SLIIT UNI
THE KNOWLEDGE UNIVERSITY

Sri Lanka Institute of Information Technology

Faculty of Computing

IT2011 - Artificial Intelligence and Machine Learning

Dr.Lakmini Abeywardhana

Year 02 and Semester 01

Lecture 3

Expert Systems and Ontologies

What is an Expert System?

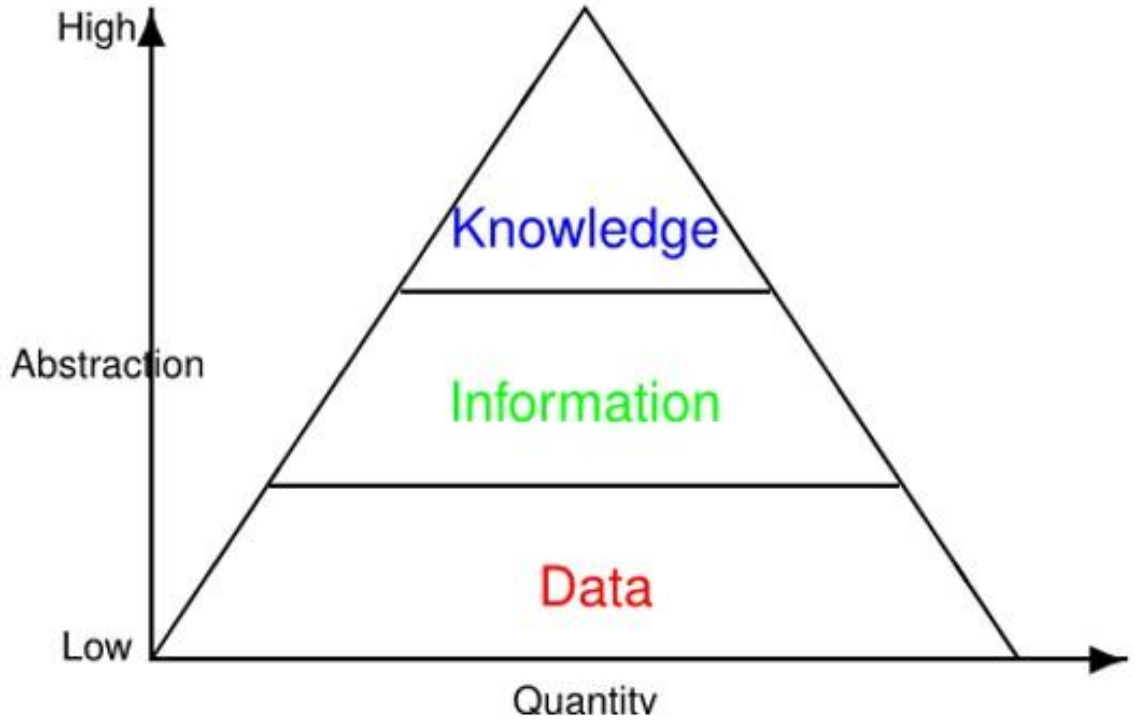
- An Expert System is a type of **computer program** designed to **simulate the decision-making ability** of a human expert in a specific **domain** (like medicine, engineering, or finance). It doesn't think like a human, but it follows **if-then rules** to reach conclusions based on **expert knowledge stored** in its system.

Think of a time when you typed your symptoms into a website like **WebMD** or **Mayo Clinic's symptom checker**. You select symptoms like fever, cough, and fatigue, and the system suggests possible illnesses like flu, COVID-19, or pneumonia

That's an example of an **expert system** in action!

Data vs. Information vs. Knowledge

Data: Raw facts (e.g., "98.6").
Information: Organized data (e.g., "Temperature = 98.6°F").
Knowledge: Meaningful insights (e.g., "98.6°F is a normal body temperature").

