MAT5024-Decision Support Systems

PPT slides for Post Mid-term Test

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

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

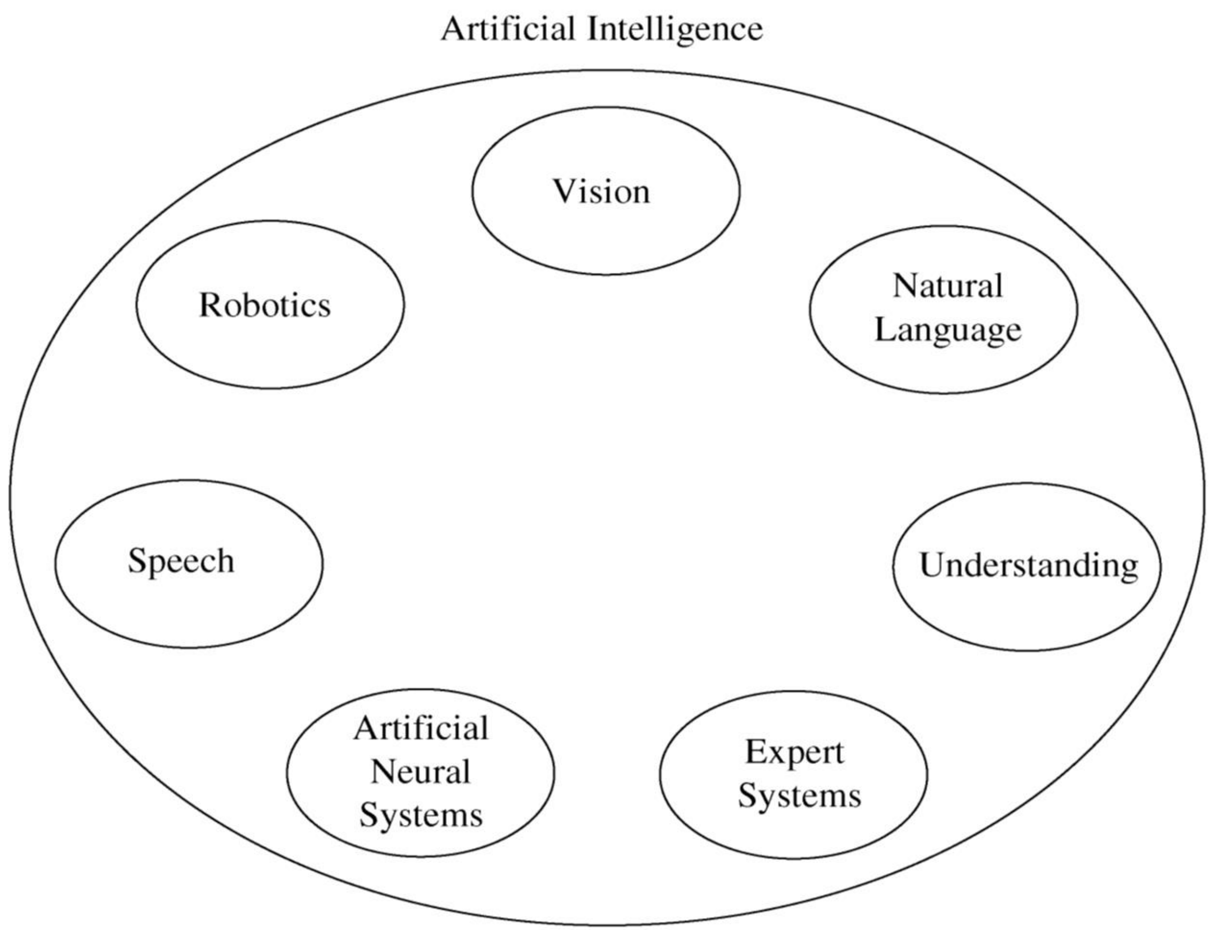
Inference

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

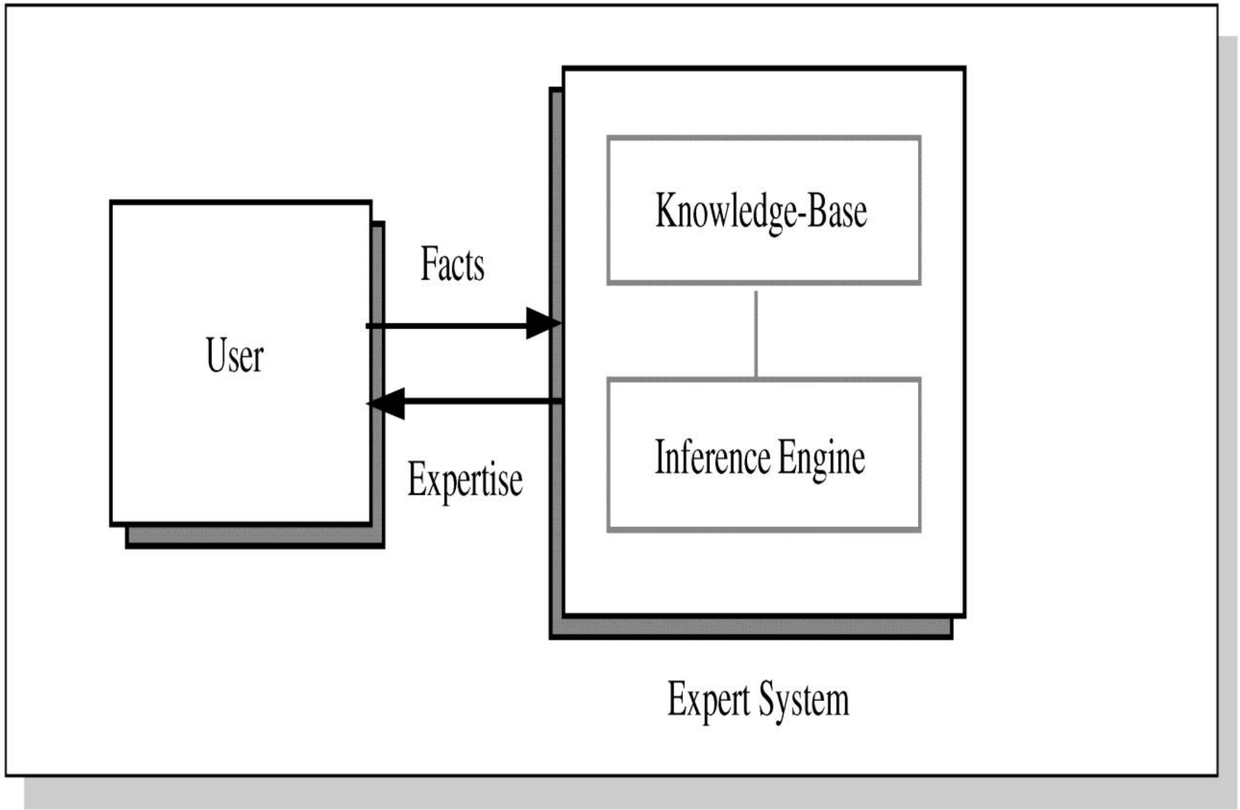
Explanation

