Overview of Expert Systems

Course outline

- Overview of Expert Systems
- ES and Knowledge Representation
- Study of Logic
- Knowledge representation techniques
- Inference methods
- Binary Decision trees and graphs
- Expert Systems Design
- Introduction to Clips

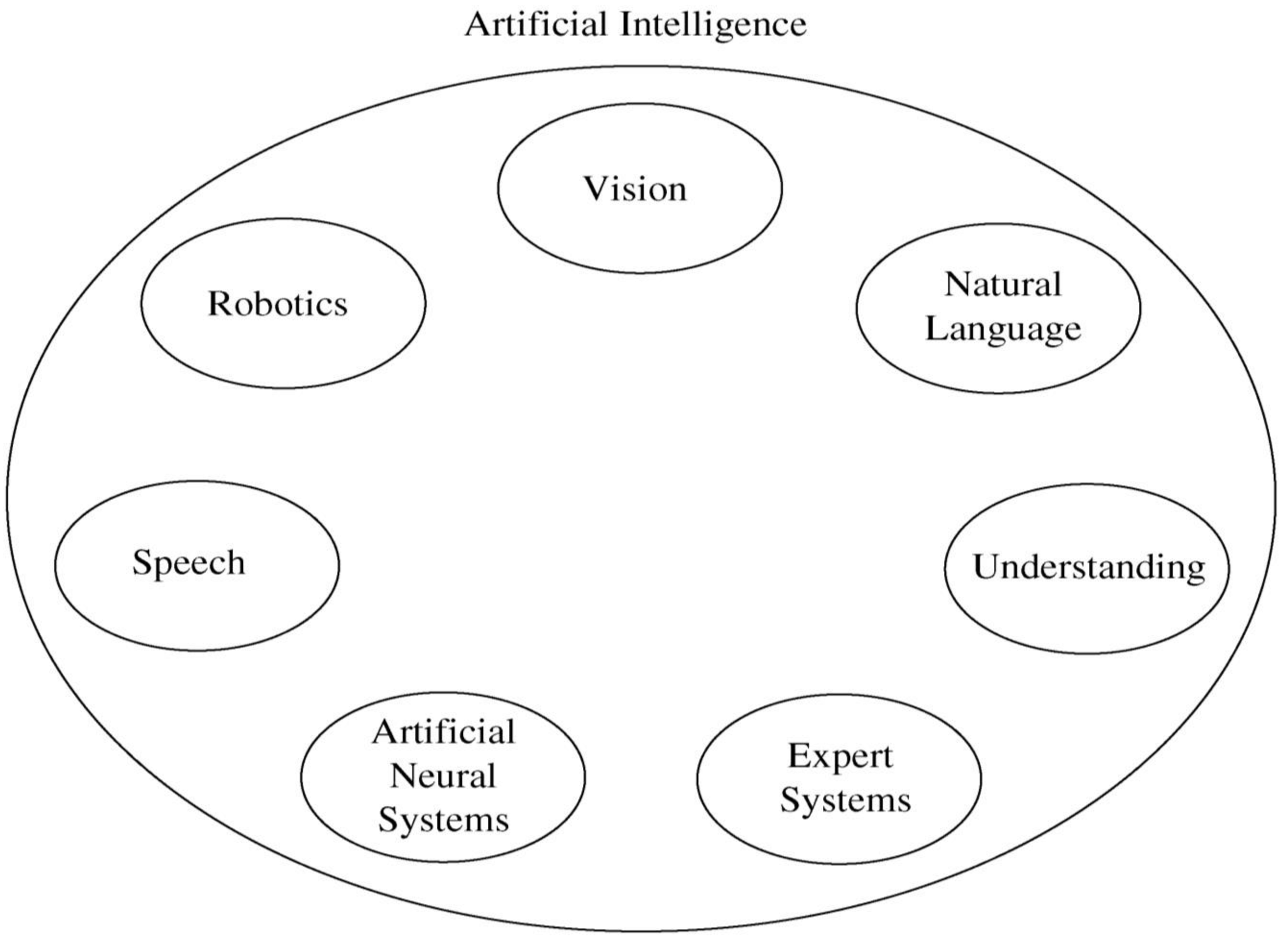
What is an expert system?

