MAT5024-Decision Support Systems

PPT slides for Mid-term Test

H.R. Vishwakarma
Senior Professor
School of Information Technology & Engineering
E-mail ID: hrvishwakarma@vit.ac.in
Cabin SJT Annex.101B

What is a system?

- The term system is used almost everywhere.
- *A* **system** *is an arrangement of parts or elements that together exhibit behaviour or meaning that the individual constituents do not.*
- We know what is "solar system", "nervous system", "computer systems", "software systems", etc.
- Each entity in a system is made up of a set of components that are someway connected among each other.

What is a system?

- Systems can be either physical or conceptual, or a combination of both.
- Systems in the physical universe are composed of matter and energy, may embody information encoded in matter-energy carriers, and exhibit observable behaviour.
- Conceptual systems are abstract systems of pure information, and do not directly exhibit behaviour, but exhibit “meaning”.

What is a system?

- In both cases, the system's properties (as a whole) result, or emerge from:
  - the parts or elements and their individual properties;
    AND
  - the relationships and interactions between and among the parts, the system and its environment.
- The goal of systems engineering is, then, to select, adjust, and arrange the parts or elements so as to achieve the desired whole-system properties when the system of interest is used as intended

Systems Engineering

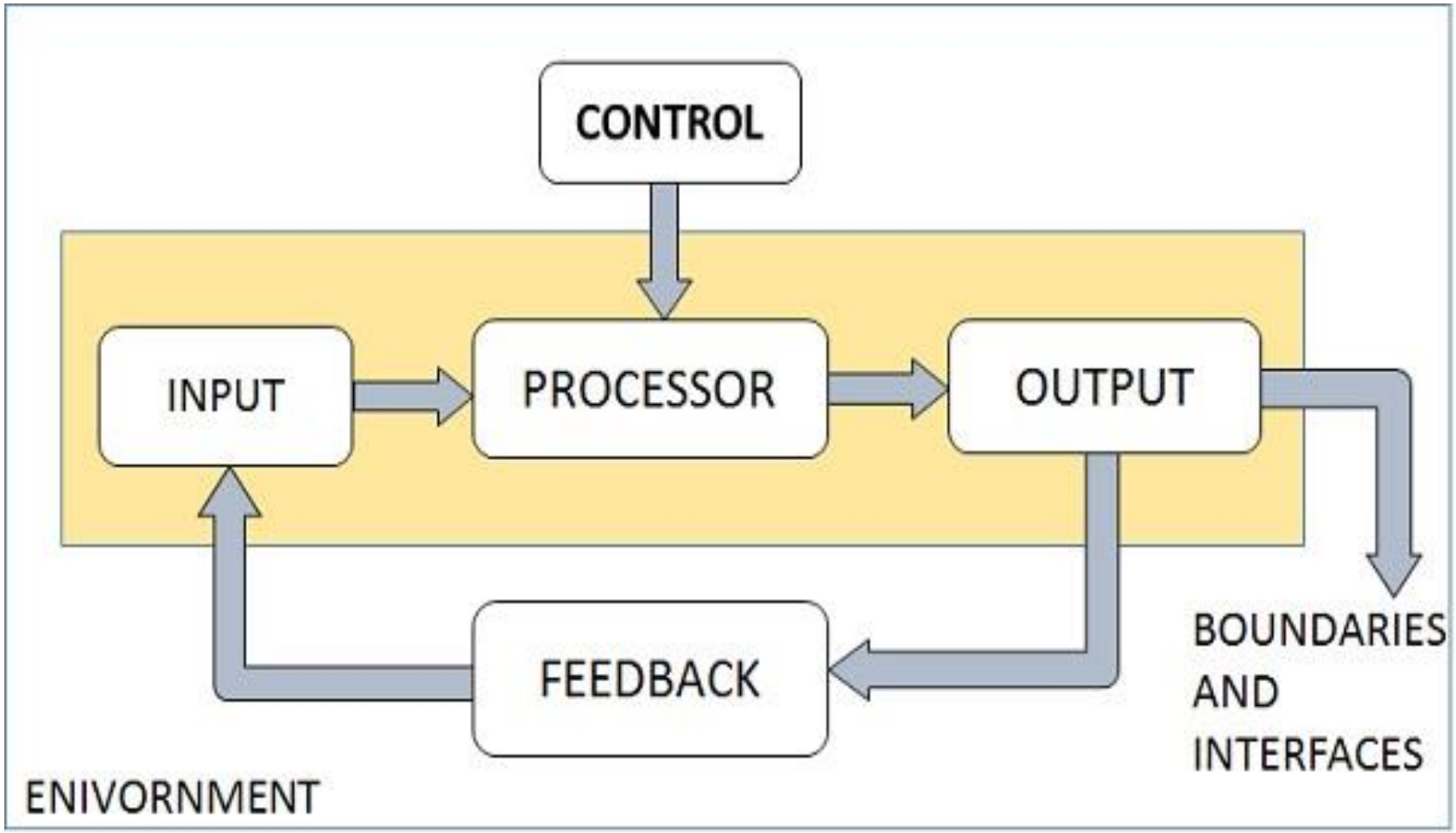
- Systems Engineering is a **transdisciplinary** and **integrative** approach
- It enables the successful
  - realization,
  - use, and
  - retirement

  of **engineered systems**
- It uses
  - **systems principles** and concepts, and
  - scientific, technological, and management methods.

Systems Engineering

- Systems principles and concepts are the ways that systems thinking and the systems sciences infuse systems engineering. A few examples of the ways are:
  - mental models,
  - system archetypes,
  - holistic thinking,
  - separation of concerns,
  - abstraction,
  - modularity and
  - encapsulation,
  - causal loop diagrams, and
  - systems mapping

Elements of a system



Elements of a system

1. Components: A component is an irreducible part or aggregation of parts that makes up a system; also called a subsystem.

2. Interrelated components: Dependence of one part of the system on one or more other system parts.

3. Boundary: A line that marks the inside and outside of a system and that sets off the system from its environment.

4. Purpose: The overall goal or function of a system.

5. Environment: Everything external to a system that interacts with the system.

Elements of a system

6. Interfaces: Point of contact where a system meets its environment or where subsystems meet each other.

7. Constraints: A limit to what a system can accomplish.

8. Input: Inputs are the information that enters into the system for processing.

9. Output: The main objective of a system is to get an output which is helpful for its user. Output is the final outcome of processing.

Characteristics of a system

- **Organization**
  - Structure and order
  - Example: Hierarchical organization in a company.
  - Computer system: organization of various components like input devices, output devices, CPU and storage devices
- **Interaction**
  - Between sub systems or the components
  - Example: the main memory holds the data that has to be operated by the ALU.

Characteristics of a system

- **Interdependence**
  - Component linkage
  - Component dependence
- **Integration**
  - How subsystems are tied together to achieve the system objective
- **Central Objective**
  - Should be known in early phases of analysis

Generic application architectures

- Application systems are designed to meet an organizational need.
- As businesses have much in common, their application systems also tend to have a common architecture that reflects the application requirements.
- A generic architecture is configured and adapted to create a system that meets specific requirements.
- Use of application architectures
  - As a starting point for architectural design.
  - As a design checklist.
  - As a way of organising the work of the development team.
  - As a means of assessing components for reuse.
  - As a vocabulary for talking about application types.

Application types

- Data processing applications
  - Data driven applications that process data in batches without explicit user intervention during the processing.
- Transaction processing applications
  - Data-centred applications that process user requests and update information in a system database.
- Event processing systems
  - Applications where system actions depend on interpreting events from the system's environment.
- Language processing systems
  - Applications where the users' intentions are specified in a formal language that is processed and interpreted by the system.

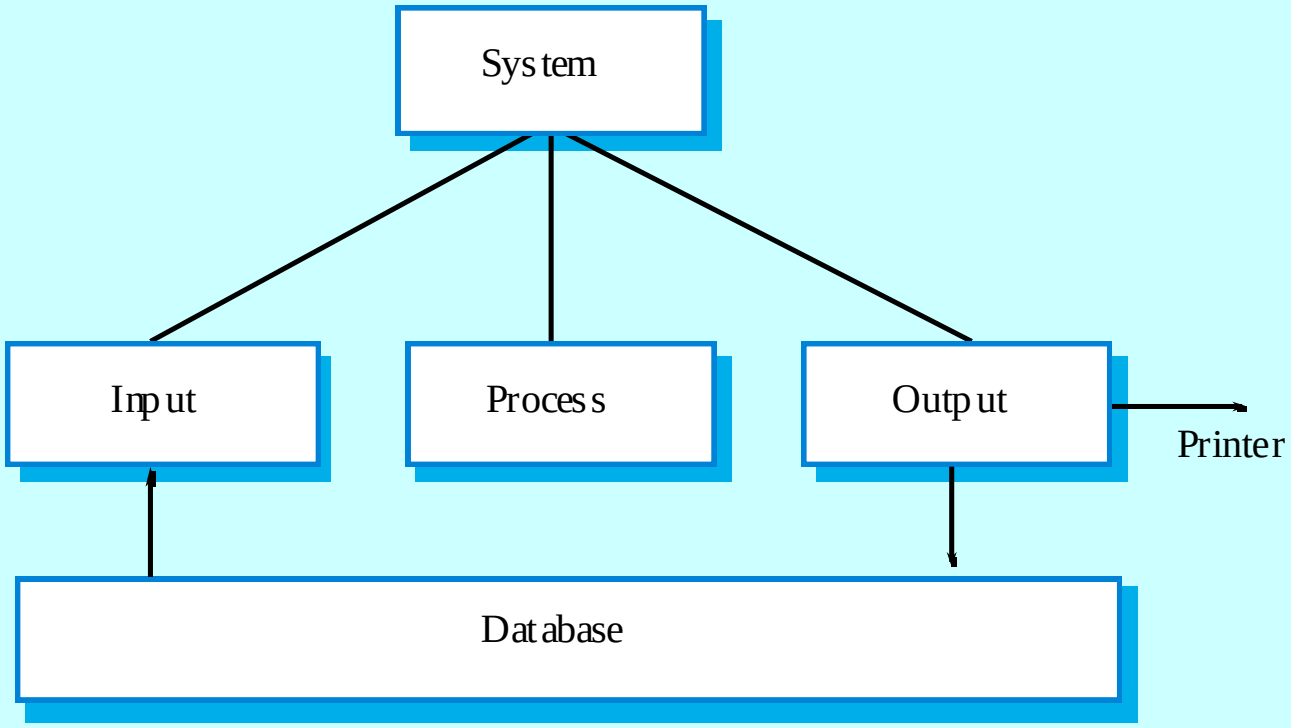
Application type examples

- Data processing systems
  - Billing systems;
  - Payroll systems.
- Transaction processing systems
  - E-commerce systems;
  - Reservation systems.
- Event processing systems
  - Word processors;
  - Real-time systems.
- Language processing systems
  - Compilers;
  - Command interpreters.

Data processing systems

- Systems that are data-centred where the databases used are usually orders of magnitude larger than the software itself.
- Data is input and output in batches
  - Input: A set of customer numbers and associated readings of an electricity meter;
  - Output: A corresponding set of bills, one for each customer number.
- Data processing systems usually have an input-process-output structure.

Input-process-output model



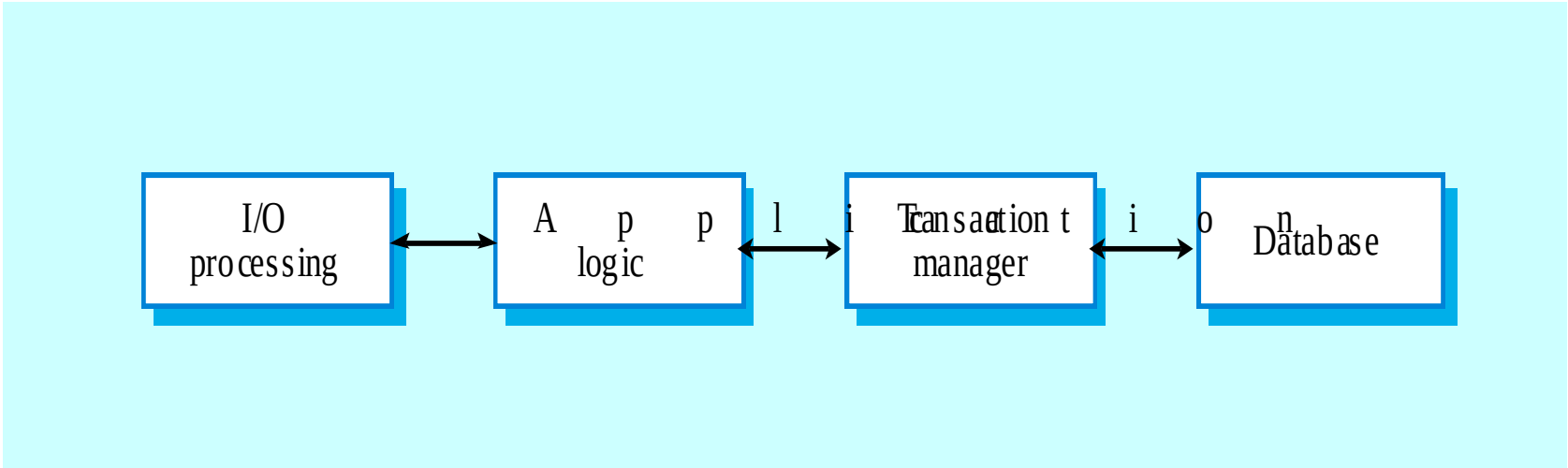
Input-process-output

- The *input* component reads data from a file or database, checks its validity and queues the valid data for processing.
- The *process* component takes a transaction from the queue (input), performs computations and creates a new record with the results of the computation.
- The *output* component reads these records, formats them accordingly and writes them to the database or sends them to a printer.

Transaction processing systems

- Process user requests for information from a database or requests to update the database.
- From a user perspective a transaction is:
  - Any coherent sequence of operations that satisfies a goal;
  - For example - find the times of flights from London to Paris.
- Users make asynchronous requests for service which are then processed by a transaction manager.

Transaction processing



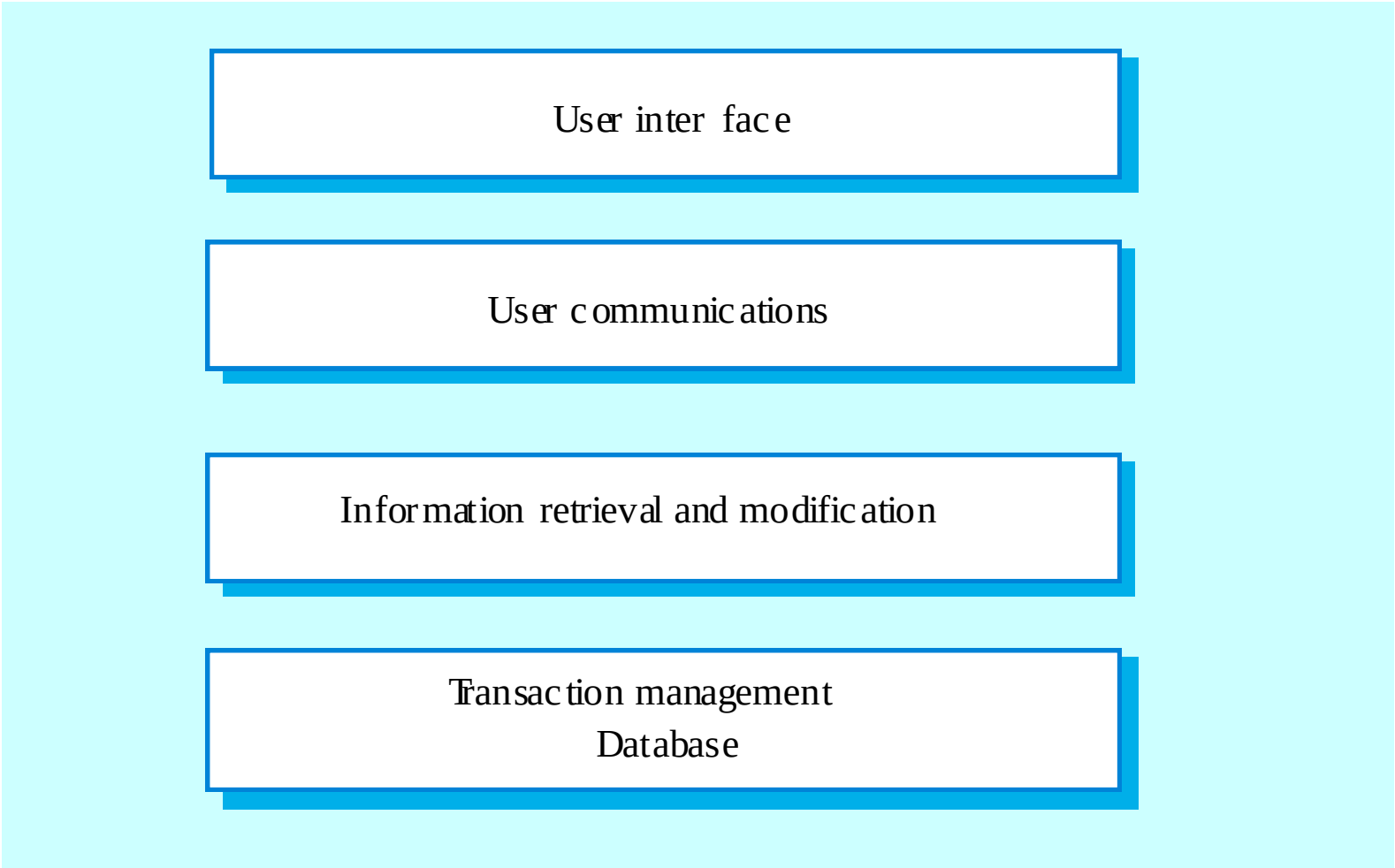
Transaction processing middleware

- Transaction management middleware or teleprocessing monitors handle communications with different terminal types (e.g. ATMs and counter terminals), serialises data and sends it for processing.
- Query processing takes place in the system database and results are sent back through the transaction manager to the user's terminal.

