UNIT - V
Artificial Intelligence

III / II CSE, R 16 - JNTUK

Dr. T. K. Rao

VVIT

1. Phases in building Expert Systems

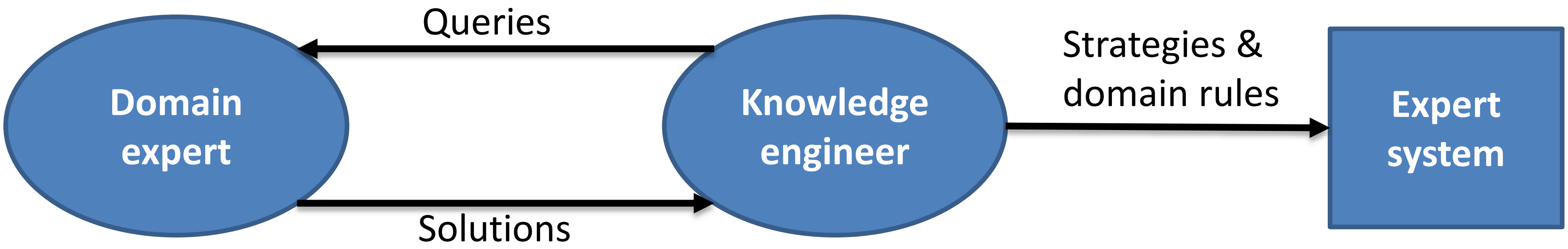
- Building Expert System (ES) requires extracting the knowledge from a human domain expert
- This extracted knowledge should be represented
- This phase is said to be ‘Knowledge Acquisition’ and this is knowledge engineering
- Following are the phases of building ES:
  - **Identification phase**
  - **Conceptualization phase**
  - **Formalization phase**
  - **Implementation phase**
  - **Testing phase**

- **Identification phase:**
  - Knowledge engineer determines the important features of the problem with the help of human domain expert
  - The parameters (determined in this phase) include the scope, resources required, and the goal of ES
- **Conceptualization phase:**
  - Knowledge engineer and domain expert decide the concepts, relations, and control mechanism needed to describe problem solving method
  - In this, issue of granularity is also addressed, which refers to the level of details required in the knowledge

- **Formalization phase:**
  - This involves expressing the key concepts and relations in some framework supported by ES building tools
  - Formalized knowledge contains DS, inference rules, control strategies, and languages required
- **Implementation phase:**
  - Formalized knowledge is converted to a working computer program, called prototype of the system
- **Testing phase:**
  - Evaluating the performance and revising the system
  - Domain experts provide feedback

1.1. Knowledge Engineering

- The whole process of building an ES is referred to as Knowledge Engineering
- It involves ES builder, or the knowledge engineer, domain experts, and potential users
- Responsibilities of a knowledge engineer are:
  - To ensure the computer has all the knowledge needed to solve a problem
  - Choosing one or more forms to represent the required knowledge
  - To ensure the computer can use the knowledge efficiently by selecting some of the reasoning methods



Interaction between Knowledge engineer and Domain Expert for creating an EX

- Role of knowledge engineer(KE) begins once the problem for developing an ES is decided
- KE interviews the domain expert to extract general rules and ensures their correctness from expert
- Then KE translates the knowledge into a computer program and design an interface

1.2. Knowledge Representation

- The power of ES is its high quality knowledge
- Domain knowledge is **knowledge base** & problem solving knowledge is **inference engine**
- Most common KR scheme for ES consists of **production rules** or simply **rules** (if-then-else)
- ES in which knowledge is represented in the form of rules is known as **Rule-based ES**
- If rules have some uncertainty; Statistical techniques are used to handle such rules

- Another widely used representation in ES is unit (also known as frame, semantic net, etc.)
- Unit is an assemblage of associated symbolic knowledge about an entity to be represented
- Unit consists of properties and values of entity
- Values of properties represent the names of other units that are linked as per the relations
- Future of ES depends on developing better methods of:
  - Knowledge acquisition and in codifying
  - Representing a large knowledge infrastructure

2. Expert System Architecture

- User interacts with ES through an interface which may have menus, natural language or any other style of interaction
- An inference engine is used to reason with expert knowledge as well as the data specific to the problem being solved
- Case specific data will be included in working memory
- Systems also have knowledge acquisition module that helps the expert or knowledge engineer to easily update and check the knowledge base

