

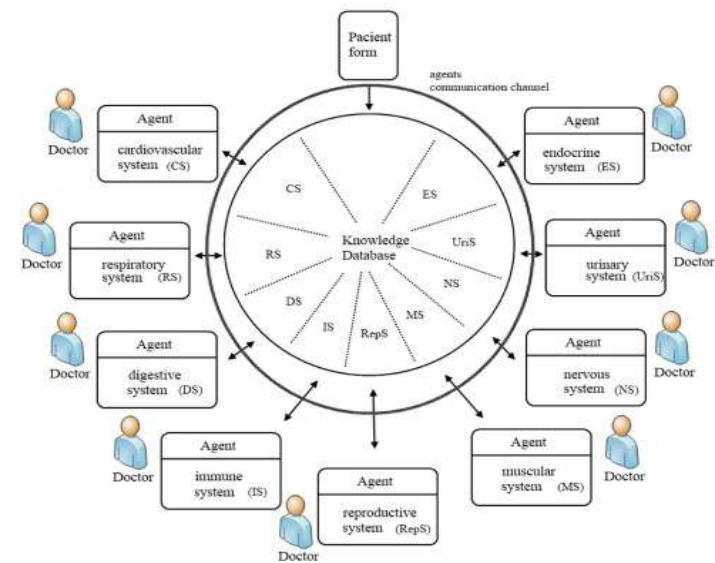
CN7050 – Intelligent Systems

Week 2: Expert Systems to DSS & Transformation to Agentic AI

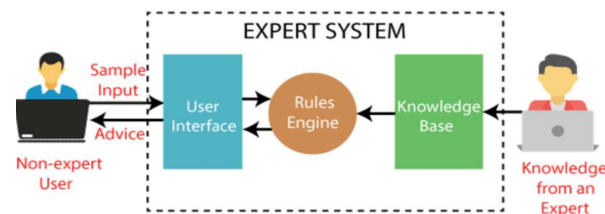
Dr Azhar Mahmood

Office: EB.1.85

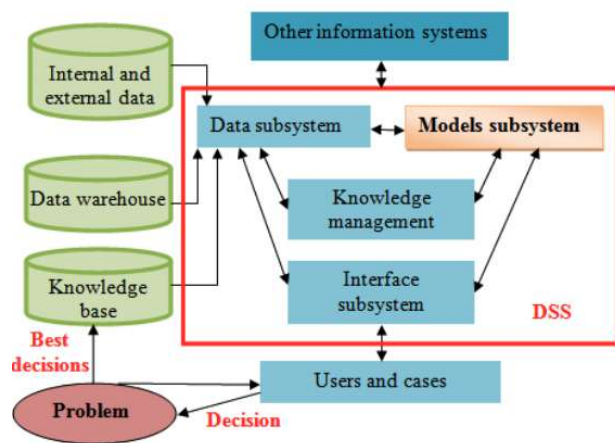
Email: a.mahmood3@uel.ac.uk



Expert Systems vs Decision Support System



Expert systems were early AI programs to capture human expertise for decision support, leading to the more general Decision Support Systems (**DSS**).



DSS combine data, models, and expert knowledge for complex problem-solving.

This evolution concludes in Agentic AI, which further advances DSS by enabling **autonomous agents** to set **goals**, make **independent decisions**, access tools, and **act independently** to achieve complex, **multi-step tasks** without constant human guidance.

Who is an Expert?

- What characterizes an ‘Expert’
 - Specialized knowledge in a certain area
 - Experience in the given area
 - Explanation of decisions
 - A skill set that enables the expert to translate the specialized knowledge gained through experience into solutions.
 - e.g. Skin specialist, heart specialist, car mechanic, architect, software designer.
- **Expert System:** *"An AI based computer program designed to model the problem solving ability of a human expert"*
- Aspects of the human expert that we wish to model
 - **Knowledge**
 - **Reasoning**

Why Expert System based Agents

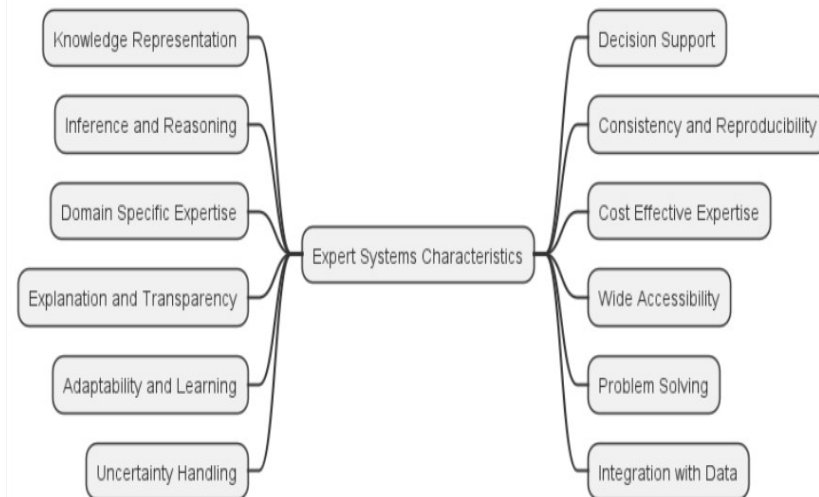
Issues	Human Expert	Expert System
Availability	Limited	Always
Geographic location	Locally available	Anywhere
Safety considerations	Irreplaceable	Can be replaced
Durability	Depends on individual	Non-perishable/ Permanent
Performance	Variable	High
Speed	Variable	High (Distributed Processing)
Cost	High	High to Low
Learning Ability	Variable/High	Low-> High (DL, Transformers)
Explanation	Variable	Exact

Example: Medical ES modelling a doctor

The ES outperforms the ‘average’ doctor and is available in regions where people may not have access to any medical care at all otherwise.

Why Expert System Agents

Why Expert System



Practical Issues

1. Is some human expert willing to cooperate?
2. Designed to solve any one specific problem.
3. Is experts knowledge especially uncertain and heuristic? if so, ES may be useful.

Roles of an Expert System

An expert system may take two main roles, relative to the human expert.

It may replace an expert or assist expert.

- **Replacement of Expert**

- Not very practical in some situations, but feasible in others
- Consider drastic situations, e.g. where safety or location is an issue, e.g. a mission to Mars.