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

Areas of Artificial Intelligence



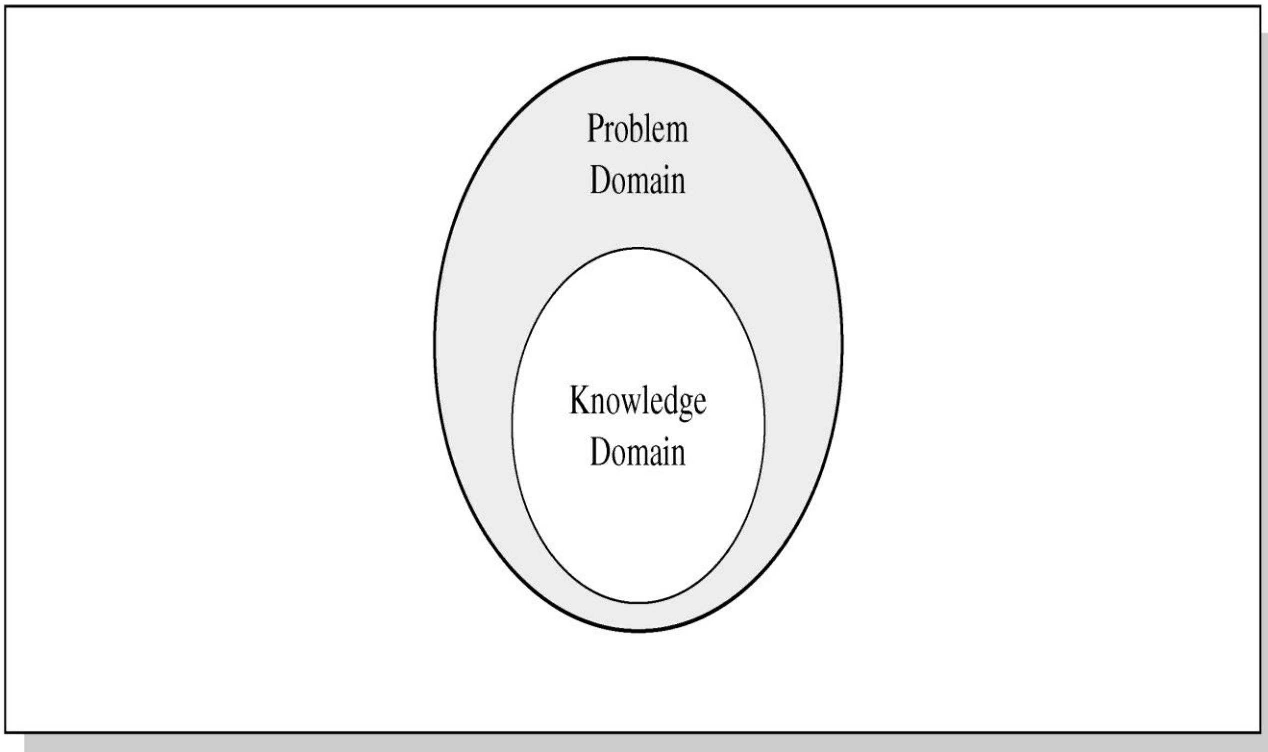
Basic Functions of Expert Systems



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`

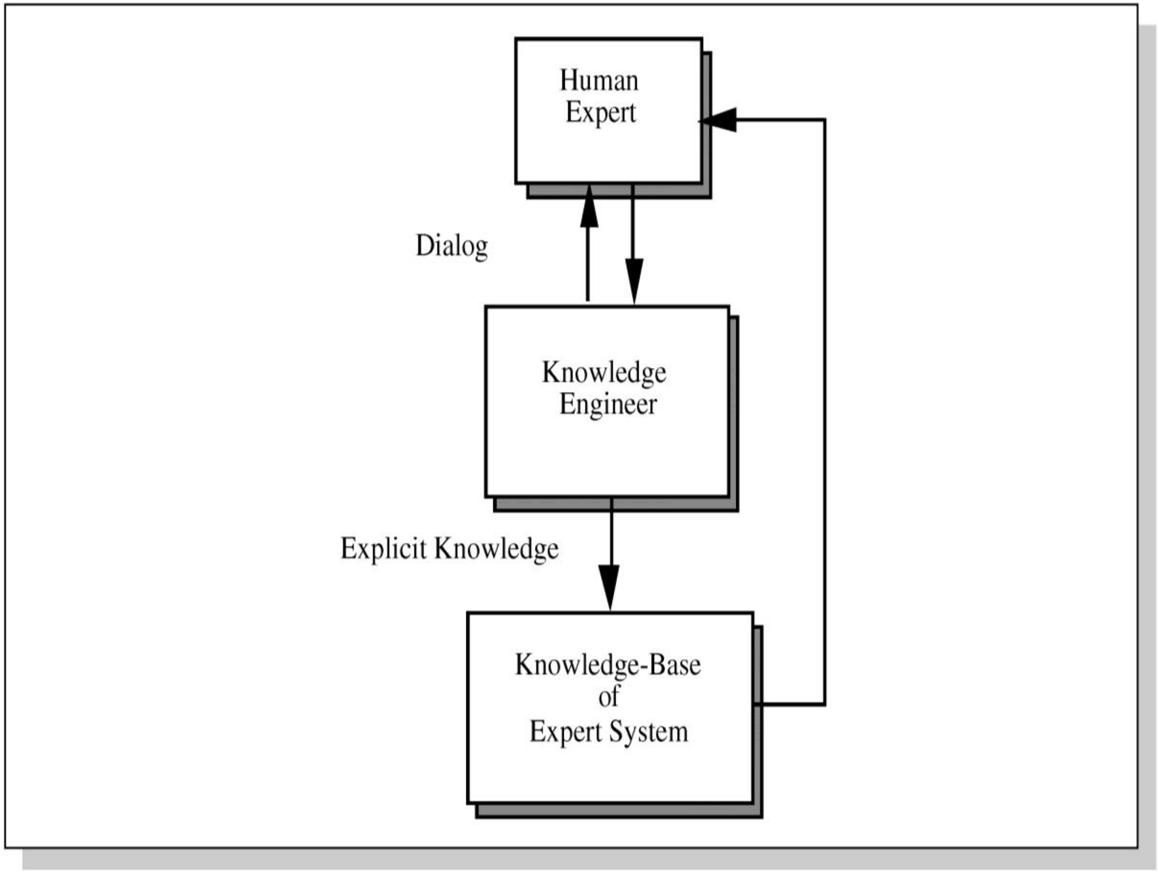
- Knowledge Engineering is the process of building an expert system:

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.