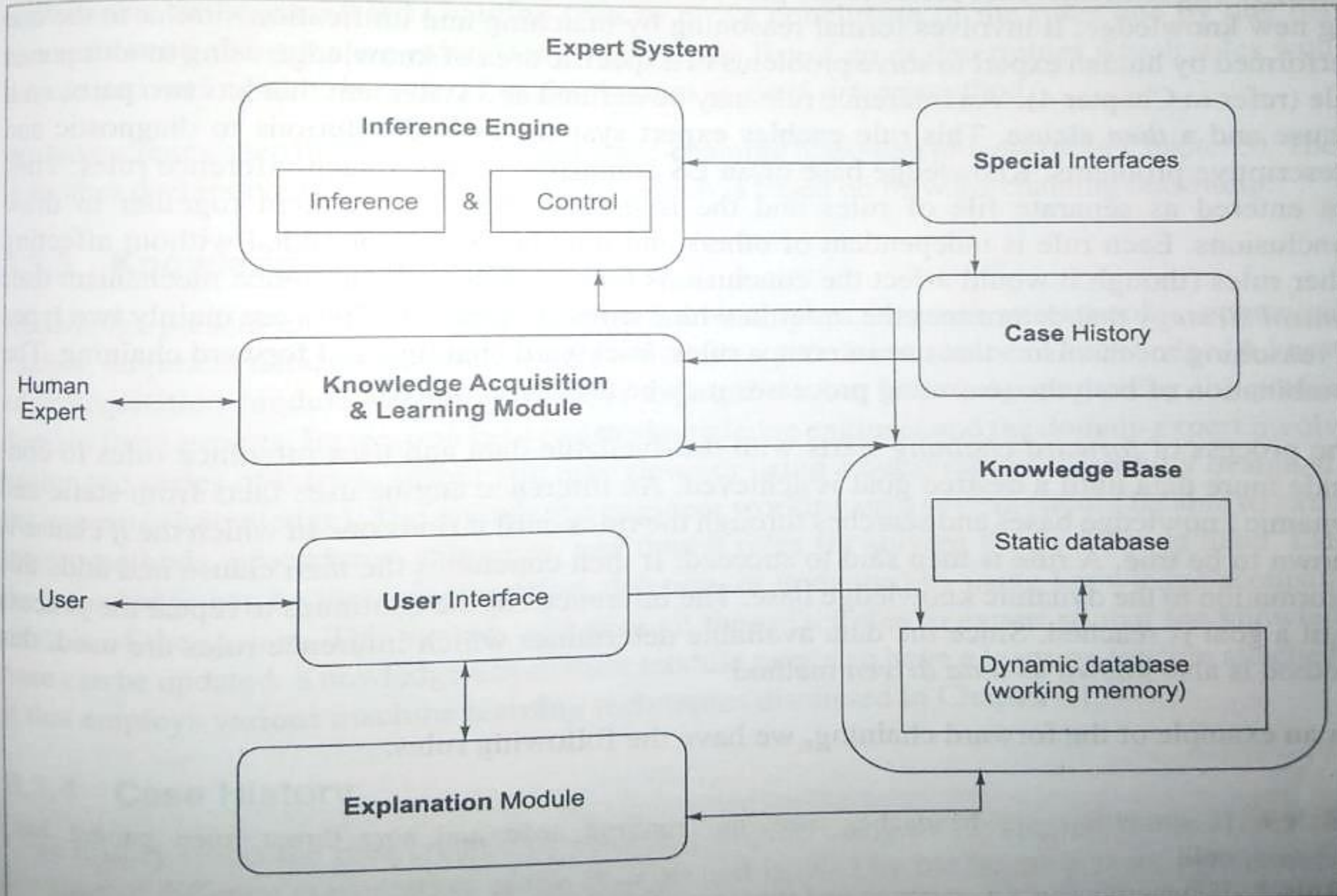


Figure 8.2 Architecture of an Expert System

- **Knowledge base**:
  - Consists knowledge of a problem domain in the form of static and dynamic databases
  - Static knowledge contains rules and facts and other types and does not change during the execution
  - Dynamic knowledge consists of facts related to a particular consultation of the system
  - At the beginning of the consultation, dynamic knowledge base is empty and grows as the consultation progresses
  - Used in decision making along with static knowledge
  - Working memory is deleted at the end of consultation