- **Assisting an Expert**

- Most commonly found application i.e. **ChatGPT**
- Aiding expert in routine task to increase productivity
- Aiding in managing complex situations (may Learn the experience of other)
- Solves by recalling problems
- A lending advisor, often a mortgage broker or financial advisor

Expert System Structure

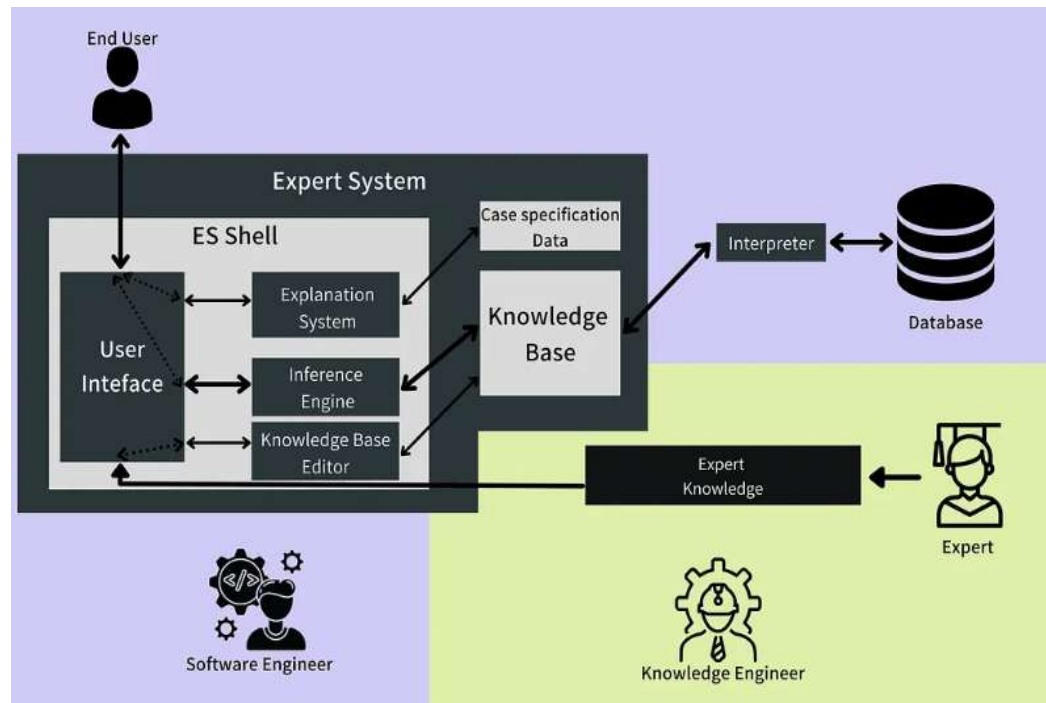
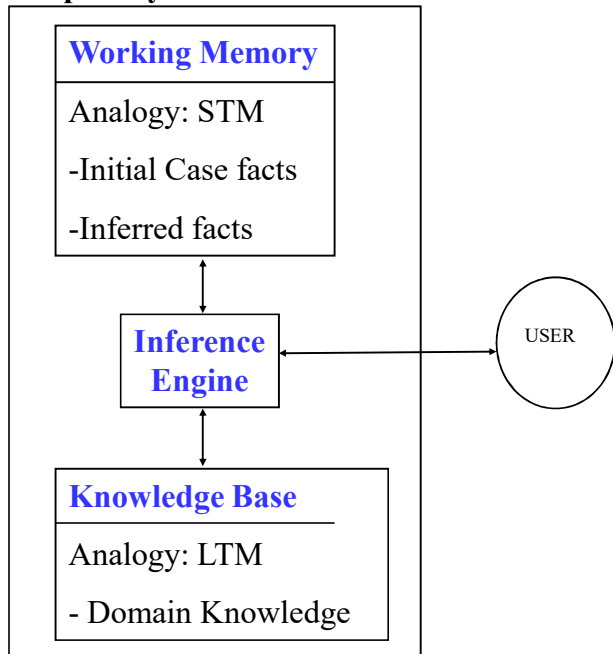
- Lets look at how an expert (say a doctor) solves a problem:
 - Focused area of expertise
 - Specialized Knowledge (Long-term Memory, **LTM**)
 - Case facts (Short-term Memory, **STM**)
 - Reasons with these to form new knowledge
 - Solves the given problem
- Lets define the corresponding concepts in an Expert System.

Expert System Structure

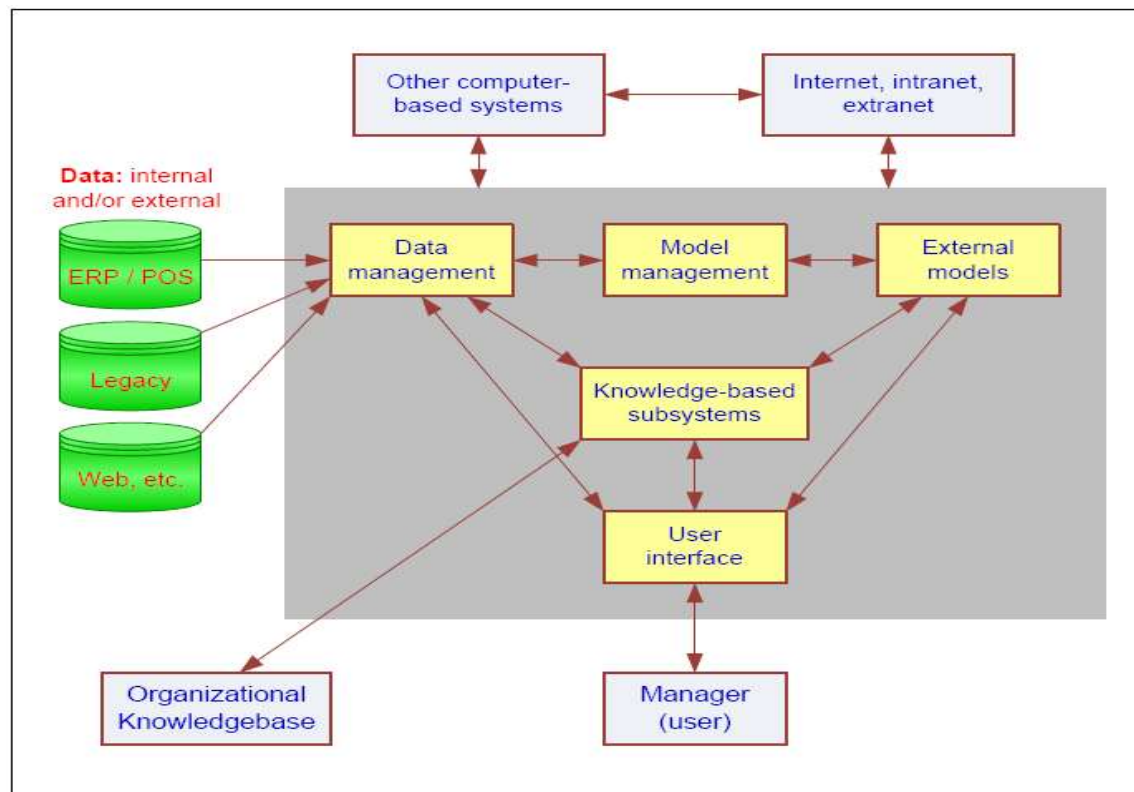
Human Expert	Expert System
Focused Area of Expertise	Domain
Specialized Knowledge (stored in LTM)	Domain Knowledge (stored in Knowledge Base)
Case Facts (stored in STM)	Case/Inferred Facts(stored in Working Memory)
Reasoning	Inference Engine
Solution	Conclusions

