ARTIFICIAL INTELLIGENT SYSTEMS

(BMA-EL-IZB-LJ-RE 1. YEAR 2024/2025)

EXPERT SYSTEMS

Simon Dobrišek

LECTURE TOPICS

- Introduction
- The concept of expert systems
- Knowledge base
- The mechanism of reasoning
- Tools for building expert systems

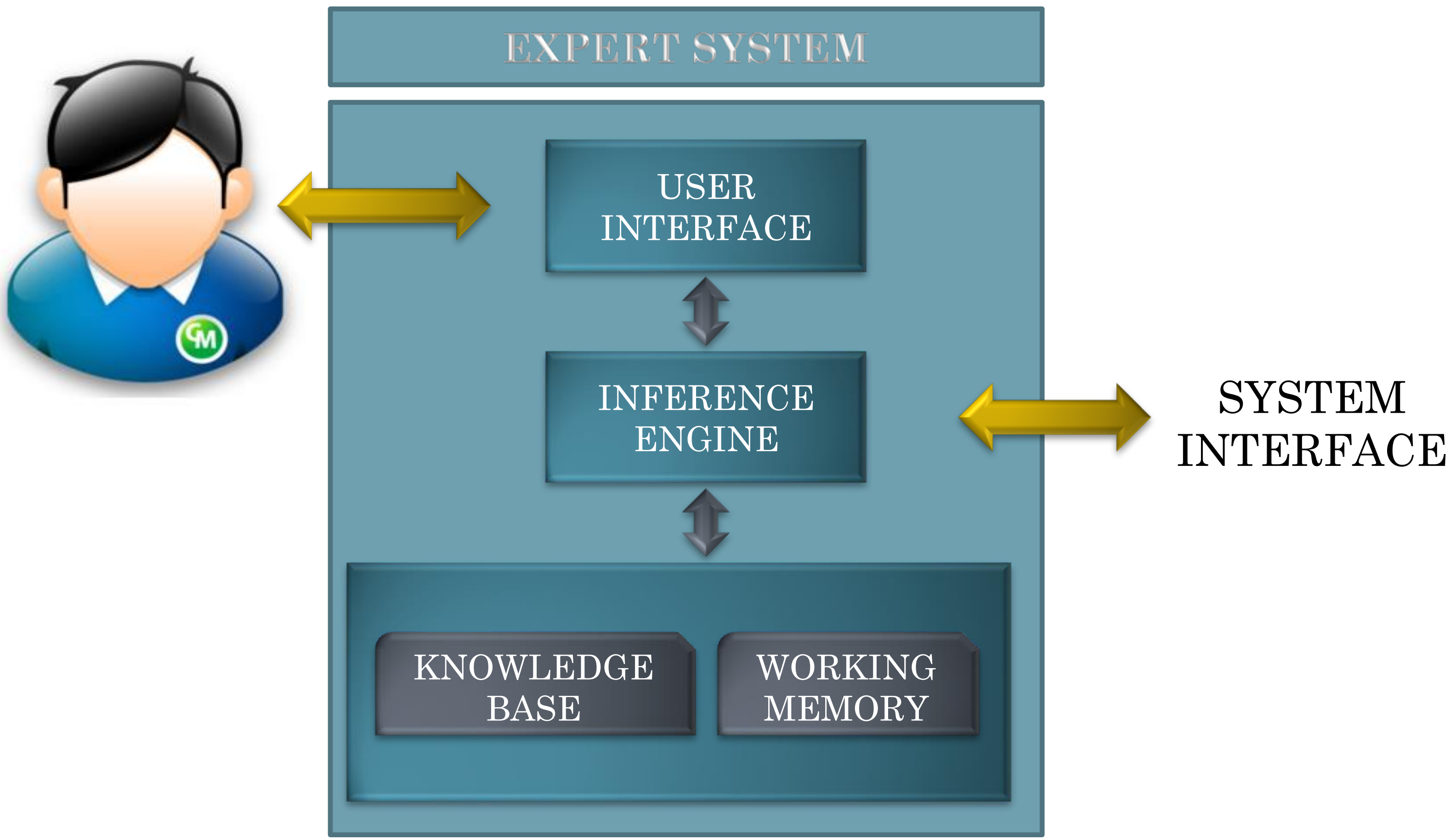
WHAT ARE EXPERT SYSTEMS?

- They try to imitate human reasoning processes.
- They embody an expertise for solving certain types of problems.
- They are based on the idea of production systems.
- They include a knowledge base and an inference mechanism.
- Their implementation usually only requires entering a knowledge base.

Components of Expert Systems

- **Knowledge base**: A declarative representation of the expertise for solving certain types of problems. Usually it contains a set of rules that provide the relations between possible facts (sensorial data and other data).
- **Working memory**: The context and the facts that have been established in solving a given problem.
- **Inference engine** – a computer program that activates the knowledge in the knowledge base.
- **User interface** – enables communication between the user and the expert system.