Information systems architecture

- Information systems have a generic architecture that can be organised as a layered architecture.
- Layers include:
  - The user interface
  - User communications
  - Information retrieval
  - System database

Information system structure



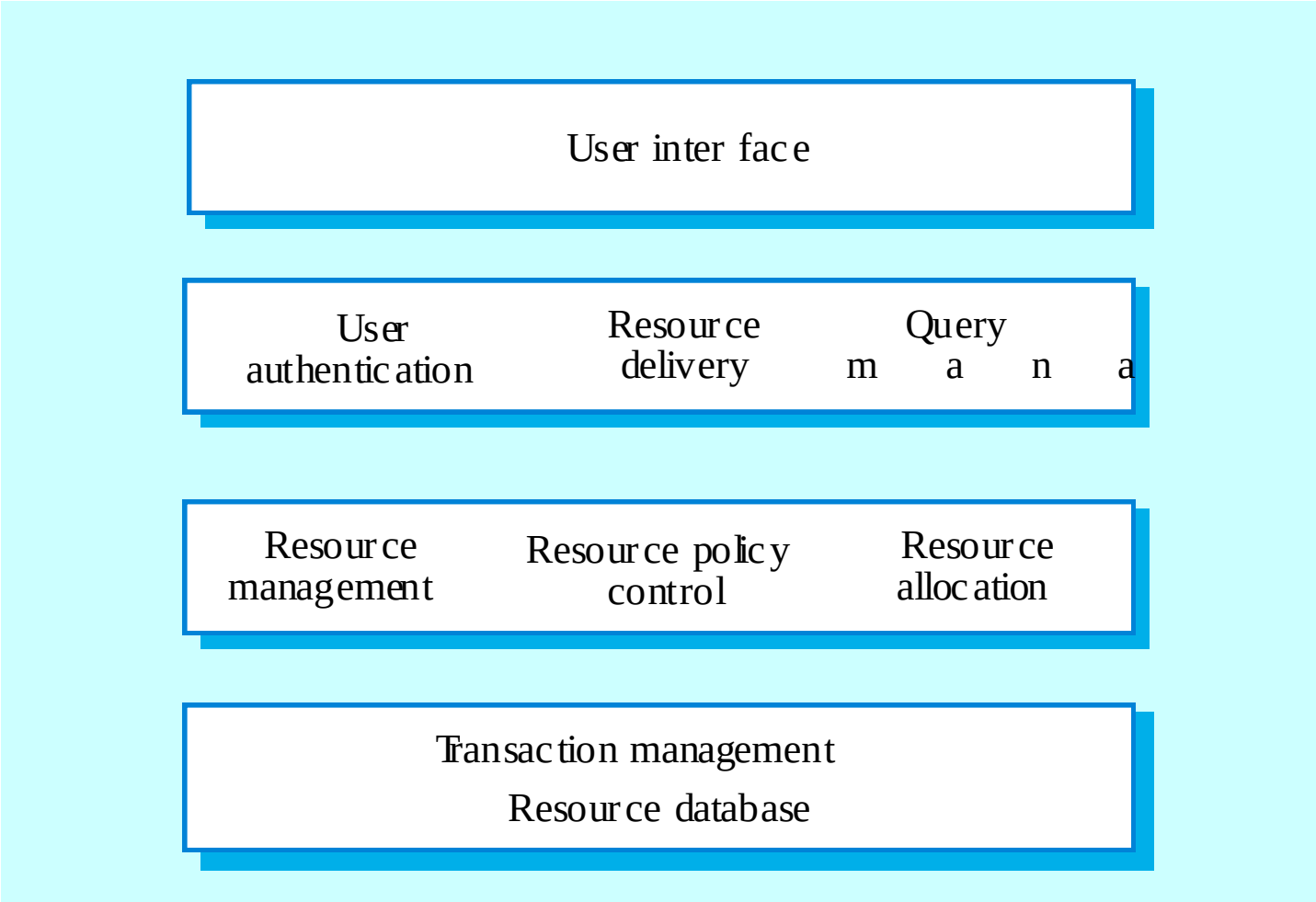
Resource allocation systems

- Systems that manage a fixed amount of some resource (football game tickets, books in a bookshop, etc.) and allocate this to users.
- Examples of resource allocation systems:
  - Timetabling systems where the resource being allocated is a time period;
  - Library systems where the resource being managed is books and other items for loan;
  - Air traffic control systems where the resource being managed is the airspace.

Resource allocation architecture

- Resource allocation systems are also layered systems that include:
  - A resource database;
  - A rule set describing how resources are allocated;
  - A resource manager;
  - A resource allocator;
  - User authentication;
  - Query management;
  - Resource delivery component;
  - User interface.

Layered resource allocation



Layered system implementation

- Each layer can be implemented as a large scale component running on a separate server. This is the most commonly used architectural model for web-based systems.
- On a single machine, the middle layers are implemented as a separate program that communicates with the database through its API.
- Fine-grain components within layers can be implemented as web services.

Event processing systems

- These systems respond to events in the system's environment.
- Their key characteristic is that event timing is unpredictable so the architecture has to be organised to handle this.
- Many common systems such as word processors, games, etc. are event processing systems.

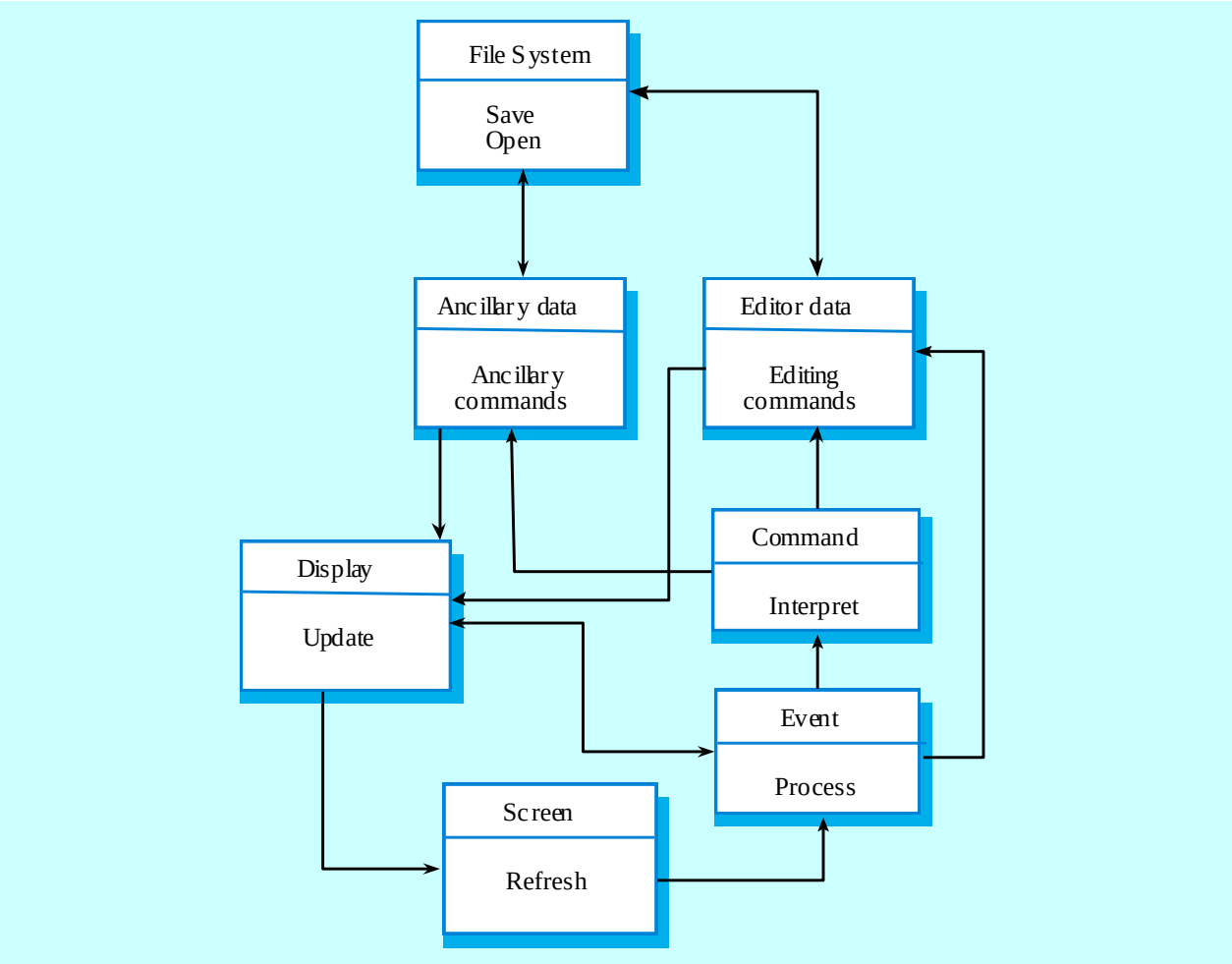
Examples of event processing systems

- Real-time systems and editing systems are the most common types of event processing system.
- Editing system characteristics:
  - Single user systems;
  - Must provide rapid feedback to user actions;
  - Organised around long transactions so may include recovery facilities.

Editing system components

- Editing systems are naturally object-oriented:
  - Screen - monitors screen memory and detects events;
  - Event - recognises events and passes them for processing;
  - Command - executes a user command;
  - Editor data - manages the editor data structure;
  - Ancillary data - manages other data such as styles and preferences;
  - File system - manages file I/O;
  - Display - updates the screen display.

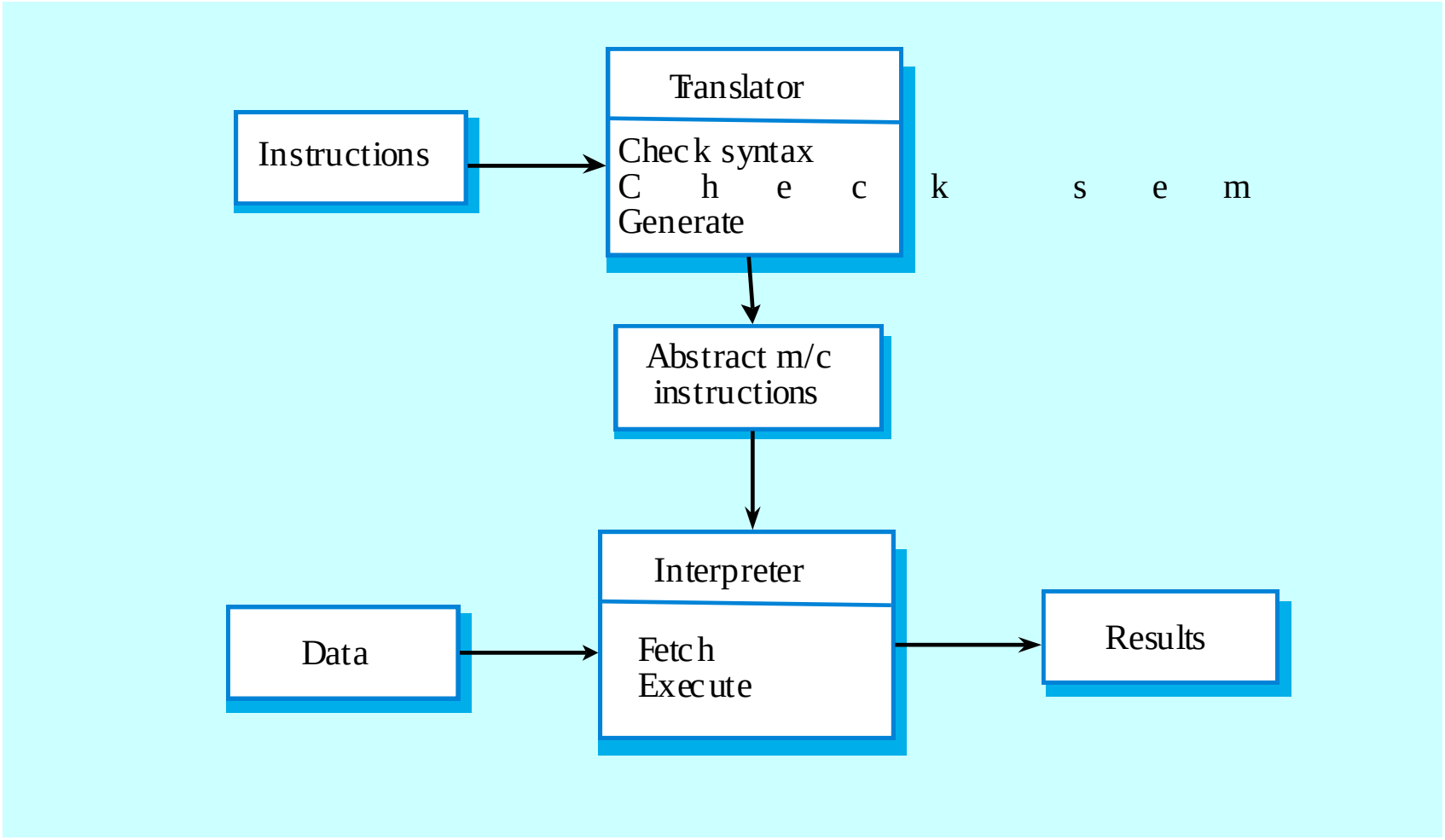
Editing system architecture



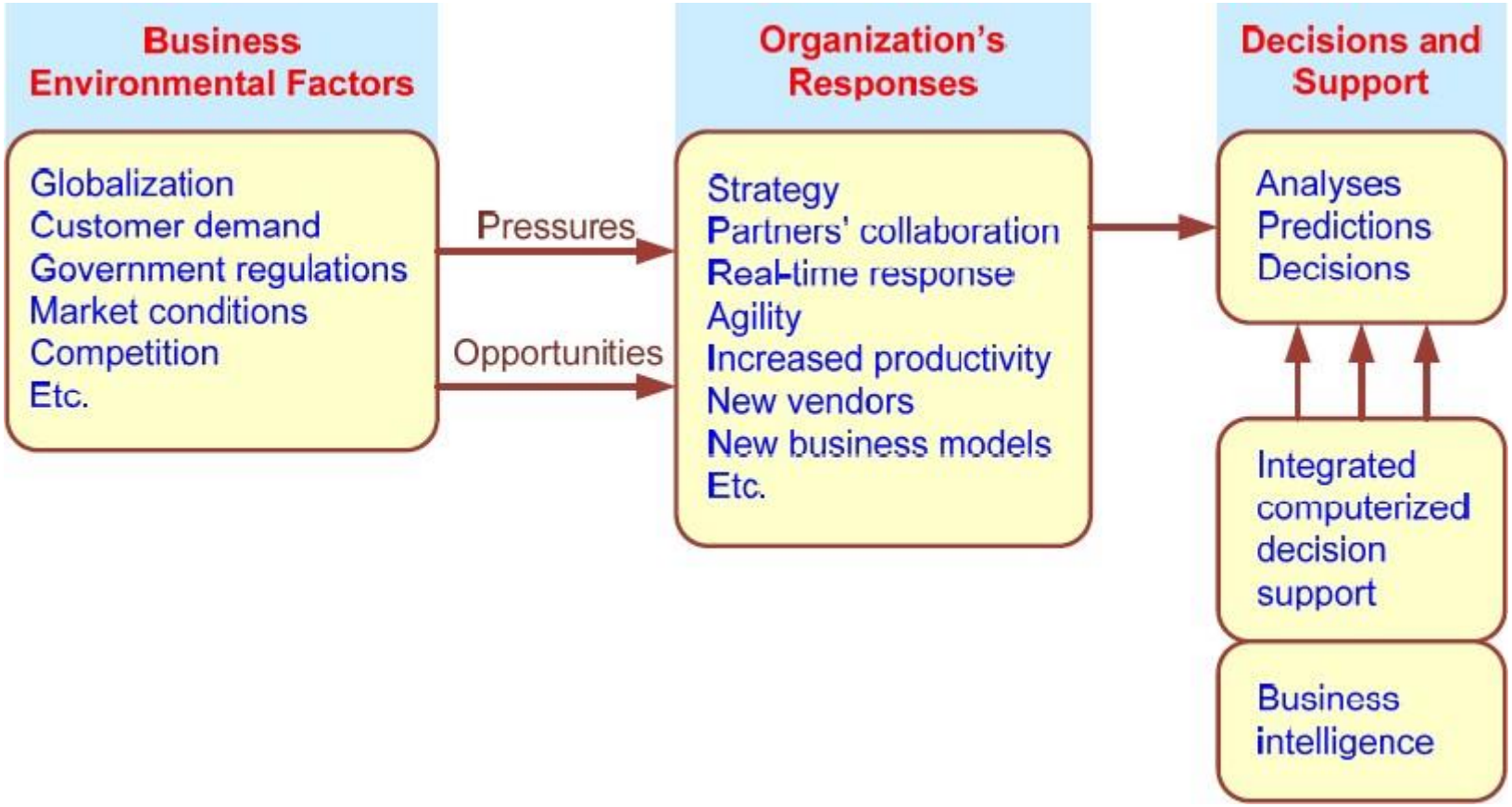
Language processing systems

- Accept a natural or artificial language as input and generate some other representation of that language.
- May include an interpreter to act on the instructions in the language that is being processed.
- Used in situations where the easiest way to solve a problem is to describe an algorithm or describe the system data
  - Meta-case tools process tool descriptions, method rules, etc and generate tools.