Expert System Structure

Expert System



DSS Components



DSS Characteristics & Capabilities

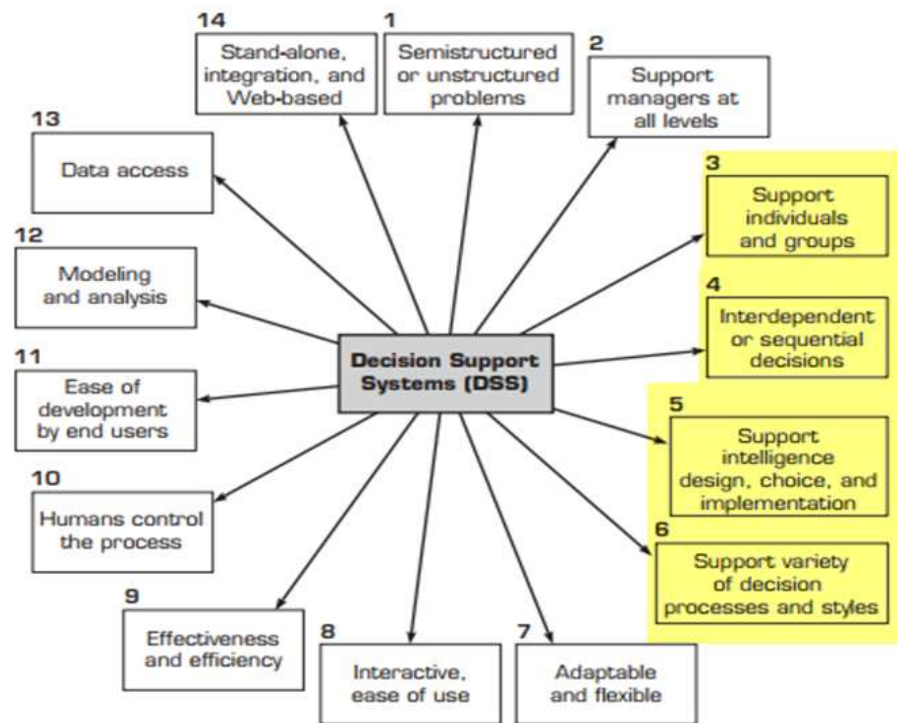
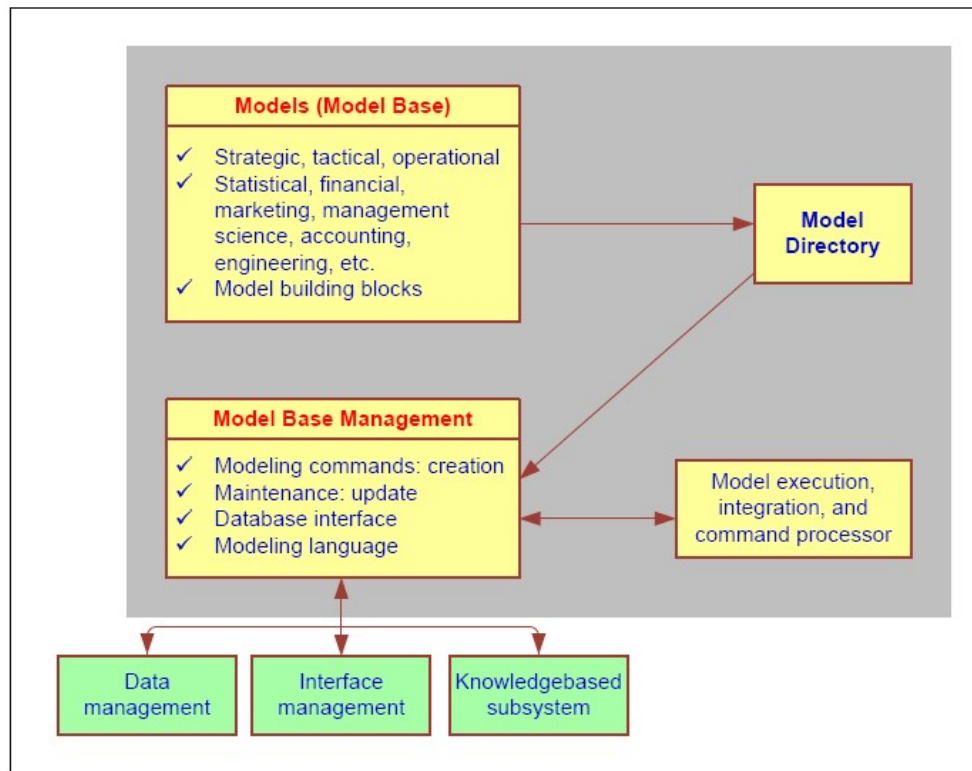