Structure of Expert Systems



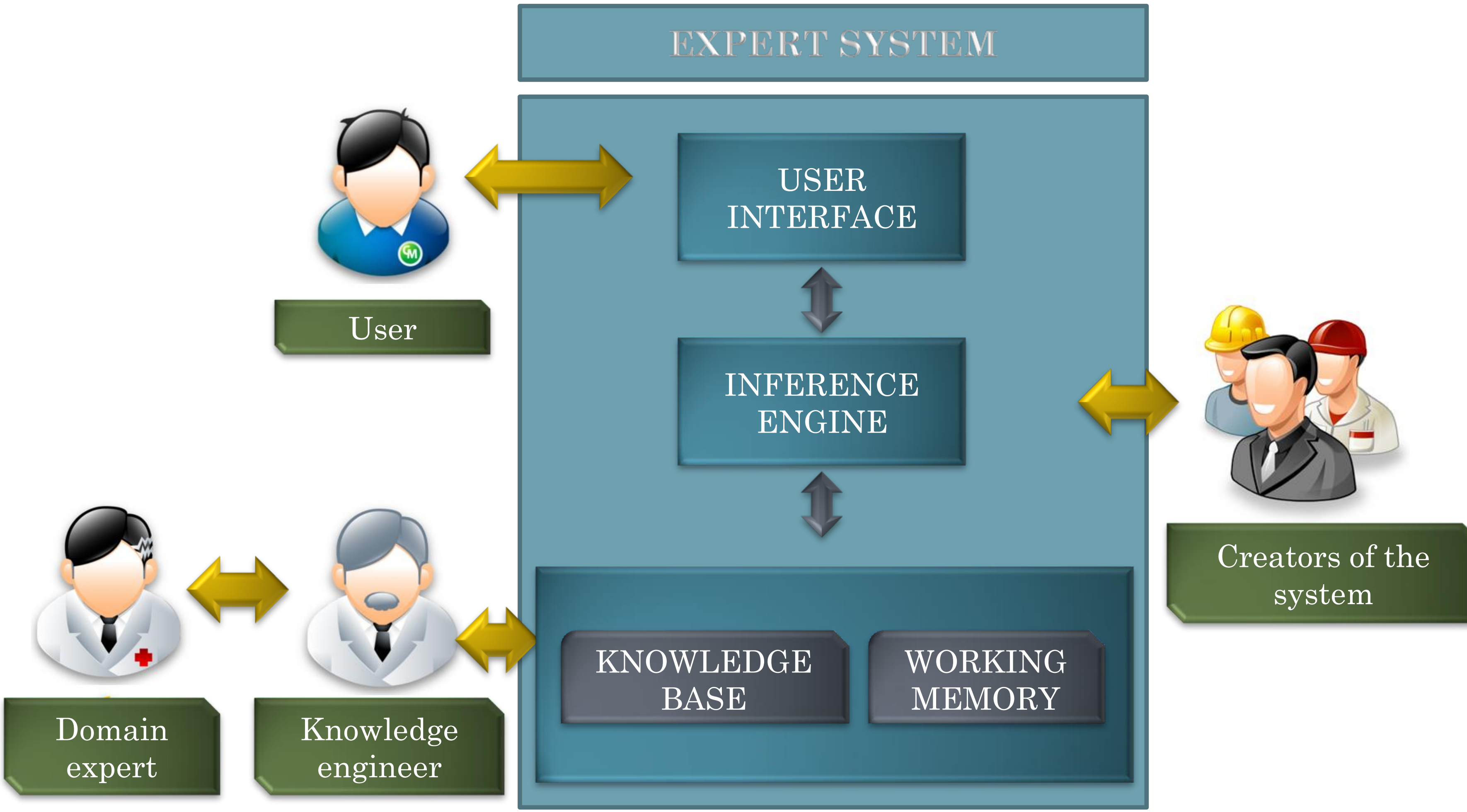
Problems solved by expert systems

- Problems that are typically dealt by experts in a narrow field of expertise.
- Problems in the field of medicine, service activities, advisory activities, providing information, etc.
- Implementation of problem-solving intelligent autonomous agents or agents in multi-agent systems.
- Wherever intelligent systems involve a lot of decision-making on the basis of a knowledge base.

When to use expert systems?

- When developing a system to solve a problem that includes many conditional branching statements that depend on the input data.
- When a decision-making process depends on many complex conditions that are in some cases interdependent.
- If the rules underpinning the decision-making process eventually change.
- If we intend to update and improve the functioning of a system with minimal intervention in the source code.
- If a client does not require a solution that is precisely tailored to their needs.

Persons Involved in the Operation of Expert Systems



Knowledge Base of Expert Systems

- A knowledge base contains a formally described idea that allows the inference engine to derive solutions to a given problem.

- Knowledge is mostly described by a production system comprising a set of IF … THEN rules

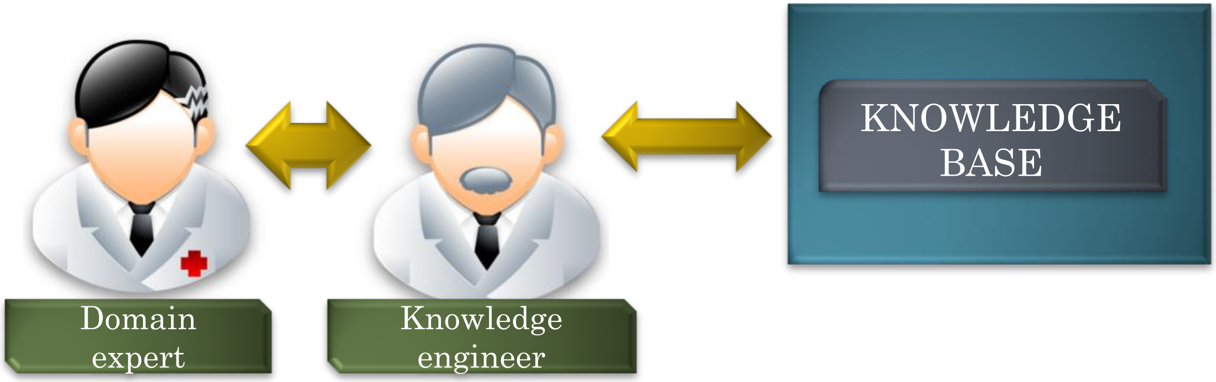
  IF *premise/antecedent* THEN *consequence/consequent*

- For example

  IF *you are hungry* THEN *you want to eat*

  (i.e., if there is the fact that *you are hungry* the premise/condition/antecedent is satisfied and the conclusion/decision/consequent is derived that *you want to eat*)

Dialogue Between a Knowledge Engineer and a Domain Expert



- The knowledge engineer must establish a dialogue with an expert in the selected field of application.
- The knowledge engineer transfers the knowledge in the knowledge base using a selected formal declarative computer programming language.
- The expert evaluate the performance of the system and provide feedback, as to enable the knowledge engineer to complement/improve the knowledge base.