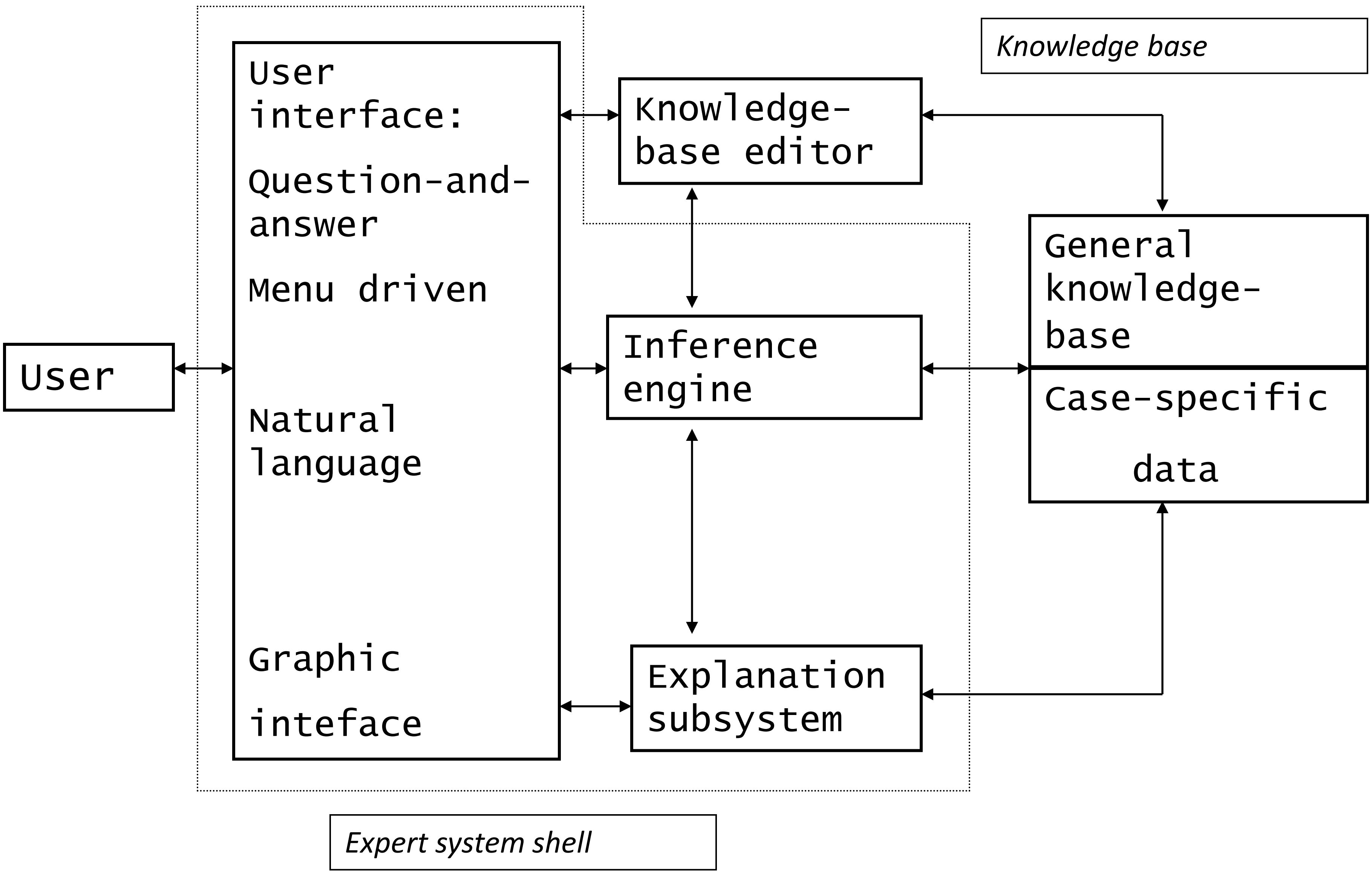
Broad Classes of Expert Systems

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.

Expert System Development Life Cycle

- A Knowledge Engineer or an AI Engineer is skilled in the art of developing Expert Systems (ES).
- A domain expert is an individual who has significant expertise in the domain of the expert system.
- It is not crucial that the domain expert understand AI or expert systems that is one of the functions of the knowledge engineer.
- The knowledge engineer and the domain expert usually work close to each other throughout the Expert System Development Life Cycle process.

Expert System Development Life Cycle

- Expert System Development Life Cycle process includes the following steps.
  - Stage 1: Identification: Determining the characteristics of the problem.
  - Stage 2: Conceptualization: Finding the concept to produce the solution.
  - Stage 3: Formalization: Designing structures to organize the knowledge
  - Stage 4: Implementation: Formulating rules which embody the knowledge.
  - Stage 5: Testing: Validating the rules.

Stage # 1. Identification

- Before we can begin to develop an expert system, it is important to describe, with as much precision as possible, the problem which the system is intended to solve.
- It is not enough simply to feel that an expert system would be helpful in a certain situation
- We must determine the exact nature of the problem and state the precise goals which indicate exactly how the expert system is expected to contribute to the solution.

Stage # 1. Identification

- To begin, the knowledge engineer, who may be unfamiliar with this particular domain, consults manuals and training guides to gain some familiarity with the subject.
- Then the domain expert describes several typical problem states.
- The knowledge engineer attempts to extract fundamental concepts from the similar cases in order to develop a more general idea of the purpose of the expert system.

Stage # 1. Identification

- After the domain expert describes several cases, the knowledge engineer develops a ‘first-pass’ problem description.
- Typically, the domain expert may feel that the description does not entirely represent the problem.
- The domain expert then suggests changes to the description and provides the knowledge engineer with additional examples to illustrate further the problem’s fine points.

Stage # 1. Identification

- Next, the knowledge engineer revises the description, and the domain expert suggests further changes.
- This process is repeated until the domain expert is satisfied that the knowledge engineer understands the problems and until both are satisfied that the description adequately portrays the problem which the expert system is expected to solve.
- The results are evaluated at each stage of the process and compared to the expectations.
- If the results do not meet the expectations, adjustments are made to that stage of the process, and the new results are evaluated.
- The process continues until satisfactory results are achieved.
- This iterative process is shown in the following figure.