“An expert system is a computer system that emulates, or acts in all respects, with the decision-making capabilities of a human expert.”

Professor Edward Feigenbaum

Stanford University

Fig 1.1 Areas of Artificial Intelligence



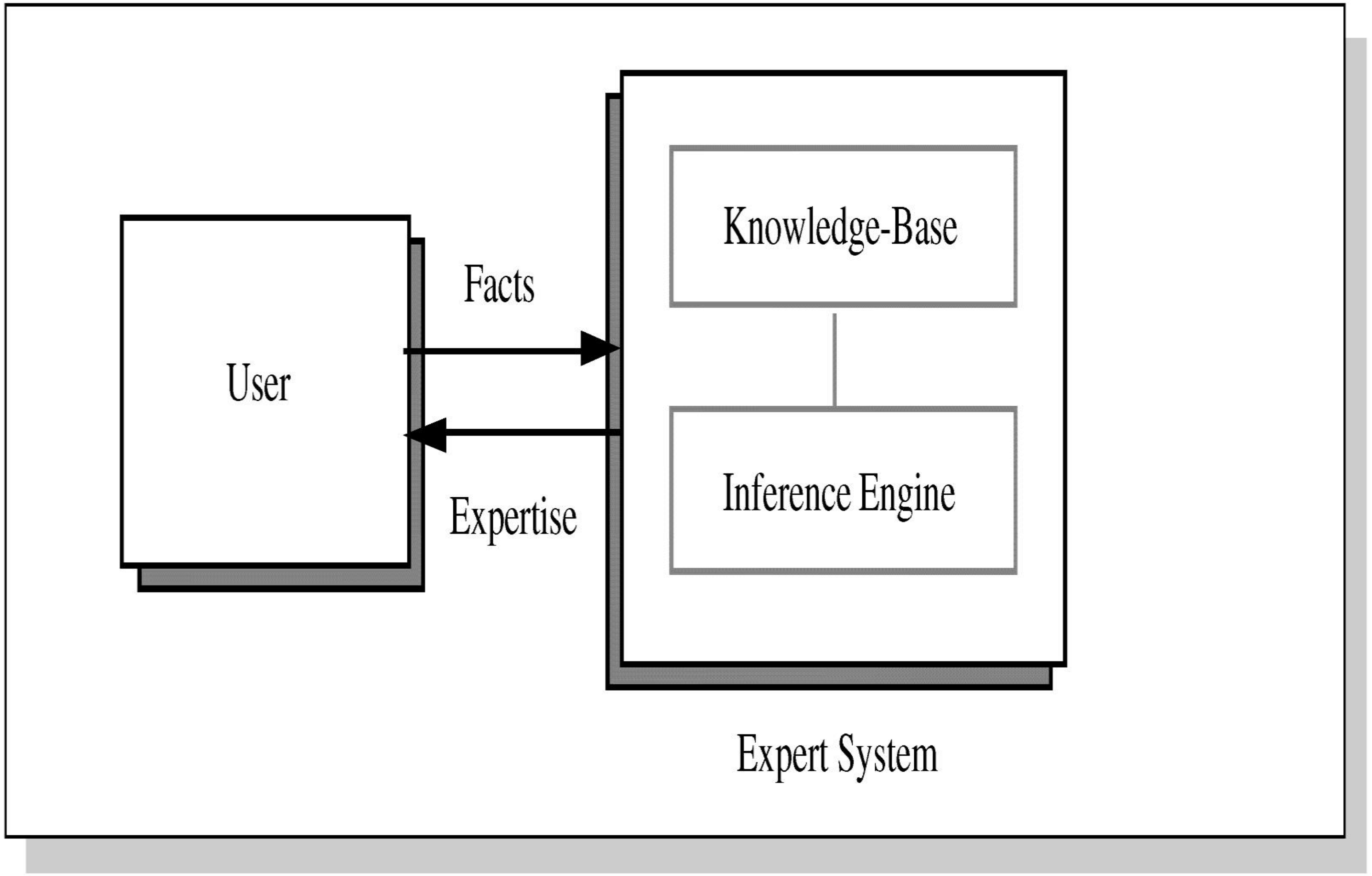
Expert system technology may include:

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

Expert System Main Components

- Knowledge base – obtainable from books, magazines, knowledgeable persons, etc.
- Inference engine – draws conclusions from the knowledge base

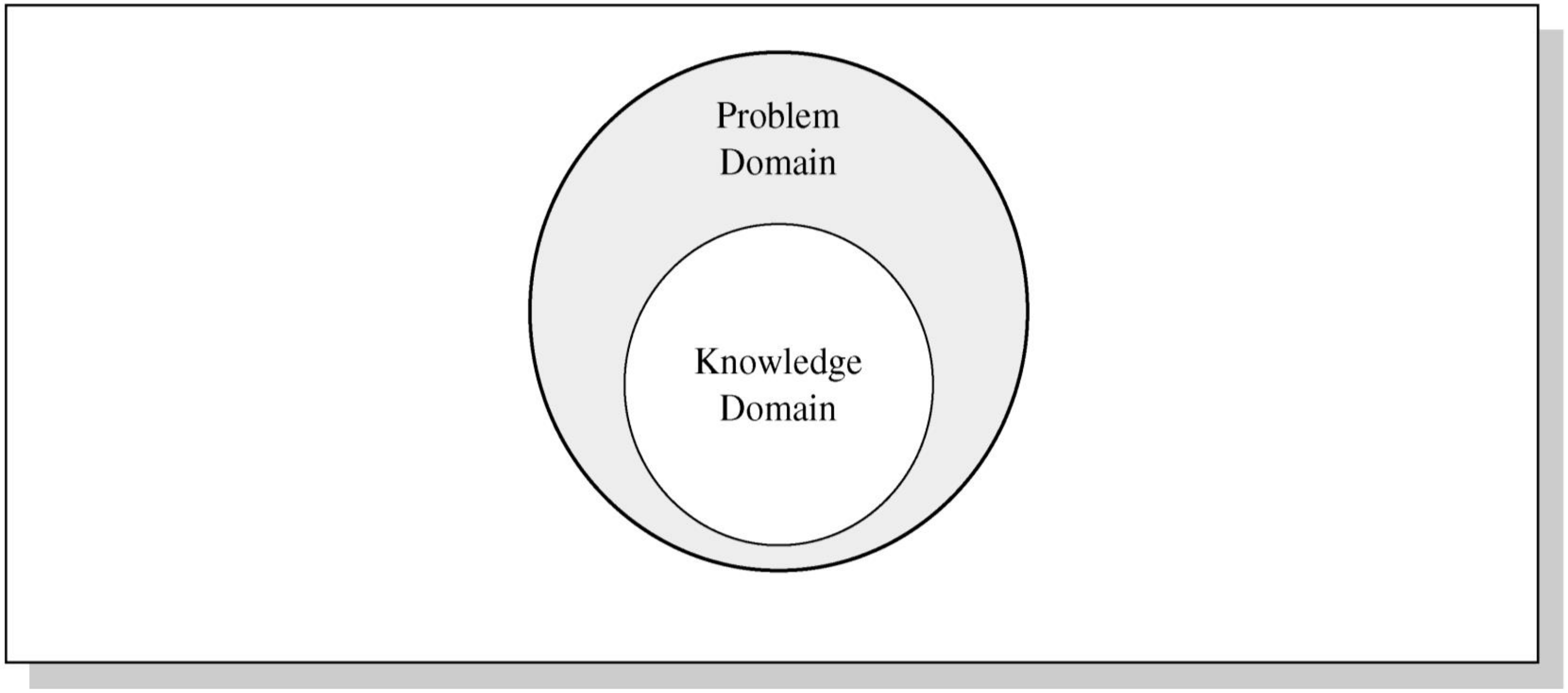
Figure 1.2 Basic Functions of Expert Systems



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:

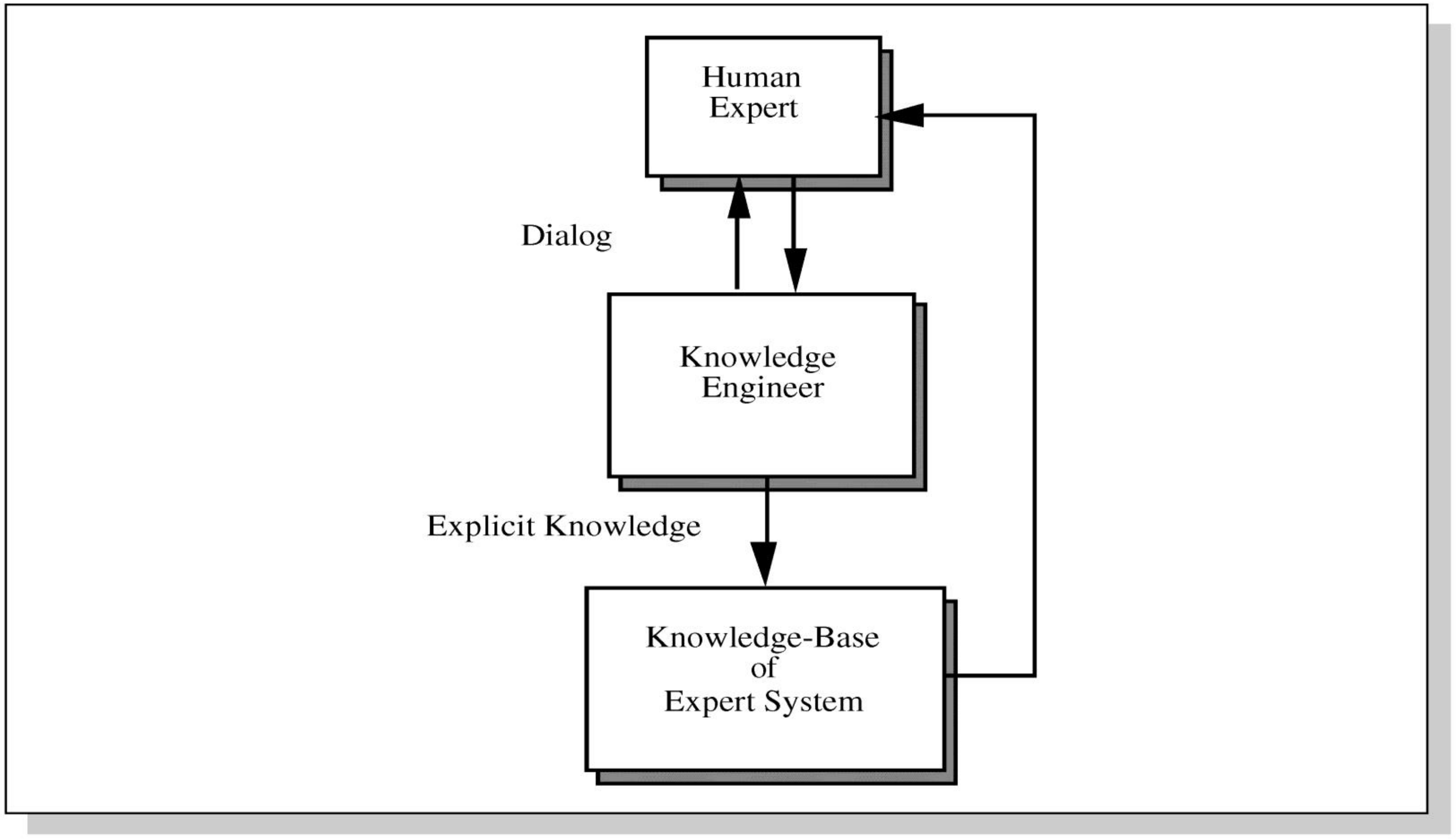
```
IF you are hungry THEN eat
```

Knowledge Engineering

The process of building an expert system:

1. The knowledge engineer establishes a dialog with the human expert to elicit knowledge.
2. The knowledge engineer codes the knowledge explicitly in the knowledge base.
3. The expert evaluates the expert system and gives a critique to the knowledge engineer.

Development of an Expert System



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.

Development of Expert Systems

- Rooted from Cognitive Studies:
  - How does human process information

- Newell/Simon Model (GPS)
  - Long Term Memory: IF-Then Rules
  - Short Term Memory: Current Facts
  - Inference Engine/Conflict Resolution

Rule Examples

- IF the car doesn’t run and the fuel gauge reads empty THEN fill the gas tank.