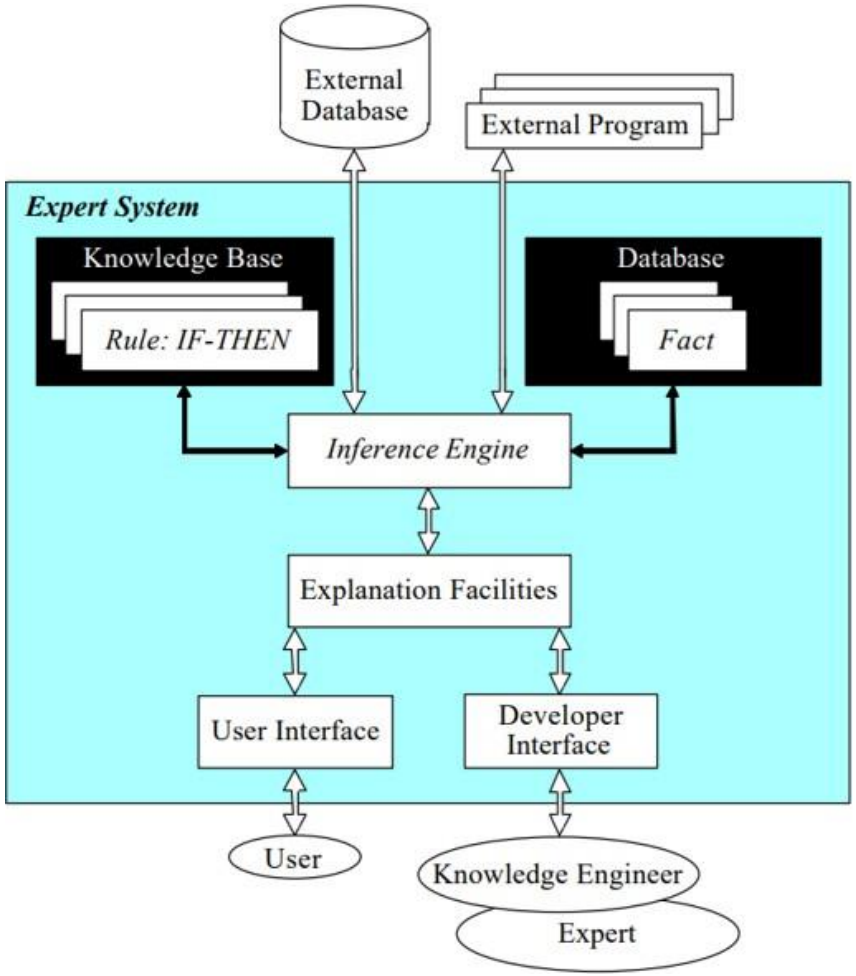


Conventional computer system vs. Expert system

Conventional computer system	Expert System
Data is used to control	Knowledge is used to control
Data and control are stored together	Knowledge is stored in a encoded form called representation
Cannot explain how the decision or conclusion is drawn	Expert system can explain how a decision or conclusion is drawn
Meta data maybe present	Meta data knowledge is always present

Key Components of Expert Systems

- **Knowledge Base**: The brain – stores facts and rules.
- **Inference Engine**: The thinker – applies rules to facts.
- **User Interface**: Allows humans to interact with the system.
- **Explanation Facility**: Explains the reasoning behind decisions.
- **Developer Interface**: Helps experts update or improve the system.



Knowledge Base

- **What it is**: A repository of **facts** and **rules** collected from domain experts.
- **Think of it as**: The brain's memory — where the expert's knowledge is stored.
- **Types of knowledge**:
  - **Declarative** (facts): "Fever is a body temp > 100.4°F"
  - **Procedural** (rules): "IF fever AND sore throat THEN possible flu"

Real-World Example:

- In a **medical expert system**, the knowledge base includes disease symptoms, treatment guidelines, and diagnostic rules.

Inference Engine

- **What it is**: The logical processing unit that applies reasoning to the knowledge base to derive conclusions.
- **Think of it as**: The brain's reasoning mechanism — it solves problems using "if-then" logic.
- **Supports:**
  - **Forward Chaining**: From symptoms → disease
  - **Backward Chaining**: From suspected disease → check for symptoms

Real-World Example:

- In a medical system, if a patient enters symptoms, the inference engine finds matching rules and concludes the likely illness.