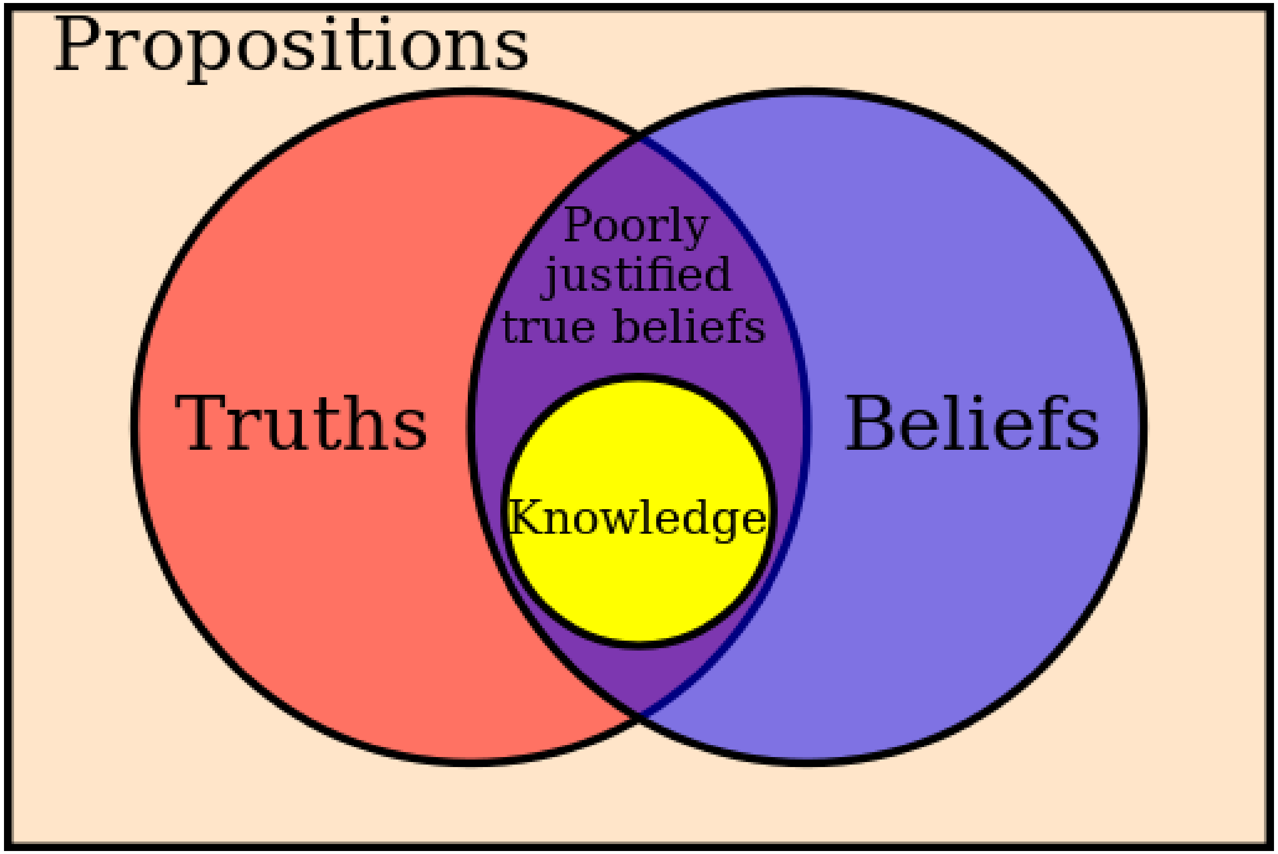
Techniques for Knowledge Acquisition

- **Protocol-generation** techniques that include many types of interviews, reporting and observational techniques.
- **Protocol-analysis** techniques that are used to identify basic knowledge objects within a protocol, such as goals, decisions, relationships and attributes.
- **Hierarchy-generation** techniques, such as laddering, are used to build taxonomies or other hierarchical structures such as goal trees and decision networks.
- **Sorting** techniques that are used for capturing the way people compare and order concepts.
- **Diagram-based** techniques including generation and use of concept maps, state transition networks, event diagrams and process maps.

Knowledge Representation in Expert Systems

- An adequate formal representation of knowledge is crucial to the success of an expert system.
- Knowledge representation in expert systems is mainly based on logical reasoning.
- The formal study of knowledge is carried out within the framework of epistemology that deals with the nature, structure and origin of the knowledge.
- Epistemology is the philosophical discipline that deals with the origin, nature, kinds, validity, and limits of human knowledge, or as the theory of knowledge, especially with regard to its scope, methods, and validity, and the distinction between justified belief and personal opinion.

EPISTEMOLOGY/GNOSEOLOGY



Euler diagram representing a definition of knowledge.

Types of Knowledge – A Priori Knowledge

- The term originates from the Latin phrase "from the earlier“ or "from the former“.
- It is independent of the concrete experiences and perceptions.
- It is universally true.
- It can not be rejected without contradiction.
- For example

“All husbands are married.”

A Posteriori (Empirical) Knowledge or Justification (Cognition)

- The term originate from the Latin phrase "*from the latter*".
- It is derived from the specific experiences and perceptions.
- It may not always be reliable.
- It is refused on the basis of new perceptions.
- For example

  "All Model T Fords are black."

Explicit, implicit, declarative, and procedural knowledge

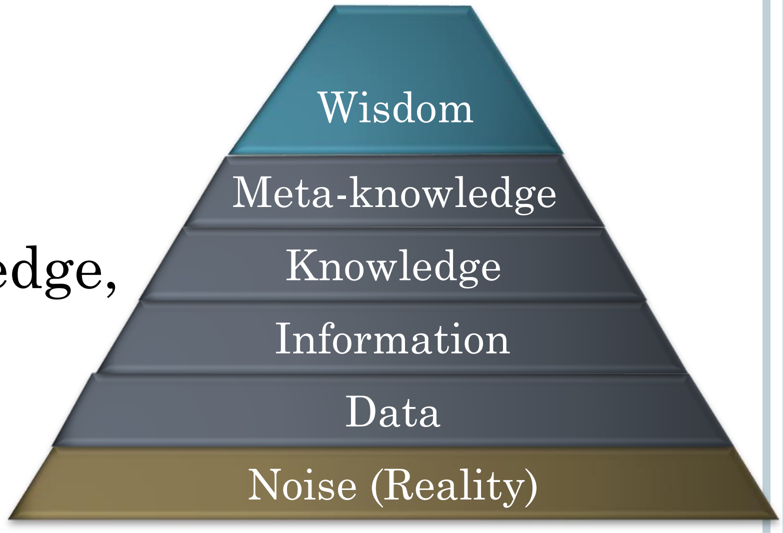
- Explicit knowledge is the knowledge that can be recorded and communicated through mediums using a communication language (*books, databases, etc*).

- Implicit (tacit) knowledge is extremely difficult to be verbalized and communicated through any medium (*e.g. different kinds of skills, such us facial recognition, swimming, skiing, riding a bicycle, driving a car, playing musical instruments, etc.*).

- Declarative (propositional, descriptive) knowledge can be expressed in declarative sentences or indicative propositions (*i.e., knowing that something is true or not, e.g., "There was a traffic accident."*).

- Procedural knowledge can be applied to solving problems (*e.g., cooking recipes, software installation guides, contingency plans, policies, etc*).

THE KNOWLEDGE PYRAMID



- Data, information, (meta-)knowledge, and wisdom are important terms that need to be understood when building expert systems.
  - **Data:** A symbolic reflection of the reality (*e.g. measurements*).
  - **Information:** Structured data that provide answers to "who", "what", "where", and "when" questions.
  - **Knowledge:** The application of data and information that provide answers to "how" questions.
  - **Meta-knowledge:** Understanding that provide answers to “why" questions.
  - **Wisdom:** Evaluated understanding that provide answers to the question “Does this make sense?”

Knowledge Representation

- Knowledge Representation (KR) is one of the principal concepts of Artificial Intelligence, and a critical part of all problem solving.

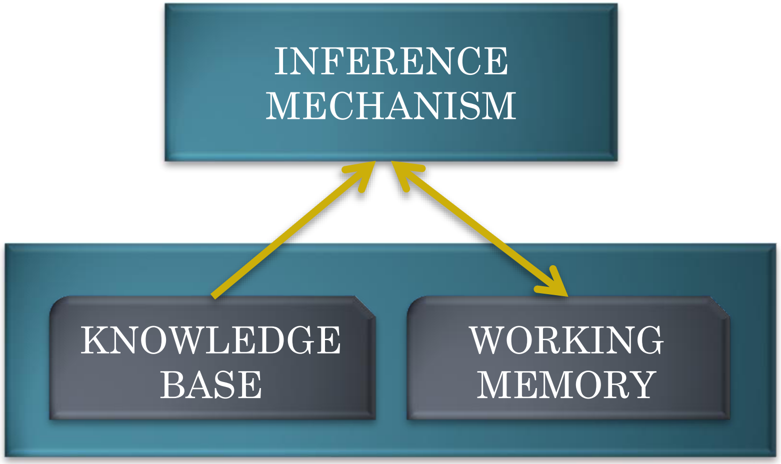
- The most common knowledge representation schemes are

  - Decision trees
  - (Logical) production rules
  - Semantic networks and frames
  - Petri nets
  - Conceptual graphs
  - Programming scripts
  - …

Knowledge representation based on production rules

- A production rule (a *production*) is a rewrite rule specifying a symbol substitution that can be recursively performed to generate new symbol sequences.
- A production system provides the inference mechanism necessary to execute production rules in order to achieve some goal.
- Knowledge can be represented by the production rules that are executed by facts (sensory preconditions) or by any other already executed rule that created new facts.
- The result is an execution of the production rules which action (consequent) are declared to be one of the goals of the system.

