أكاديمية طويق
Tuwaiq Academy

Introduction to Artificial Intelligence and Applications

Semester One | Course One

Unit 2 : AI Concepts, Terminology and Application Domains

Recap From Unit 1

In the early days of **AI**, the top-down approach to creating intelligent systems. The idea was to extract the knowledge from people into some machine-readable form, and then use it to automatically solve problems. This approach was based on two big ideas:

______ - ____________________________ ______

Knowledge Representation

Reasoning

- **It is important to differentiate knowledge from information or data.**

✅ **Knowledge** is something which is contained in our head and represents our understanding of the world.

✅ It is obtained by an active **learning** process, which integrates pieces of information that we receive into our active model of the world.

Recap From Unit 1

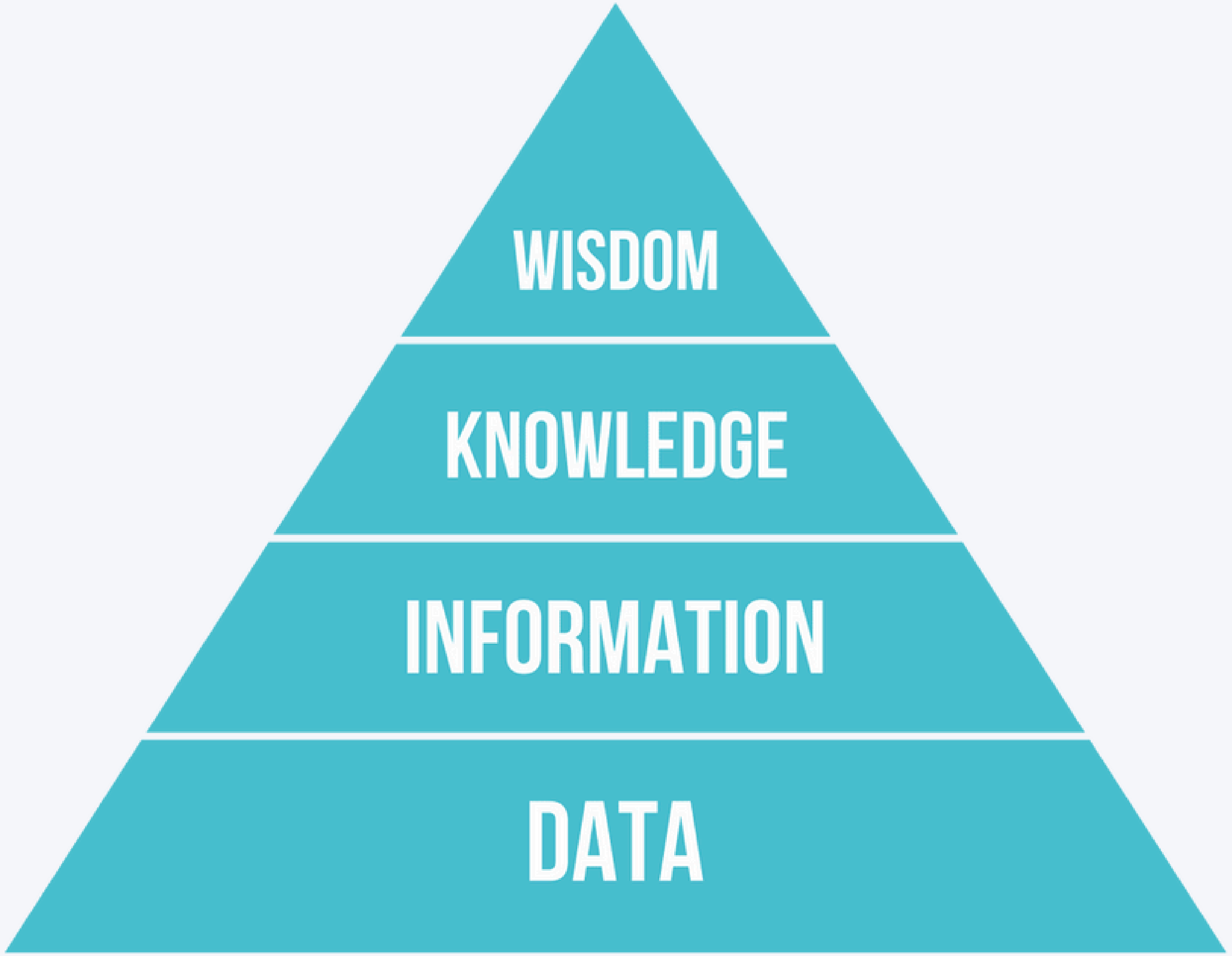
- **It is important to differentiate knowledge from information or data.**