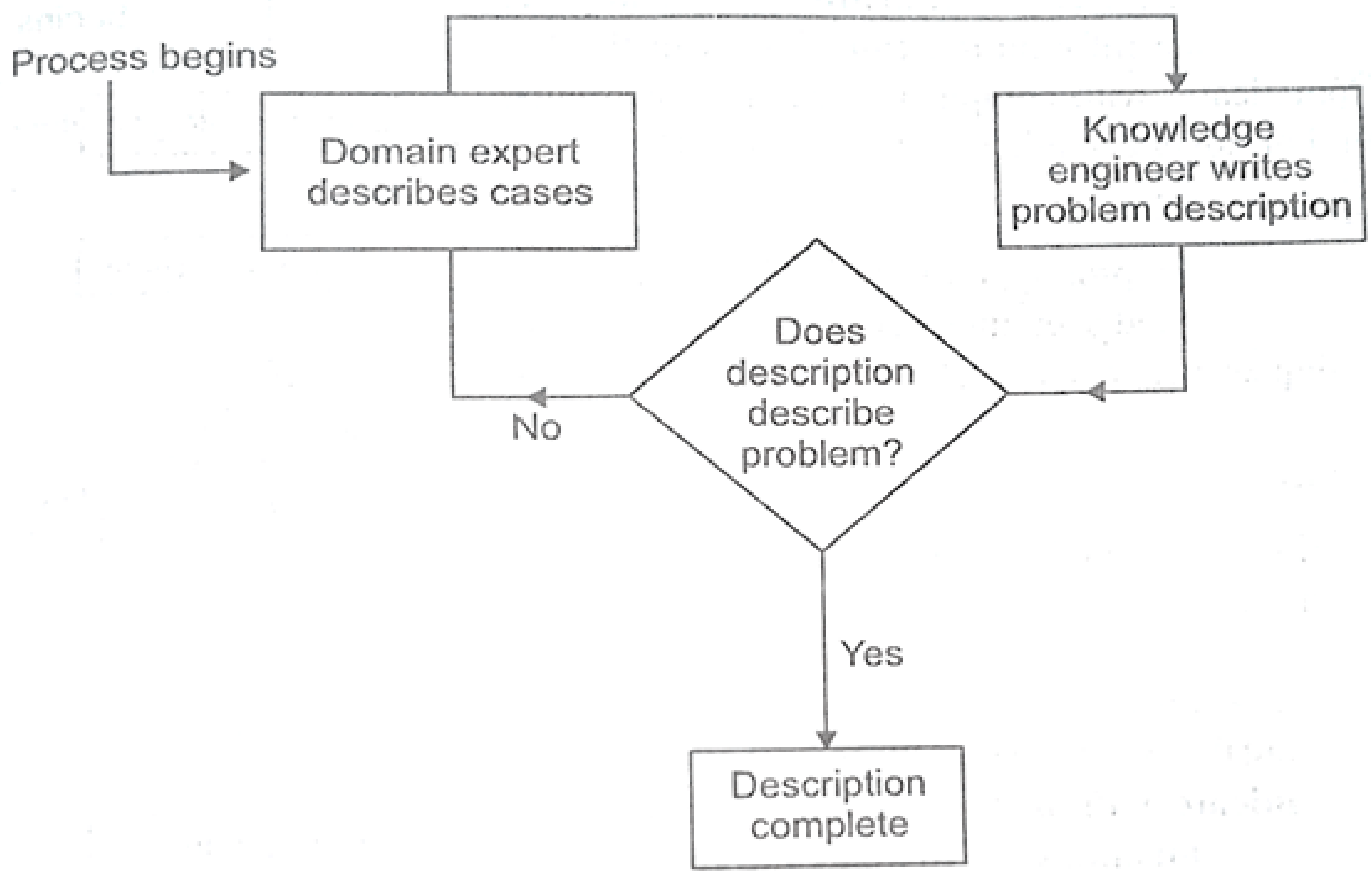


Fig: Iterative Process of identifying the problem in Expert System

Stage # 1. Identification

- It is also important to identify our resources.
- Who is to participate in the development process?
- Does a single domain expert possess all the necessary expertise, or is the domain knowledge distributed over several people in an organisation?
- Can a single knowledge engineer develop the system in a timely fashion, or is it necessary to provide additional technical assistance?
- Domain experts are not the only resources which must be identified.
- It is unusual for all domain knowledge to be embodied in human experts; therefore, more tangible sources of information, such as reference books and manuals, are usually identical and located.

Stage # 2. Conceptualisation

- Once it has been identified for the problem an expert system is to solve, the next stage involves analysing the problem further to ensure that its specifics, as well as generalities, are understood.
- In the conceptualiation stage, the knowledge engineer frequently creates a diagram of the problem to depict graphically the relationships between the objects and processes in the problem domain.
- It is often helpful at this stage to divide the problem into a series of sub-problems and to diagram both the relationships among the pieces of each sub-problem and the relationships among the various sub-problems.

Stage # 2. Conceptualisation

- As in the identification stage, the conceptualization stage involves a circular procedure of iteration and reiteration between the knowledge engineer and the domain expert. When both agree that the key concepts-and the relationships among them-have been adequately conceptualized, this stage is complete.
- Not only is each stage in the expert system development process circular, the relationships among the stages may be circular as well.
- Since each stage of the development process adds a level of detail to the previous stage, any stage may expose a weakness in a previous stage.

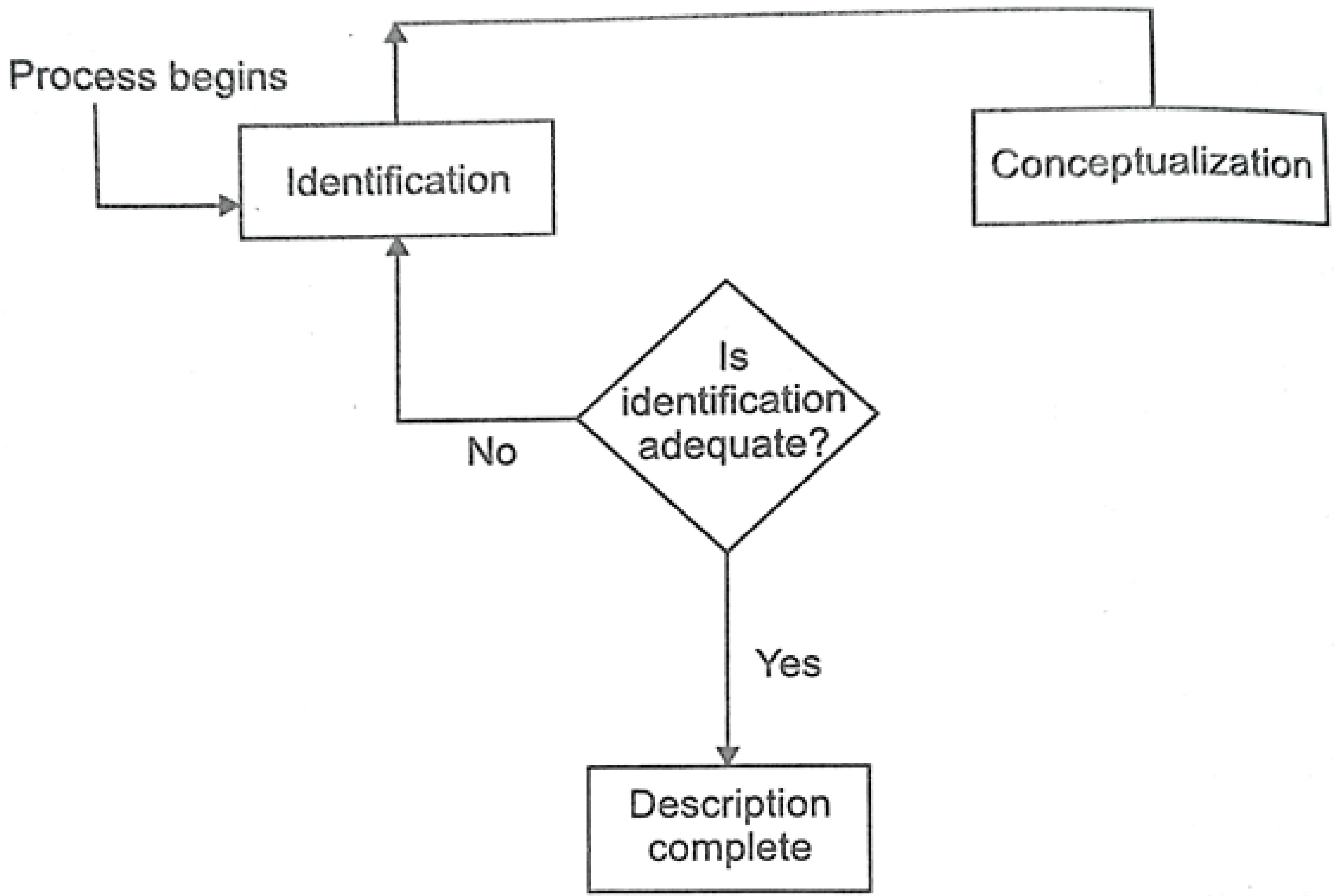


Fig: Conceptual Phase of Expert System Development Life Cycle

Stage # 3. Formalisation (Designing)

- In the two preceding stages, no effort has been made to relate the domain problem to the artificial intelligence technology which may solve it.
- During the identification and formalization stages, the focus is entirely on understanding the problem.
- Now, during the formalization stage, the problem is connected to its proposed solution, an expert system is supplied by analyzing the relationships depicted in the conceptualization stage.
- The knowledge engineer begins to select the techniques which are appropriate for developing this particular expert system.

Stage # 3. Formalisation (Designing)

During formalization, it is important that the knowledge engineer be familiar with the following:

1. The various techniques of knowledge representation and intelligent search techniques used in expert systems.

2. The expert system tools which can greatly expedite the development process.

3. Other expert systems which may solve similar problems and thus may be adaptable to problem at hand.

Stage # 3. Formalisation (Designing)

- Often it is desirable to select a single development technique or tool which can be used throughout all segments of the expert system.
- However, the knowledge engineer may determine that no particular technique is appropriate for the entire expert system, making it necessary to use different techniques for different sub-problems.
- Once it has been determined which technique(s) will be used the knowledge engineer starts to develop a formal specification which can used to develop a prototype expert system.