A language processing system



Decision Support



Organizational Responses

- Might be
  - Reactive,
  - Anticipative,
  - Adaptive, and
  - Proactive
- Managers may take actions, such as
  - Employ strategic planning.
  - Use new and innovative business models.
  - Restructure business processes.
  - Participate in business alliances.
  - Improve corporate information systems.

Decisions Everywhere

- We take numerous decisions daily
- What to eat everyday, which dress to wear, what time to start to class, when to study, etc.
- These are all trivial decisions that do not require much effort or knowledge
- But there are lots of decisions to be taken by managers and executives that require forethoughts as it could be very critical

Decision sciences

- Decision Sciences is an interdisciplinary field that draws on
  - economics,
  - machine learning,
  - statistical decision theory,
  - operations research,
  - forecasting,
  - behavioral decision theory and
  - cognitive psychology

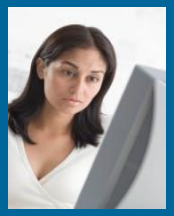
Executive Management

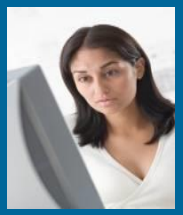
Decision Support System (DSS)
- Strategic decision making

Mid-level Management

Management Information System (MIS) - Mid-range performance control & planning

Supervisory Management





Transaction Processing system (TPS) - Daily operations control

Knowledge Work Systems (professionals)	**Group Support Systems** (any teams)	**Communication Systems** (everybody)	**Office Automation Systems** (clerks, others)

Knowledge Work Systems

Theses are specialized systems designed for knowledge workers such as engineers, scientists, and other knowledgeable individuals.

Examples of these include:

- 3D Visualization tools,
- Virtual/Augmented reality tools,
- Electronic design automation (EDA) tools,
- Simulation tools,
- Computer-Aided Design (CAD) systems,
- Computer-Aided Software Engineering (CASE) tools.

Group support systems

Group support systems are a collection of applications aimed at facilitating group work and communication similar to groupware.

These can include:

- Videoconferencing - making it possible for multiple participants to see and talk to each other in real-time.
- Group scheduling - allowing participants to share schedules and plan joint activities.
- Project management software - providing a way to plan for and keep track of group activities

Office communications systems

- Office communication systems can be any tool – digital or analogy – that helps employees communicate.
- Companies today rely on these systems to stay connected with their co-workers as well as clients and customers.
- Examples of these include:
  - phone calling, video calling, visual voicemail, email, instant messaging, live chat, and much more.

Office automation systems (OAS)

- Office automation systems (OAS) are systems that are designed to increase the productivity of clerical workers and knowledge workers and enhance communication in the workplace.
- Examples of OAS include: -
  - word processing,
  - desktop publishing,
  - workflow automation systems,
  - multimedia systems

Data Processing System (DPS)

- DPS is the manipulation of data by computers. It represents the automation of routines processing to support operations.
- Basically, DPS converts raw data into readable format which can be easily utilized by the people in the organization.
- The data processing functions are data collection, manipulation, storage as used to report and analyze business activities.
- It is oriented primarily to processing transaction data for day-to-day transactions.

Data Processing System (DPS)

- There are six stages of data processing :
  - Data Collection
  - Data Preparation
  - Data Input
  - Processing
  - Data Output
  - Data Storage

Transaction Processing System (TPS)

- A transaction processing system or TPS refers to an information processing system used for business transactions that involve the retrieval, collection, and modification of transaction data.
- It offers an execution environment that ensures data availability, security, and integrity.
- It has four components — storage, processing system, inputs, and outputs.
- There are various benefits of transaction processing systems. e.g. increased transaction speeds, better cost efficiency, automated management, and reliability.

Management Information System

- Management Information System (MIS) is an application of information technologies to programs.
- It provides managers with information and support for effective decision-making and provides the feedback on daily operations.
- The outputs or reports are usually generated through accumulation of transaction processing data.
- It ensures that appropriate data is collected from the valid sources, processed and passed to needy destinations.

Management Information System

- It satisfies the needs through query systems, analysis systems, modelling systems.
- It supports data processing functions.
- It uses an integrated database and supports a variety of functional areas.
- It provides operational, tactical and strategic levels of organization.
- It is flexible and therefore it can adapt to the changing needs of the organization

Decision Support System (DSS)

- DSS is an interactive, flexible computer based information system or sub-system intended to help decision makers.
- Decision makers use communication technologies, data, documents to identify and solve problems, complete decision process tasks and make decision.
- A DSS is an information system that aids a business in decision-making activities that require judgment, determination, and a sequence of actions.
- A DSS is either human-powered, automated, or a combination of both.

Decision Support System (DSS)

- A decision support system produces detailed information reports by gathering and analyzing data.
- In an organization, a DSS is used by the planning departments – such as the operations department – which collects data and creates a report that can be used by managers for decision-making.
- Theoretically, a DSS can be employed in various knowledge domains from an organization to forest management and the medical field.
- One of the main applications of a DSS in an organization is real-time reporting.

Decision Support System (DSS)

- DSS is an interactive computer-based systems, which helps decision makers utilize data and models to solve unstructured problems
- DSS couple the intellectual resources of individuals with the capabilities of the computer to improve the quality of decisions.
- DSS is sometimes used as an Umbrella Term
- Evolution of DSS into Business Intelligence

Business Intelligence (BI)

- Generally, BI is considered to be a set of tools and techniques applied to gather data and transform it into information that can be used in business analysis for the purposes of business development.
- BI technologies usually deal with large amounts of unstructured data via the use of data warehousing and online analytical processing (OLAP).
- BI enables organizations to integrate data across the enterprise, unlock the information and empower knowledge worker to make better (and faster) decisions – it focuses on explicit knowledge

DSS versus BI

- Decision support system:
  - A DSS helps with decision-making within an organization.
  - The primary goal of a DSS is to analyze more extensive data areas and compile that information.
  - A DSS produces far-reaching data.
- Business Intelligence (BI)
  - BI includes various software programs that aid with the organization and management of data and other valuable information within an organization.
  - BI involves data mining, online processing, reporting, and querying.

Decision Support System

- A decision support system (DSS) is **designed, developed and deployed to support determinations, judgments, and courses of action in an organization or a business.**
- A DSS sifts through and analyzes massive amounts of data, compiling comprehensive information that can be used to solve problems and in decision-making.
- Typical information used by a DSS includes target or projected revenue, sales figures or past ones from different time periods, and other inventory- or operations-related data
- There are two types of decisions - programmed and non-programmed decisions.

Decision Support System

- Programmed decisions are basically automated processes, general routine work, where –
  1. These decisions have been taken several times.
  2. These decisions follow some guidelines or rules
- Non-programmed decisions occur in unusual and non-addressed situations, so
  1. It would be a new decision.
  2. There will not be any rules to follow.
  3. These decisions are made based on available information.
  4. These decisions are based on the manger's discretion, instinct, perception and judgment
- Decision support systems generally involve non-programmed decisions. Adaptability and flexibility are the key attributes.

Types of Decision Support Systems

- Communication-Driven DSS
- Data-Driven DSS
- Document-Driven DSS
- Knowledge-Driven DSS
- Model-Driven DSS

Types of Decision Support Systems

- Communication-Driven DSS
  - These are used by internal teams, including partners.
  - The main purpose is to help communication between different levels of operations.
  - Examples include chats and instant messaging software, online collaboration, and net-meeting systems.

Types of Decision Support Systems

- Data-Driven DSS
  - These are used by managers, staff, and product/service suppliers.
  - These are typically used to query a database or data warehouse and find specific data bits for particular purposes.
  - A Data-Driven DSS is controlled by a mainframe system, client/server link, or the cloud.
  - A computer-based database is one example.

Types of Decision Support Systems

- Document-Driven DSS
  - Document-driven DSSs are more common and used by various user groups.
  - The purpose of these systems is to search web pages and look for documents on specific keywords or search terms.

Types of Decision Support Systems

- Knowledge-Driven DSS
  - Knowledge-driven DSS or Knowledge-based systems cover a broad range of designs with users inside and outside the organization.
  - They systems can be used by anyone who requires knowledge about a particular subject, for example, business customers.
  - A Knowledge-Driven DSS is used to advise managers or customers about products/services.

Types of Decision Support Systems

- Model-Driven DSS
  - These DSSs are complex systems that help analyze decisions or choose between numerous options.
  - These systems are used by managers and staff members of a business or people who interact with the organization for several purposes, such as scheduling, decision analyses, etc.
  - These can be controlled through software/ hardware stored on PCs, client/server systems, or the cloud.

Methods of Decision Making and Problem Solving

- Problem-solving and decision-making are two aspects of critical thinking.
- The above go hand in hand because without solving a problem, it becomes difficult to make a decision. And even vice versa in some cases.
- Two types of problem-solvers and decision-makers:
  - The first type is the intuitive one. They usually go with their gut feeling or intuition and put their emotions above all.
  - The second type is the ones who think more rationally. They have a systematic approach to everything.

Methods of Decision Making and Problem Solving

- Problem-solving is a method where we identify the best possible solutions to a given problem.
- When we are solving a problem, we also end up making a few decisions necessary for the right outcome.
- The main aim of problem-solving is to come up with the right solution.
- Problem-solving involves gathering up all the right facts, finding related causes, asking the right questions, and more.

Methods of Decision Making and Problem Solving

- Decision-making is when we choose a solution as per our judgment, facts, knowledge, and more.
- The main criteria of decision-making are to ensure we avoid any difficulties that may otherwise arise and to identify the right opportunities along the way.
- Generally, both problem-solving and decision-making are used together.
- The ability to solve problems and make decisions quickly and effectively can mean the difference between success and failure.

Methods of Decision Making and Problem Solving

- Two main approaches to problem-solving and decision-making:
  - vertical thinking and
  - horizontal thinking.
- Both approaches have strengths and weaknesses, so understanding the differences between them can help we apply the right method at the right time.

Methods of Decision Making and Problem Solving

- Vertical Thinking
  - The vertical thinking is about making a well-informed decision based on a thorough analysis of the data.
  - Vertical thinking is especially useful in situations where there is a clear goal and a need for a precise, data-driven approach.
  - Experts often use it in fields like finance, where decisions depend heavily on facts and figures.

Methods of Decision Making and Problem Solving

- Horizontal Thinking
  - This is horizontal thinking: working with a team to generate a variety of ideas and consider different perspectives to find an innovative solution.
  - Horizontal thinking is a great approach for problem-solving when the problem is complex and there may be multiple solutions or approaches.
  - Creative professionals, especially in marketing, advertising and designing, highly value this approach.

Managerial Decision Making

- Problem-solving often involves decision-making, and decision making is especially important for management and leadership.
- Management is a process by which organizational goals are achieved by using resources.
  - Inputs: resources
  - Output: attainment of goals
  - Measure of success: outputs / inputs
- Managing ≅ Decision Making
- Decision making: selecting the best solution from two or more alternatives

Most important roles of managers

- Interpersonal
  - Figurehead
  - Leader
  - Liaison
- Informational
  - Monitor
  - Disseminator
  - Spokesperson
- Decisional
  - Entrepreneur
  - Disturbance handler
  - Resource allocator
  - Negotiator

Decision-making process

- Managers make decisions using a four-step process (a.k.a. the scientific approach)
  1. Define the problem (or opportunity)
  2. Construct a model that describes the real-world problem.
  3. Identify possible solutions to the modeled problem and evaluate the solutions.
  4. Compare, choose, and recommend a potential solution to the problem.

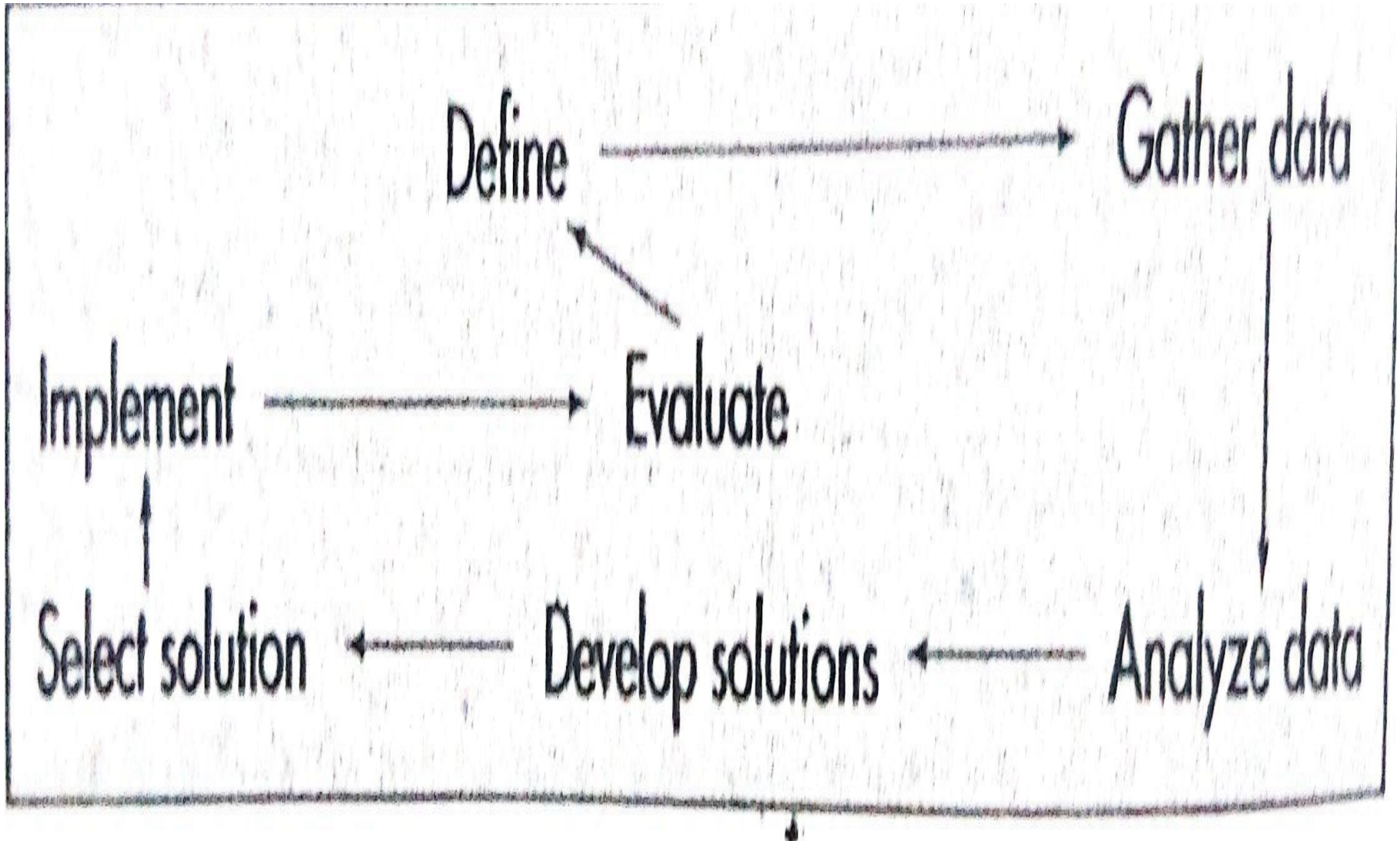
Decision-making necessities

- Group communication and collaboration
- Improved data management
- Managing data warehouses and Big Data
- Analytical support
- Overcoming cognitive limits in processing and storing information
- Knowledge management
- Anywhere, anytime support