FIGURE 3.5 Key Characteristics and Capabilities of DSS

DSS Components: Model Management Subsystem

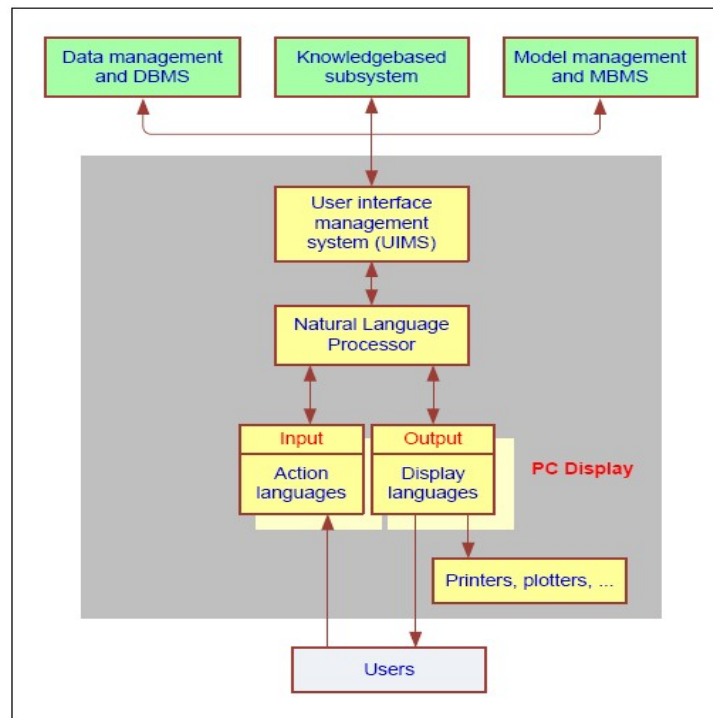


Model Types:

Strategic models
Tactical models
Operational models
Analytic models

DSS Components: User Interface (Dialog) Subsystem

- Interface
 - Application interface
 - User Interface
 - Graphical User Interface (GUI)
- DSS User Interface
 - Portal
 - Graphical icons
 - Dashboard
 - Color coding
- Interfacing with PDAs, cell phones, etc.



Knowledge Base

- Part of the expert system that contains the domain knowledge
 - Problem facts, Rules
 - Concepts
 - Relationships
- One of the roles of the expert system designer is to act as a **knowledge engineer**.
- **Knowledge acquisition bottleneck (Expert opinion is required)**
- You have to get information from the expert and encode it in the knowledge base, using one of the knowledge representation techniques such as Knowledge Representation(logic).
- One way of encoding that knowledge is in the form of **IF-THEN rules**.

Working Memory

- ‘Part of the expert system that contains the **problem facts** that are discovered during the **session**’ *Durkin*.
- A session is one consultation (in doctor example).
- During a consultation:
 - User presents some facts about the situation.
 - These are stored in the working memory.
 - Using these and the knowledge in the knowledge base, new information is inferred and also added to the working memory.

Inference Engine

- **Processor in an expert system** that matches the **facts** contained in the working memory with the **domain knowledge** contained in the knowledge base, to **draw conclusions** about the problem
- The inference engine works with the knowledge base and the working memory, and **draws on both to add new facts to the working memory.**
- If our knowledge is represented in the form of IF-THEN rules, the Inference Engine has the following mechanism
 - Match given facts in working memory to the premises of the rules in the knowledge base, if match found, **'fire'** the conclusion of the rule, i.e. add the conclusion to the working memory.

Expert System Example: Family

Knowledge Base

Rule 1:

IF father (X, Y)

AND father (X, Z)

THEN brother (Y, Z)

Rule 2:

IF father (X, Y)

THEN payTuition (X, Y)

Rule 3:

IF brother (Y, Z)

THEN like (Y, Z)

Working Memory

father (M.Tariq, Ali)

father (M.Tariq, Ahmed)

brother (Ali, Ahmed)

payTuition (M.Tariq, Ali)

payTuition(M.Tariq,Ahmed)

like (Ali, Ahmed)

New Session:

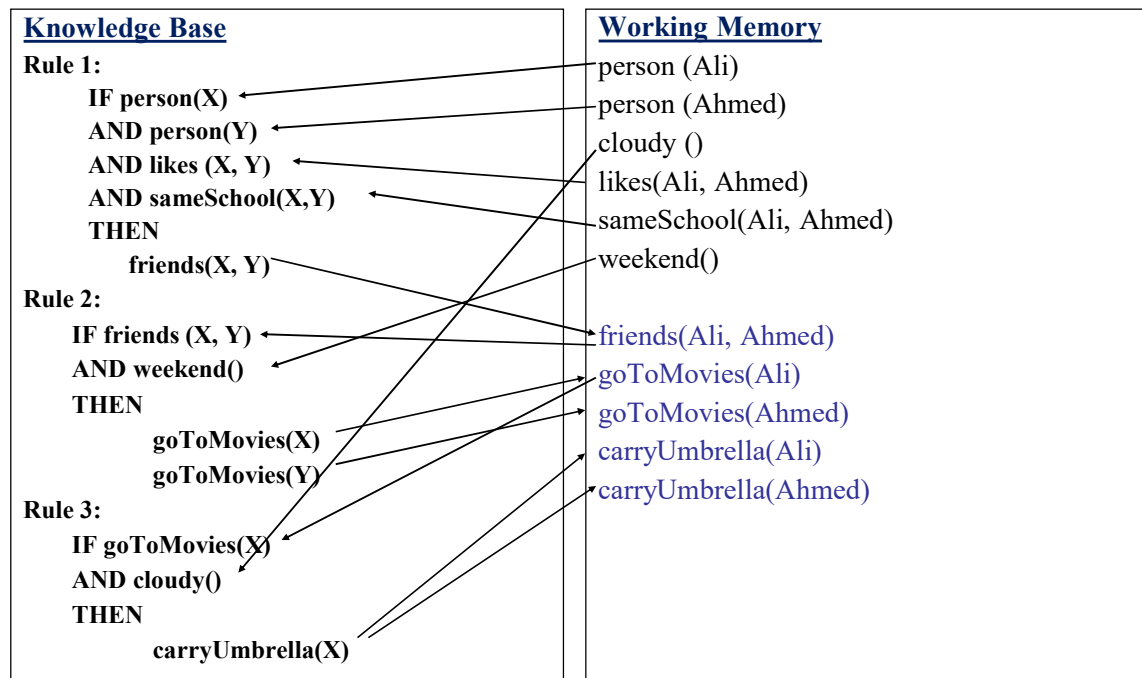
User **asserts** the following facts:

M. Tariq is Ali's father

M. Tariq is Ahmeds's father

The inference engine matches the premises of the rules in the knowledge base to the facts in the working memory, if premises match, the rule is 'fired' i.e. its conclusion is added to the working memory

Expert System Example: Raining



Explanation Facility

Module of an expert system, that allows transparency of operation, by providing an explanation of how it reached the conclusion.