Stage # 3. Formalisation (Designing)

- In the case of a rule-based system, for example, the knowledge engineer develops a set of rules designed to represent the knowledge communicated by the domain expert.
- This is a critical part of the development process, requiring, great skill on the part-of the knowledge engineer.
- Many domain experts can explain what they do but not why; therefore, one of the knowledge engineer’s primary responsibilities is to analye example situations and filter in from those examples a set of rules which describe the domain expert’s knowledge.

Stage # 3. Formalisation (Designing)

- The formalisation process is often the most interactive stage of expert system development, as well as the most time consuming.
- The knowledge engineer must develop a set of rules and ask the domain expert if those rules adequately represent the expert's knowledge.
- The domain expert reviews the rules proposed by the knowledge engineer and suggests changes, which are then incorporated into the knowledge base by the knowledge engineer.
- As in the other development stages, this process also is iterative: the rule review is repeated and the rules are refined continually until the results are satisfactory.
- It is not unusual for the formalisation process of a complex expert system to last for several years.

Stage # 4. Implementation

- During the implementation stage the formalised concepts are programmed into the computer which has been chosen for system development, using the predetermined techniques and tools to implement a ‘first-pass’ (prototype) of the expert system.
- Theoretically, if the methods of the previous stages have been followed with diligence and care, the implementation of the prototype should proceed smoothly.
- In practice, the development of an expert system may be as much an art as it is a science, because following all the rules does not guarantee that the system will work the first time it is implemented.
- In fact, experience suggests the opposite. Many scientists actually consider the prototype to be a ‘throw-away’ system, useful for evaluating progress but hardly a usable expert system.

Stage # 4. Implementation

- If the prototype works at all, the knowledge engineer may be able to determine if the techniques chosen to implement the expert system were the appropriate ones.
- On the other hand, the knowledge engineer may discover that the chosen techniques simply cannot be implemented.
- It may not be possible, for example, to integrate the knowledge representation techniques selected for different sub-problems.
- At that point, the concepts may have to be re-formalised, or it even may be necessary to create new development tools to implement the system efficiently.
- Once the prototype system has been refined sufficiently to allow it to be executed, the expert system is ready to be tested thoroughly to ensure that it expertise's correctly.

Stage # 5. Testing (Validation, Verification and Maintenance)

- The chance of prototype expert system executing flawlessly the first time it is tested are so slim as to be virtually non-existent.
- A knowledge engineer does not expect the testing process to verify that the system has been constructed entirely correctly.
- Rather, testing provides an opportunity to identify the weaknesses in the structure and implementation of the system and to make the appropriate corrections.
- Depending on the types of problems encountered, the testing procedure may indicate that the system was implemented incorrectly, or perhaps that the rules were implemented correctly but were poorly or incompletely formulated.
- Results from the tests are used as ‘feedback’ to return to a previous stage and adjust the performance of the system.

Stage # 5. Testing (Validation, Verification and Maintenance)

- The chance of prototype expert system executing flawlessly the first time it is tested are so slim as to be virtually non-existent.
- A knowledge engineer does not expect the testing process to verify that the system has been constructed entirely correctly.
- Rather, testing provides an opportunity to identify the weaknesses in the structure and implementation of the system and to make the appropriate corrections.
- Depending on the types of problems encountered, the testing procedure may indicate that the system was implemented incorrectly, or perhaps that the rules were implemented correctly but were poorly or incompletely formulated.
- Results from the tests are used as ‘feedback’ to return to a previous stage and adjust the performance of the system.

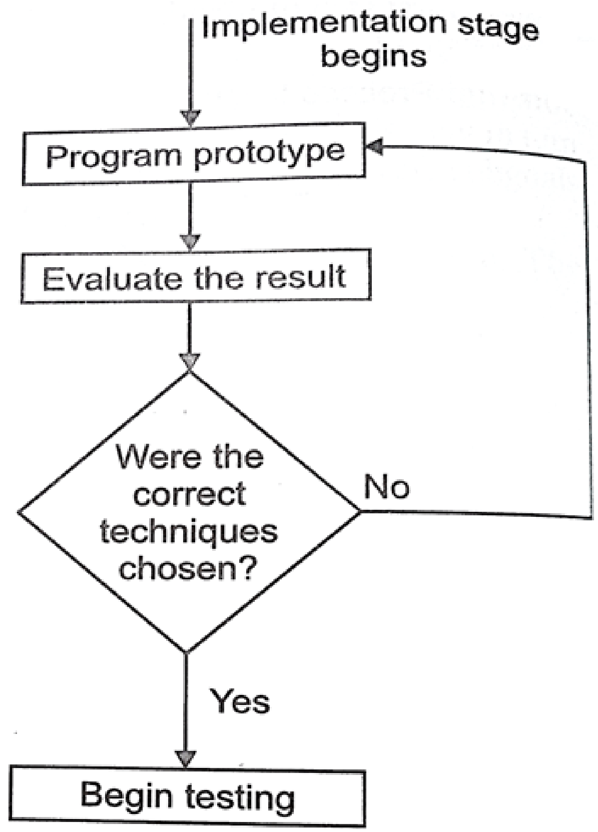


Fig: Implementation Phase of Expert System Development Life Cycle

Step5: Testing: Validating the rules.

- The chances of a prototype expert system executing flawlessly are very less initially.
- A knowledge engineer doesn’t expect the testing process to verify that the system has been constructed entirely correctly.
- Testing provides an opportunity to identify the weaknesses in the structure and the implementation of the system.
- Testing includes the following steps.

Step5: Testing: Validating the rules.

- i. The system implements correctly or incorrectly.
- ii. Rules implement correctly or not.
- iii. The System uses for testing for both simple and complex problems by domain experts to uncover more defects.
- iv. An Expert System is finally tested to be successful only when it is operated at the level of a human expert.
- v. The testing process is **NOT** complete until it indicates that the solutions suggested by the expert system are consistently valid.
- vi. Expert systems are typically interactive, they work in question-and-answer form. This interaction between users and the Experts system continues until the system can conclude.

Step5: Testing: Validating the rules.

- i. The system implements correctly or incorrectly.
- ii. Rules implement correctly or not.
- iii. The System uses for testing for both simple and complex problems by domain experts to uncover more defects.
- iv. An Expert System is finally tested to be successful only when it is operated at the level of a human expert.
- v. The testing process is **NOT** complete until it indicates that the solutions suggested by the expert system are consistently valid.
- vi. Expert systems are typically interactive, they work in question-and-answer form. This interaction between users and the Experts system continues until the system can conclude.

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.

Building Model-Driven Decision Support System (MDSS)

- As the name itself speaks, a model-driven decision support system utilizes a model to solve problems or aid in decision making.
- A model can be
  - statistical,
  - financial,
  - mathematical,
  - analytical,
  - simulation
  - optimization.

Building Model-Driven Decision Support System (MDSS)

- A model-driven DSS may employ a single model or a combination of models, depending upon the specific needs of its users.
- Simple models provide basic functionality while combination of two or more models lets users analyze complex data.
- Model-driven DSS are generally not data intensive. Rather they use parameters entered by decision makers and help them analyze a situation.
- They generate optimal solutions that are consistent with time and resource constraints.