PRODUCTION SYSTEM

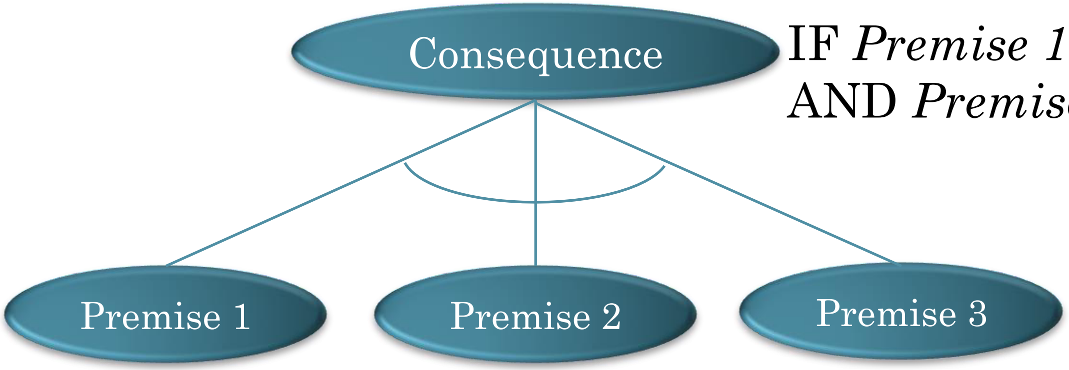


- **Knowledge base**: A database of production rules that provide the relations between the possible facts :

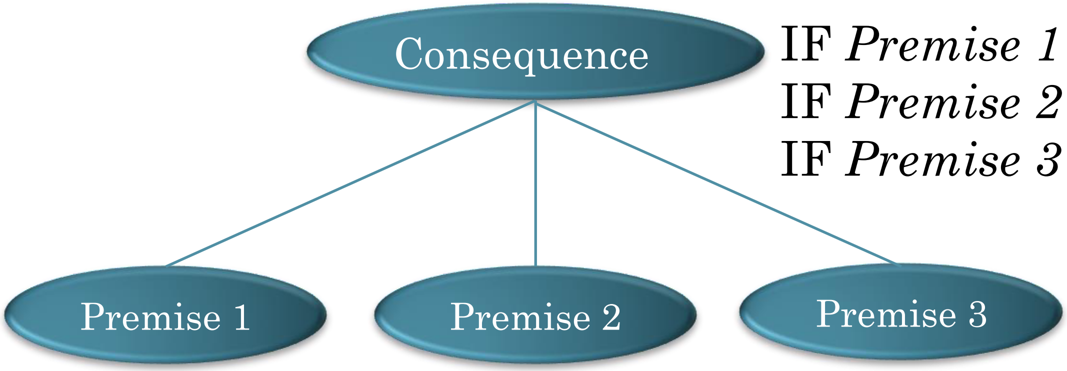
  IF *premise/condition* THEN *consequence/action*

- **Working memory**: A set/list of facts that may satisfy premises/conditions of the rules and cause their execution.
- **Inference mechanism**: An algorithm that selects and executes the production rules with respect to the facts (data) in the working memory.

PRESENTATION OF THE PRODUCTION RULES



IF *Premise 1* AND *Premise 2* AND *Premise 3* THEN *Consequence*



IF *Premise 1* THEN *Consequence*
IF *Premise 2* THEN *Consequence*
IF *Premise 3* THEN *Consequence*

Inference Mechanism

- The inference engine activates the knowledge in the knowledge base to deduce results from it, with respect to the input (sensorial) data.
- A knowledge base is usually large and it is necessary to have an inferencing mechanism that search through the database and deduce results in an organized manner.
- Inferences are drawn from the knowledge base using different search techniques, such as
  - Logical tree search
  - Forward chaining of rules
  - Backward chaining of rules

(Data-Driven) Forward Chaining of Rules

- This method can be described logically as a repeated application of modus ponens (*a logical rule of inference - P implies Q; P is asserted to be true, so therefore Q must be true*).
- Searching for the rules which premises (antecedents) are considered to be true with respect to the facts in the working memory.
- Such rules are executed and their conclusions (consequents) are stored in the working memory as new facts.
- Searching and executing rules take place as long as
  - The inferred consequent does not match the desired goal of inference, or
  - No more rules can be executed with respect to the facts in the working memory.