In the below example, how does the expert system draw the conclusion that Ali likes Ahmed?

Knowledge Base

Rule 1:

IF father (X, Y)
AND father (X, Z)
THEN brother (Y, Z)

Rule 2:

IF father (X, Y)
THEN payTuition (X, Y)

Rule 3:

IF brother (X, Y)
THEN like (X, Y)

Working Memory

father (M.Tariq, Ali)
father (M.Tariq, Ahmed)

brother (Ali, Ahmed)
payTuition (M.Tariq, Ali)
payTuition (M.Tariq, Ahmed)
like (Ali, Ahmed)

How?

Characteristics of an Expert System

- **Separates Knowledge From Control**

- Knowledge Base, Working Memory and Inference Engine
- Comparison to traditional programs that mix a program's knowledge and control.
- *Separation allows changes to the knowledge to be independent of changes in control and vice versa. (Same like DB and Application layers)*

- **Focuses Expertise**

- The larger the domain, the more complex the expert system becomes
- E.g. Car Diagnosis Expert (more easily handled if we make separate ES components for engine problems, electricity problems, etc.)

Characteristics of an Expert System

- Reasons with symbols (Knowledge Representation)
- Reasons heuristically
- Permits inexact reasoning (approximation)
- Is limited to solvable problems
- Thrives on reasonable complexity
- **Makes Mistakes (conflicting facts and rules)**

Inference Strategies: Forward Chaining

- How does a doctor diagnose a patient?
 1. Asks for symptoms
 2. Infers diagnosis from symptoms
- **Data-driven approach**
- Forward Chaining: “Inference strategy that begins with a **set of known facts**, **derives new facts** using rules whose premises match the known facts, and continues this process until a goal state is reached or until no further rules have premises that match the known or derived facts.

Inference Strategies: Forward Chaining (Doctor Example)

- Rules

- Rule 1

IF The patient has deep cough
AND We suspect an infection
THEN The patient has Pneumonia

- Rule 2

IF The patient's temperature is above 100
THEN Patient has fever

- Rule 3

IF The patient has been sick for over a fortnight
AND The patient has a fever
THEN We suspect an infection

Inference Strategies: Forward Chaining (Doctor Example)

■ Case facts:

- Patients temperature= 103
- Patient has been sick for over a month
- Patient has violent coughing fits

■ Approach

1. Add facts to working memory (WM)
2. Take each rule in turn and check to see if any of its premises match the facts in the WM
3. When matches found for all premises of a rule, place the conclusion of the rule in WM.
4. Repeat this process till no more facts can be added.

Doctor Example: First Pass

Rule, premise	Status	Working Memory
1, 1 Deep cough	True	Temp= 103 Sick for a month Coughing fits
1, 2 Suspect infection	Unknown	Temp= 103 Sick for a month Coughing fits
2, 1 Temperature>100	True, fire rule	Temp= 103 Sick for a month Coughing fits Patient has fever

▪ Rule 1

IF The patient has deep cough
AND We suspect an infection
THEN The patient has Pneumonia

▪ Rule 2

IF The patient's temperature is above 100
THEN Patient has fever

▪ Rule 3

IF The patient has been sick for over a fortnight
AND The patient has a fever
THEN We suspect an infection

Doctor Example Second Pass

Rule, premise	Status	Working Memory
1, 1 Deep cough	True	Temp= 103 Sick for a month Coughing fits Patient has fever
1, 2 Suspect infection	Unknown	Temp= 103 Sick for a month Coughing fits Patient has fever
3, 1 Sick for over fortnight	True	Temp= 103 Sick for a month Coughing fits Patient has fever
3, 2 Patient has fever	True, fire	Temp= 103 Sick for a month Coughing fits Patient has fever Infection

▪ Rule 1

IF The patient has deep cough
AND We suspect an infection
THEN The patient has Pneumonia

▪ Rule 2

IF The patient's temperature is above 100
THEN Patient has fever

▪ Rule 3

IF The patient has been sick for over a fortnight
AND The patient has a fever
THEN We suspect an infection

Doctor Example Third Pass

Rule, premise	Status	Working Memory
1, 1 Deep cough	True	Temp= 103 Sick for a month Coughing fits Patient has fever Infection
1, 2 Suspect infection	True, fire	Temp= 103 Sick for a month Coughing fits Patient has fever Infection Pneumonia

▪ Rule 1

IF The patient has deep cough
AND We suspect an infection
THEN The patient has Pneumonia

▪ Rule 2

IF The patient's temperature is above 100
THEN Patient has fever

▪ Rule 3

IF The patient has been sick for over a fortnight
AND The patient has a fever
THEN We suspect an infection

Now no more facts can be added to the Working Memory. Diagnosis: **Patient has Pneumonia**

Issues & Conflict Resolution

- The forward chaining inference engine **infers all possible facts from the given facts**
- Has no way of distinguishing between **important and unimportant facts**.
- Therefore, equal **time spent on trivial evidence as well as crucial facts**
- **Conflict:** what happens when the premises of two rules match the given fact. Which should be fired? If we fire both, they may add conflicting facts. E.g.
 - *IF you are bored*
AND you have no cash
THEN go to a friend's place
 - *IF you are bored*
AND you have a credit card
THEN go watch a movie
- **If both rules are fired?**, you will add conflicting recommendations to the working memory.