- **Data** is something represented in physical media, such as written text or spoken words. Data exists independently of human beings and can be passed between people.

- **Information** is how we interpret data in our head. For example, when we hear the word computer, we have some understanding of what it is.

- **Knowledge** is information being integrated into our world model. For example, once we learn what a computer is, we start having some ideas about how it works, how much it costs, and what it can be used for. This network of interrelated concepts forms our knowledge.

- **Wisdom** is yet one more level of our understanding of the world, and it represents meta-knowledge, eg. some notion on how and when the knowledge should be used.

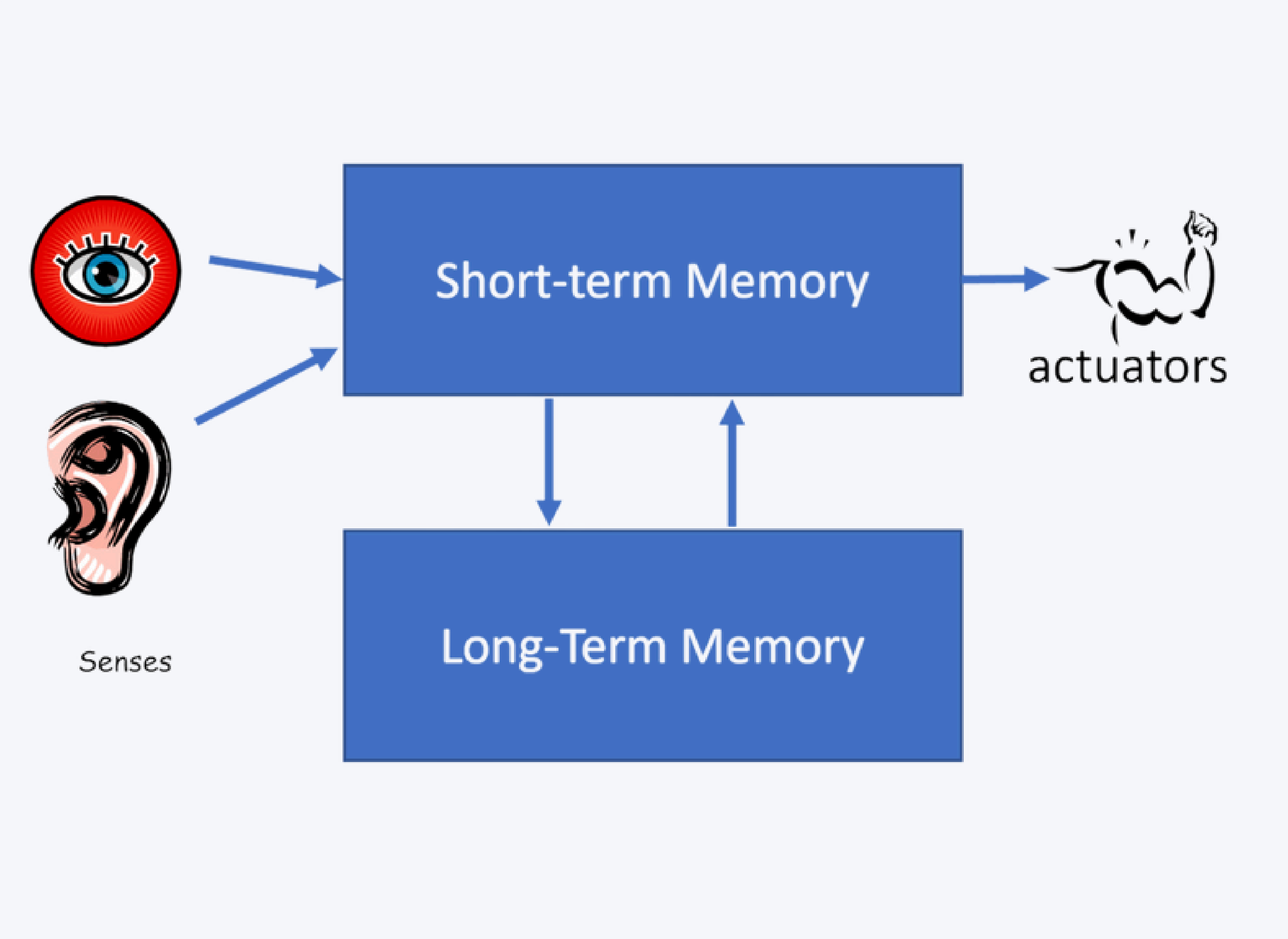


Expert Systems

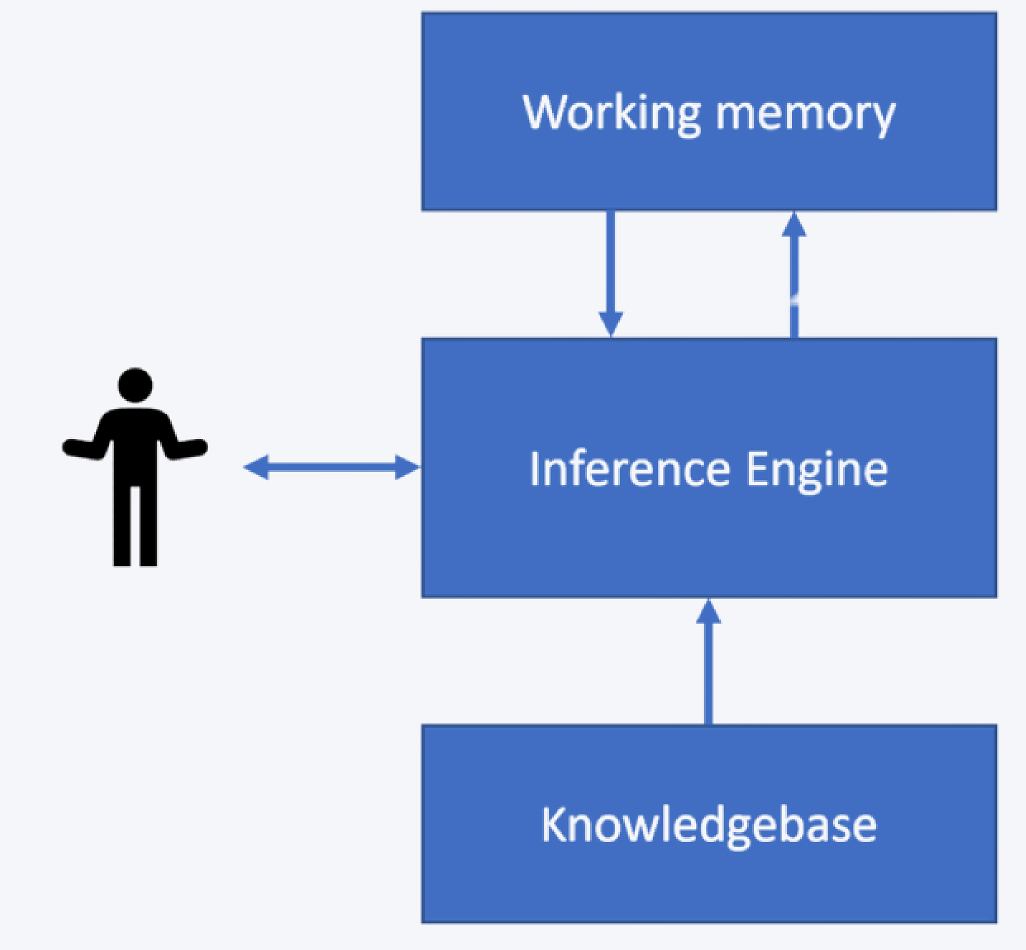
Expert Systems

- One of the early successes of **AI** were so-called **expert systems**
- Computer systems that were designed to act as an **expert** in some **limited problem domain.**

- They were based on a **knowledge base** extracted from one or more **human** experts, and they contained an **inference engine** that performed some **reasoning** on top of it.