An Illustration of Forward Chaining

KNOWLEDGE BASE

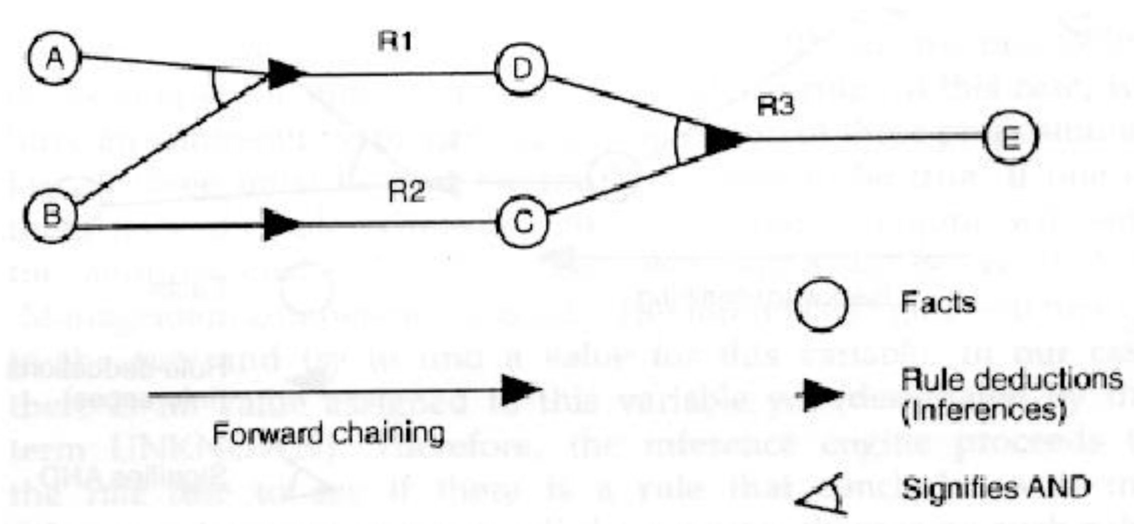
WORKING MEMORY

R1: IF A AND B THEN D

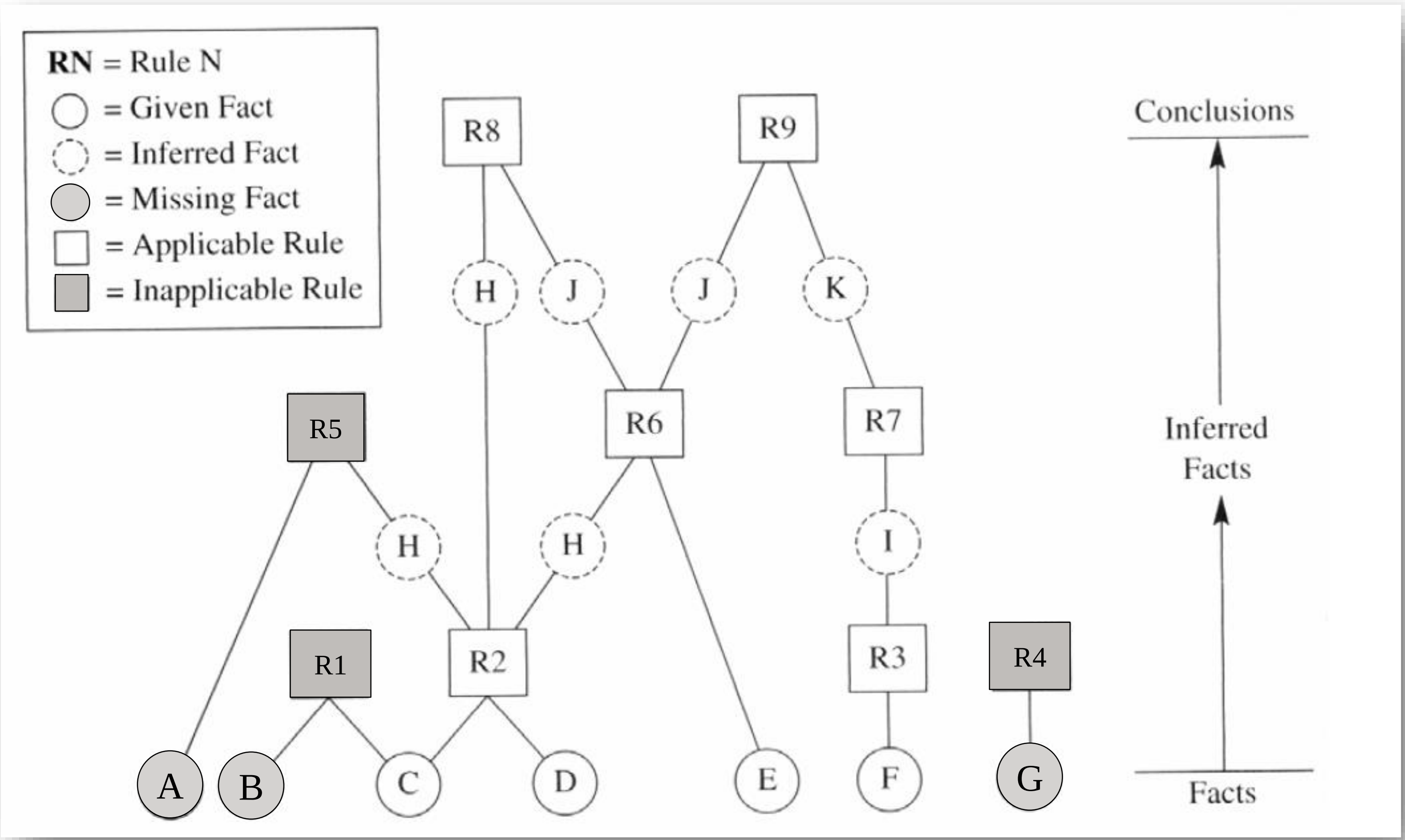
R2: IF B THEN C

R3: IF C AND D THEN E

A,B



Forward Chaining Seen as a Breath-First Bottom-up Data-driven Search



(Goal-Driven) Backward Chaining of Rules

- This method can be described as searching backward from the desired goal(s) of inference and repeatedly from the consequents to the antecedents to see if there are facts available in the working memory that support the goal(s) (and the intermediate consequents).
- Searching for the rules which consequents match the desired goal of inference.
- If the premise (antecedent) of that rule is not known to be true, with respect to the working memory, then it is added to the list of goals.
- Backward chaining systems usually employ a depth-first search strategy.

AN ILLUSTRATION OF BACKWARD CHAINING

KNOWLEDGE BASE

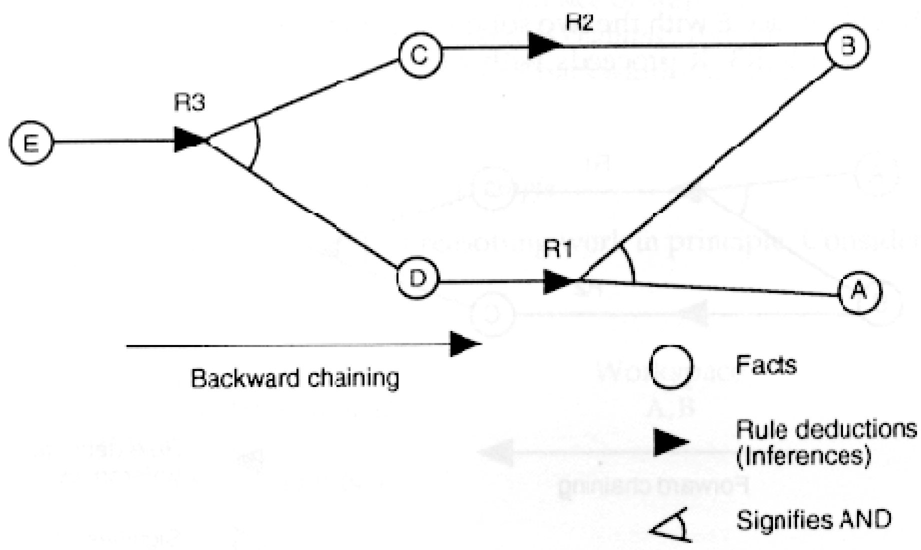
WORKING MEMORY

R1: IF A AND B THEN D

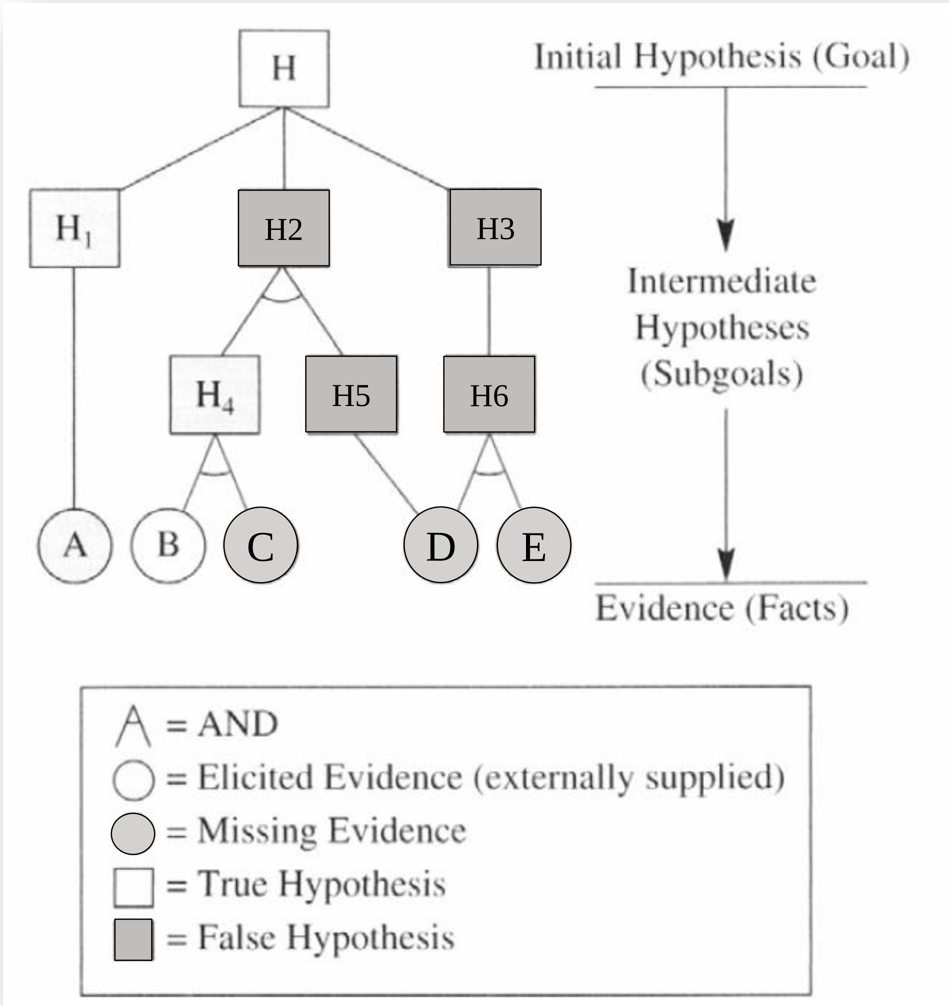
R2: IF B THEN C

R3: IF C AND D THEN E

A,B



BACKWARD CHAINING SEEN AS A DEPTH-FIRST TOP-DOWN GOAL-DRIVEN SEARCH STRATEGY



Differences between the two strategies

- Forward chaining employs a breath-first bottom-up data-driven search strategy.
- Backward chaining employs a depth-first top-down goal-driven search strategy.