Conflict Resolution Strategies

- **Fire first rule in sequence:** Using this strategy all the rules in the list are ordered (the ordering imposes prioritization). When more than one rule matches, we simply fire the first in the sequence.
- **Assign rule priorities (rule ordering by importance):** Using this approach we assign explicit priorities to rules to allow conflict resolution.
- **More specific rules:** (more premises) are preferred over general rules. This strategy is based on the observation that a rule with more premises, in a sense, more evidence or votes from its premises, therefore it should be fired in preference to a rule that has less premises.
- **Prefer rules:** whose premises were added more recently to WM (**timestamping**). This allows prioritizing recently added facts over older facts.
- **Parallel Strategy (view-points):** Using this strategy, we do not actually resolve the conflict by selecting one rule to fire. Instead, we branch out our execution into a tree, with each branch operation in parallel on multiple threads of reasoning. This allows us to maintain multiple view-points on the argument concurrently.

Inference Strategies: Backward Chaining

- Backward chaining is an inference strategy that works backward from a **hypothesis to a proof**.
 1. You begin with a hypothesis about what the situation might be.
 2. Then you prove it using given facts, e.g. a doctor may suspect some disease and proceed by inspection of symptoms.
 3. In backward chaining terminology, the hypothesis to prove is called the **goal**.

Inference Strategies: Backward Chaining

■ Approach

1. Start with the goal.
 2. Goal may be in WM initially, so check and you are done if found!
 3. If not, then search for goal in the THEN part of the rules (**match conclusions, rather than premises**). This type of rule is called **goal rule**.
 4. Check to see if the goal rule's premises are listed in the working memory.
 5. Premises not listed become sub-goals to prove.
 6. Process continues in a recursive fashion until a premise is found that is not supported by a rule, i.e. a premise is called a **primitive**, if it cannot be concluded by any rule .
 7. When a primitive is found, ask user for information about it. Back track and use this information to prove sub-goals and subsequently the goal.
- As you look at the example for backward chaining, notice how the approach of backward chaining is like **depth first search**.

Inference Strategies: Backward Chaining

- Consider the same example of doctor and patient that we looked at previously

Rule 1

IF The patient has deep cough

AND We suspect an infection

THEN The patient has Pneumonia

Rule 2

IF The patient's temperature is above 100

THEN Patient has fever

Rule 3

IF The patient has been sick for over a fortnight

AND The patient has fever

THEN We suspect an infection

Inference Strategies: Backward Chaining

■ Goal : Patient has Pneumonia

Step	Description	Working Memory
1	Goal: Patient has pneumonia. Not in working memory	
2	Find rules with goal in conclusion: Rule 1	
3	See if rule 1, premise 1 is known, "the patient has a deep cough"	
4	Find rules with this statement in conclusion. No rule found. "The patient has a deep cough" is a primitive. Prompt patient. Response: Yes.	Deep cough
5	See if rule 1, premise 2 is known, "We suspect an infection"	Deep cough
6	This is in conclusion of rule 3. See if rule 3, premise 1 is known, "The patient has been sick for over a fortnight"	Deep cough
7	This is a primitive. Prompt patient. Response: Yes	Deep cough Sick over a month
8	See if rule 3, premise 2 is known, "The patient has a fever"	Deep cough Sick over a month
9	This is conclusion of rule 2. See if rule 2, premise 1 is known, "Then patients temperature is above 100"	Deep cough Sick over a month

■ Rule 1

IF The patient has deep cough
AND We suspect an infection
THEN The patient has Pneumonia

■ Rule 2

IF The patient's temperature is above 100
THEN Patient has fever

■ Rule 3

IF The patient has been sick for over a fortnight
AND The patient has a fever
THEN We suspect an infection

Inference Strategies: Backward Chaining

10	This is a primitive. Prompt patient. Response: Yes. Fire Rule	Deep cough Sick over a month Fever
11	Rule 3 fires	Deep cough Sick over a month Fever Infection
12	Rule 1 fires	Deep cough Sick over a month Fever Infection Pneumonia

- Rule 1

IF The patient has deep cough
AND We suspect an infection
THEN The patient has Pneumonia