- **Inference engine**:
  - Consists inference mechanisms and control strategy
  - Inference refers the process of searching through knowledge base and derives new knowledge
  - It use Modus Ponen’s rules, if-then-else rule, etc.
  - inference mechanism uses control strategy that determines the order in which rules are applied
  - Rules are entered as a file and the inference engine uses them together to draw conclusions
  - It uses two types of reasoning mechanisms viz., Backward chaining and forward chaining

- **Knowledge acquisition**:
  - It may be done from many resources like books, reports, case studies, empirical data, domain expert, etc.
  - Knowledge can be updated (insert, delete, update)
  - This module may also have a learning module attached to various Machine Learning techniques
- **Case history**:
  - Stores the files created by inference engine using dynamic DB and is used by the learning module
  - It enriches the knowledge base

- **User interfaces**:
  - Allows user to interact with the ES and creates working knowledge for the problem to be solved
  - Presents questions and information and forwards the responses of the user to inference engine
  - Validates the answers of user as per the rules
  - Questions are generated as per the information present in the system and knowledge contained in knowledge base

- **Explanation module**:
  - Enables to query the ES about why it asked some question and how it reached some conclusion
  - These modules are called how and why modules
  - How module tells uses regarding the process through which the system has given a solution
  - Why module tells the users that the reasoning behind the solution
- **Special interfaces**:
  - Used for specialized activities
  - Used for uncertain and incomplete knowledge

3. ES vs Traditional Systems

- Basic difference is ES manipulates knowledge where as TS manipulates data
- Ess allows the use of confidences or certainty factors, which is similar to human reasoning
- E.g. if weather is humid, then it might rain probably.
  - It contains: if, then, might, probably, etc. which indicates uncertainty
- This type of reasoning can be imitated by using numeric values called confidences

- E.g. if the weather is humid, it might be concluded with 0.9 confidences that it rains
- They are meant to imitate the confidences humans use in reasoning rather than to follow the mathematical definitions
- TSs are designed to produce always correct answers, whereas ES designed to behave like human
- Ess may sometimes give incorrect results

3.1. Characteristics of ES

- **Expertise**: have high level of skill and robustness
- **Symbolic reasoning**: knowledge is represented symbolically which can be easily reasoned
- **Self knowledge**: a system should be able to explain and examine its own reasoning
- **Learning capability**: a system should learn from its mistakes and improves as it grows
- **Ability to provide training**: is capable of provide training by explaining the reasoning process
- **Predictive modelling power**: it can explain how new situation led to the change

3.2. Evaluation of ES

- Consists of performance and utility evaluations
- Performance evaluation consists of answering various questions such as:
  - Does the system make decisions that experts generally agree to?
  - Are the inference rules correct and complete?
  - Does the control strategy allow the system to consider items in the natural order that the expert prefers?
  - Are relevant questions asked to the user in proper order?
  - Are the explanation given by the ES adequate for describing How and Why conclusions?

- Utility evaluation consists of answering the questions such as:
  - Does the system help user in a significant way?
  - Are the conclusions of the system organized and ordered in a meaningful way?
  - Is the system fast enough to satisfy the user?
  - Is the user interface friendly enough?

3.3. Advantages & disadvantages of ES

- **Advantages**:
  - Provides consistent answers for repetitive decisions, processes, and tasks
  - Fastens the pace of human professional work
  - Holds and maintains significant levels of info
  - Provides improved quality of decision making
  - Domain experts are not always able to explain their logic and reasoning unlike ES
  - Leads to major internal cost savings in companies
  - Causes introduction of new products
  - Never forgets to ask a question

- **Disadvantages**:
  - Unable to make creative responses as human experts in unusual circumstances
  - Lacks common sense needed in some decision making
  - May cause errors in the knowledge base, and lead to wrong decisions
  - Cannot adopt to changing environments, unless knowledge abase in changed