Decision Making and Problem Solving

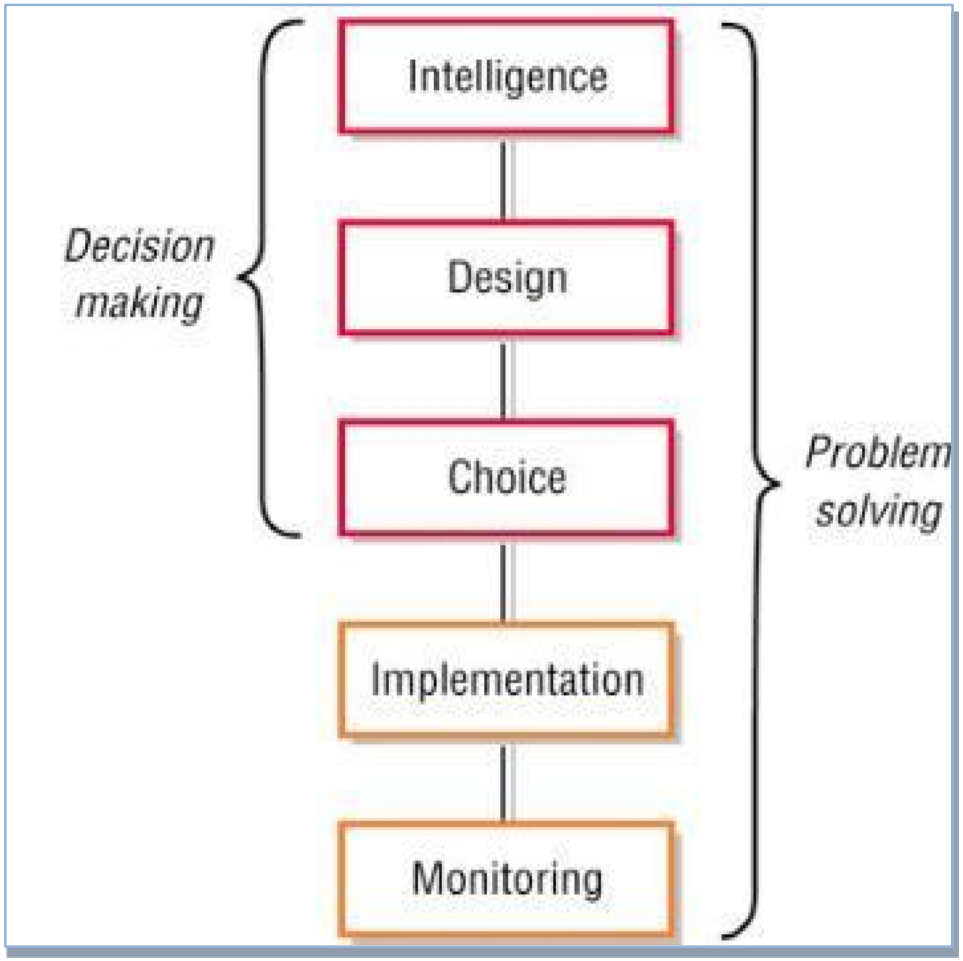
- Problem solving process?
  - It is the systematic process that focuses on analyzing problem to reach for root cause and selecting appropriate solutions.
- Problem solving includes Decision making process
- Decision making is a complex, cognitive process often defined as “process of choosing on course of action over another”

Problem solving process

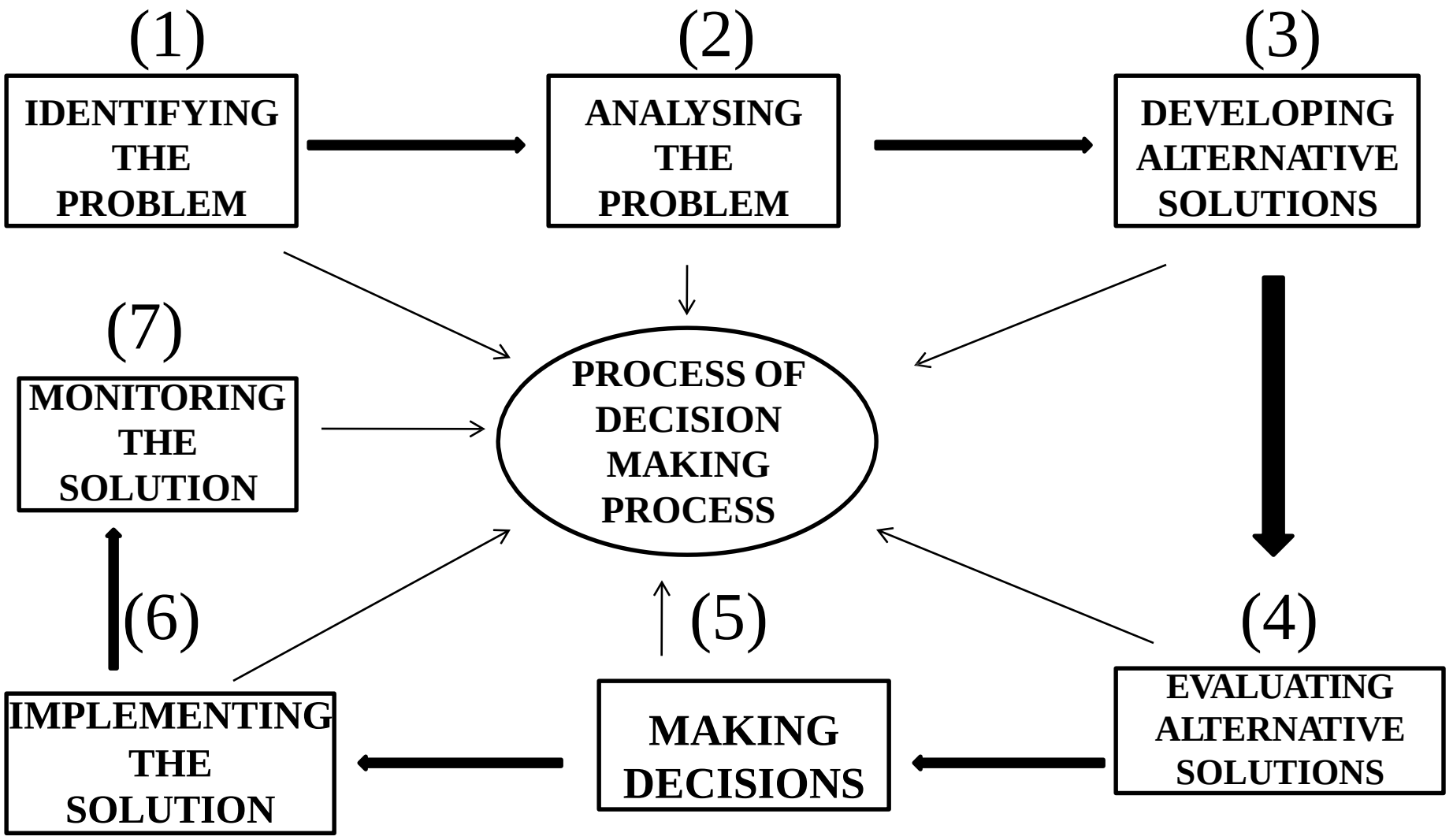


Decision making

- It is the key component of the problem-solving process that takes place in three stages:
  - intelligence,
  - design, and
  - choice



DECISION MAKING PROCESS



Decision making process

1. **Identifying or Defining the Problem:**
   - The first step towards a decision making process is to define the problem.
   - Obviously, there would be no need to make a decision without having a problem. So, the first thing one has to do is to state the underlying problem that has to be solved.
   - We also have to clearly state the outcome or goal that we desire after we have made the decision.
   - This is a good way to start, because stating our goals would help us in clarifying our thoughts.

Decision making process

2. Analyzing the Problem:

- The most important aspect of this step is to find the limiting factor or the critical factor.
- This limiting factor has to be changed or removed so that the decision or problem solving can be taken and the desired goal is accomplished.
- The ability of manager to locate this limiting factor in the problem is crucial to his success in finding optimum solution to the problem.
- e.g. if sales of a business are reduced ,the sales manager has to find the limiting factor which could be ineffective advertisement, faulty distribution, price, after sales services.

Decision making process

3. Developing Alternatives:

- The situation of making a decision arises because there are many alternatives available for it. Hence, the next step after defining the main problem would be to state out the alternatives available for that particular situation.
- Here, we do not have to restrict ourselves to think about the very obvious options, rather we can use our creative skills and come out with alternatives that may look a little irrelevant.
- This is important because sometimes solutions can come out from these out-of-the-box ideas.
- We would also have to do adequate research to come up with the necessary facts that would aid in solving the problem.

Decision making process

4. Evaluating the Alternatives:

- This can be said to be the one of the most important stages of the decision making process.
- This is the stage where we have to analyze each alternative we have come up with.
- We have to find out the advantages and disadvantages of each option.
- This can be done as per the research we have done on that particular alternative.
- At this stage, we can also filter out the options that we think are impossible or do not serve our purpose.
- Rating each option with a numerical digit would also help in the filtration process.

Decision making process

5. Making the Decision:

- This is the stage where the hard work we have put in analyzing would lead to.
- The evaluation process would help us in looking at the available options clearly and we have to pick which we think is the most applicable.
- We can also club some of the alternatives to come out with a better solution instead of just picking out any one of them.

Decision making process

6. Implementing the Solution:

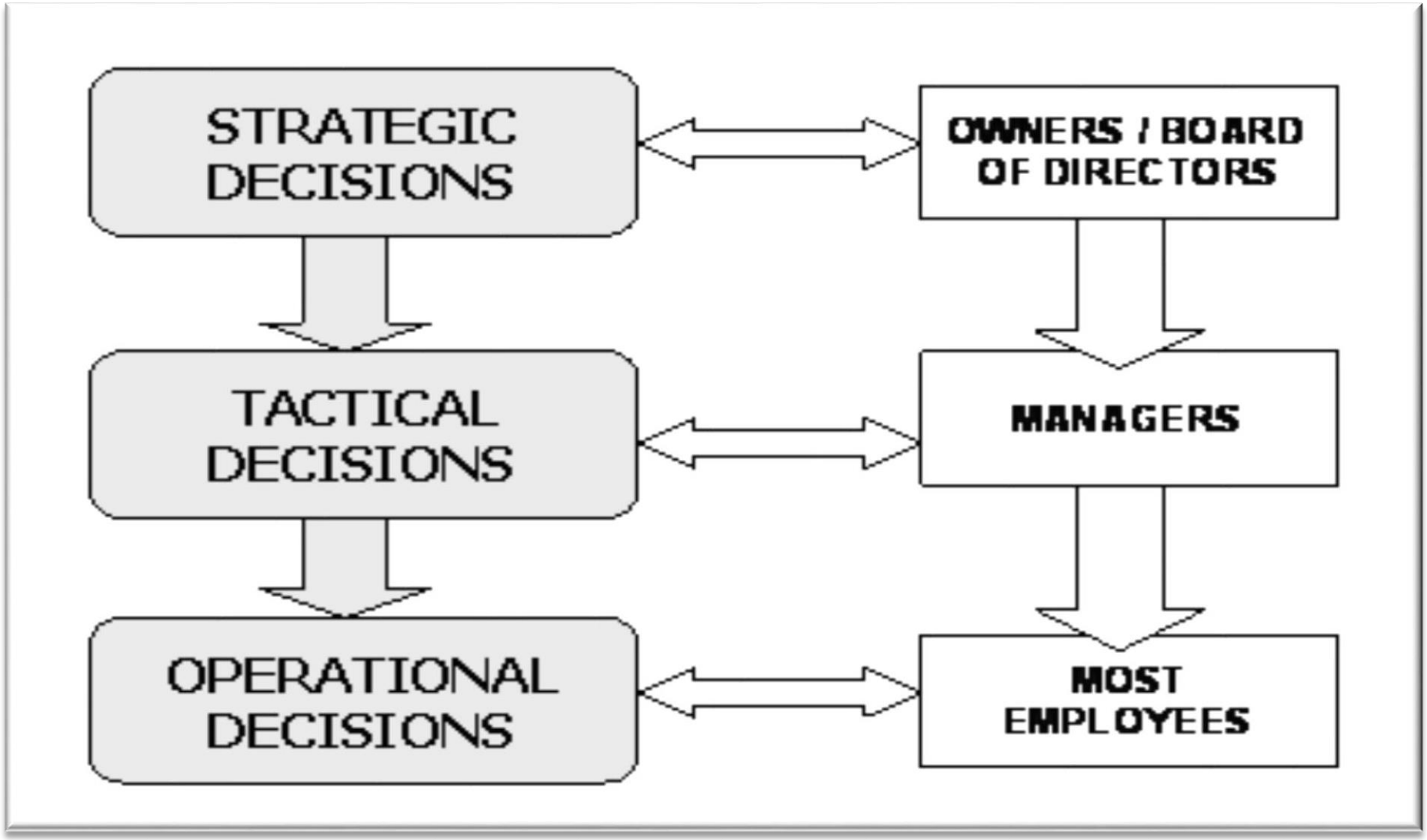
- The next obvious step after choosing an option would be implementing the solution. Just making the decision would not give the result we want.
- Rather, we have to carry out on the decision we have made.
- This is a very crucial step because all the people involved in the implementation of a solution should know about the implications of making the decision.
- This is very essential for the decision to give successful results.

Decision making process

7. Monitoring the Solution:

- Just making the decision and implementing it is not the end of the decision making process, it is very important to monitor our decision regularly.
- At this stage, we have to keep a close eye on the progress of the solution taken and also whether it has led to the results we expected.

TYPES OF DECISIONS



Strategic decisions

- Strategic decisions deals with the big picture of the business.
- The focus of strategic decisions is typically external to the business and usually future oriented.
- Strategic decision- making creates the forward thrust in the business.
- **It includes decisions about:**
  - What business are we in?
  - What is our vision for the business?
  - What's our business' identity?
  - What do we stand for?
  - Which direction is the business headed?
  - How will the business compete?

Tactical decisions

- Tactical decisions involve the establishment of key initiatives to achieve the overall strategy.
- For example, if we have decided to be the Number 1 provider in our market (a strategic decision) then we will develop tactics (e.g. implement a marketing system) to achieve that outcome.
- In a small business we may have 4 or 5 key tactics that we are going to use to achieve our overall strategy.

Tactical decisions

- Tactical decision-making can sometimes be over-looked yet it is the glue that creates a strong connection between long-term vision and day-to-day activities.
- Usually tactical decision-making is the domain of 'mission' statements.
- We need to think in terms of the battlefields from which the term has emerged.
- The overall strategy, that is, what the army is there to do, is to win the war.
- Then we have a number of 'missions' to send troops on, preferably diplomatic ones, the cumulative effect of which is intended to win the war.

Operational decisions

- Operational decisions determine how activities actually get done.
- These are typically the 'grass roots' decisions about who is going to do what and when.
- Examples of these include:
  - How will we spend our money this month?
  - How will we service that client?
  - What is our procedure for delivering an order?
  - Who will be doing quality control?

Operational versus Strategic Decisions

Basis	Operational / Programmed Decisions	Strategic / Non-Programmed Decisions
1. Meaning	Repetitive decision that can be handled by a routine approach.	Decisions are unique and non-recurring
2. Simple versus Difficult	Decision- making is relatively simple and tends to rely heavily on previous solutions.	Decision-making is relatively difficult and is different from the previous organisational decisions.
3. Horion	Considers short-term horizon and feedback is important.	Considers long-term horizon.

Basis	Operational / Programmed Decisions	Strategic / Non-Programmed Decisions
4. Uncertainities	Ignores uncertainties-considering as minor obstacles, which can be overcome through experience.	Account for uncertainty- decision maker should have knowledge of risk return trade off.
5. Action versus Thinking Oriented	Decisions are thinking-oriented and mistakes can lead to jeopardy.	Decisions are action-oriented and mistakes are too costly.
6. Resources	Less resources required	More resources required
7. Management	Middle/Lower-level	Top-level
8. Problem type	Problem well-structured.	Problem ill-structured.