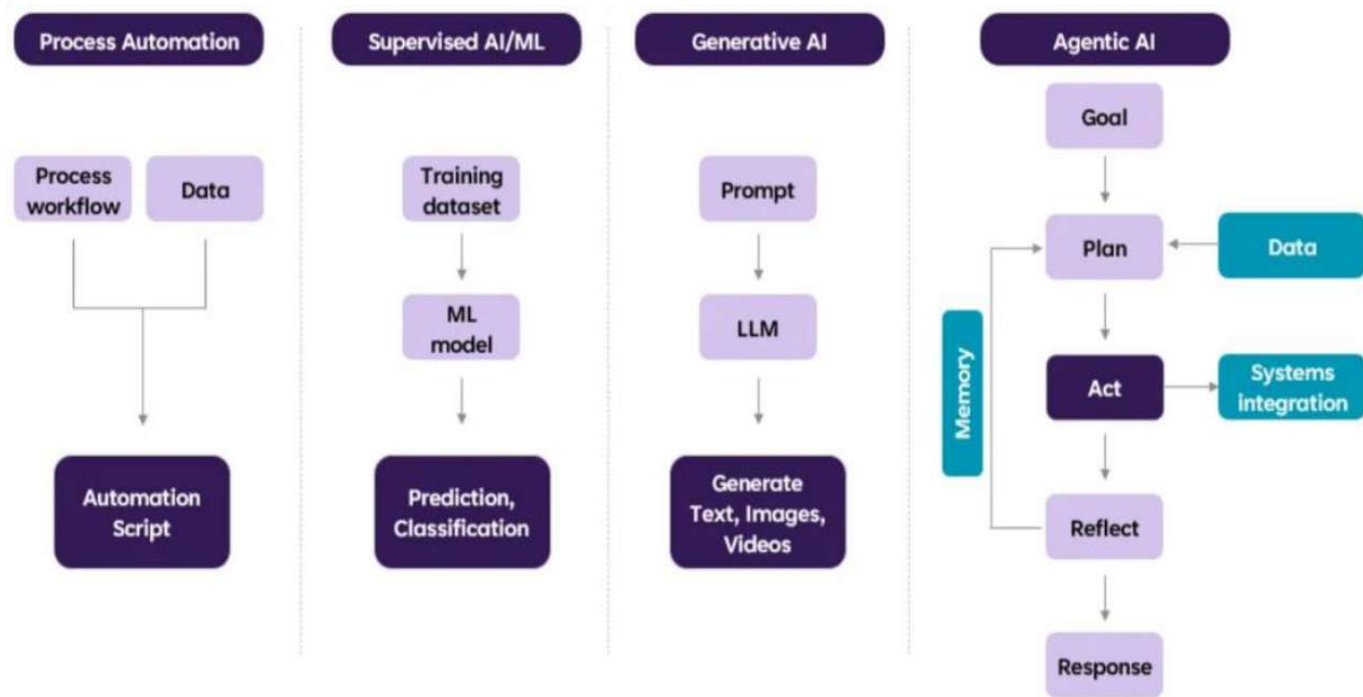
- Rule 2

IF The patient's temperature is above 100
THEN Patient has fever

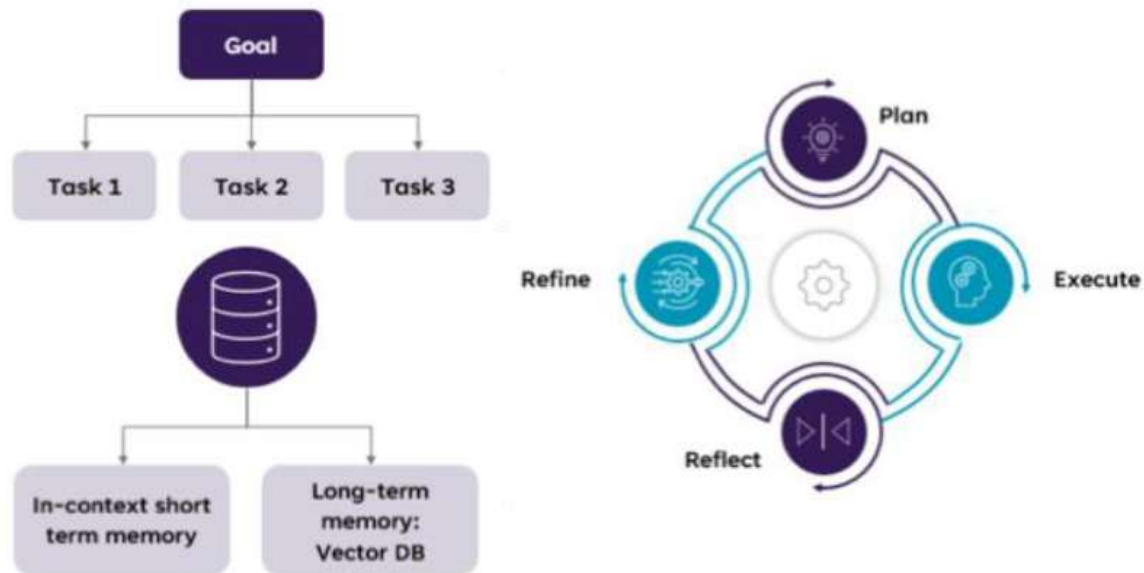
- Rule 3

IF The patient has been sick for over a fortnight
AND The patient has a fever
THEN We suspect an infection