4. Rule based ES

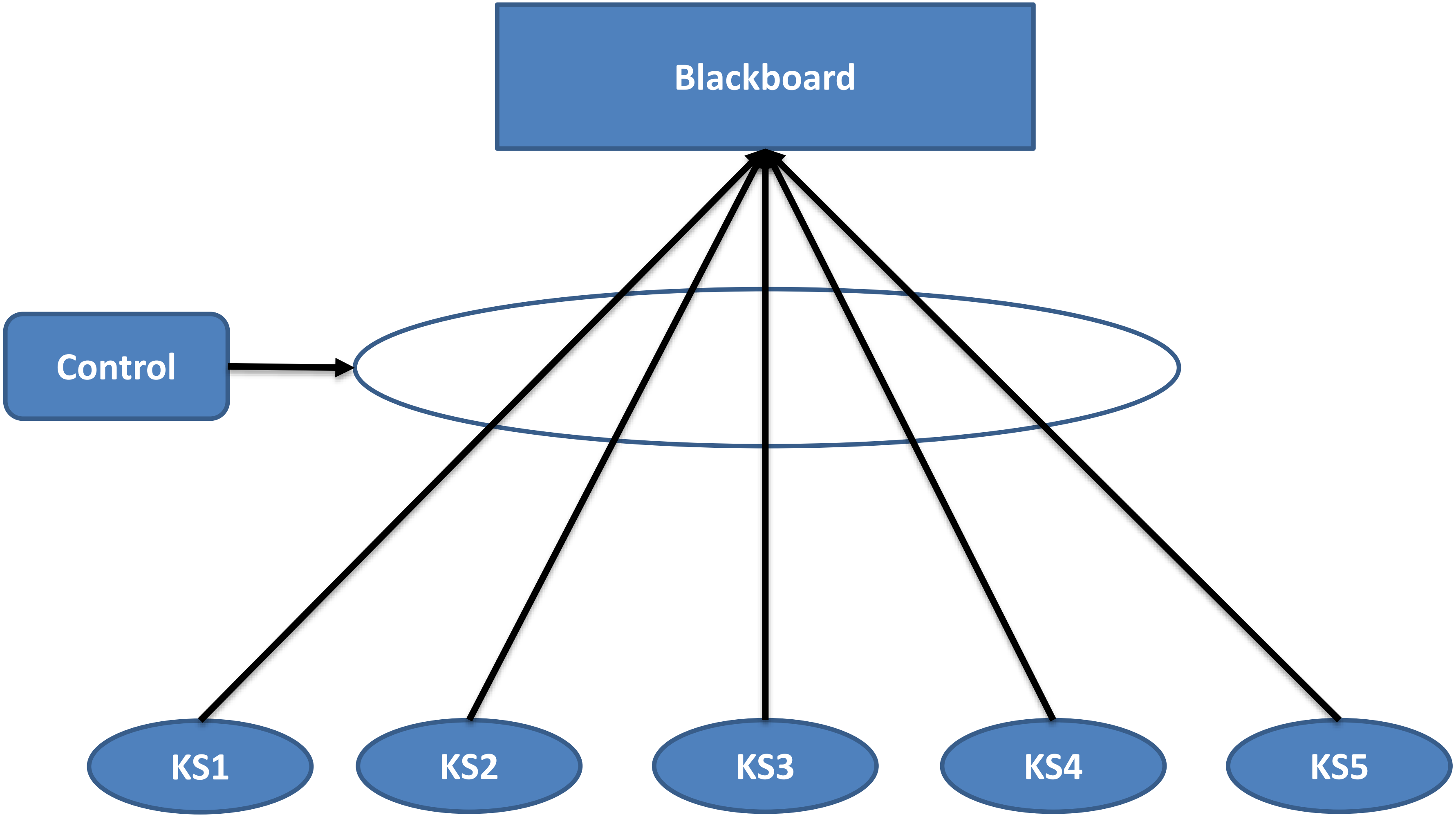
- Expert system shell in prolog
  - Here we will develop a simple expert system
  - We define a special syntax for the rules using operator declarations (:-op)
  - Using op, any standard system operator declaration can be changed or new operator can be defined by the user
  - A goal op with three arguments:
    - **:- op (Prec, Type, Atom)**

- ***Prec*** is an unsigned integer in the range [1,1200]
- ***Type*** Є {fx, fy, xfx, xfy, yfx, yfy} is a specification representing associativity
- ***Atom*** is a symbol or a name declared as an operator with precedence ***Prec*** and ***Type***
- For declaring infix operators, Type are xfx, xfy, yfx, yfy, here f - an operator and x & y are arguments
- The choice of x & y in the positions mentioned above convey associativity information
- x means the operator in the argument must have strictly lower precedence value of operator f
- y means the argument can contain operator with the same/lower precedence value than operator f