Decision Support Framework

Type of Decision	Type of Control: Operational Control	Type of Control: Managerial Control	Type of Control: Strategic Planning
Structured	1 Accounts receivable Accounts payable Order entry	2 Budget analysis Short-term forecasting Personnel reports Make-or-buy	3 Financial management Investment portfolio Warehouse location Distribution systems
Semistructured	4 Production scheduling Inventory control	5 Credit evaluation Budget preparation Plant layout Project scheduling Reward system design Inventory categorization	6 Building a new plant Mergers & acquisitions New product planning Compensation planning Quality assurance HR policies Inventory planning
Unstructured	7 Buying software Approving loans Operating a help desk Selecting a cover for a magazine	8 Negotiating Recruiting an executive Buying hardware Lobbying	9 R & D planning New tech. development Social responsibility planning

Decision Support Systems (DSS)

- Decision support systems are interactive computer-based systems, which help decision makers utilize data and models to solve unstructured problems
- These systems couple the intellectual resources of individuals with the capabilities of the computer to improve the quality of decisions.
- Decision support system is sometimes used as an Umbrella Term
- Decision support system may evolve into Business Intelligence

Important System Concepts

- Systems analysts need to know several other important systems concepts:
  - Decomposition
  - Modularity
  - Coupling
  - Cohesion

Important System Concepts

- Decomposition
  - It is the process of breaking down a system into its smaller components.
  - These components may themselves be systems (subsystems) and can be broken down into their components as well.
  - Decomposing a system also allows us to focus on one particular part of a system, making it easier to think of how to modify that one part independently of the entire system.

Important System Concepts

- Decomposition is a technique that allows the systems analyst to:
  - Break a system into small, manageable, and understandable subsystems
  - Focus attention on one area (subsystem) at a time, without interference from other areas.
  - Concentrate on the part of the system pertinent to a particular group of users, without confusing users with unnecessary details
  - Build different parts of the system at independent times and have the help of different analysts

Important System Concepts

- Modularity
  - It is a direct result of decomposition.
  - It refers to dividing a system into chunks or modules of a relatively uniform size.
  - Modules can represent a system simply, making it easier to understand and easier to redesign and rebuild.
  - For example, each of the separate subsystem modules for the MP3 player shows how decomposition makes it easier to understand the overall system.

Important System Concepts

- Coupling
  - It means that subsystems are dependent on each other.
  - In other words, subsystems should be as independent as possible.
  - If one subsystem fails and other subsystems are highly dependent on it, the others will either fail themselves or have problems functioning.
  - For example, components of a portable MP3 player are tightly coupled.
  - The best example is the control system, made up of the printed circuit board and its chips.

Important System Concepts

- Coupling
  - The best example is the control system, made up of the printed circuit board and its chips.
  - Every function the MP3 player can perform is enabled by the board and the chips.
  - A failure in one part of the circuit board would typically lead to replacing the entire board rather than attempting to isolate the problem on the board and fix it.

Important System Concepts

- Coupling
  - Even though repairing a circuit board in an MP3 player is certainly possible, it is typically not cost-effective; the cost of the labor expended to diagnose and fix the problem may be worth more than the value of the circuit board itself.
  - In a home stereo system, the components are loosely coupled because the subsystems, such as the speakers, the amplifier, the receiver, and the CD player, are all physically separate and function independently.
  - If the amplifier in a home stereo system fails, only the amplifier needs to be repaired.

Important System Concepts

- Cohesion
  - It is defined as the extent to which a subsystem performs a single function.
  - In the MP3 player example, supplying power is a single function.

Designing and Developing DSS

- An effective DSS provides users with unbiased data analysis, real time monitoring and rich reporting, supporting the users make an informed decision in the least possible time span.
- A meticulously designed DSS makes use of analytical models, various statistical and econometric tools and of course, human intelligence and insights to support decision making.
- This proves the importance of technology, methodology and approach behind designing and development of a decision support system.

Designing and Developing DSS

- Designers and developers need to be extremely careful in selecting the DSS technology and developmental methodology.
- They need to get the basics right, in order to get a reliable tool to support diverse information and decision-making needs.
- Because designing and development of a decision support system is a complex process, you may face a number of issues, such as

Designing and Developing DSS

- Issues or Challenges:
  - Different perspectives of DSS among stakeholders
  - Identification of specific requirements
  - Technology selection
  - Approach to software design and development
- Most of these issues can be resolved by:
  - Identifying clear agendas
  - Brainstorming with team members
  - Communicating the expectations clearly to developers
  - Educating about various technologies for DSS development
  - Trying to reach at a common platform with the developers

Designing and Developing DSS

- Pre-Design Diagnosis help gains clarity on
  - Problem identification process
  - Data collection techniques
  - How decisions are made
  - Who all are involved in decision making
  - Norms pertaining to decision making in an organization

Designing and Developing DSS

- A diagnostic study of decision making process involves
  - Defining the types of decisions
  - Charting out the formal decision making process basis
  - Identifying the primary role of decision maker
  - Interviewing decision makers
  - Assessing the effectiveness of data collection techniques
  - Evaluating the performance of decision making process
  - Need for computerized aid to support these decisions
  - Identifying what needs to be improved
  - Determining what will remain unchanged

Designing and Developing DSS

- A diagnostic study of decision making proves to be very useful to assess the overall process.
- It helps in
  - redesigning the decision making process and
  - charting out the objectives for DSS development,
  - listing out functionality and expectations from a decision support system
- Testing feasibility of a proposed decision support system succeeds the diagnostic study of a decision making process.

Designing and Developing DSS

- The feasibility study aims to:
  - Uncover viability or achievability of a proposed DSS
  - Discover the target users of a proposed decision support system
  - Identify opportunities and threats to a proposed DSS
  - Determine if a proposed DSS with desired features falls within the budget of the organization
  - Determine the technologies that can be used to develop a DSS
  - Measure how well a proposed DSS will be able to solve problems
  - Identify time available to build the system
  - Uncover the amount of resources required to build the system
  - Identify the limitations and risks associated with the proposed system

Designing and Developing DSS

- Both decision maker and programmer attain clarity on whether to go ahead with the system development or not, once the aforesaid factors are assessed
- Choosing a System Development Approach follows the feasibility study that could be very challenging
  A. SDLC - System Development Life Cycle Approach
  B. Rapid Prototyping Approach
  C. End-User DSS Development Approach

Designing and Developing DSS

A. The systems development life cycle (SDLC) approach is based on a series of formal steps, including these seven steps:

1. Confirm user requirements;
2. Systems analysis;
3. System design;
4. Programming;
5. Testing;
6. Implementation; and
7. Use and Evaluation.

This formal SDLC approach is sometimes called the "waterfall" model because of the sequential flow from one step to another.

Designing and Developing DSS

B. The prototyping development approach evolved in response to perceived deficiencies and limitations of the SDLC approach.

A typical prototyping methodology usually includes five steps:

1. Identify user requirements.
2. Develop and test a first iteration DSS prototype.
3. Create the next iteration DSS prototype.
4. Test the DSS prototype and return to step 3 if needed.
5. Pilot testing, phased or full-scale implementation.

Designing and Developing DSS

C. End-user development approach puts the responsibility for building and maintaining a DSS on the manager who builds it.

- Powerful end-user software is available to managers, and many managers have the ability—and feel the need—to develop their own desktop DSS using tools e.g. spreadsheets, 4GL-based tools, Visual programming, Screen generators, application generators, report generators etc.
- The major advantage of encouraging end-user DSS development is that the person who wants computer support will be involved in creating it. The manager/builder controls the situation and the solution that is developed.
- This approach usually results in faster development and cost savings. However, it is not suitable for complex DSS.

Designing and Developing DSS

- Almost DSS configurations:
  - Support individuals and teams
  - Used repeatedly and constantly
  - Two major components: data and models
  - Web-based
  - Uses subjective, personal, and objective data
  - Has a simulation model
  - Used in public and private sectors
  - Has what-if capabilities
  - Use quantitative and qualitative models

Designing and Developing DSS

- Almost DSS configurations:
  - Support individuals and teams
  - Used repeatedly and constantly
  - Two major components: data and models
  - Web-based
  - Uses subjective, personal, and objective data
  - Has a simulation model
  - Used in public and private sectors
  - Has what-if capabilities
  - Use quantitative and qualitative models

Characteristics and Capabilities of DSS

1. Provide support in semi-structured and unstructured situations, includes human judgment and computerized information
2. Support for various managerial levels
3. Support to individuals and groups
4. Support to interdependent and/or sequential decisions
5. Support all phases of the decision-making process
6. Support a variety of decision-making processes and styles
7. Are adaptive

Characteristics and Capabilities of DSS

8. Have user friendly interfaces
9. Goal: improve effectiveness of decision making
10. The decision maker controls the decision-making process
11. End-users can build simple systems
12. Utilizes models for analysis
13. Provides access to a variety of data sources, formats, and types

DSS Components

1. Data Management Subsystem
2. Model Management Subsystem
3. Knowledge-based (Management) Subsystem
4. User Interface Subsystem
5. The User

The Data Management Subsystem

- DSS database
- Database management system
- Data directory
- Query facility

DSS Database Issues

- Data warehouse
- Data mining
- Special independent DSS databases
- Extraction of data from internal, external, and private sources
- Web browser data access
- Web database servers
- Multimedia databases
- Special GSS databases (like Lotus Notes / Domino Server)
- Online Analytical Processing (OLAP)
- Object-oriented databases
- Commercial database management systems (DBMS)

The Model Management Subsystem

- Analogous to the database management subsystem.
- It has the following constituents
  - Model base
  - Model base management system
  - Modeling language
  - Model directory
  - Model execution, integration, and command processor

Model Management Issues

- Model level:
  - Strategic,
  - managerial (tactical), and
  - operational
- Modeling languages
- Lack of standard MBMS activities. WHY?
- Use of AI and fuzzy logic in MBMS

Knowledge Based (Management) Subsystem

- Provides expertise in solving complex unstructured and semi-structured problems
- Expertise provided by an expert system or other intelligent system
- Advanced DSS have a *knowledge* based (*management)* component
- Leads to intelligent DSS
- Example: Data mining

The User Interface (Dialog) Subsystem

- Includes all communication between a user and the MSS
- Graphical user interfaces (GUI)
- Voice recognition and speech synthesis possible
- To most users, the user interface *is* the system

The User

Different usage patterns for the *user*, the *manager*, or the *decision maker*

- Managers
- Staff specialists
- Intermediaries
    1. *Staff assistant*
    2. *Expert tool user*
    3. *Business (system) analyst*
    4. Group Support System (*GSS) Facilitator*

DSS Hardware

Evolved with computer hardware and software technologies

- Major Hardware Options
  - Mainframe
  - Workstation
  - Personal computer
- Web server system
  - Internet
  - Intranet
  - Extranet

DSS Classifications

- A decision support system may present information graphically and may include an expert system or artificial intelligence (AI).
- It may be aimed at business executives or some other group of knowledge workers.
- Typical information that a decision support application might gather and present would be,

  (a) Accessing all information assets, including legacy and relational data sources;

  (b) Comparative data figures;

  (c) Projected figures based on new data or assumptions;

  (d) Consequences of different decision alternatives, given past experience in a specific context.

DSS Classifications

- There are several ways to classify DSS. From a data representation perspective 5 major classifications can be made.
  - Text Oriented DSS
  - Database Oriented DSS
  - Spreadsheet Oriented DSS
  - Solver Oriented DSS
  - Rules Oriented DSS

DSS Classifications

1. ***Text-oriented DSS*** contains information in text form allowing documents to be electronically created, revised and viewed as needed
2. ***Database-oriented DSS*** is driven by a database of organized and highly structured data.
3. ***Spreadsheet-oriented DSS*** contains information in spread sheets. A spread sheet makes it easy to create, view, or modify procedural knowledge. It is also possible to instruct the system to execute self-contained instructions.

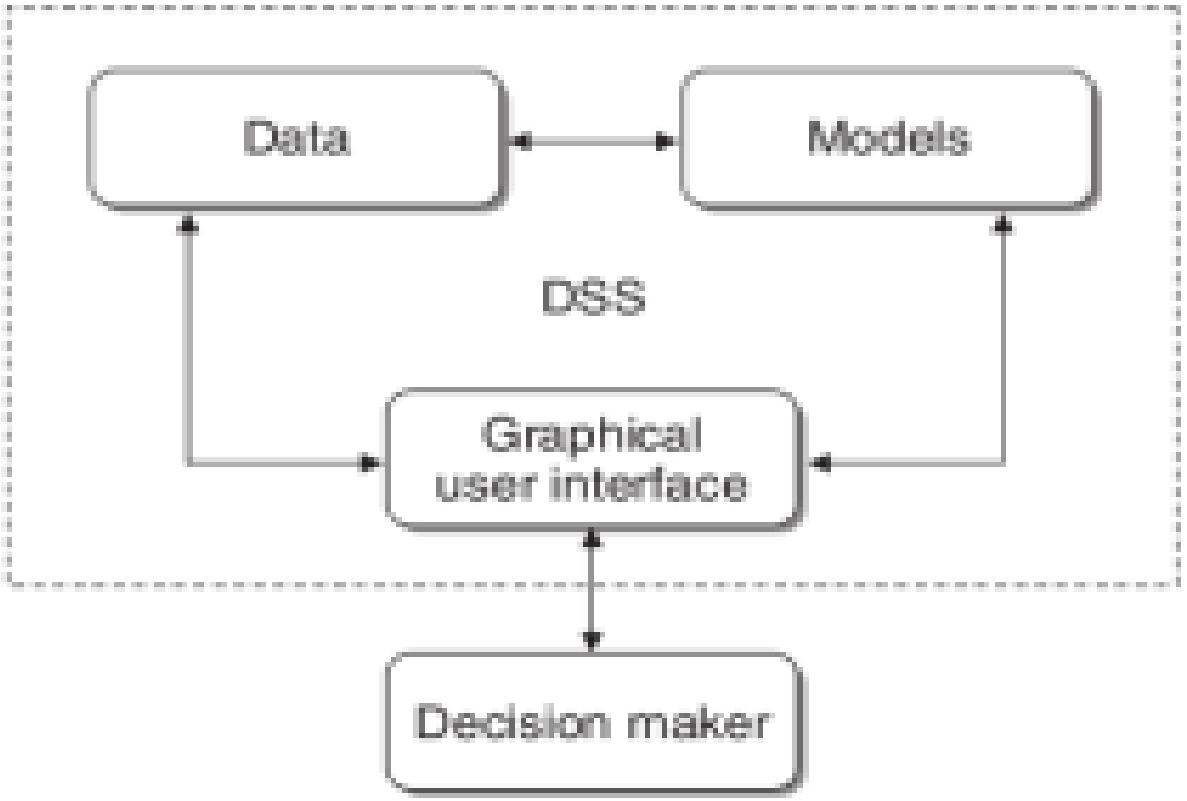
DSS Classifications

4. ***Solver-oriented DSS*** is based on an algorithm or procedure written for performing certain calculations and executing a particular program type
5. ***Rule-oriented DSS*** follows certain procedures adopted as rules
6. ***Compound DSS*** is any combination of the above

Despite these differences in classification, all DSS systems have the same general architecture as shown in the figure.

DSS: General Architecture

Despite these differences in classification, all DSS systems have the same general architecture as shown below:



DSS Classifications

- Another way to classify a DSS is to look at it in terms of the processes and purposes that drive it and it's target audience. From this perspective several major categories present themselves.
  - Communication-driven DSS
  - Data-driven DSS
  - Document-driven DSS
  - Knowledge-driven DSS
  - Model-driven DSS

DSS Classifications

- Communication-driven DSS
  - These are often called group decision support systems (GDSS)
  - They are a special type of hybrid DSS that emphasizes the use of communications and decision models intended to facilitate the solution of problems by decision makers working together as a group.
  - Most communications-driven DSS's are targeted at internal teams, including partners.
  - GDSS supports electronic communication, scheduling, document sharing and other group productivity and decision enhancing activities and involves technologies such as two-way interactive video, bulletin boards, e-mail, etc. Examples: chats and instant messaging software, online collaboration and net-meeting systems.

DSS Classifications

- Data-driven DSS
  - These support decision making by enabling users to extract useful information that was previously buried in large quantities of data.
  - Often data from transaction processing systems are collected in data warehouses for this purpose.
  - Most data-driven DSSs are targeted at managers, staff and also product/service suppliers who need to query a database or data warehouse for a specific answer for specific purposes.

DSS Classifications

- **Document-driven DSS**
  - These systems help managers retrieve and mange unstructured documents and web pages by integrating a variety of storage and processing technologies to provide complete document retrieval and analysis.
  - These systems provide access to all kinds of documents such as company policies and procedures, product specification, catalogues, corporate historical documents, minutes of meetings, important correspondence, corporate records, etc. and are usually driven by a task-specific search engine.
  - Document-driven DSSs are very common, and are targeted at a broad base of user groups.

DSS Classifications

- Knowledge-driven DSS
  - These are also known as or a ‘knowledge-base’ , a catch-all category, covering a broad range of systems users within the organization setting it up.
  - It may also include others interacting with the organization – for example, consumers of a business.
  - It is essentially used to provide management advice or to choose products/services.
  - The typical deployment technology used to set up such systems could be silent/server systems, the web, or software running on stand-alone PCs.