Simplified structure of a human neural system



Architecture of a knowledge-based system

https://github.com/microsoft/AI-For-Beginners/tree/main

Expert Systems

Expert systems are built like the human reasoning system, which contains **short-term memory** and **long-term memory.** Similarly, in knowledge-based systems we distinguish the following components:

- **Problem Memory:**
  - Stores the current information about the problem being solved.
  - **Example**: A patient's temperature, blood pressure, or whether they have inflammation.
  - Known as **static knowledge** because it represents what we know right now—a snapshot of the problem.
- **Knowledge Base:**
  - Holds long-term, general knowledge about the problem domain.
  - This knowledge is gathered from experts and remains the same for all consultations.
  - Called **dynamic knowledge** because it helps transition from one state of the problem to another.
- **Inference Engine:**
  - Manages the entire problem-solving process.
  - **Tasks**:
    - Searches through the current problem state.
    - Asks questions to gather missing information.
    - Identifies and applies the correct rules to move forward.

Expert Systems

As an example, let's consider the following expert system of determining an animal based on its physical characteristics:

- This diagram is called an **AND-OR tree,** and it is a graphical representation of a set of production rules

.

- Drawing a tree is useful at the beginning of extracting knowledge from the expert.

- To represent the knowledge inside the computer it is more convenient to use rules.

