rules:

```
IF you are hungry THEN eat
```

# Development of an Expert System



*Expert System*
*Development Team*

Project Manager

Domain Expert

Knowledge Engineer

Programmer

*Expert System*

End-user

- The domain expert is a knowledgeable and skilled person capable of solving problems in a specific area or domain.  This person has the greatest expertise in a given domain.  This expertise is to be captured in the expert system.  Therefore, the expert must be able to communicate his or her knowledge, be willing to participate in the expert system development and commit a substantial amount of time to the project.  The domain expert is the most important player in the expert system development team.

- The knowledge engineer is someone who is capable of designing, building and testing an expert system. He or she interviews the domain expert to find out how a particular problem is solved.
- The knowledge engineer establishes what reasoning methods the expert uses to handle facts and rules and decides how to represent them in the expert system.
- The knowledge engineer then chooses some development software or an expert system shell, or looks at programming languages for encoding the knowledge.  And finally, the knowledge engineer is responsible for testing, revising and integrating the expert system into the workplace.

- The programmer is the person responsible for the actual programming, describing the domain knowledge in terms that a computer can understand.
- The project manager is the leader of the expert system development team, responsible for keeping the project on track. He or she makes sure that all deliverables and milestones are met, interacts with the expert, knowledge engineer, programmer and end-user.
- The end-user, often called just the user, is a person who uses the expert system when it is developed. The user must not only be confident in the expert system performance but also feel comfortable using it.

# The Role of AI

- An algorithm is an ideal solution guaranteed to yield a solution in a finite amount of time.
- When an algorithm is not available or is insufficient, we rely on artificial intelligence (AI).
- Expert system relies on inference – we accept a “reasonable solution.”

# Uncertainty

- Both human experts and expert systems must be able to deal with uncertainty.
- It is easier to program expert systems with shallow knowledge than with deep knowledge.
- Shallow knowledge – based on empirical and heuristic knowledge.
- Deep knowledge – based on basic structure, function, and behavior of objects.

# Limitations of Expert Systems

- Typical expert systems cannot generalize through analogy to reason about new situations in the way people can.

- A knowledge acquisition bottleneck results from the time-consuming and labor intensive task of building an expert system.

# Early Expert Systems

- DENDRAL – used in chemical mass spectroscopy to identify chemical constituents
- MYCIN – medical diagnosis of illness
- DIPMETER – geological data analysis for oil
- PROSPECTOR – geological data analysis for minerals
- XCON/R1 – configuring computer systems

# Problems with Algorithmic Solutions

- Conventional computer programs generally solve problems having algorithmic solutions.
- Algorithmic languages include C, Java, and C#.
- Classical AI languages include LISP and PROLOG.

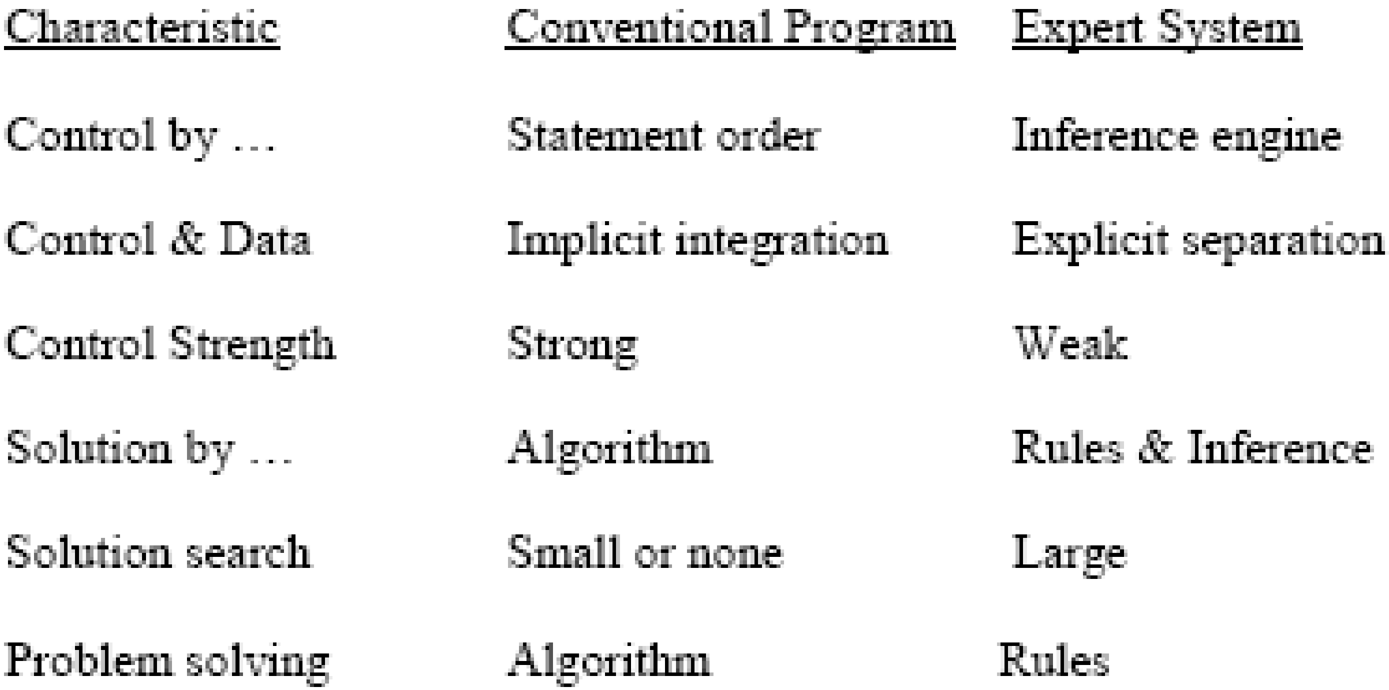
# Considerations for Building Expert Systems