Forward Chaining (Data-Driven Reasoning)

Concept:

- Start with **known facts** (input data).
- Apply rules to infer **new facts** until a goal is reached.

Real-World Example: Diagnosing Flu Known facts (what the user enters):

- The patient has a fever
- The patient has a headache
- The patient has a runny nose

Rule in Knowledge Base:

- IF **fever** AND **headache** AND **runny nose**, THEN **possible flu**

How Forward Chaining Works:

- The system **matches these facts** with rules.
- When it finds a rule whose conditions are all true, it **fires** the rule.
- The **conclusion** (flu diagnosis) is added to the known facts.

Conclusion Reached: "The patient might have flu."

Backward Chaining (Goal-Driven Reasoning)

Concept:

- Start with a goal (e.g., ”Does the patient have flu?”)
- Look for rules that support this goal.
- Ask questions to verify the conditions of those rules.

Real-World Example:Confirming Flu Diagnosis

Goal: “Is the patient likely to have flu?”

System searches for a rule like:

IF **fever** AND **headache** AND **runny nose**, THEN **flu**

Now the system asks:

1. “Does the patient have a fever?” ✅
2. “Does the patient have a headache?” ✅
3. “Does the patient have a runny nose?” ✅

If all conditions are confirmed, it **proves the goal** is true.

✅ **Conclusion**: “Yes, flu is likely.”

Forward chaining vs Backward chaining

Feature	Forward Chaining	Backward Chaining
Starts From	Known facts (data)	Goal (hypothesis)
Moves Toward	Conclusions (goal)	Supporting facts (data)
Common In	Monitoring, control systems	Diagnostic and expert advice systems
Example	From symptoms → find disease	From disease → check if symptoms fit

User Interface

- **What it is**: The **communication bridge** between the user and the system.
- **Think of it as**: A chatbot or form where users input information and receive decisions or suggestions.
- **Can be**: Text-based, menu-driven, or graphical.

Real-World Example:

- In WebMD's symptom checker, users click checkboxes for symptoms, and the system provides likely diagnoses.

Knowledge Acquisition Module (a.k.a. Developer Interface)

- **What it is**: The tool that helps **experts and engineers** input and update knowledge.
- **Think of it as**: The backstage control panel — where new rules or data are added or removed.

Real-World Example:

- Doctors collaborate with developers to add new disease rules into a medical expert system through a developer interface.

Explanation Facility

- **What it is**: The part of the system that **explains how and why** a certain conclusion was reached.
- **Think of it as**: A teacher that justifies answers — "I diagnosed flu because fever + cough were observed."
- Builds **user trust** and ensures transparency.

Real-World Example:

- In legal expert systems, the explanation facility shows which laws and facts supported the given legal recommendation.

Working Memory / Fact Base

- **What it is**: Temporary memory that stores **current facts** during a reasoning session.
- Used by the inference engine to **track progress** of reasoning.

Real-World Example:

- When diagnosing a patient, the working memory stores entered symptoms like "fever," "headache," "nausea."

External Interfaces (optional)

- **What it is**:Links to **external databases or software programs.**
- Used when the expert system needs to **fetch live data** or send results to other systems.

Real-World Example:

- In a financial expert system, it might access **real-time stock prices** or **customer account data.**

How it Works (Step-by-Step)

Component	Role	Example (WebMD-style system)
Knowledge Base	Stores facts and rules	Rule: "IF fever AND cough THEN flu"
Inference Engine	Applies logic to rules	Checks symptoms and finds matching diagnosis
User Interface	Lets you enter symptoms	You tick boxes or enter symptoms
Explanation Facility *(optional)*	Explains decision	"This is likely flu because you have fever, cough, and muscle ache"

Inference Engine

Reasoning in expert systems refers to the process of **drawing conclusions** from known facts and rules. The system uses logical steps to figure out a solution to a problem — similar to how a human expert would ”reason” through a case.

- Expert systems typically use two types of reasoning:

Expert Systems in the Real World

MYCIN (Healthcare)

- Diagnosed blood infections.
- Example of rule-based treatment suggestions.