5. Blackboard Systems

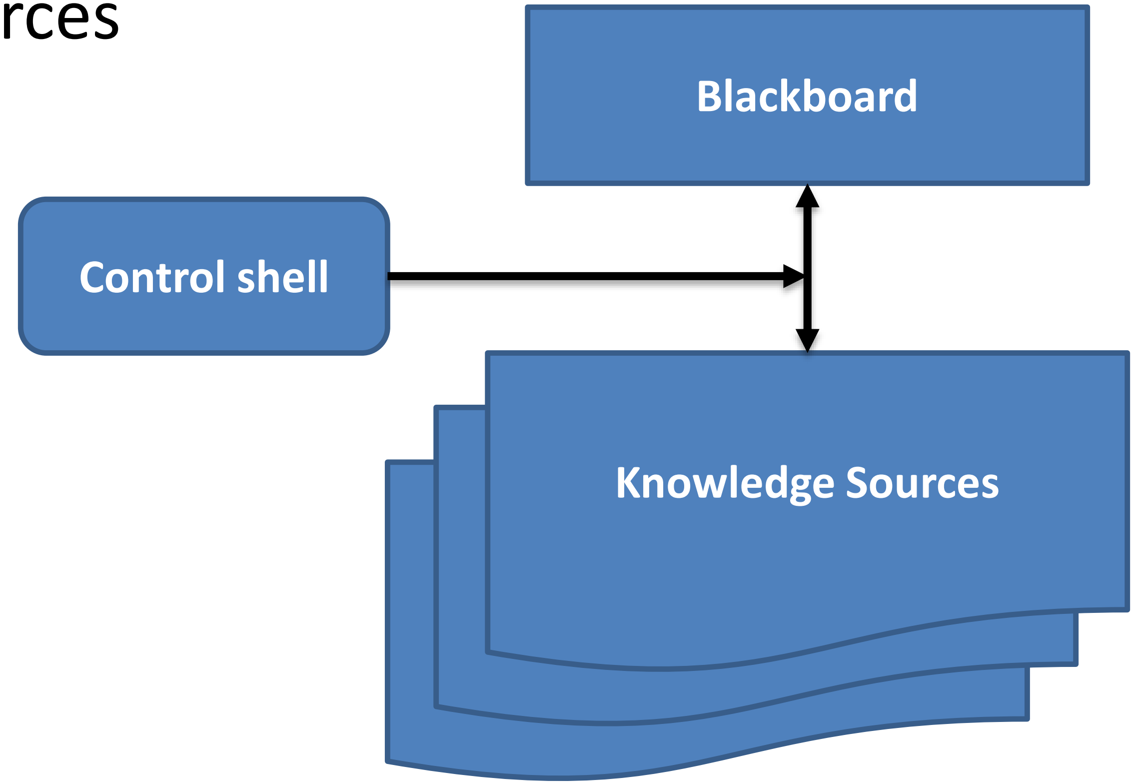
- Combining diverse s/w modules as per their data-flow requirements is a traditional way
- This approach works well when communication among the modules is static
- In dynamic environment, specific modules and their ordering will be changed
- Hence cannot be determined until specific data values are known at the time of execution
- Direct interaction induces inflexibility and the system becomes unusable

- Hence, indirect and anonymous communication among modules with the help of blackboard data repository proves to be extremely useful
- Blackboard data repository is an intermediary process, in which all processing paths are possible
- A separate moderator mechanism dynamically selects a path among all possible paths
- The information placed on the blackboard is public and is made available to all modules
- BB systems are developed to solve complex, difficult and ill-structured problems in a wide range of application areas