- Can the problem be solved effectively by conventional programming?
- Is there a need and a desire for an expert system?
- Is there at least one human expert who is willing to cooperate?
- Can the expert explain the knowledge to the knowledge engineer can understand it.
- Is the problem-solving knowledge mainly heuristic and uncertain?

# Languages, Shells, and Tools

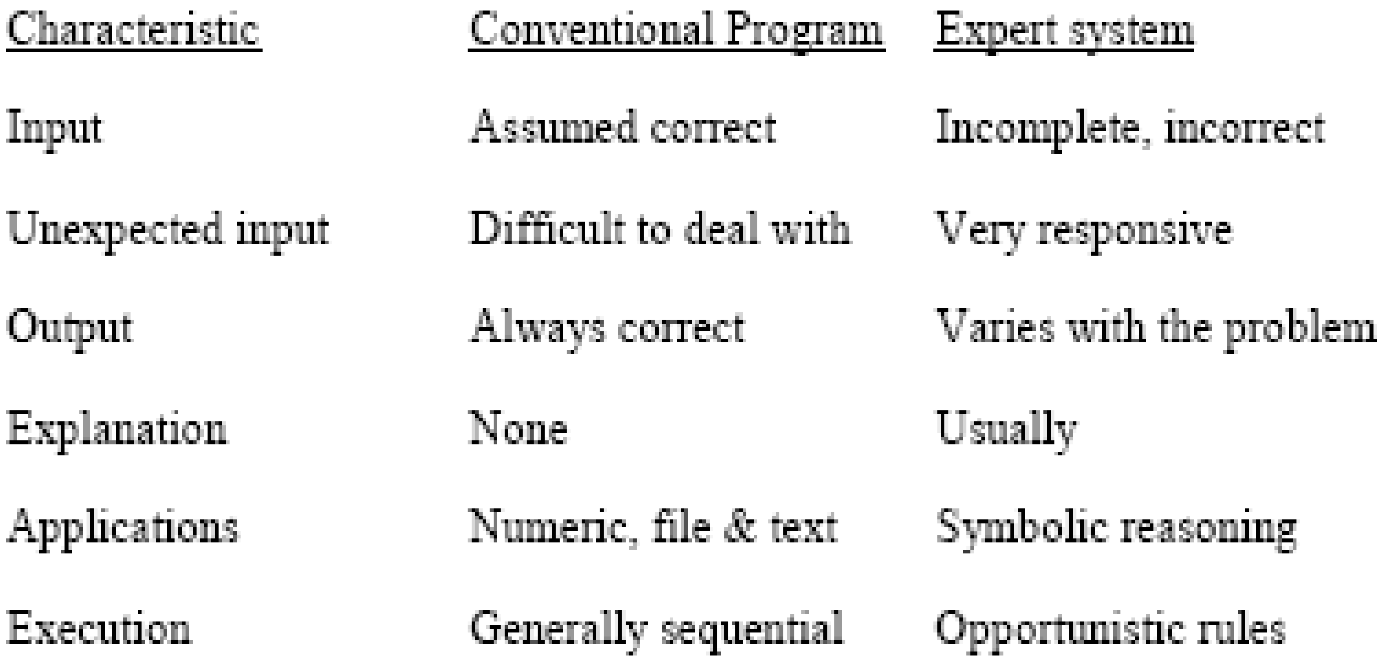
- Expert system languages are post-third generation.
- Procedural languages (e.g., C) focus on techniques to represent data.
- More modern languages (e.g., Java) focus on data abstraction.
- Expert system languages (e.g. CLIPS) focus on ways to represent knowledge.