DENDRAL (Chemistry)

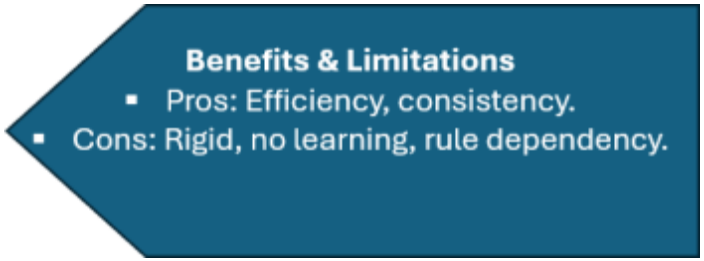
- Predicted molecular structures using spectra data

XCON (Computing)

- Automated configuration of computer systems.
- Saved $25 million for DEC.

HEARSAY (Military/Voice)

- Voice recognition → submarine sonar detection.



Limitations of Expert Systems:

- Built entirely using **manual rules** created by human experts.
- Hard to keep up-to-date as **domains evolve**.
- Cannot **learn from data** — no improvement over time.
- Breaks easily with **incomplete or conflicting rules**.
- **Not scalable** to complex problems like vision, language, or natural speech.

Ontologies

What is an Ontology?

- An **ontology** is a **structured way to represent knowledge**, including **concepts (things), properties (features), and relationships** between those concepts — in a way that computers can understand and reason with.

> "A formal, explicit specification of a shared conceptualization."
> — *Gruber, 1993*

This means:

- **Formal**: It follows a specific structure and rules.
- **Explicit**: All terms and relationships are clearly defined.
- **Shared**: Agreed upon by a group of users or systems.
- **Conceptualization**: It reflects how we view a part of the world.

Real-World Analogy 1: Library Catalog (Dewey Decimal System)

Imagine you walk into a library:

- Books are **classified into categories** (Science, History, Fiction).
- Each book has a **code** and **belongs to a subject**.
- Some books are **related** (e.g., "History of Science" belongs to both History and Science).

This structure is **ontology in action** — organizing knowledge so it's easy to search, browse, and retrieve based on **defined relationships**.

Real-World Analogy 2: Family Tree

In a family tree:

- **Classes**: Parent, Child, Sibling, Grandparent.
- **Instances**: "John" is a Person. "Anna" is John's sister.
- **Relationships**: "hasSibling," "isParentOf," "isChildOf."

This is a simple **ontology of family relationships**:

- You can **infer** that "John's mother is Anna's mother" without it being explicitly stated.
- The structure helps you reason about **who is related to whom.**

Why Ontologies Are Useful in AI

- They give **meaning to data** — machines don't just see words; they see relationships and structure.
- They help in **reasoning**: infer new facts from existing ones.
- They make **data sharable and understandable** across systems (semantic interoperability).