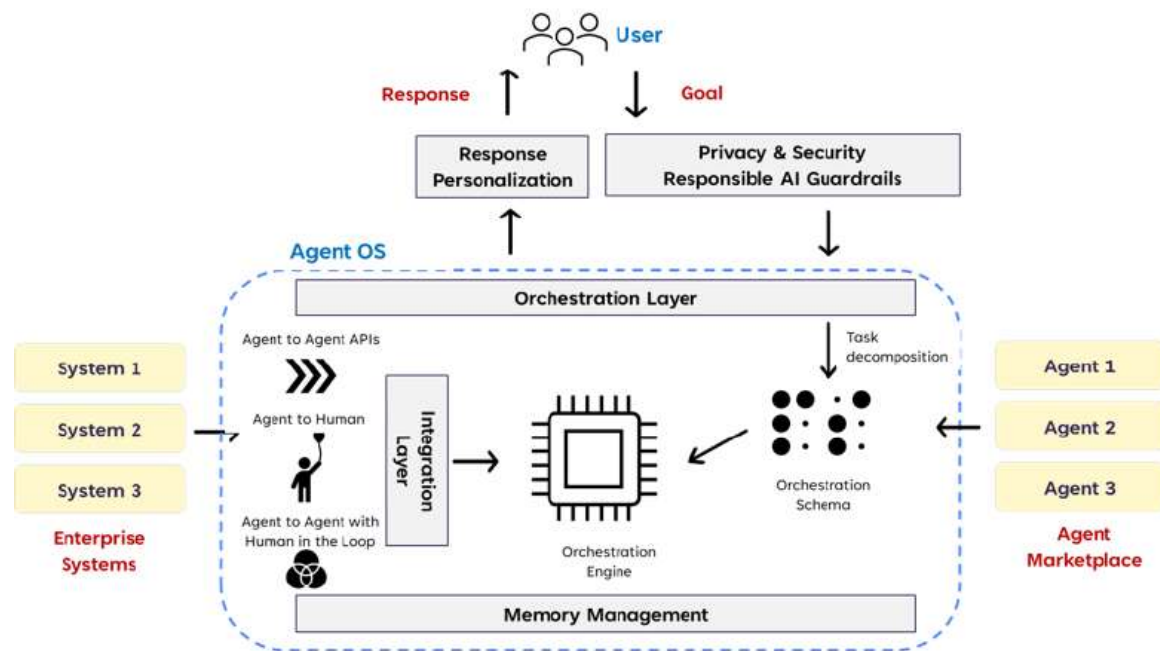
Agentic AI Evolution



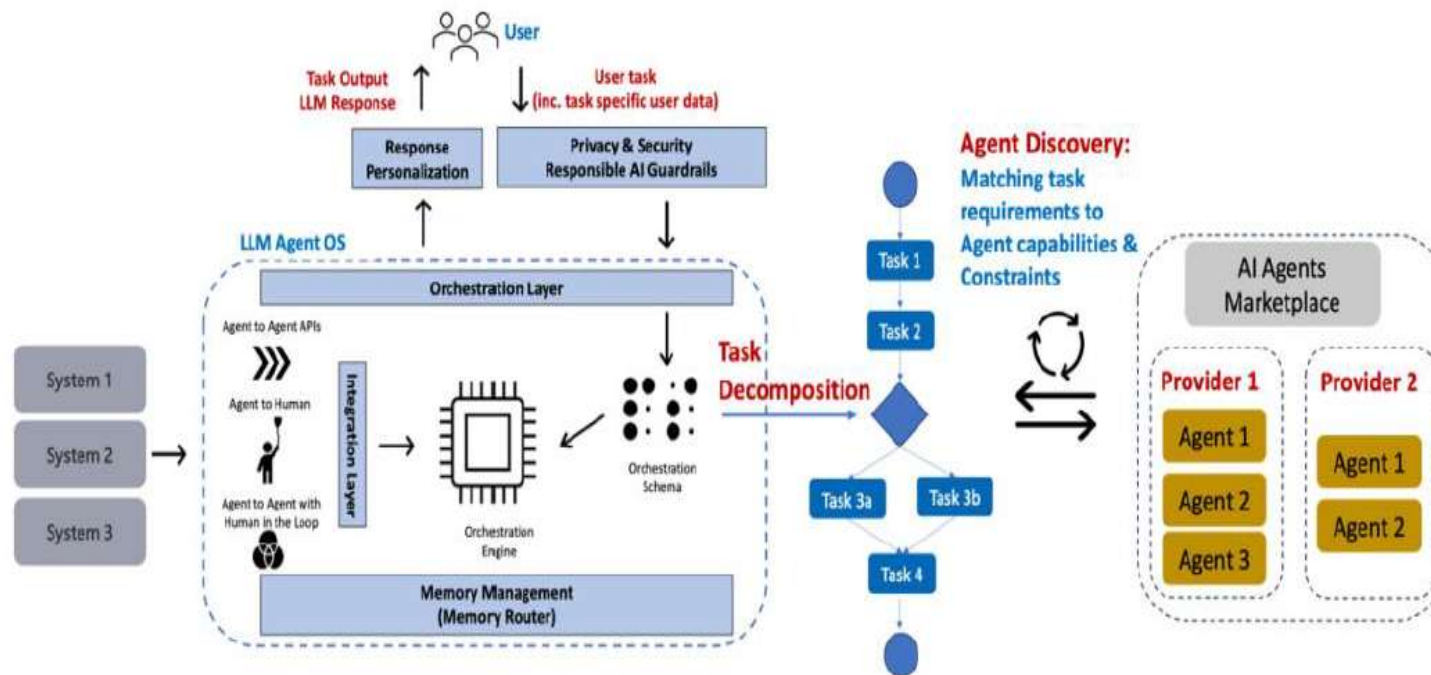
Agentic AI Core Capabilities



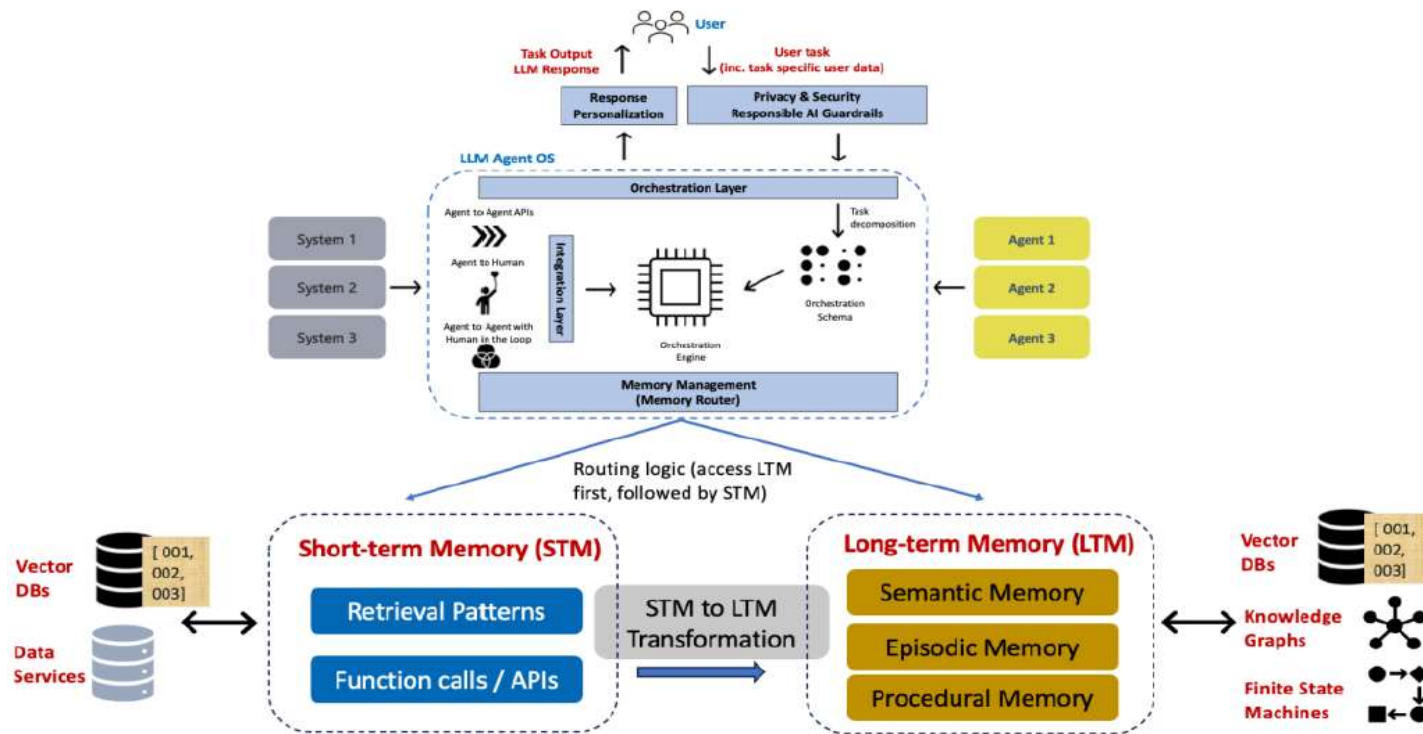
AI Agent Platform: Architecture



Agent Discovery & Matchmaking for Agentic AI



Agentic AI Memory Management



Summary

- Overview of ES, DSS and how Agentic AI evolve
- DSS components and explanation
- Memory Structure and Models
- The main components of an ES
- Explanation Facility
- Characteristics of ES
- Inference Mechanisms
 - Forward Chaining
 - Backward Chaining
- Agentic AI arch and processing
- Alignment with LLM and Agents