Building Model-Driven Decision Support System (MDSS)

- The scope of model-driven DSS is huge and can be further enhanced by integrating web-based applications.
- When developing proprietary MDSS, it's important to understand modeling and analytical tools, their working and scope.
- Building model-driven DSS requires a considerable level of expertise.
- Managers and DSS analysts need to work closely to develop an efficient system, which is scalable, versatile and easy to integrate and use.

Modeling Decision Situations

- Model-driven DSS can be used to aid decision making in a variety of situations.
- It can assist managers in making:
  - Credit and lending decisions
  - Product demand forecasting
  - Budgeting decisions
  - Marketing decisions
  - Production forecasting decisions
  - Resource allocation decisions
  - Project planning
  - Investment decisions

Building Model-Driven Decision Support System (MDSS)

- Each MDSS has a clear objective and specific purpose.
- It deploys a model.
- Consequently, a lot of thought goes into deciding what models should be included in a model-driven DSS.
- MDSS usually carries out sensitivity analysis or ‘what if’ analysis.
- However, the users must remember that the system doesn’t make a decision.
- It only generates alternatives that are to be analyzed and assessed by decision-makers.

Building Model-Driven Decision Support System (MDSS)

- The most important aspect of a model-driven DSS is the model it uses for decision making.
- This means that the selection of a model is the most crucial step in building an MDSS.
- **Modeling**
    - It is the process of identifying an appropriate model for a prospective model-driven decision support system.
    - It goes through various phases in a chronological manner, beginning from problem identification.
    - Once modeling is done, it’s vital to validate the selected model, to ensure it works well and generates appropriate results.
    - Model validation is done by comparing model’s output and the actual behavior of the event.

Building Model-Driven Decision Support System (MDSS)

- **Assumptions & Forecasts**
  - Assumptions are predictions or best guesses.
  - Each model has certain assumptions about the time and risk involved in a particular situation.
  - These results are tested through sensitivity or what if analysis.
  - Assumptions play an important role in defining a problem and identifying and dealing with uncertainty.
  - Decision makers form a hypothesis and attempt to predict results.
  - Basis the outcome, a hypothesis is either accepted or rejected.
- Model-driven DSS are designed assuming any of the analyses – static and dynamic.

Building Model-Driven Decision Support System (MDSS)

- **Static Analysis:**
  - This type of analysis doesn't take into consideration the long term response of a system.
  - It takes a single snapshot of a situation and assumes that it will remain stable all through and won't change.
  - Static analysis is done when a situation in which company makes a decision is static in nature.
- **Dynamic Analysis:**
  - Dynamic analysis is testing a program or a software system in real-time.
  - This method considers that the situation changes over time, due to any reason, such as cost, rules and regulations, time, etc.

Building Model-Driven Decision Support System (MDSS)

- Decision makers and DSS analysts must identify whether it is appropriate to assume certainty, uncertainty or risk in a situation.
- **Certainty:**
  - When adequate information about a situation is available.
  - Models based on certainty/static analysis tend to yield optimal solutions.
- **Uncertainty:**
  - When information available is vague, unpredictable or unreliable.
  - It’s important to acquire more information to find an appropriate model.
- **Risk:**
  - When information is missing.
  - What if analysis is carried to aid decision making.

Building Model-Driven Decision Support System (MDSS)

- **Model Types:** A DSS deploys one or combination of following models:
  - **Explanatory/Descriptive Model:** Describes and explains why something is the way it is and why and how it works.
  - **Contemplative Model:** Forecasts results or outcomes that may be produced from a specific set of parameters.
  - **Algebraic Model:** A high-level modeling system for solving complex equations. It is employed to optimize a variable or equation. The best part is that it can handle several simultaneous equations.

Building Model-Driven Decision Support System (MDSS)

- A DSS with any one of above models performs a single function whichever it is meant to do while a DSS with multiple models is a complete system to perform all three tasks, including:
  - Identifying relationships between variables
  - Forecasting results based on changes or parameters
  - Deciding to what extent a variable can be manipulated

Modeling Decision Situations

- Mathematical and analytical models are the dominant component in a model-driven Decision Support System.
- If a model is needed to understand a situation, then a model-driven DSS can potentially deliver the needed representation to managers.
- DSS Analysts can create a wide variety of alternative model-driven DSS.
- So actually building a model-driven DSS involves resolving a number of important design and development questions.

Modeling Decision Situations

- Models can help managers understand financial, marketing, and many other business decisions.
- One major issue that must be resolved is the purpose of a proposed model-driven DSS.
  - Is the purpose to assist in credit and lending decisions, budgeting, or product demand forecasting?
  - Will the system be used routinely in a decision process or as part of a special study?

Modeling Decision Situations

- Each model-driven DSS should have a clearly stated and specific purpose.
- To accomplish the specific purpose of a system, more than one type of model is sometimes used in building the model-driven DSS.
- So, a second issue is what models should be included in a specific system.
- The tasks involved in building model-driven DSS are complex enough that a modeling specialist is usually needed on a development team for a large-scale system.

Modeling Decision Situations

- End users should only develop model-driven DSS for one-time and special purpose decision support needs.
- Therefore, managers must confront the issue of who should build a planned or contemplated model-driven DSS.
- In many specific DSS, a model produces outputs displayed for users.
- Also, the decision variables of model-driven DSS are frequently manipulated directly by managers.

Approaches for Developing Model-driven DSS

- Model-driven DSS have been built using
  - statistical software packages,
  - forecasting software,
  - modeling packages, and
  - end-user tools like spreadsheets.
- In all of these development environments, the goal is the same: to build a model that can be manipulated and tested.

Approaches for Developing Model-driven DSS

- This capability of changing a parameter in a model-driven DSS is called “What if?” analysis and expanded testing of model parameters is called sensitivity analysis.
- The results from using a model-driven DSS in a situation are analyzed and evaluated by decision makers.

A typical modeling process

- Usually a modeling process begins with identification of a problem and analysis of the requirements of the situation.
- It is advisable to analyze the scope of the problem domain and the forces and dynamics of the environment.
- The next step is to identify the variables for the model. The identification of decision variables and their relationships is very important.
- One should always ask if using a model is appropriate.
- If a model is appropriate, then one asks what variables and relationships need to be specified, using an appropriate modeling tool.

A typical modeling process

- An influence diagram can be used to examine the variables and relationships.
- An influence diagram is an intuitive visual display of a decision problem.
- It depicts the key elements, including decisions, uncertainties, and objectives as nodes of various shapes and colors. It shows influences among them as arrows.
- After examining the variables and relationships among them, a solution method or methods need to be chosen.

A typical modeling process

- Also, an analyst need to specify assumptions and make any needed forecasts.
- Forecasting variables or parameters is sometimes part of the construction of a model.
- Building a computerized system also involves integrating models and other DSS components like data files and data analysis procedures.
- Model-driven DSS need to be validated, evaluated, and managed.

A typical modeling process

- Model validation is the process of comparing a model’s output with the actual behavior of the phenomenon that has been modeled.
- Validation attempts to answer the question, “Have we built the right model of the situation?”
- In other words, model validation refers to the process of confirming that the model actually achieves its intended purpose.
- In most situations, this will involve confirmation that the model is predictive under the conditions of its intended use.

Importance of model assumptions

- Assumptions are untested beliefs or predictions.
- Assumptions are important in building many models because one is projecting or anticipating results.
- A decision maker can test assumptions using “what if” or sensitivity analysis before accepting the results of the model.
- DSS analysts and managers need to make assumptions about the time and risk dimensions for a situation.
- Model driven DSS can be designed assuming either a static or dynamic analysis.