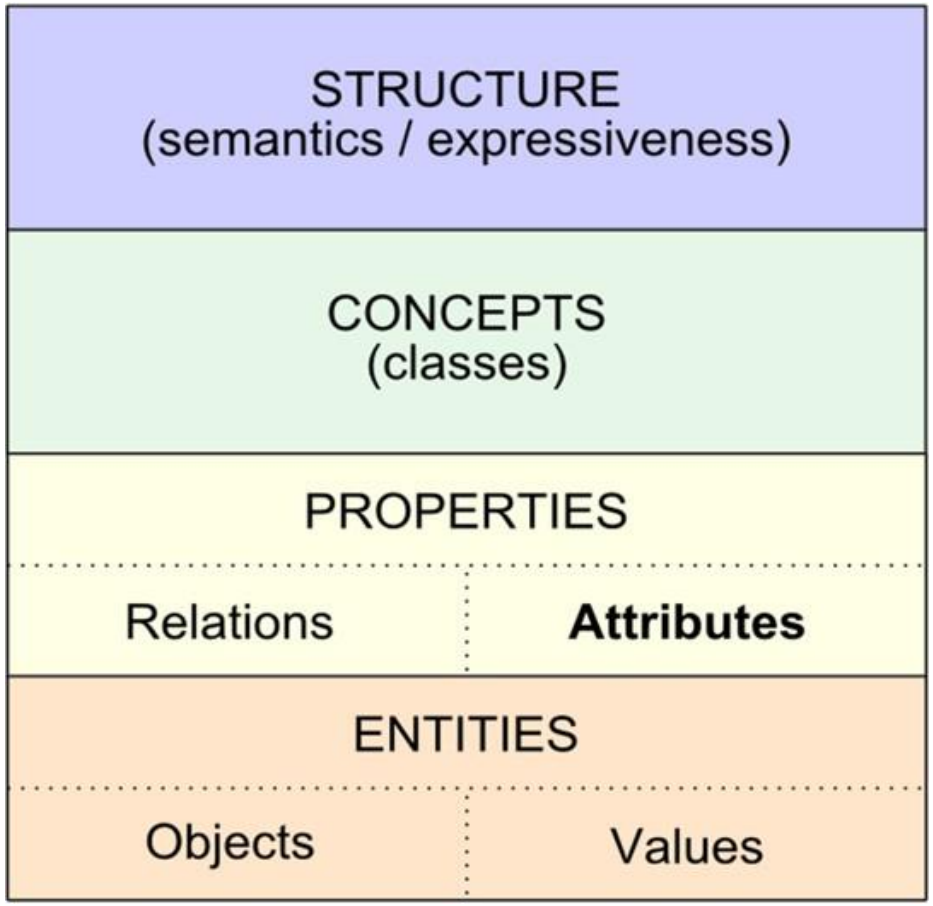
Real-world applications

Domain	Use of Ontologies
Healthcare	SNOMED CT: standardizes medical terms across systems
E-commerce	Amazon's product categories and filters
Search Engines	Google Knowledge Graph for smart answers
Smart Farming	AgroVoc: crop, soil, and weather data links

Components of an Ontology

Element	Description	Example (University Domain)
Class	A group/category of things	Student, Course, Instructor
Instance	Specific member of a class	John (a Student), CS101 (a Course)
Property	Feature or attribute of a class	hasName, hasCredits, hasAge
Relationship	Link between classes or instances	enrolledIn (Student → Course)
Axiom	Logical rule that must always be true	"A course must have at least 1 instructor"
Restriction	Constraint on properties or relationships	"Students must enroll in ≤ 5 courses"

The Ontology Stack



Source: https://www.mkbergman.com/1842/conceptual-and-practical-distinctions-in-the-attributes-ontology/

Challenges of Ontologies:

Challenges of Ontologies:

- Require **domain experts** to build and maintain.
- Best for **well-structured knowledge** (e.g., anatomy, curriculum).
- Not suitable for **unstructured data** like text, images, audio.
- Limited in capturing **uncertainty** or **probabilistic relationships**.

From Expert Systems Ontologies to Modern AI Techniques

Why Move Beyond Expert Systems and Ontologies?

- Rule-based systems are **limited in flexibility and scalability**
- Ontologies require **manual creation** and **don't adapt to changing data**
- Both **lack learning capability** — they don't improve with experience
- They **struggle with unstructured, ambiguous, or noisy real-world data**(e.g., images, social media text, sensor data)

Modern AI Techniques That Have Taken Over

Machine Learning (ML)

Learns patterns from data automatically

Scales to new problems without explicit reprogramming

Powers applications like email filtering, fraud detection

Deep Learning (DL)

Uses neural networks to handle complex unstructured data

Excels in image recognition, speech processing, NLP

Examples: ChatGPT, facial recognition, medical imaging

Reinforcement Learning (RL)

Learns optimal decisions through **trial and error**

Used in robotics, autonomous driving, game AI (e.g., AlphaGo)

Hybrid AI Systems

Combine **symbolic reasoning** (from ontologies) with **data-driven learning** (from ML)

Enhance **explainability + adaptability**

Used in explainable AI and smart assistants

Summary