DSS Classifications

- Knowledge-driven DSS
  - These are also known as or a ‘knowledge-base’ , a catch-all category, covering a broad range of systems users within the organization setting it up.
  - It may also include others interacting with the organization – for example, consumers of a business.
  - It is essentially used to provide management advice or to choose products/services.
  - The typical deployment technology used to set up such systems could be silent/server systems, the web, or software running on stand-alone PCs.

DSS Classifications

- Model-driven DSS
  - The underlying model that drives the DSS can come from various disciplines or areas of speciality and might include accounting models, financial models, representation models, optimization models, etc.
  - Model-driven DSSs are complex systems that help analyse decisions or choose between different options with the emphasize on access to and manipulation of a model, rather than data, i.e. it uses data and parameters to aid decision makers in analysing a situation.
  - These systems usually are not data intensive and consequently are not linked to very large databases.

DSS Classifications

- Model-driven DSS
  - A model-driven DSS will most commonly be used by managers and staff members of a business, or people who interact with the organization, for a number of purposes depending on how the model is set up – scheduling, decision analyses etc.
  - These DSSs can be deployed via software/hardware in stand-alone PCs, client/server systems, or the web.

Intelligent DSS Categories

- Descriptive
- Procedural
- Reasoning
- Linguistic
- Presentation
- Assimilative

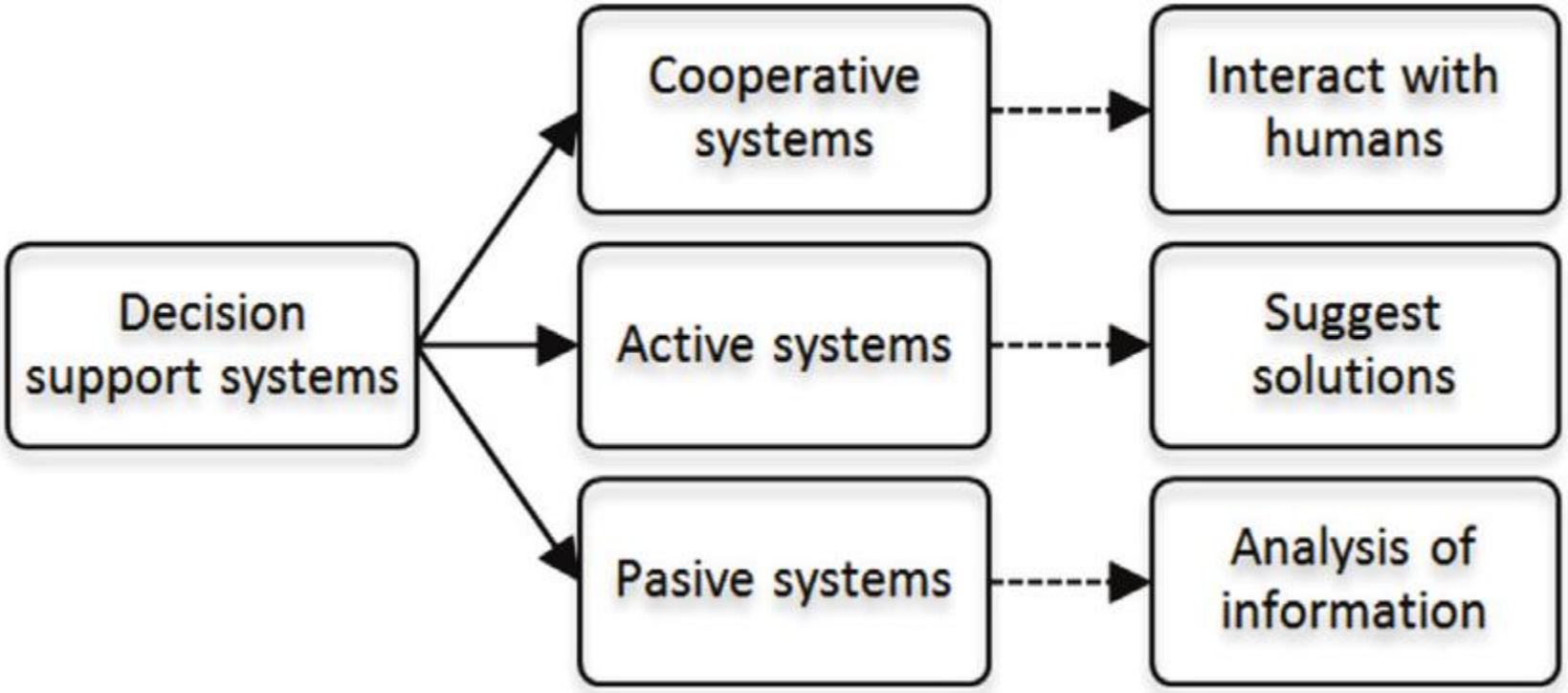
Alternate Categories of Intelligent DSS

- Symbiotic
- Expert-system based
- Adaptive
- Holistic

Other classifications

- Institutional DSS vs. Ad Hoc DSS
  - Institutional DSS deals with decisions of a recurring nature
  - Ad Hoc DSS deals with specific problems that are usually neither anticipated nor recurring
- Degree of non-procedurality
- Personal, group, and organizational support
- Individual versus group support systems (GSS)
- Custom-made versus ready-made systems

DSS classification and scope in decision making



DSS classification and scope in decision making

- The passive decision support systems are responsible only for collecting and organizing information for its use by the people responsible for the decision; therefore, these systems do not suggest any specific response.
- The active decision support systems are responsible for collecting information, upon which they base an explicit presentation of one or more solutions to the decision problem.
- A cooperative DSS is responsible for gathering the information, analyzing it, and then delivering it to the people responsible for decision making, and is also tasked to revise or refine the information. The name “cooperative” is derived from the cooperative work between software and people, with the intention of making the best possible decision.

DSS Project Participants

- A complex DSS, built using either an SDLC or a prototyping approach, requires a team approach to development.
- Once the system is developed, a group may also need to maintain the system.
- Some large-scale DSS are built with teams of two or three people, or with a larger group of ten or more.
- Members of DSS teams are drawn from many areas in an organization, including the IS group.
- Any DSS development project requires a mix of complementary skills. Usually, one does not find all of the needed skills in one person

DSS Project Participants

- Project Manager
- Executive Sponsor or Project Champion
- Potential DSS Users
- DSS Analyst
- Technical Support Staff
  - e.g. data warehouse architects, data modelers, data quality analysts, database administrators, network specialists, application architects, operations researchers, and developers
- DSS Toolsmith

DSS Project Participants

- ***Project Manager:*** This is a business or technical manager who can organize and manage the resources needed to complete the DSS project.
- ***Executive Sponsor or Project Champion:*** This is a senior manager who has access to other senior executives and has the influence to help resolve major resource and political problems. The sponsor is occasionally actively involved in the development tasks.
- ***Potential DSS Users:*** The persons who make decisions that a proposed DSS will support. Users are often nontechnical people in functional areas of a business such as marketing and finance.

DSS Project Participants

- ***DSS Analyst:*** This is the MIS specialist who acts as an intermediary or liaison between users and DSS developers. He/She has a great deal of experience who understands both the business problem and the available technologies.
- A DSS analyst may make the decisions about software tools to use, hardware platforms to use, models and/or databases to incorporate into the DSS, and how they will be integrated with each other.
- A DSS analyst often works with users to define and document system requirements for Decision Support Systems.
- A DSS analyst may help redesign business processes to better use a computerized Decision Support System. This person sometimes also assumes the role of project manager

DSS Project Participants

- ***Technical Support Staff***: A number of MIS professionals are involved as technical support staff including data warehouse architects, network specialists, application architects, operations researchers, and developers.
- A data modeler and data quality analyst are often involved in building data-driven DSS.
- The data quality analyst is concerned with data integration, metadata, and data scrubbing.
- A database administrator is an integral part of a development team for a data-driven DSS project.
- Data administrators, systems administrators, and networking specialists are often consulted on DSS projects and may join a development team for some projects.

DSS Project Participants

- ***DSS Toolsmith:*** This is a specialist with the tools and technologies that will be used in the construction of the DSS and the packages that will be combined to create the DSS.
- He/She is an expert on these tools and packages, and their effective use.
- This is the person who creates underlying capabilities and integrates existing packages into one overall system and carries out custom programming that contributes directly to DSS functionality.
- His/Her responsibility begins with the packages that will comprise part of the DSS and ends with completion of a specific DSS.

Knowledge-based systems

- A knowledge-based system (KBS) aims at capturing the knowledge of human experts to support decision-making
- The foundation of a successful KBS depends on a series of technical procedures and development that may be designed by certain related experts.
- Knowledge-Based Systems (KBS) are based on the interaction between two major components: the inference engine and the knowledge base.
- Examples of knowledge-based systems include **expert systems**, which are so called because of their reliance on human expertise

Reasoning in Knowledge-based systems

- The inference engine of KBS implements a reasoning model and makes use of the knowledge base to explicit solutions to problems
- The following techniques for reasoning have been found useful:
  - Rule-based reasoning
  - Case-based reasoning
  - Constraint-based reasoning
  - Model-based reasoning
  - Diagrammatic reasoning
  - Black-board reasoning
  - Truth Maintenance System

Reasoning in Knowledge-based systems

- A rule-based system may be viewed as consisting of three basic components: a set of **rules** [rule base], a **data base** [fact base], and an **interpreter** for the rules
- Case-based reasoning (CBR) is an experience-based approach to solving new problems by adapting previously successful solutions to similar problems
- Rule-based reasoning (RBR) and case-based reasoning (CBR) have emerged as two important and complementary reasoning methodologies in artificial intelligence (AI).
- For problem solving in complex, real-world situations, it is useful to integrate RBR and CBR.

Rule Based Reasoning (RBR) versus Case Based Reasoning (CBR)

- Rule Based Reasoning (RBR)
  - It requires us to elicit an explicit model of the domain.
  - As we all know and have experienced, knowledge acquisition has a set of associated problems.
- Case Based Reasoning (CBR)
  - Unlike, it does not require an explicit model.
  - Cases that identify the significant features are gathered and added to the case base during development and after deployment.
  - This is easier than creating an explicit model, as it is possible to develop case bases without passing through the knowledge-acquisition bottleneck.

Rule Based Reasoning (RBR) versus Case Based Reasoning (CBR)

- Domain experts would have accumulated knowledge over the years through experience.
- It would be a blunder mistake if we do not use them to our advantage.
- But the difficulty lies in getting the experts to list down the decision rules which they use.
- It is a Herculean task to comprehensively recall all the tacit rules which they have come to adopt.
- However, they usually have little difficulty in recalling concrete cases, which they have encountered in practice.
- Thus, their mental set appears to be oriented towards a Case Based Reasoning approach.

Rule Based Reasoning (RBR) versus Case Based Reasoning (CBR)

- Developing a CBR system is much faster and easier than constructing a rule-based equivalent.
- Case bases do not have to be complete when they are deployed for use, as even non-computer experts can add cases to the existing structure.
- Maintenance with a Rule Based System may be a nightmare.
- If the rules are not written clearly, it would lead to many sleepless nights of debugging.

Rule Based Reasoning (RBR) versus Case Based Reasoning (CBR)

- Maintenance with Case Based Systems are much easier and straightforward.
- When rules are added or deleted from a rule-based system, the system has to be checked for conflicting rules and redundant rules.
- An addition or deletion of a case from the case base does not any further checking or debugging.
- But it have to be noted that while it does not affect the system's functioning, it may have an impact on the outcome of the system.

CBR Applications

- The following are some applications where Case-Based Reasoning (CBR) has been used:
  - Fault Diagnosis
  - Medical Diagnosis
  - Credit Card Risk Assessment
  - Design of bridges
  - Customer Service Hotline
  - Helpdesk
  - Troubleshooting Applications

When to use CBR?

- Case Based Reasoning (CBR) is more suitable for the following problems:
  - The problem domain is complex and not amenable to complete mathematical modeling.
  - An explicit model is extremely difficult to elicit and represent with rules.
  - Historical data exists within the organization.
  - The domain experts have considerable difficulty in writing down the decision rules. But, they are comfortable in providing well-proven heuristics and experiences, to incorporate into the case base as cases. They have little difficulty in recalling concrete cases, which they have encountered in the past.
  - The domain is such that it needs reasoning for the system's solutions.

Constraint-based reasoning

- Constraint-based reasoning is an important area of automated reasoning in artificial intelligence, with many applications such as:
  - Configuration and design problems,
  - Planning and scheduling,
  - Temporal and spatial reasoning,
  - Defeasible and causal reasoning,
  - Machine vision and language understanding,
  - Qualitative and diagnostic reasoning, and expert systems.

Constraint-based reasoning

- Constraint-Based Reasoning presents current work in the field at several levels:
  - theory,
  - algorithms,
  - languages,
  - applications, and
  - hardware.

Constraint-based reasoning

- Constraint-based reasoning has connections to a wide variety of fields, including
  - formal logic,
  - graph theory,
  - relational databases,
  - combinatorial algorithms,
  - operations research,
  - neural networks,
  - truth maintenance, and
  - logic programming.

Constraint-based reasoning

- The ideal of describing a problem domain in natural, declarative terms and then letting general deductive mechanisms synthesize individual solutions has to some extent been realized, and even embodied, in programming languages.
- Constraint-based reasoning is applicable in domains that are defined by constraints, or what cannot be done.

Model-based Reasoning

- In artificial intelligence, model-based reasoning (MBR) refers to an inference method used in expert systems based on a model of the physical world .
- With this approach, the main focus of application development is developing the model.
- Then at run time, an "engine" combines this model knowledge with observed data to derive conclusions such as a diagnosis or a prediction.
- In a model-based reasoning system knowledge can be represented using causal rules.

Model-based Reasoning

- Model-based reasoning is the use of a working model and accompanying real-world observations to draw conclusions.
- It plays an important role in artificial logic systems as well as reasoning in the sciences.
- The creation of the model is the time-consuming aspect of this approach, as it is necessary to make the model as deep, complex, and detailed as possible to achieve the best results.
- Once a working model has been established, it may also require periodic updates

Model-based Reasoning

- A model-based system is based on a model of the structure and behavior of the device that the system is designed to simulate
- Used for well structured problems – Not for stock pricing /modeling, not well structured – Engineering Problems
- Based on written documentation
- The problem is extracting knowledge
- Observed behavior (what the device is actually doing) versus predicted behavior (what the device should do)
- The difference between them is called a discrepancy, indicating a defect
- Then a process is initiated to diagnose the nature and location of the defect
- Could be a mathematical equation

Model-based Reasoning

- MBR-based systems can have a wide range of applications in the sciences.
- These systems can allow researchers to explore and test hypotheses.
- Model-based reasoning can also be the backbone of a monitoring system that sends alerts based on inputs.
- Climate modeling, for example, allows computers to take information about current weather conditions and run it through a model to provide information about budding tropical storms and other meteorological events of concern.
- Automation of some tasks can allow researchers to focus on other topics that require more complex reasoning.

Diagrammatic reasoning

- Diagrammatic reasoning is an artificial intelligence technique that aims to understand concepts and ideas using diagrams that represent knowledge.
- A common motivation for developing computational frameworks for diagrammatic reasoning is the hope that they might serve as re-configurable tools for studying human problem solving performance.
- Despite the ongoing debate as to the precise mechanisms by which diagrams, or any other external representation, are used in human problem solving, there is little doubt that diagrammatic representations considerably help humans solve certain classes of problems.

Diagrammatic reasoning

- In fact, there are a host of applications of diagrams and diagrammatic representations in computing, from data presentation to visual programming languages.
- In contrast to both the use of diagrams in human problem solving and the ubiquitous use of diagrams in the computing industry, the use of diagrammatic representations has been explored in automated problem solving.

Diagrammatic reasoning

- Within artificial intelligence, systems that have claimed to comprise some degree of diagrammatic reasoning capability have tended to be restricted to two problem domains:
  - geometry theorem proving and discovery, and
  - reasoning about physical systems.

Black-board reasoning

- *Imagine a group of human specialists seated next to a large blackboard. The specialists are working cooperatively to solve a problem, using the blackboard as the workplace for developing the solution.*

- *Problem solving begins when the problem and initial data are written onto the blackboard. The specialists watch the blackboard, looking for an opportunity to apply their expertise to the developing solution. When a specialist finds sufficient information to make a contribution, she records the contribution on the blackboard, hopefully enabling other specialists to apply their expertise. This process of adding contributions to the blackboard continues until the problem has been solved.*

Black-board reasoning

- A blackboard system is an artificial intelligence application based on the blackboard architectural model.
- In this, a common knowledge base, the "blackboard", is iteratively updated by a diverse group of specialist knowledge sources, starting with a problem specification and ending with a solution.
- Each *knowledge source* updates the blackboard with a partial solution when its internal constraints match the blackboard state.
- In this way, the specialists work together to solve the problem.
- The blackboard model was originally designed as a way to handle complex, ill- defined problems, where the solution is the sum of its parts.