Importance of model assumptions

- Making either assumption about changes in a decision situation has advantages and disadvantages.
- Static analysis is based on a “single snapshot” of a situation.
- Everything occurs in a single interval, which can be a short or long duration.
- A decision about whether a company should make or buy a product can be considered static in nature.
- A quarterly or annual income statement is static.
- During a static analysis, it is assumed that there is stability in the decision situation.

Importance of model assumptions

- Dynamic analysis is used for situations that change over time.
- A simple example would be a five-year profit projection, where the input data, such as costs, prices, and quantities change from year to year.
- Dynamic models are time dependent, e.g., in determining how many cash registers should be open in a supermarket, it is necessary to consider the time of day.
- This time dependence occurs because in most supermarkets there are changes in the number of people that arrive at the market at different hours of the day.

Importance of model assumptions

- Dynamic models are important because they show trends and patterns over time.
- Also, they can be used to calculate averages per period or moving averages, and to prepare comparative analyses.
- A comparative analysis might examine profit this quarter versus profit in the same quarter of last year.
- Dynamic analysis can provide an understanding of the changes occurring within a business enterprise.

Importance of model assumptions

- The analyses may identify possible solutions to specific business challenges
- The above may facilitate the development of
  - business plans,
  - strategies, and
  - tactics.
- DSS analysts and managers also must examine whether it is appropriate to assume certainty about model parameters in the decision situation.

Importance of model assumptions

- Many financial models are constructed under assumed certainty. “What if” analysis is the primary means of considering risk and uncertainty.
- As previously noted, “what if” analysis is the capability of “asking” or manipulating a model-driven DSS to determine what the effect will be on result variables of changing some of the decision variables.
- The assumptions of DSS analysts and managers limit or constrain the types of models that can be used to build a DSS for the situation.

General Types of Models

- Models transform user inputs and data into useful information.
- A model represents a real situation as an abstract framework.
- A model may be specified in mathematical expressions, in natural language statements or as a computer program.
- Managers can manipulate the input to a model to change outputs.
- Models update files, provide responses to user actions, and perform recurring analytical tasks.

General Types of Models

- Tool labels like optimization and simulation are often used to describe categories or types of models
- Explanatory model
  - It describes what has occurred to create current results or outcomes, and it provides an explanation or analysis of a situation.
  - For example, the model Sales = f (Advertising, Number of Salespersons) may be based on a correlation of advertising and the number of salespersons with sales in prior quarters

General Types of Models

- Algebraic model
  - It indicates which values must be introduced into a system of simultaneous equations to create a specific outcome.
  - A manager specifies an outcome and a starting point, and then runs the model.
  - This type of model helps managers gain insight about what variables must be manipulated and to what extent.

General Types of Models

- Explanatory models versus Algebraic models
  - Explanatory models are descriptive models that describe situations.
  - Algebraic models are predictive models
- DSS with multiple model types
  - A model-driven DSS may include multiple model types, e.g., a specific model-driven DSS may include:
    1. An explanatory regression model that identifies relationships among variables,
    2. A financial model of a pro forma income statement, and
    3. An algebraic optimization model like linear programming.

General Types of Models

- Some models are standard components in DSS development packages, and some must be custom-programmed.
- A DSS analyst chooses appropriate models.
- Once models have been chosen, a decision must be made to build the models, to use “ready-made” models, or to modify existing models.
- The software used for creating the model component also needs to be linked to any data and user interface.
- The user interface provides the functionality so that a decision support analyst or a decision maker can interact with the model.

General Problem Types

- Management Scientists have been analyzing and trying to solve business problems for more than 50 years.
- During that time a variety of problem types that can potentially be analyzed with quantitative models have been identified.
- Some of these include:

1. Cost-benefit analysis
2. Forecasting
3. Finance and investment
4. Inventory control and stockout
5. Location, allocation, distribution & transportation
6. Manpower planning and assignment
7. Project planning & control
8. Queuing and congestion
9. Reliability and replacement policy
10. Sequencing and scheduling

General Types of Models

1. Cost-benefit analysis: Given the decision maker's assessment of costs and benefits, which choice should be recommended?
2. Forecasting: Using time series analysis to answer questions such as: What will demand be for a product? What are the sales patterns? How will sales affect profits?
3. Finance and investment: How much capital is needed? How much will the capital cost?
4. Inventory control and stockout: How much stock should be held? When to order more? How much should be ordered?
5. Location, allocation, distribution & transportation: Where is the best location for an operation? How big should facilities be? What resources are needed? Are there shortages?

General Types of Models

6. Manpower planning and assignment: How many employees are needed? When?
7. Project planning and control: How long will a project take? What activities are most important? How should resources be used?
8. Queuing and congestion: How long are waiting lines? How many servers are needed? What service level is provided?
9. Reliability and replacement policy: How well is equipment working? How reliable is it? When should it be replaced?
10. Sequencing and scheduling: What job is most important? In what order should jobs be completed?

General Types of Models

- Eeach of these general problem types can occur in situations where a model-driven DSS could support one or more decision makers.
- These 10 common Management Science problem types can be analyzed using five general categories of quantitative models:
  1. accounting and financial models,
  2. decision analysis models,
  3. forecasting models,
  4. network and optimization models, and
  5. simulation models.

Accounting and Financial Models

- These model-driven decision support systems aid in decision making in various situations related to accounting and financial management.
- The examples include:
  - Break-Even Analysis
  - Budget Financial Model
  - Pro Forma Financial Statements
  - Ratio Analysis

Accounting and Financial Models

- **Break-Even Analysis**
  - It aids managers in determining a break-even point for a product.
  - It helps establish a what-if selling price and analyzing the relationship between various related components – prices, marketing spend and profits.
  - The process begins by assuming fixed and variable costs. Profit is set at zero.
  - It helps determine a break even cost of a product at which the company is neither loss nor makes profit.

Accounting and Financial Models

- **Budget Financial Model:**
  - It is typically an enterprise-wide application. Many companies use such systems for budget planning and forecasting.
- **Pro Forma Financial Statements:**
  - This model summarizes the anticipated financial results for a specific time period in future.
  - Costs are estimated based on past data, gross sales are predicted and profit or loss is then calculated on these relationships.
- **Ratio Analysis:**
  - This helps a business in evaluating its financial statements.
  - It makes financial data more meaningful, by showing logical relationships between data.

Accounting and Financial Models

- Basic steps for building a financial model

1. Gather historical data: Get last three years of financial data for the company.

2. Calculate ratios and metrics: Using the historical data, calculate historical ratios and metrics, like growth margins and rates, asset turnover ratios, and inventory changes.

3. Make informed assumptions: Using historical data, ratios, and metrics, projections and assumptions, calculate future growth margins and rates, assets that may turnover, and projected changes in inventory.

4. Create a forecast: Use the above data and reports to forecast the usual accounting documents, such as future income, balance sheet, and cash flow statements.

5. Value the company: Value the company using Discounted Cash Flow (DCF) method

6. Review: Use the drafted statements to decide how different scenarios may play out.

Decision Analysis Models

- The main job of decision analysis models is to identify and evaluate alternatives with their respective pros and cons.
- The decision makers then evaluate all the alternatives and pick the one that they think is the best.
- The aim of decision analysis techniques is to:
  - Decompose and restructure the problems
  - Help decision makers gain in-depth understanding of the problem
  - Separate facts and figures from preferences and priorities
  - Help users study the performance of decision alternatives
  - Avoid citing priorities that don’t help in decision making

Decision Analysis Models

- Types of Decision Analysis Models
    - Analytical Hierarchy Process (AHP)
    - Decision Trees
    - Multi-Attribute Utility Analysis (MAUA)
    - Influence Diagrams

Forecasting Models

- Forecasting models form an integral part of a large number of decision support systems. Their main job is to predict the value of interrelated variables at some point of time in future.
- The two main categories of forecasts are:
  - **Short run forecasts:** where the prediction will be used anytime soon mainly in deterministic models
  - **Long run forecasts:** where the prediction is used for long term investment/planning decisions
- Forecasting may include ambiguity as factors on which decisions depend are uncontrollable and dynamic in nature.
- This means that the accuracy of data and time taken in making near-perfect predictions matter a lot.