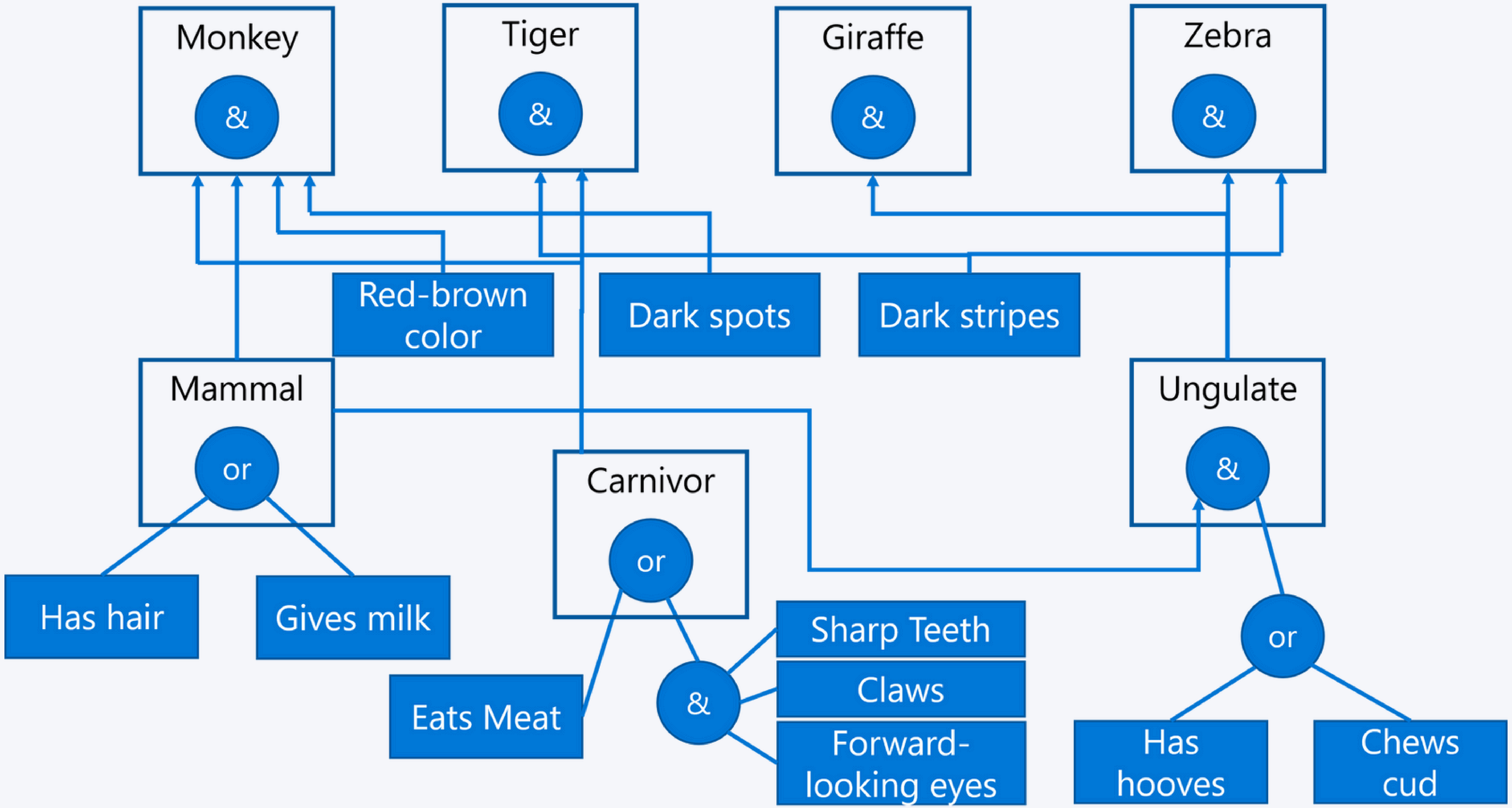


Image by Dmitry Soshnikov

Expert Systems

You can notice that each condition on the of the **rule and the action are essentially object-attribute-value (OAV) triplets**.

Working memory contains the set of **OAV triplets** that correspond to the problem currently being solved. A **rules engine** looks for rules for which a condition is satisfied and applies them, adding another triplet to the working memory.

```
IF the animal eats meat
OR (animal has sharp teeth
    AND animal has claws
    AND animal has forward-looking
eyes ) THEN the animal is a
carnivore
```

Expert Systems

Conditions:

1. Animal - Eats - Meat
2. (If the animal eats meat)
3. Animal - Has Teeth - Sharp
4. (If the animal has sharp teeth)
5. Animal - Has Claws - True
6. (If the animal has claws)
7. Animal - Has Eyes - Forward-Looking
8. (If the animal has forward-looking eyes)

Action:

1. Animal - Is - Carnivore
2. (Then the animal is a carnivore)

```
IF the animal eats meat
OR (animal has sharp teeth
    AND animal has claws
    AND animal has forward-looking
eyes ) THEN the animal is a
carnivore
```

These triplets form the basis for reasoning in the working memory, where conditions are evaluated, and new knowledge (e.g., "Animal - Is - Carnivore") is added when rules are satisfied.

Forward vs. Backward Inference

Forward Inference (Data-Driven Reasoning)

- Starts with what we already know (initial data in working memory).
- Works step by step to add new knowledge until we reach the answer.

Steps:

1. Check if the goal is already known: If yes, stop and return the result.
2. Find rules to apply: Look for rules whose conditions match the current data (this creates a "conflict set").
3. Choose a rule to apply (Conflict Resolution):
    - Pick the first matching rule.
    - Choose randomly.
    - Pick the rule with the most specific conditions.
4. Apply the rule: Use the rule to add new knowledge to the working memory.
5. Repeat: Go back to step 1 until the goal is reached.

Example: Diagnosing a patient based on existing lab results.

Backward Inference (Goal-Driven Reasoning)

- Starts with a specific goal (e.g., "What is the diagnosis?").
- Asks targeted questions to find the information needed to reach the goal.

Steps:

1. Identify rules for the goal: Find rules that have the desired answer on the Right-Hand Side (RHS).
2. If no rules exist, ask for missing data: If no rules cover the goal, ask the user for the information.
3. Test one rule as a hypothesis: Try to prove the rule by checking its conditions (Left-Hand Side, LHS).
4. Repeat for sub-goals: If the rule's conditions depend on other unknowns, repeat the process for those sub-goals.
5. If a rule fails: Try another rule from step 3.