- IF there is flame, THEN there is a fire.
- IF there is smoke, THEN there may be a fire.
- IF there is a siren, THEN there may be a fire.

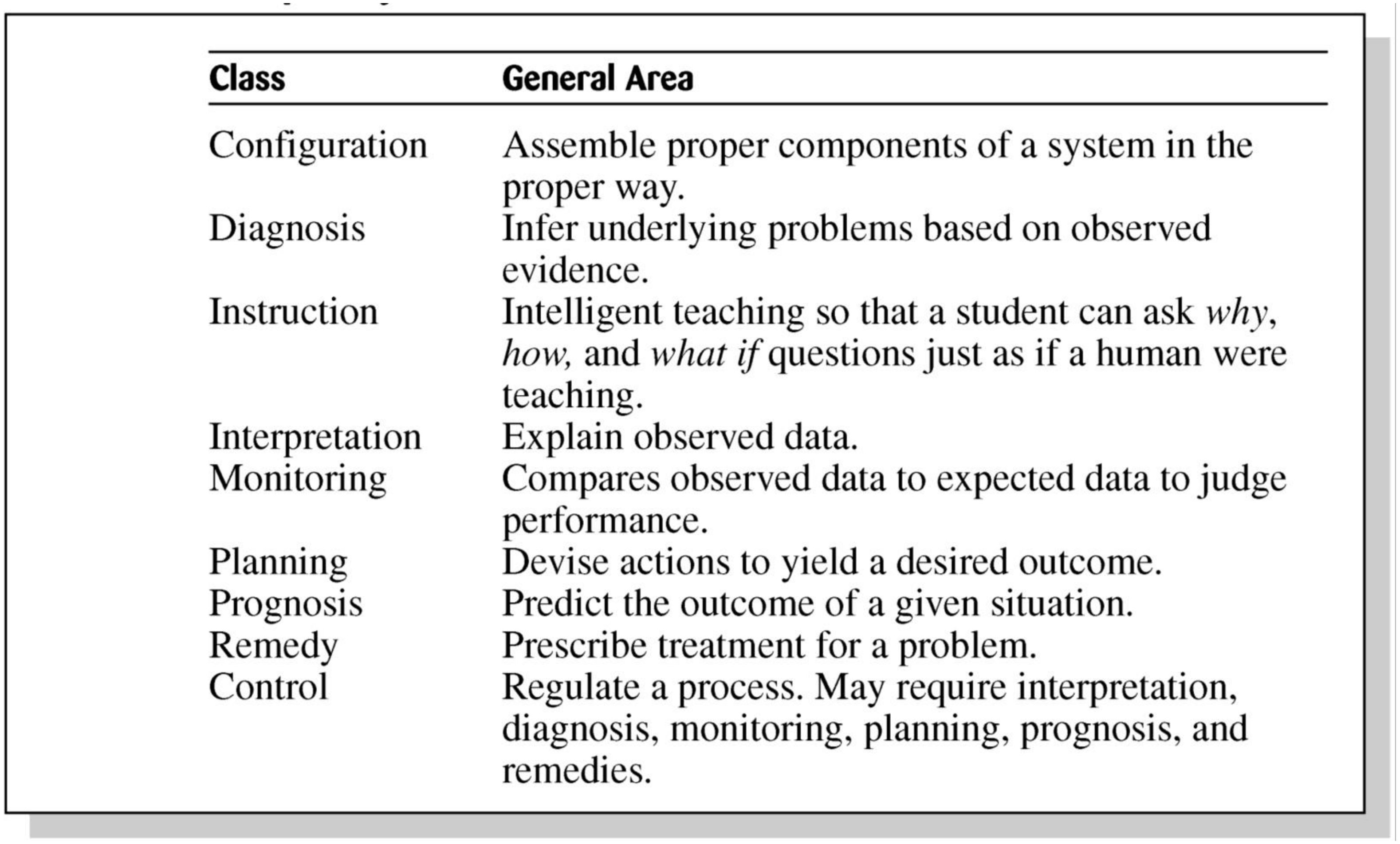
Expert Knowledge

- Base Knowledge / Expert Knowledge
  - Book Rules / Heuristics and Experiences (secrets!)
    - Experts usually score almost similar to novices in brand new problems.
  - Chess Rules / Chess Master Patterns