Black-board reasoning

- Building blocks
  - Knowledge sources
  - A common place to share knowledge, called Blackboard
  - Control Mechanism
- Problem Solving Steps
  - Initiation by Knowledge Sources
  - Control decides what to focus attention on
  - Knowledge Sources modify solutions
  - Terminated when either solution is found or no further action is possible

Black-board reasoning

- A blackboard system is an artificial intelligence application based on the blackboard architectural model.
- In this, a common knowledge base, the "blackboard", is iteratively updated by a diverse group of specialist knowledge sources, starting with a problem specification and ending with a solution.
- Each knowledge source updates the blackboard with a partial solution when its internal constraints match the blackboard state.
- In this way, the specialists work together to solve the problem.
- The blackboard model was originally designed as a way to handle complex, ill- defined problems, where the solution is the sum of its parts

Truth Maintenance System (TMS)

- Truth Maintenance Systems (TMS), also called Reason Maintenance Systems, are used within Problem Solving Systems, in conjunction with Inference Engines (IE) such as rule-based inference systems, to manage as a Dependency Network the inference engine's beliefs in given sentences
- TMS is a problem solver subsystem for performing these functions by recording and maintaining the reasons for program beliefs.
- Such recorded reasons are useful in constructing explanations of program actions and in guiding the course of action of a problem solver.

Truth Maintenance System (TMS)

- Necessary when changes in the fact-base lead to inconsistency / incorrectness among the facts non-monotonic reasoning
- A TMS tries to adjust the Knowledge Base or Fact Base upon changes to keep it consistent and correct.
- It uses dependencies among facts to keep track of conclusions and allow revision / retraction of facts and conclusions.
- TMS can have different characteristics:
  - Justification-Based Truth Maintenance System
  - Assumption-Based Truth Maintenance System
  - Logical-Based Truth Maintenance System

Truth Maintenance System (TMS)

- A TMS is intended to satisfy a number of goals:
  - Provide justifications for conclusions
  - Recognise inconsistencies
  - Support default reasoning
  - Remember derivations computed previously
  - Support dependency driven backtracking
- Truth Maintenance Systems are significant as a mechanism for implementing dependency directed backtracking during search

Types of Knowledge-based systems

- There are seven main types of KBS:
  - Expert systems
  - Neural networks
  - Case-based reasoning
  - Genetic algorithms
  - Intelligent agents
  - Data mining
  - Intelligent tutoring systems

Roles in KBS development

- Knowledge provider
- Knowledge engineer/analyst
- Knowledge system developer
- Knowledge user
- Project manager
- Knowledge manager

Knowledge provider/specialist

- “Traditional” expert
- Person with extensive experience in an application domain
- Can provide also plan for domain familiarization
  - “where would you advise a beginner to start?”
- Inter-provider differences are common
- Needs to assure cooperation

Knowledge engineer

- Specific kind of system analyst
- Should avoid becoming an "expert"
- Plays a liaison function between application domain and system

Knowledge-based system developer

- Person that implements a knowledge-based system on a particular target platform
- Needs to have general design/implementation expertise
- Needs to understand knowledge analysis
  - but only on the “use”-level
- Role is often played by knowledge engineer

Knowledge user

- Primary users
  - interact with the prospective system
- Secondary users
  - are affected indirectly by the system
- Level of skill/knowledge is important factor
- May need extensive interacting facilities
  - explanation
- His/her work is often affected by the system
  - consider attitude / active tole

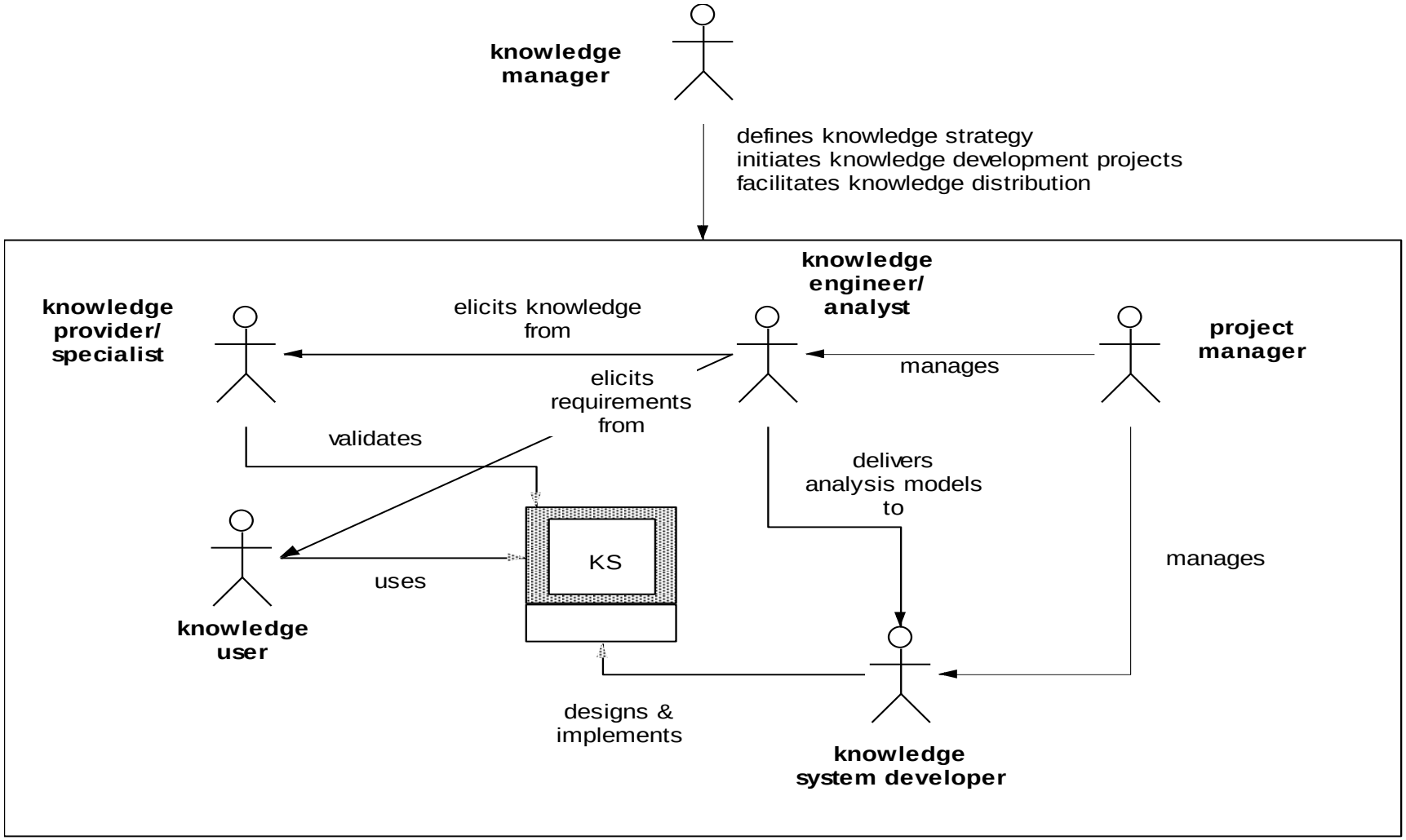
Project manager

- Responsible for planning, scheduling and monitoring development work
- Liaises with client
- Typically, medium-size projects (4-6 people)
- Profits from structured approach

Knowledge manager

- background role
- monitors organizational purpose of
  - system(s) developed in a project
  - knowledge assets developed/refined
- initiates (follow-up) projects
- should play key role in reuse
- may help in setting up the right project team

Roles in knowledge-based system development



Terminology

- Domain
  - some area of interest

    banking, food industry, photocopiers, car manufacturing
- Task
  - something that needs to be done by an agent

    monitor a process; create a plan; analyze deviant behavior
- Agent
  - the executor of a task in a domain

    typically, either a human or some software system

Terminology

- Application
  - The context provided by the combination of a task and a domain in which this task is carried out by agents
- Application domain
  - The particular area of interest involved in an application
- Application task
  - The (top-level) task that needs to be performed in a certain application
- Knowledge-based system (KBS)
  - system that solves a real-life problem using knowledge about the application domain and the application task
- Expert system
  - Knowledge-based system that solves a problem which requires a considerable amount of expertise, when solved by humans.

An Introduction to Expert Systems

- Expert Systems are computer programs that exhibit intelligent behavior.
- These systems are concerned with the concepts and methods of symbolic inference, or reasoning, by a computer, and how the knowledge used to make those inferences will be represented
- Achieving expert-level competence in solving problems in task areas by bringing to bear a body of knowledge about specific tasks is called *knowledge-based* or *expert system*.

An Introduction to Expert Systems

- The term expert system is reserved for programs whose knowledge base contains the knowledge used by human experts.
- Expert systems and knowledge-based systems are used synonymously.
- The area of human intellectual endeavor to be captured in an expert system is called the *task domain*.
- *Task* refers to some goal-oriented, problem-solving activity.
- *Domain* refers to the area within which the task is being performed.
- Typical tasks are diagnosis, planning, scheduling, configuration and design.

An Introduction to Expert Systems

- Building an expert system is known as *knowledge engineering* and its practitioners are called *knowledge engineers*.
- The knowledge engineer must make sure that the computer has all the knowledge needed to solve a problem.
- The knowledge engineer must choose one or more forms in which to represent the required knowledge as symbol patterns in the memory of the computer -- that is, he (or she) must choose a *knowledge representation*.
- He/She must also ensure that the computer can use the knowledge efficiently by selecting from a handful of *reasoning methods*.

Building Blocks of Expert Systems

- Every expert system consists of two principal parts:
  - the knowledge base; and
  - the reasoning, or inference, engine.
- The *knowledge base* of expert systems contains both
  - factual knowledge and
  - heuristic knowledge.
- *Factual knowledge* is that knowledge of the task domain that is widely shared, typically found in textbooks or journals, and commonly agreed upon by those knowledgeable in a particular field.

Building Blocks of Expert Systems

- *Heuristic knowledge* is the less rigorous, more experiential, more judgmental knowledge of performance.
- In contrast to factual knowledge, heuristic knowledge is rarely discussed, and is largely individualistic.
- Heuristic knowledge is the knowledge of good practice, good judgment, and plausible reasoning in the field.
- It is the knowledge that underlies the "art of good guessing."

Components of Expert Systems

Knowledge base

- contains essential information about the problem domain
- often represented as facts and rules

Inference engine

- mechanism to derive new knowledge from the knowledge base and the information provided by the user
- often based on the use of rules

User interface

- interaction with end users
- development and maintenance of the knowledge base

Expert System: Concepts & Characteristics

Knowledge acquisition

- transfer of knowledge from humans to computers
- sometimes knowledge can be acquired directly from the environment
  - machine learning, neural networks

Knowledge representation

- suitable for storing and processing knowledge in computers

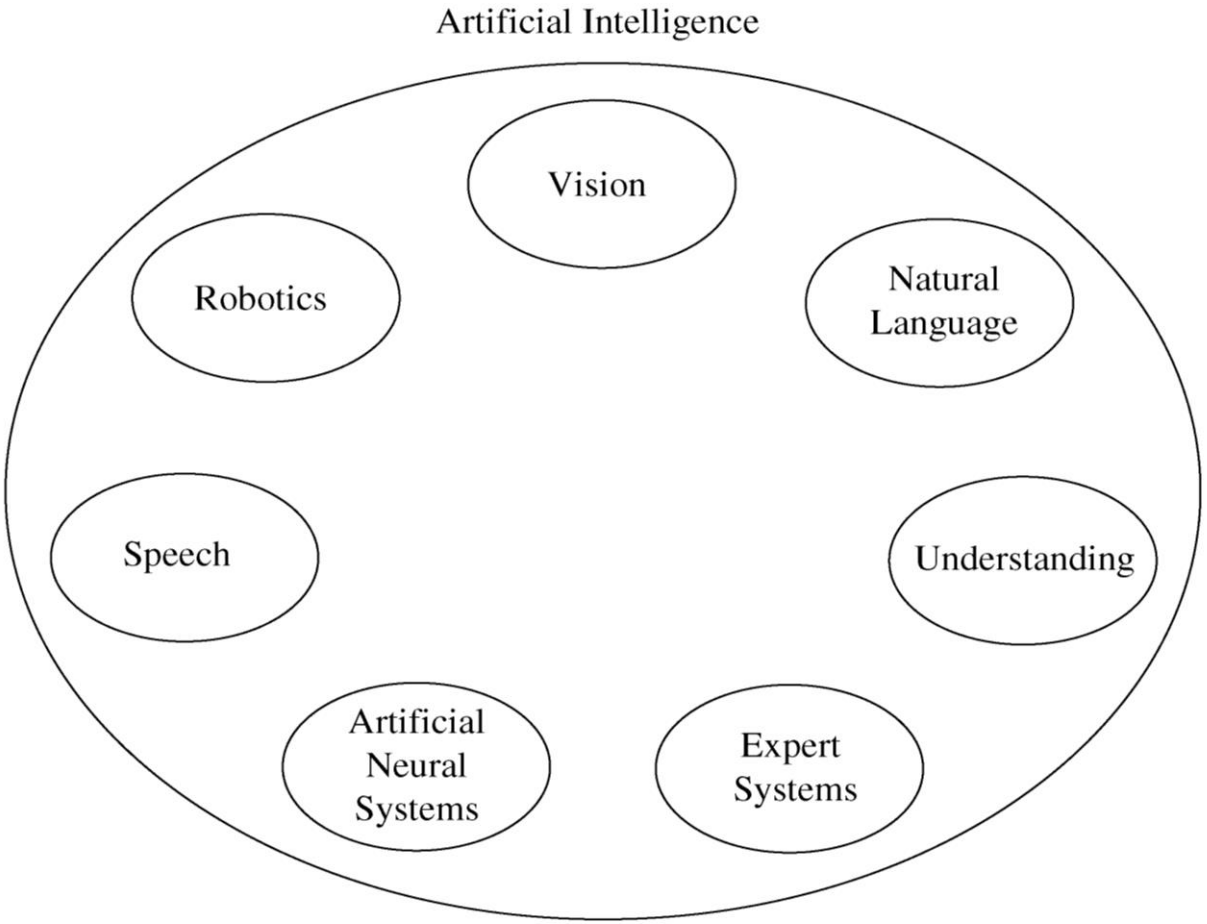
Inference

- mechanism that allows the generation of new conclusions from existing knowledge in a computer

Explanation

- illustrates to the user how and why a particular solution was generated

Areas of Artificial Intelligence



Expert systems overview

What is an expert system?

"An expert system is a computer system that emulates, or acts in all respects, with the decision-making capabilities of a human expert."

Expert system technology may include:

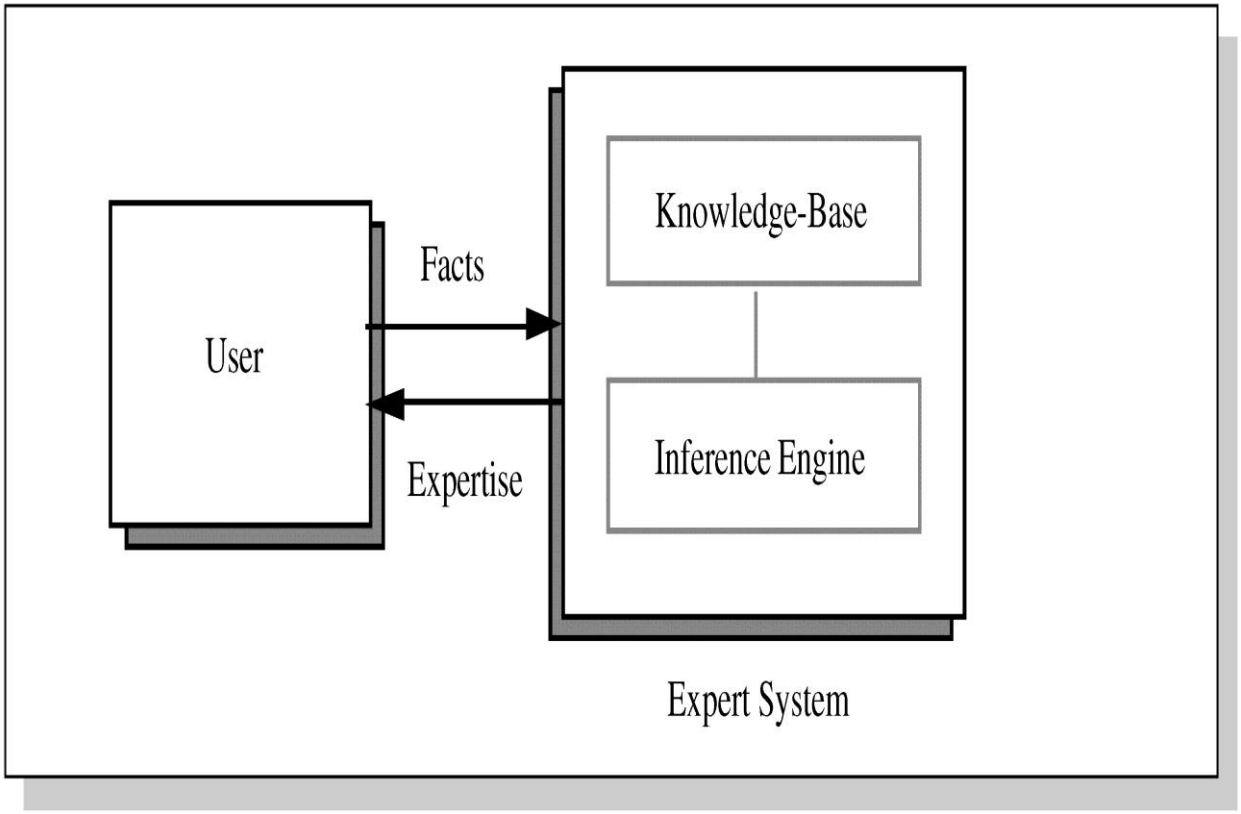
- Special expert system languages –
  - PROLOG,
  - LISP,
  - The C Language Integrated Production System (CLIPS)
- Programs
- Hardware designed to facilitate the implementation of those systems

Expert System Main Components

- Knowledge base – obtainable from books, magazines, knowledgeable persons, etc.