Example: In medical diagnosis, asking for specific tests or symptoms only when needed to narrow down the cause.

Implementing Expert Systems

Expert systems can be implemented using different tools:

- Programming them directly in some **high level programming language**. This is not the best idea, because the main advantage of a knowledge-based system is that knowledge is separated from inference, and potentially a problem domain expert should be able to write rules without understanding the details of the inference process

- Using **expert systems** shell, i.e. a system specifically designed to be populated by knowledge using some knowledge representation language.

Let us implement some code

Ontologies and the Semantic Web

Background:

- By the end of the 20th century, there was a push to improve how we find information on the Internet.

The idea:

- Use knowledge representation to annotate web resources so users can make very specific queries.
- This idea became known as the **Semantic Web.**

Smart Knowledge Representation

- A way to describe knowledge using **Description Logics (DL)**, which provide a formal, structured, and logical framework for understanding data.
- Helps in organizing information into hierarchies and assigning properties to objects (like categories and subcategories).
- Beyond organizing, DL allows systems to reason or infer new facts automatically.
  - **Ontology about transportation:**
    - "All cars are vehicles."
    - "Toyota Camry is a car."
    - The system infers: "Toyota Camry is a vehicle."

Ontologies and the Semantic Web

Is it usable today?

- Yes. Modern AI and search engines use similar techniques to infer relationships and provide better recommendations or search results.
    - **Google** uses ontologies to link related search terms (e.g., searching "Canine" may show results for "Dog").

Distributed Knowledge

- Knowledge is distributed across multiple systems or websites, but concepts are uniquely identified using global **URIs (Uniform Resource Identifiers).**
  A "Dog" ontology in one database and a "Veterinary Care" ontology in another **can link information seamlessly**.
- Makes data interoperable **(easily shared and understood across systems)**. ______________________________ .
  Enables linking knowledge from diverse domains—biology, transportation, healthcare, etc.—on the global web.
-
- _____________ ________ _

Ontologies and the Semantic Web

Special XML-Based Languages

- **Tools to describe, structure, and share knowledge in a machine-readable format:**
    - RDF (Resource Description Framework): A standard way to represent information about resources.
        - Example: Describe "Paris" with RDF:
            - "Paris is a city. It is in France."
    - RDFS (RDF Schema): Adds hierarchies and relationships.
        - Example: Define "City" as a subclass of "Location."
    - OWL (Ontology Web Language): More expressive, allowing for rules and reasoning.
        - Example: Define "City" and infer, "If Paris is a city, it is also a location."

- **Is it usable today?**
    - Yes. These languages are actively used in:
        - **Knowledge Graphs:** Google, Microsoft, and Amazon build these to link data intelligently.
        - **AI and Research:** Fields like healthcare use OWL to create ontologies for diseases and treatments.

Core Concept: Ontology

What is an Ontology?

- A formal specification of a domain (a specific topic or area).

- **Organizes knowledge as:**
  - A hierarchy of objects (simple version).
  - Rules and relationships for inference (advanced version).

- **Purpose of Ontologies in the Semantic Web:**
  - Help machines understand the meaning of data.
  - Example: In an ontology about animals:
    - Define "Dog" as a type of "Mammal."
    - Add rules: "Mammals have hair" → Automatically know "Dogs have hair."

Why is This Important?

- Enables the web to move beyond simple keyword searches to understanding user queries.
- Makes the web smarter, linking knowledge across sites.
- Supports applications like intelligent search engines, AI assistants, and data integration tools.

Core Concept: Ontology

Real-World Applications of Ontologies in Food and Nutrition:

- **Health Apps:**
  - An ontology can link food items to their nutritional content.
  - Query: "Which foods are rich in Vitamin C?"
    - The system uses the ontology to find and suggest items like oranges, apples, and strawberries.

- **E-commerce:**
  - When you search for "Fruits" on a grocery website, it can automatically group apples, bananas, and other fruits without needing manual categorization.

Let’s Code !!

Practise Quiz !!

~~https://forms.gle/wSSmQejDyq5G9dae7~~