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

Expert Systems Applications and Domains



Class	General Area
Configuration	Assemble proper components of a system in the proper way.
Diagnosis	Infer underlying problems based on observed evidence.
Instruction	Intelligent teaching so that a student can ask *why*, *how,* and *what if* questions just as if a human were teaching.
Interpretation	Explain observed data.
Monitoring	Compares observed data to expected data to judge performance.
Planning	Devise actions to yield a desired outcome.
Prognosis	Predict the outcome of a given situation.
Remedy	Prescribe treatment for a problem.
Control	Regulate a process. May require interpretation, diagnosis, monitoring, planning, prognosis, and remedies.

Considerations for Building Expert Systems

- Can the problem be solved effectively by conventional programming?
  - Ill-Structured Problems / Rigid Control

- Is the domain well bound?
  - Headache: Neurochemistry, biochemistry, chemistry, molecular biology, physics, yoga, exercise, stress management, psychiatry, …

- Is there a need and a desire for an expert system?
  - The Traffic Light Example

Considerations for Building Expert Systems

- Is there at least one human expert who is willing to cooperate?
  - Their faults may be revealed.
  - Their secrets are revealed.
  - They have different ideas.
- Can the expert explain the knowledge to the knowledge engineer can understand it.
  - How do you move your finger?
  - Medicine
- Is the problem-solving knowledge mainly heuristic and uncertain?
  - If not, why expert system?

Expert Systems Languages, Shells, and Tools

- Conventional computer programs generally solve problems having algorithmic solutions.
- Tight interweaving of data and knowledge results in rigid control flow control.
- More advance languages limit the usage, but are easier for the limited area.

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.

Elements of an Expert System

- User interface – mechanism by which user and system communicate.
- Exploration facility – explains reasoning of expert system to user.
- Working memory – global database of facts used by rules.
- Inference engine – makes inferences deciding which rules are satisfied and prioritizing.

Elements Continued

- Agenda – a prioritized list of rules created by the inference engine, whose patterns are satisfied by facts or objects in working memory.
- Knowledge acquisition facility – automatic way for the user to enter knowledge in the system bypassing the explicit coding by knowledge engineer.
- Knowledge Base!

Production Rules

- Knowledge base is also called production memory.
- Production rules can be expressed in IF-THEN pseudocode format.
- In rule-based systems, the inference engine determines which rule antecedents are satisfied by the facts.