- Inference engine – draws conclusions from the knowledge base

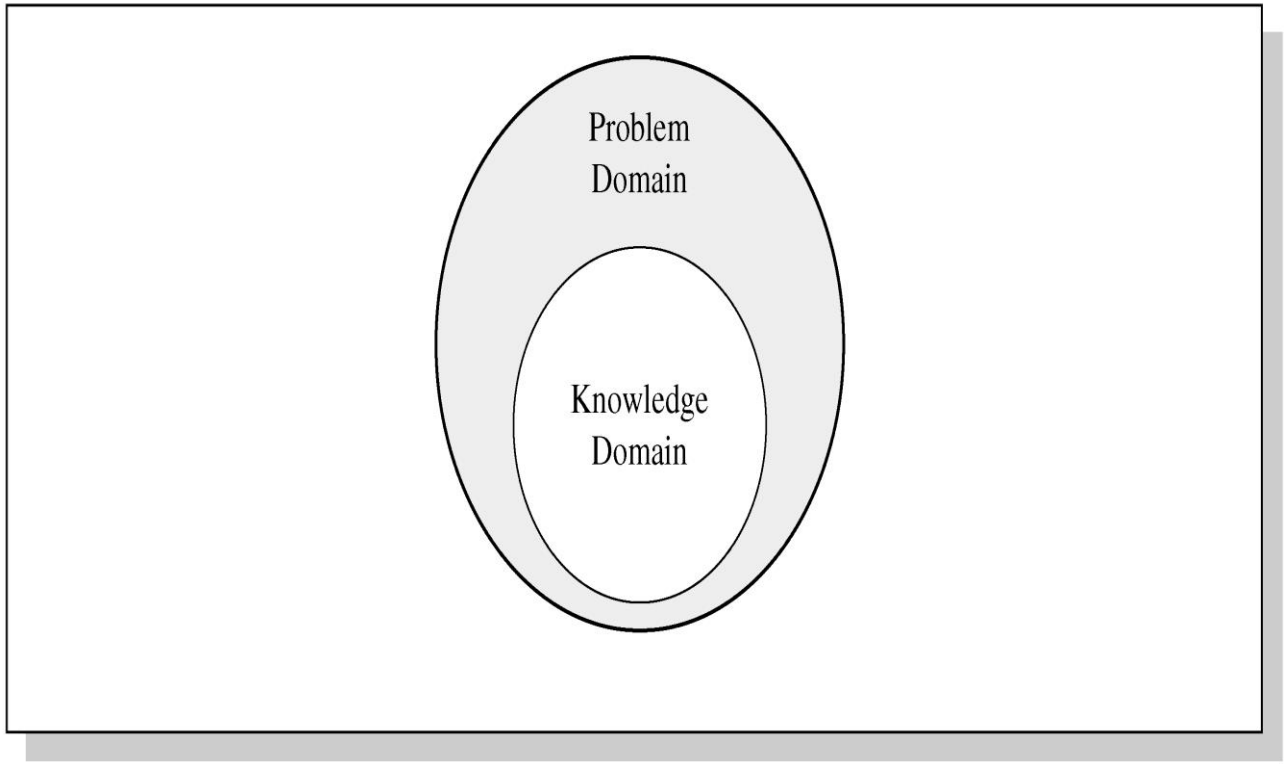
Basic Functions of Expert Systems



Problem Domain vs. Knowledge Domain

- An expert's knowledge is specific to one problem domain –
  - medicine,
  - finance,
  - science,
  - engineering, etc.
- The expert's knowledge about solving specific problems is called the knowledge domain.
- The problem domain is always a superset of the knowledge domain.

Problem and Knowledge Domain Relationship



Advantages of Expert Systems

- Increased availability
- Reduced cost
- Reduced danger
- Performance
- Multiple expertise
- Increased reliability
- Explanation
- Fast response
- Steady, unemotional, and complete responses at all times
- Intelligent tutor
- Intelligent database

Representing the Knowledge

The knowledge of an expert system can be represented in a number of ways, including IF-THEN rules:

```
IF you are hungry THEN eat
```

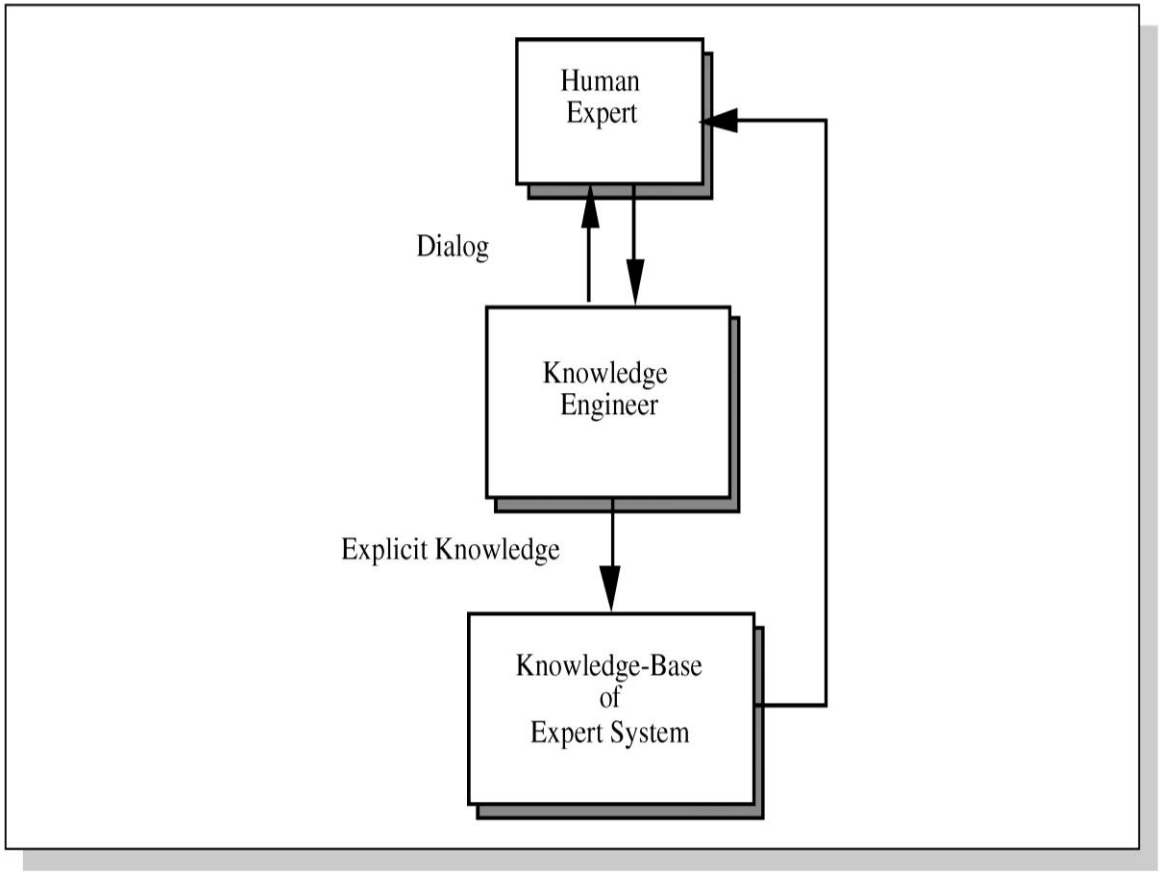
- Knowledge Engineering is the process of building an expert system:

1. The knowledge engineer establishes a dialog with the human expert to elicit knowledge.
2. The knowledge engineer codes the knowledge explicitly in the knowledge base.
3. The expert evaluates the expert system and gives a critique to the knowledge engineer.

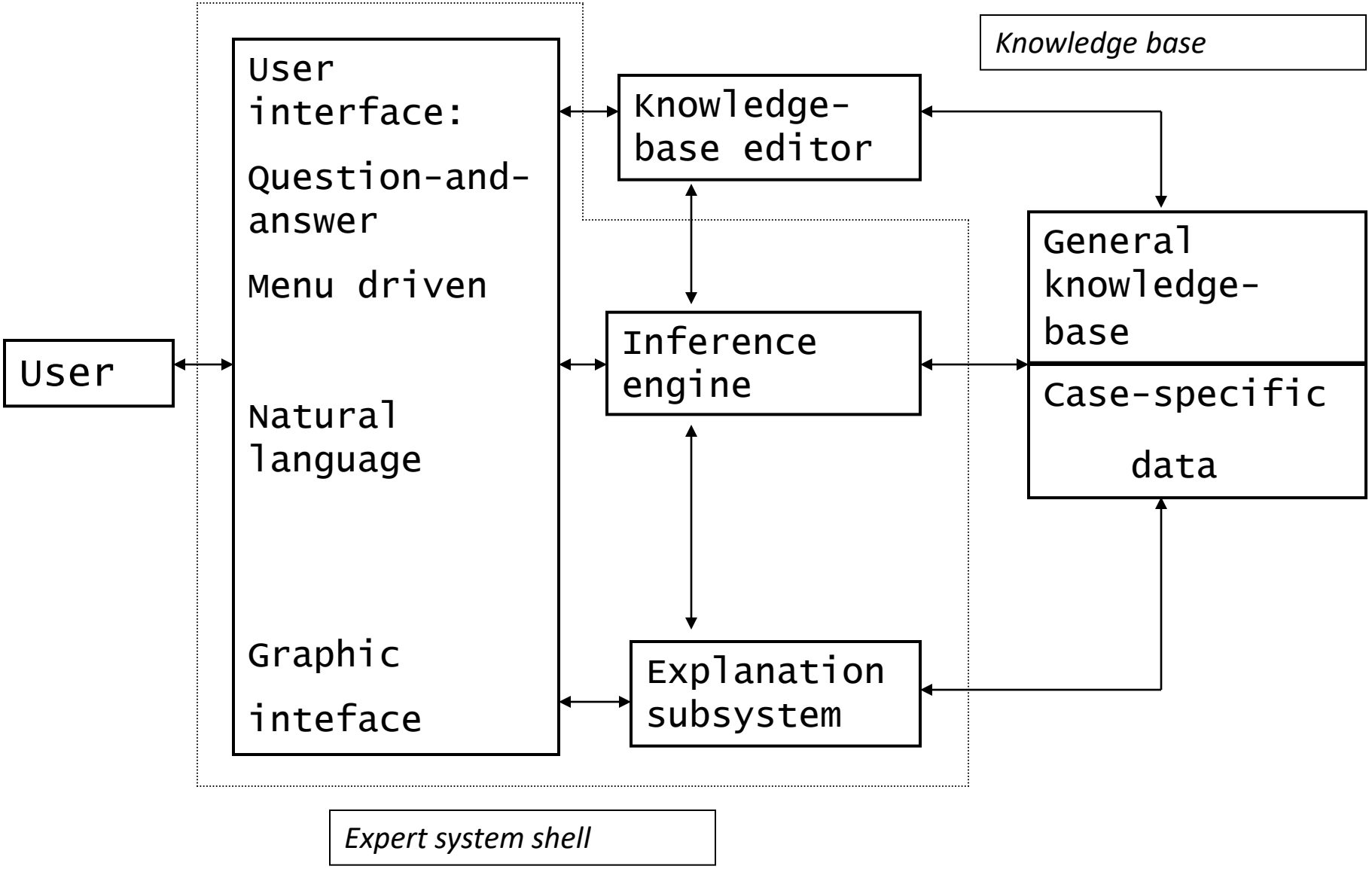
Broad Classes of Expert Systems

Class	General Area
Configuration	Assemble proper components of a system in the proper way.
Diagnosis	Infer underlying problems based on observed evidence.
Instruction	Intelligent teaching so that a student can ask *why*, *how,* and *what if* questions just as if a human were teaching.
Interpretation	Explain observed data.
Monitoring	Compares observed data to expected data to judge performance.
Planning	Devise actions to yield a desired outcome.
Prognosis	Predict the outcome of a given situation.
Remedy	Prescribe treatment for a problem.
Control	Regulate a process. May require interpretation, diagnosis, monitoring, planning, prognosis, and remedies.

Development of an Expert System



Architecture of a typical expert system



Software development: Conventional systems and Knowledge-based system

- We are already familiar with a standard model of the software development life cycle. It is likely to be something like this:

→ Feasibility study

→ Analysis

→ Requirements definition

→ Design

→ Implementation

→ Testing

→ Maintenance & review

Software development: Conventional systems and Knowledge-based system

- Knowledge-based systems require special approaches to systems analysis, especially to the collection of the data (or rather knowledge) on which the system is based.
- The conventional system development model needs to be modified to take account of these special features.
- The term "knowledge engineering" is often used to mean the process of designing, building, installing knowledge-based system (including expert systems). In other words, the whole process of making a KBS, from beginning to end.
- Some authors use the term to mean just the phase in which the knowledge base is built.

Building the knowledge base

Five processes can be identified:

1. Knowledge acquisition
2. Knowledge analysis & representation
3. Knowledge validation
4. Inference design
5. Explanation and justification

These are not stages that have to follow each other - some of them will run concurrently.

Knowledge Acquisition

- Knowledge acquisition is the process of gathering the knowledge to stock the expert system's knowledge base.
- This has proved to be the most difficult component of the knowledge engineering process.
- It's become known as the *'knowledge acquisition bottleneck'*, and KBS projects are more likely to fail at this stage than any other.
- This is the principle reason why KBS including expert systems have not become more widespread.

Knowledge Acquisition

- Sources of knowledge:
  - Human experts
  - Documents: textbooks, journal articles, technical reports, records containing case histories, etc.
- Documents will almost never be sufficient to provide the knowledge base for a real-world expert system.
- The range of problems which the sources like a textbook examine and solve is always smaller than the range of problems that a human expert is master of.

Knowledge Acquisition

- The most important part of knowledge acquisition is *knowledge elicitation* - obtaining knowledge from a human expert (or human experts) for use in an expert system.
- It is necessary to find out what the expert(s) know, and how they use their knowledge.
- Expert knowledge includes:
  - domain-related facts and principles;
  - problem-solving strategies;
  - meta-knowledge - for instance, knowledge about when to use a particular piece of knowledge;
  - explanations and justifications.

Knowledge Elicitation

- The knowledge elicitation/analysis task involves
  - finding at least one expert in the domain who:
    - *is willing* to provide his/her knowledge;
    - *has the time* to provide his/her knowledge;
    - *is able* to provide his/her knowledge.

  - any or all of these are liable to prove difficult.
- The knowledge elicitation/analysis task involves
  - repeated interviews with the expert(s), probably combined with other, non-interview, techniques.

Knowledge Elicitation

- One major obstacle to knowledge elicitation: experts cannot easily describe all they know about their subject.
- They do not necessarily have much insight into the methods they use to solve problems.
- Their knowledge is "*compiled*" (like a compiled computer program - fast and efficient, but unreadable).

Knowledge analysis and Representation

- Simultaneously with the knowledge acquisition process, a knowledge analysis process takes place.
- The knowledge engineer (KE) uses the data - the transcripts and protocols, etc - from the knowledge acquisition sessions to build a good **model of the expertise** that the domain expert (DE) is using to solve problems in the domain.
- The raw data (taken from the DE) is converted into **intermediate representations**. These are structured representations of the knowledge, but not yet the sort of coded knowledge that can be put into the knowledge base.
- This will improve the knowledge engineer's understanding of the subject;
- This will probably provide knowledge in a form that can be shown to the domain experts (DEs), for criticism and correction;
- This provides easily-accessible knowledge for future (knowledge engineers (KEs) to work from (**knowledge archiving**).
- The intermediate representation is then converted into the knowledge representation formalism which is to be used in the KBS.