Indirect and anonymous communication

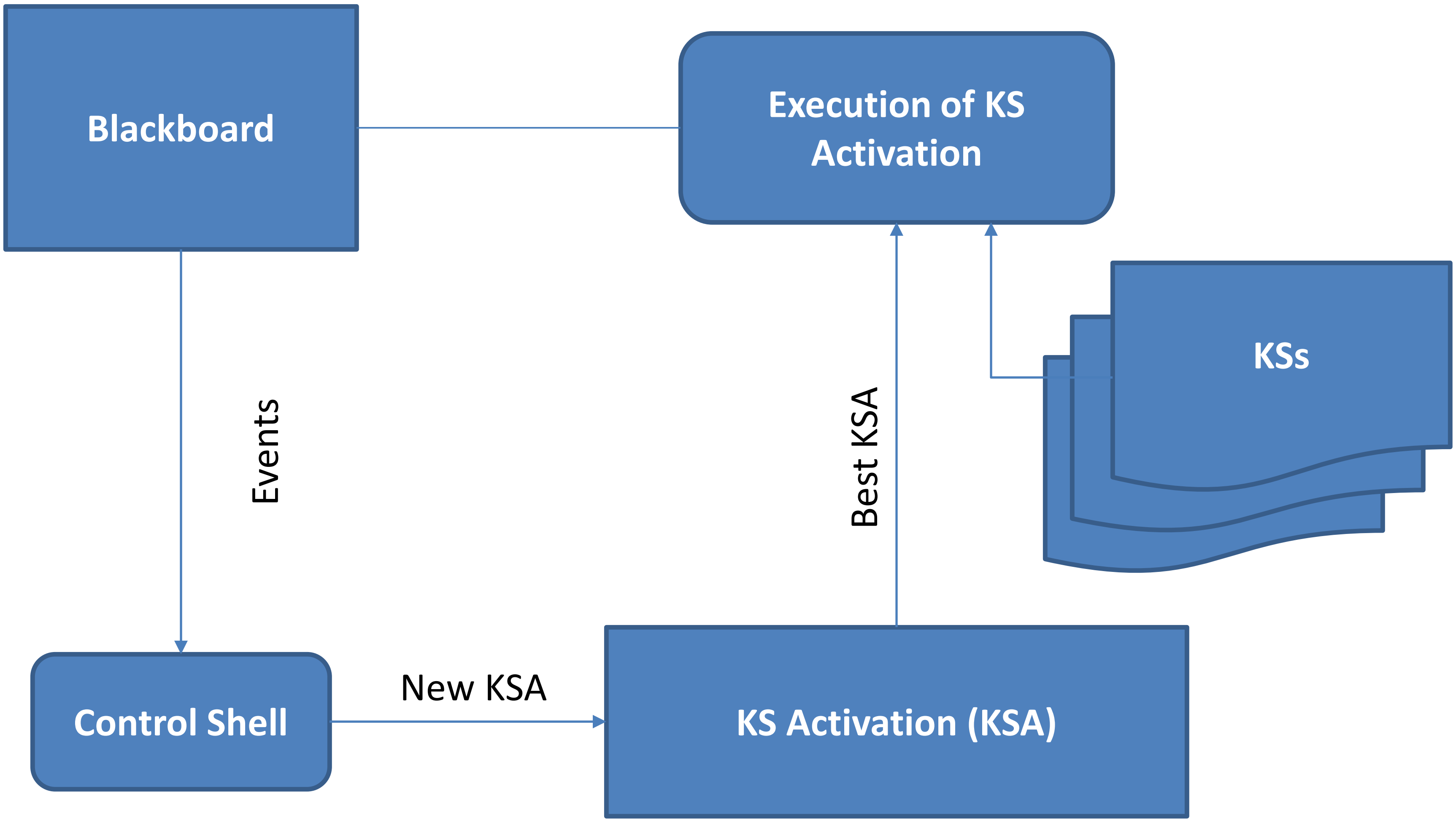
- Blackboard architecture represents controlling the knowledge bases, using independent groups of rules, called Knowledge Sources (KSs)
- BBS consists the following components:
  - Knowledge sources
  - Blackboard
  - Control shell



- **Knowledge Sources (KSs)**:
  - BB uses a functional modularization of expertise knowledge in the form of KSs
  - These are independent computational modules that contain expert knowledge to solve a problem
  - These are diverse in their internal representation and computational techniques
  - Do not interact directly with each other
  - KSs can be added, updated and deleted
  - Each KS is aware of the conditions under which it can contribute to solve problem
  - This awareness is known as triggering condition

- **Blackboard**:
  - It is a global data repository and shared DS available to all KSs
  - It contains raw input data, partial /final solutions, alternatives, control info, communication medium, buffer and a KS trigger mechanism
  - If the no.of contributions placing on the BB is large, it creates problems in locating the required info
  - This issue is solved by sub-dividing the BB in to regions; and each region links with a particular info
  - Important characteristic of BB is to integrate the contributions dynamically
  - It can retain the results of the problems that were solved earlier

- **Control component**:
  - It helps in making runtime decisions about the problem solving and the expenditure of the problem-solving resources
  - It is also called control shell sometimes
  - BBS uses the process of incremental reasoning
  - Solution to the problem is built one step at a time
  - In BBS, currently executing KS activation (KSA) generates events as it makes changes on BB
  - These events are ranked, best KSA is selected for execution
  - KS execution cycle continues until the problem is solved