Forward chaining	Backward chaining
Infer from the present to the future	Infer from the present to the past
Infer from ancestors/antecedents to descendants/consequents	Infer from descendants/consequents to ancestors/antecedents
Planning and controlling systems	System diagnostics
Data-driven bottom-up search/reasoning	Goal-driven top-down search/reasoning
Search the conclusions that follow from the facts	Search the facts that support the conclusions
Explanation of the chain of reasoning is not possible	Explanation of the chain of reasoning is possible

An example dialogue

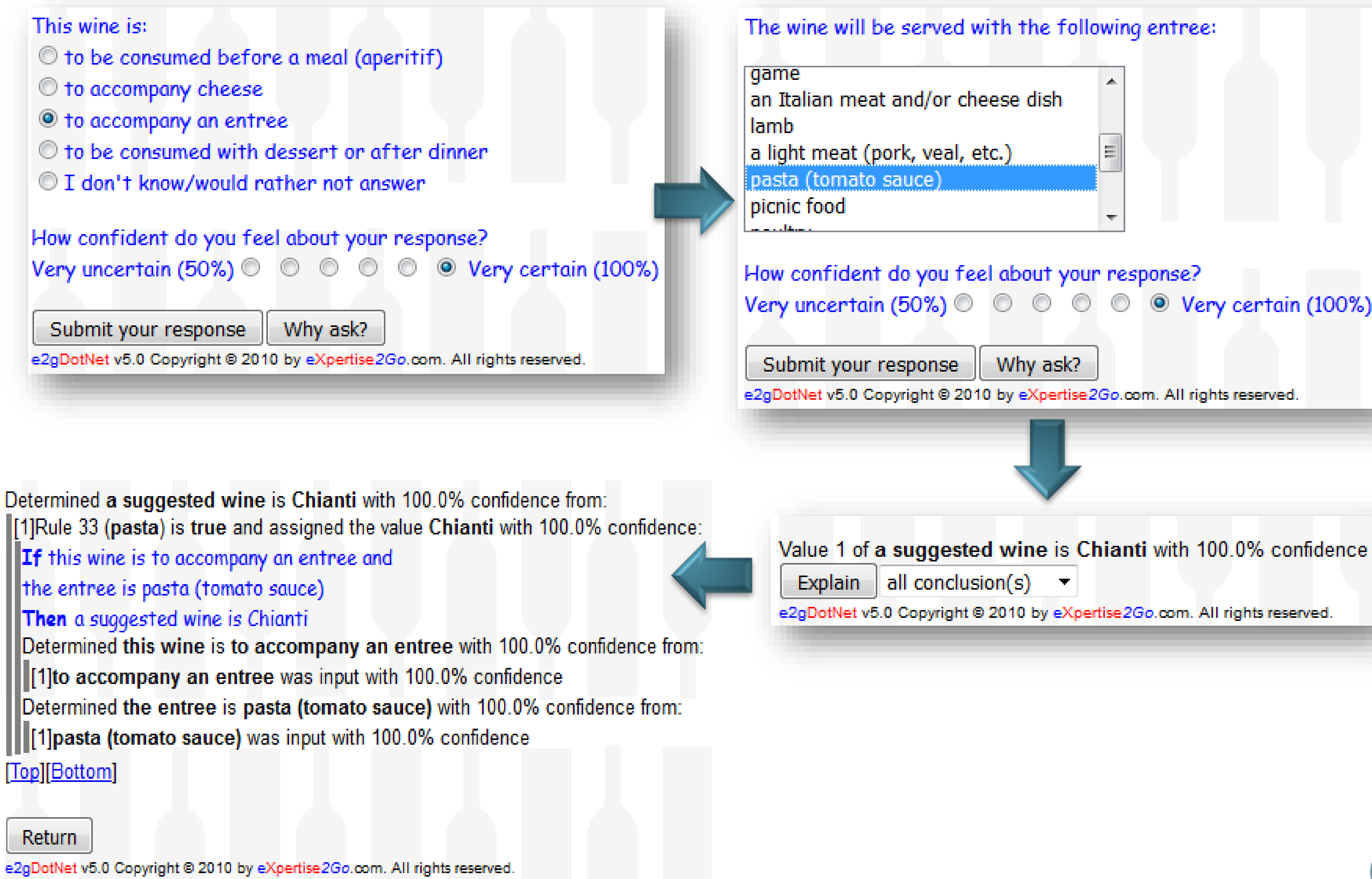
Dialog with an expert system designed to give advice on car problems.

System	Does the car start ?
User	No.
System	Does the engine turn over ?
User	Yes.
System	Do you smell gas ?
User	Yes
System	Recommendation - Wait 5 minutes and try again.
User	Why ?
System	I used the rule: If not start, and engine_turn_over, and smell_gas Then recommendation is 'Wait 5 minutes and try again'
User	

Note : The rule gives the correct advice for a flooded car, and knows the questions to be ask to determine if the car is flooded, but it does not contain the knowledge of **what** a flooded car is and **why** waiting will help.