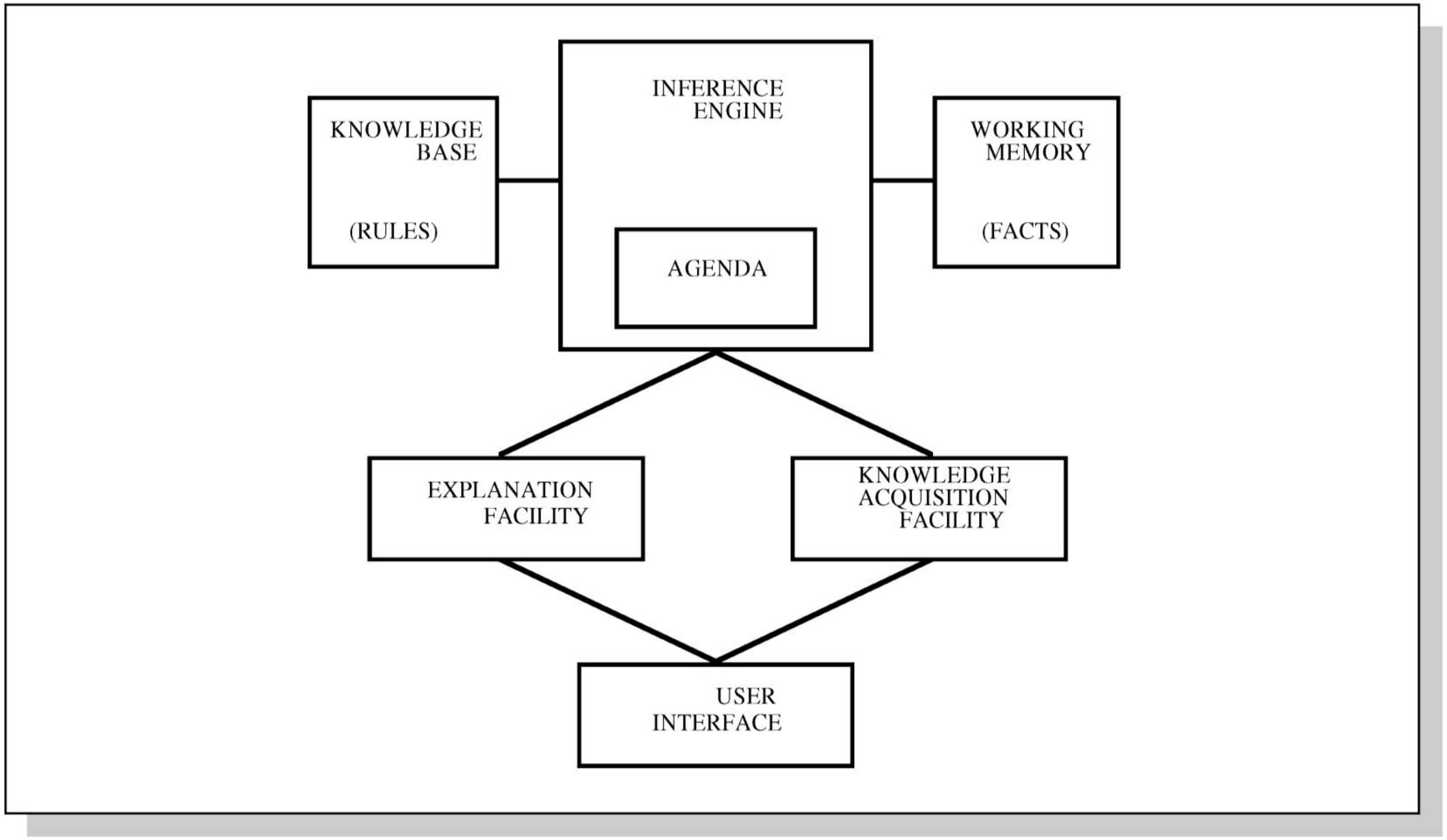
An Example from MYCIN

- IF
  - The site of the culture is blood and
  - The identity of the organism is not known with certainty, and
  - The stain of the organism is gramnegm and
  - The morphology of the organism is rod, and
  - The patient is seriously burned.
- THEN
  - There is a weakly suggestive evidence (.4) that the identity of the organism is pesudomonas.

An Example from XCON/R1

- 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 each supports one drive.

Structure of a Rule-Based Expert System



General Methods of Inferencing

- Forward chaining – reasoning from facts to the conclusions resulting from those facts – best for prognosis, monitoring, and control.
  - primarily data-driven
- Backward chaining – reasoning in reverse from a hypothesis, a potential conclusion to be proved to the facts that support the hypothesis – best for diagnosis problems.
  - primarily goal driven

Main Inference Engine Cycle

- While Not DONE
  - If there are active rules, Conflict Resolution. Else DONE.
  - Act
  - Match
  - Check for Halt
- End of While
- Accept a new user command.

Programming Paradigms

- Procedural (sequential)
  - Functional/Imperative
- None Procedural

Procedural Paradigms

- Algorithm – method of solving a problem in a finite number of steps.
- Procedural programs are also called sequential programs.
- The programmer specifies exactly how a problem solution must be coded.