Blackboard-System Control Cycle

6. Truth Maintenance Systems

- Also known as Reason Maintenance System
- TMS maintains the belief for general problem solving systems
- Inference engine explores alternatives, makes choices, and examines their consequences
- If a contradiction is noticed during this process, the TMS eliminates it by revising the KB
- TMSs work with inference engines for solving problems within large search spaces
- Inference engine & TMS together can solve problems in which algorithmic solutions do not exist

6.1. Monotonic System and Logic

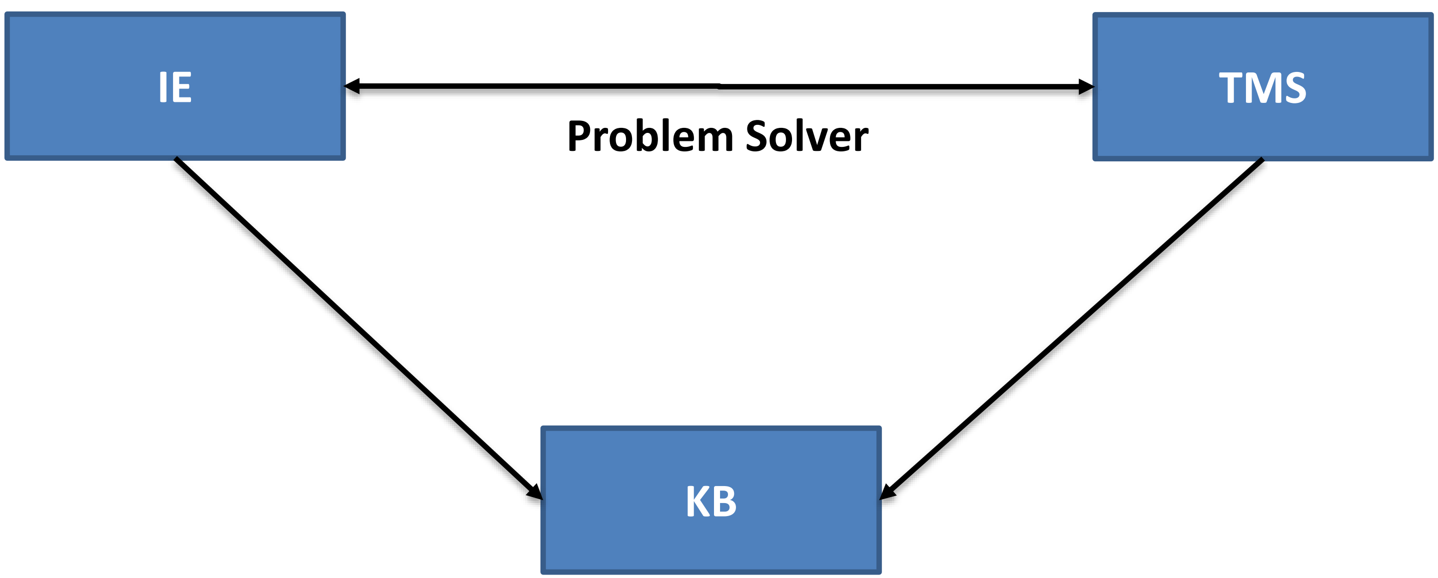
- Axioms are not allowed to change, once a fact is confirmed to be true
- It must always remain true and can never be modified
- All the axioms used are either basic or derived from other facts that are also known to be true
- Making assumptions, such as probably, likely, etc. are not allowed

- Def: if a formula is a theorem, for a formal theory, then the formula remains a theorem for any augmented theory obtained by adding axioms to the theory
  - E.g. if a property P is a theorem of T and if T is augmented to T1 by additional axioms, then P remains a theorem of T1
- If an axiom A is added to a theory T to build a theory T1, then all of the theorems of T are also theorems of T1
- In monotonic reasoning, the world of axioms continually increases in size and keeps on expanding

6.2. Non-Monotonic System and Logic

- Truths that are present in the system can be retracted whenever contradiction arise
- Hence, number of axioms can increase as well as decrease
- The system is continually updated depending on the knowledge base
- In non-monotonic logic, formula of a theorem for a formal theory, need not be a theorem for an augmented theory

- Common sense reasoning is an example of non-monotonic reasoning
- Non-monotonic reasoning is based on inferences made by applying N-M logic
- Monotonic and non-monotonic reasoning can be best implemented using TMS



Interaction b/w Different Components in a Problem Solver