RC Chakraborty, www.myreaders.info

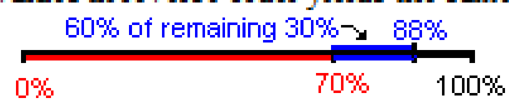
AN EXAMPLE DIALOGUE



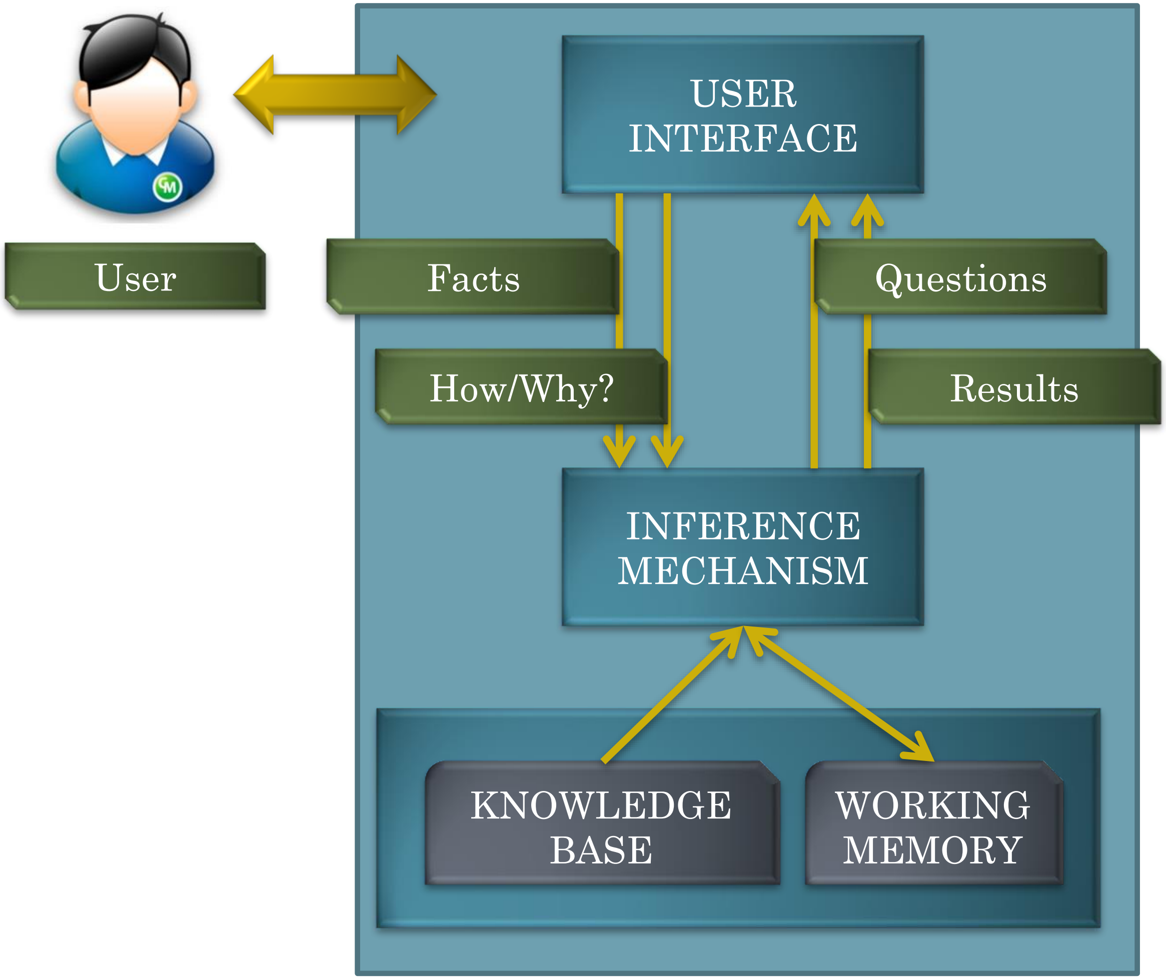
Coping with Uncertainty

- Often the knowledge is imperfect which causes uncertainty.
- To work in the real world, expert systems must be able to deal with uncertainty.
- This can be achieved by:
  - Taking into account the uncertainties in presenting the facts.
  - Using probabilistic theories that take into account the uncertainty (Bayes, Shannon, ...).
  - Assigning confidence factors (CF) to the facts, premises and consequents.
  - The use of fuzzy logic and fuzzy rules.

An example of coping with uncertainty

Example	Method	Result	Justification
IF a AND b CFa=60% CFb=80%	Minimum	Premise is true, CF=60%	*Both* a and b must be true for the premise to be true: the overall certainty is driven by the weakest certainty.
IF a OR b CFa=60% CFb=80%	Maximum	Premise is true, CF=80%	The premise is true if *either* a or b is true: the overall certainty is driven by the strongest certainty.
IF a AND b IF a OR b CFa=60% CFb=80%	Average	Premise is true, CF=70%	A compromise between the Maximum and Minimum methods for either premise.
CFa=70% from 1st source CFa=60% from 2nd source	Probability Sum	Final CFa=88% 70+(60/100)x(100-70)	30% uncertainty remains from the first certainty factor of 70%. The second CF explains 60% of this remaining 30%. (Note: applying the two certainty values in reverse order yields the same result.) 
IF a AND b Then c=5 with 80% confidence (premise CF=90%)	Multiplication	c is 5 with 72% confidence	The confidence assigned to the attribute in the rule conclusion is assumed independent of the premise confidence level, so the two are combined by multiplying as independent probabilities would be combined.

Expert System User Interface



User Interface

- The user interface is a means of communication with the user.
- The user interface is generally not a part of the expert system technology.
- The user interface can make a critical difference in the perceived utility of an expert system.
- Most expert systems have explanation facilities that allow the user to ask questions - why and how it reached some conclusion.
- The questions are answered by referring to the system goals, the rules being used, and the problem being solved.
- By looking at explanations, the knowledge engineer can see how the system is behaving, and how the rules and data in the working memory are interacting.

EXPERT SYSTEM SHELLS

- An expert system shell is a software development environment that contains the basic components of expert systems.
- A shell is associated with a prescribed method for building applications by configuring and instantiating the implemented components.
- There are several software frameworks and shells for the development of expert systems.
- They contain the main expert system components, but without any built-in knowledge.

Expert System Shells

- Examples of software frameworks and expert system shells are:
  - CLIPS - A Tool for Building Expert Systems
    https://www.clipsrules.net/
  - Drools - A business-rule management system
    https://www.drools.org/
  - JLisa - A Clips-like Rule engine
    http://jlisa.sourceforge.net/
  - FuzzyClips
    http://en.wikipedia.org/wiki/FuzzyCLIPS
  - Prolog is often used to build expert systems
    e.g., http://www.swi-prolog.org

CLIPS – C Language Integrated Production System

- CLIPS was designed using the C language at the NASA/Johnson Space Center https://www.clipsrules.net/
- CLIPS is a rule-based programing language that:
  - Can create an manage a fact list;
  - Can create and manage a rule list;
  - Has an inference engine that matches facts against rules.
- CLIPS is also an object-oriented language that can define classes, create different set of instances and interface rules and objects.

CLIPS – Basics

- In CLIPS, facts can be asserted

```
CLIPS> (assert (today is Thursday))
<Fact-0>
```

- listed

```
CLIPS> (facts)
f-0 (today is Thursday)
```

- and retracted

```
CLIPS> (retract 0)
CLIPS> (facts)
```

CLIPS – Basics

- The syntax for defining a rule is

```
(defrule rule-name "comment"
     (if-condition)
     (if-condition)
     (etc.)
=>
     (action 1)
     (action 2)
     (etc.)
)
```