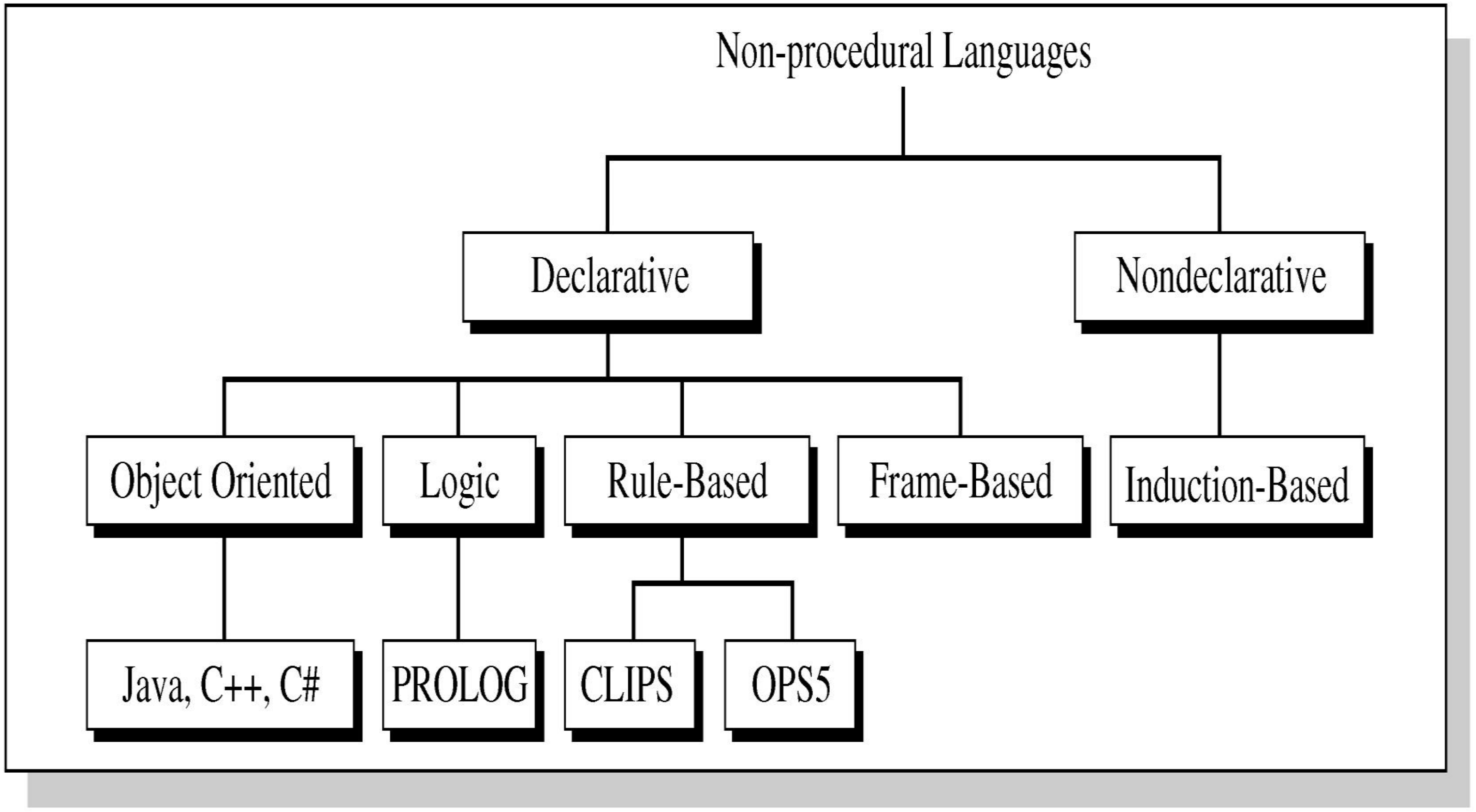
Imperative Programming

- Focuses on the concept of modifiable store – variables and assignments.
- During execution, program makes transition from the initial state to the final state by passing through series of intermediate states.
- Provide for top-down-design.
- Not efficient for directly implementing expert systems.

Nonprocedural Paradigms

- Do not depend on the programmer giving exact details how the program is to be solved.
- Declarative programming – goal is separated from the method to achieve it.
- Object-oriented programming – partly imperative and partly declarative – uses objects and methods that act on those objects.
- Inheritance – (OOP) subclasses derived from parent classes.

Nonprocedural Languages


Non-procedural Languages
Declarative
Nondeclarative
Object Oriented
Logic
Rule-Based
Frame-Based
Induction-Based
Java, C++, C#
PROLOG
CLIPS
OPS5