# Expert systems Vs conventional programs I



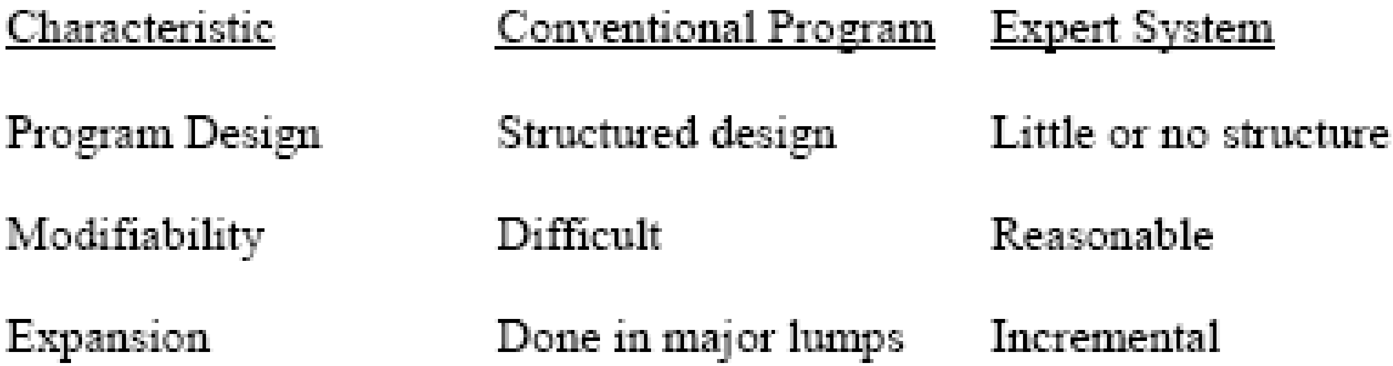
| Characteristic | Conventional Program | Expert System |
|---|---|---|
| Control by … | Statement order | Inference engine |
| Control & Data | Implicit integration | Explicit separation |
| Control Strength | Strong | Weak |
| Solution by … | Algorithm | Rules & Inference |
| Solution search | Small or none | Large |
| Problem solving | Algorithm | Rules |

# Expert systems Vs conventional programs II



| Characteristic | Conventional Program | Expert system |
|---|---|---|
| Input | Assumed correct | Incomplete, incorrect |
| Unexpected input | Difficult to deal with | Very responsive |
| Output | Always correct | Varies with the problem |
| Explanation | None | Usually |
| Applications | Numeric, file & text | Symbolic reasoning |
| Execution | Generally sequential | Opportunistic rules |

# Expert systems Vs conventional programs III



| Characteristic | Conventional Program | Expert System |
|---|---|---|
| Program Design | Structured design | Little or no structure |
| Modifiability | Difficult | Reasonable |
| Expansion | Done in major lumps | Incremental |

# Comparison of expert systems with conventional systems and human experts

| *Human Experts* | *Expert Systems* | *Conventional Programs* |
|---|---|---|
| Use knowledge in the form of rules of thumb or heuristics to solve problems in a narrow domain. | Process knowledge expressed in the form of rules and use symbolic reasoning to solve problems in a *narrow domain*. | Process data and use algorithms, a series of well-defined operations, to solve general numerical problems. |
| In a human brain, knowledge exists in a compiled form. | Provide a *clear separation of knowledge from its processing*. | Do not separate knowledge from the control structure to process this knowledge. |
| Capable of explaining a line of reasoning and providing the details. | *Trace the rules fired* during a problem-solving session and *explain how* a particular conclusion was reached and *why* specific data was needed. | Do not explain how a particular result was obtained and why input data was needed. |

# Comparison of expert systems with conventional systems and human experts (Continued)

| *Human Experts* | *Expert Systems* | *Conventional Programs* |
|---|---|---|
| Use inexact reasoning and can deal with incomplete, uncertain and fuzzy information. | Permit *inexact reasoning* and can deal with incomplete, uncertain and fuzzy data. | Work only on problems where data is complete and exact. |
| Can make mistakes when information is incomplete or fuzzy. | *Can make mistakes* when data is incomplete or fuzzy. | Provide no solution at all, or a wrong one, when data is incomplete or fuzzy. |
| Enhance the quality of problem solving via years of learning and practical training.  This process is slow, inefficient and expensive. | Enhance the quality of problem solving by adding new rules or adjusting old ones in the knowledge base.  When new knowledge is acquired, *changes are easy* to accomplish. | Enhance the quality of problem solving by changing the program code, which affects both the knowledge and its processing, making changes difficult. |