CLIPS – Basics

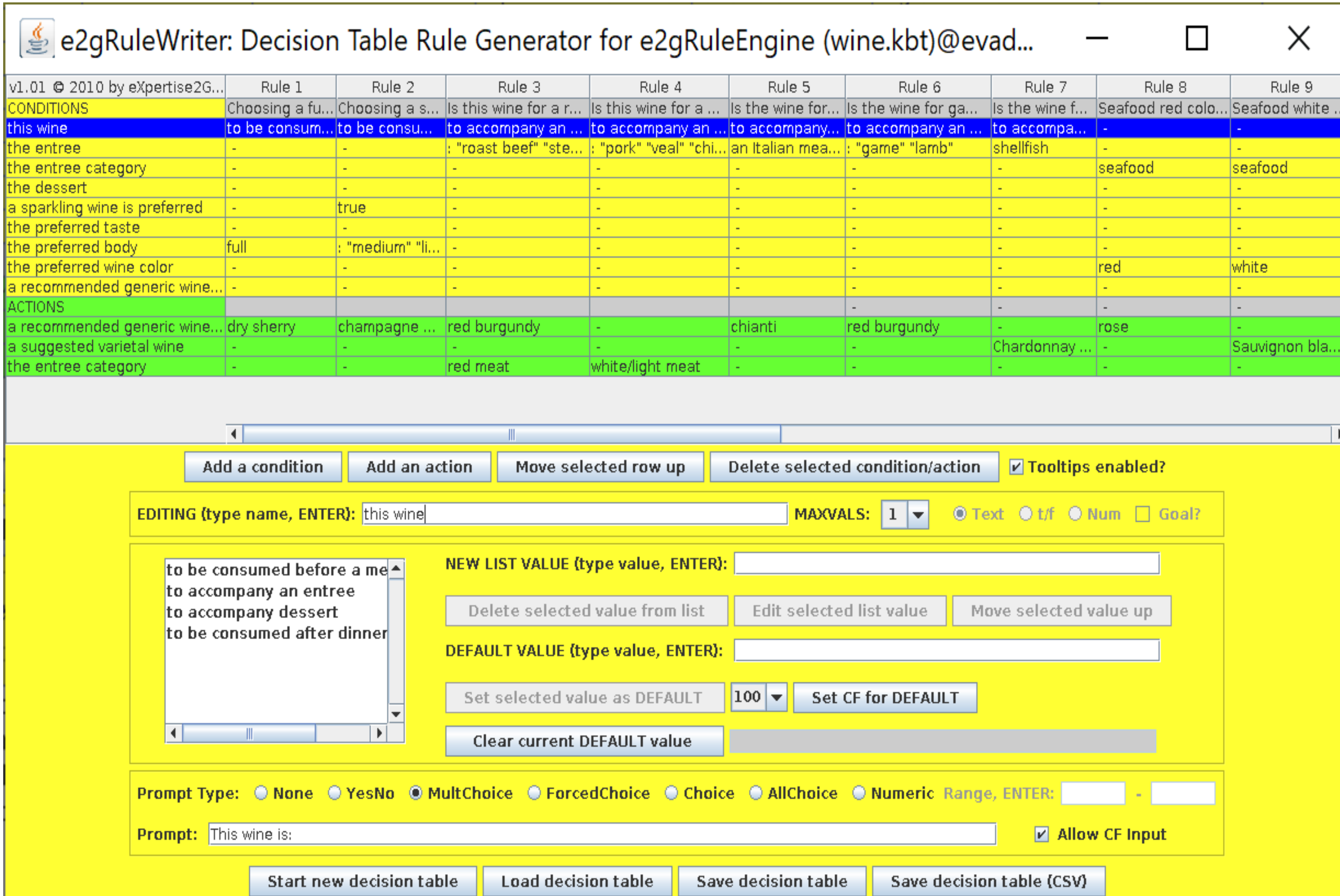
- Defining a production rule in CLIPS

```
(defrule useful-rule "a useful rule"
    (animal fierce)
    (animal big)
    (animal hungry)
 =>
    (assert (run away))
 )
```

- Running CLIPS

```
CLIPS>(clear)
CLIPS>(load "knowledge.clp")
CLIPS>(reset)
CLIPS>(run)
CLIPS>(exit)
```

An Example Knowledge Base (Experise2Go)



v1.01 © 2010 by eXpertise2G...	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8	Rule 9
CONDITIONS	Choosing a fu...	Choosing a s...	Is this wine for a r...	Is this wine for a ...	Is the wine for...	Is the wine for ga...	Is the wine f...	Seafood red colo...	Seafood white ...
this wine	to be consum...	to be consu...	to accompany an ...	to accompany an ...	to accompany...	to accompany an ...	to accompa...	-	-
the entree	-	-	: "roast beef" "ste...	: "pork" "veal" "chi...	an Italian mea...	: "game" "lamb"	shellfish	-	-
the entree category	-	-	-	-	-	-	-	seafood	seafood
the dessert	-	-	-	-	-	-	-	-	-
a sparkling wine is preferred	-	true	-	-	-	-	-	-	-
the preferred taste	-	-	-	-	-	-	-	-	-
the preferred body	full	: "medium" "li...	-	-	-	-	-	-	-
the preferred wine color	-	-	-	-	-	-	-	red	white
a recommended generic wine...	-	-	-	-	-	-	-	-	-
ACTIONS						-	-	-	-
a recommended generic wine...	dry sherry	champagne ...	red burgundy	-	chianti	red burgundy	-	rose	-
a suggested varietal wine	-	-	-	-	-	-	Chardonnay ...	-	Sauvignon bla...
the entree category	-	-	red meat	white/light meat	-	-	-	-	-

An Example Knowledge Base

```
RULE [Choosing a full bodied aperitif]
If [this wine] = "to be consumed before a meal (aperitif)" and
[the preferred body] = "full"
Then [a recommended generic wine type] = "dry sherry"

RULE [Choosing a sparkling aperitif]
If [this wine] = "to be consumed before a meal (aperitif)" and
[a sparkling wine is preferred] = true and
[the preferred body] : "medium" "light"
Then [a recommended generic wine type] = "champagne or sparkling white"

RULE [Is this wine for a red meat entree?]
If [this wine] = "to accompany an entree" and
[the entree] : "roast beef" "steak"
Then [a recommended generic wine type] = "red burgundy" and
[the entree category] = "red meat"

RULE [Is this wine for a white/light meat entree?]
If [this wine] = "to accompany an entree" and
[the entree] : "pork" "veal" "chicken" "turkey"
Then [the entree category] = "white/light meat"

RULE [Is the wine for Italian meat & cheese dish?]
If [this wine] = "to accompany an entree" and
[the entree] = "an Italian meat and cheese dish"
Then [a recommended generic wine type] = "chianti"

…
```

An Example Knowledge Base

```
…
PROMPT [a sparkling wine is preferred] YesNo
"Do you prefer a sparkline wine?:"

PROMPT [the preferred taste] MultChoice CF
"Which taste do you prefer in a wine?"
"sweet"
"medium dry"
"dry"

PROMPT [the preferred body] MultChoice CF
"Which body do you prefer in a wine?"
"light"
"medium"
"full"

PROMPT [the preferred wine color] MultChoice CF
"Which color of wine do you prefer?"
"red"
"white"

MAXVALS [a recommended generic wine type] 2
MAXVALS [a suggested varietal wine] 2

GOAL [a recommended generic wine type]
GOAL [a suggested varietal wine]

MINCF 80
```

Application of Expert Systems

- Diagnosis and troubleshooting of devices and systems.
- Planning and scheduling, such as manufacturing process planning and job scheduling.
- Configuration of manufactured objects from sub-assemblies.
- Financial decision making, e.g. advisory programs assisting bankers and insurance agents.
- Knowledge publishing, e.g. a tax advisor on tax strategy.
- Process monitoring and control, e.g. analysis of real-time data from physical devices, looking for anomalies, predicting trends, controlling optimality and failure correction.

STRENGTHS OF EXPERT SYSTEMS

- Low development costs.
- All knowledge of the system is stored in the knowledge base.
- The possibility of easily understandable explanation of the decisions made for the users who are not computer experts.
- The same system can be used for different problems.
- Knowledge of expert staff can be captured to some extent before they move on.
- Can be used as a training aid to increase the expertise of staff.

WEAKNESSES OF EXPERT SYSTEMS

- They require a lot of detailed knowledge.
- Not all domain knowledge fits rule format.
- Knowledge acquisition is time consuming.
- The results are completely dependent on the accuracy of the knowledge.
- They do not have the ability to learn from the experiences with users.
- Useless in unexpected operating conditions.
- Limited to less extensive knowledge base.

QUESTIONS

- What are expert systems?
- What are the basic components of expert systems?
- What are the strengths and weakness of expert systems?
- Describe the production system.
- Describe forward and backward chaining rules.
- How expert systems deal with uncertainty?
- Give examples expert system shells.