Forecasting Models

- Types of Forecasting Models
  - Naive Exploration
  - Judgment Methods
  - Moving Average
  - Exponential Smoothing
  - Time Series Extrapolation
  - Regression and Econometric Models

Forecasting Models

Types of Forecasting Models

- **Naive Exploration:** As is explained by the name itself, naive exploration is not a sophisticated prediction. Rather it is simple forecasting that provides limited accuracy. The technique is implemented using a spreadsheet.
- **Judgment Methods:** The predictions or forecasting are based on the perceptions and opinions of experts instead on hard data. It's a subjective estimate used for long-run forecasts where external environment plays a critical role. The results are not very accurate.
- **Moving Average:** Used for short-run forecasts, the predictions are based upon the historical values. DSS with this model is inexpensive and easy to use.

Forecasting Models

Types of Forecasting Models

- **Exponential Smoothing:** Used for short-term forecasts, it alters the historical data mathematically to better reflect the assumptions of a decision maker. Similar to moving average model but claims to obtain better results using exponential smoothing.
- **Time Series Extrapolation:** This method takes into account the economic variables that are measured at consecutive intervals of time. It is believed that the knowledge of past behavior of the variable at successive intervals of time will help understand the behavior of the variables in future better.
- **Regression and Econometric Models:** These types of forecasting models make use of linear and multiple regressions to establish cause and effect relationships. These methods are considered more powerful than time-series but also complex at the same time. They are complex because they use sophisticated models and include more variables. The results obtained are more accurate.

Forecasting Models

Steps involved in developing forecasting models

- There are five major steps that are used to develop a forecasting system:
    - finding a problem,
    - gathering information,
    - choosing the right forecasting model,
    - analyzing the data and
    - verifying performance.

Network and Optimization Models

- Network and optimization models are integrated into a DSS when decisions regarding resource allocation, project control, location, scheduling, transportation, distribution, size, shortages, multinational cash flow management, inventory management and distribution and network need to be made.
- For example:
  - The best location for an operation or manufacturing
  - The resources needed to carry out the operations
  - Most suitable aircraft route to transport products

Simulation Models

- In a DSS context, simulation generally refers to a technique for conducting experiments with a computer-based model.
- One method of simulating a system involves identifying the various states of a system and then modifying those states by executing specific events.
- A wide variety of problems can be evaluated using simulation, including inventory control and stock-out, manpower planning and assignment, queuing and congestion, reliability and replacement policy, and sequencing and scheduling.

Simulation Models

- Often, companies are faced with planning the production of a new product or building a new factory.
- Although these may seem like straightforward analyses, managers need to make many interrelated decisions.
- For example, production of a new product involves decisions regarding equipment, scheduling and control, and manufacturing philosophy.
- Many factors influence these decisions, including the need to meet production volume goals and costs associated with achieving these goals.
- Simulations can help evaluate complex, interrelated decision issues.

Simulation Models

- In a DSS context, simulation generally refers to a technique for conducting experiments with a computer-based model.
- One method of simulating a system involves identifying the various states of a system and then modifying those states by executing specific events.
- A wide variety of problems can be evaluated using simulation, including inventory control and stock-out, manpower planning and assignment, queuing and congestion, reliability and replacement policy, and sequencing and scheduling.
- There are several types of simulation. The major types are probabilistic, time-dependent, and visual simulation.

Simulation Models

- Steps involved in developing a simulation model

1. Identify the problem
2. Formulate the problem
3. Outline a model
4. Model the inputs for the simulation model
5. Translate the model into code or software
6. Verify and validate the model
7. Experiment with alternative models
8. Document and report the simulation’s performance
9. Implement and update the model

Modeling Languages and Spreadsheets

- As models are computerized software program, a number of programming languages can be used for coding.
- Typically the languages used are C++ and Java.
- Moreover, the decision support systems make use of spreadsheets, allowing users to
  - Write values
  - Manipulate data
  - Apply mathematical and statistical formulas
  - Create graphs and visuals
  - Prepare, consolidate and sort reports
- A DSS can be built in two ways: using a DSS generator or using primary DSS development tools.

Modeling Languages and Spreadsheets

- A DSS generator is a package of related hardware and software which provides a set of capabilities to quickly and easily build a Specific DSS.
- DSS development tools are a set of hardware and software elements which facilitate the development of a specific DSS or a DSS generator
- Decision support systems that are built using spreadsheet software can be called spreadsheet-based DSS.
- Spreadsheets can be used to build data-driven, model-driven DSS and solver-based DSS.

Modeling Languages and Spreadsheets

- Spreadsheet are appropriate for building a DSS with small models, for building a DSS prototype or for testing the DSS models, etc.
- The user interface can be build using buttons, lists and the operations can be automated using Macros or Visual Basic.
- Choosing Excel as a DSS generator has a number of advantages: is accessible, is known by a great number of business users, is easy to use, can perform a lot of analysis – optimization, simulation, sensitivity analysis, what-if analysis.

Spreadsheet-Based DSS

- A DSS uses the data residing in spreadsheets and/or databases, models it, processes or analyzes it using problem-specific methodologies, and assists the user in the decision-making process through a graphical user interface
- Spreadsheets are a great option for developing many DSS applications since they are available with almost any operational system and have many features that are relatively easy to learn and can be implemented for a large variety of problems.
- Spreadsheets can be used to create charts, graphs, and other visualizations that help managers to understand complex data and trends.
- This information can help managers to make decisions about resource allocation, marketing strategies, and other business decisions.

Spreadsheet-Based DSS

Spreadsheets application scenarios

- Spreadsheets can be used for decision making in the following scenarios:
  - Budgeting and forecasting
  - Cost analysis
  - Break-even analysis
  - Sensitivity analysis
  - Scenario analysis
  - Data visualization

Spreadsheet-Based DSS

Spreadsheets application scenarios

- Budgeting and forecasting: Managers can create detailed spreadsheets that outline revenue and expenses, and use formulas to calculate project profits and cash flows. The information can be used to make decisions about resource allocation and strategic planning.
- Cost analysis: Managers can use spreadsheets to track and analyze costs associated with specific projects, products or departments. This information can help managers to make decisions about pricing, product mix, and resource allocation.

Spreadsheet-Based DSS

Spreadsheets application scenarios

- Break-even analysis: Spreadsheets can be used to conduct break-even analysis, which helps managers to determine the minimum level of sales required to cover all of the costs associated with producing a product or service. This information can help managers to make decisions about pricing and production levels.

Spreadsheet-Based DSS

Spreadsheets application scenarios

- Sensitivity analysis: Spreadsheets can be used to conduct sensitivity analysis, which involves analyzing the impact of changes in assumptions or variables on financial outcomes. This information can help managers to make decisions about risk management, investments and strategic planning.
- Scenario: Spreadsheets can be used to conduct scenario analysis, which involves analyzing the potential outcomes of different scenarios or situations. This information can help managers to make decisions about strategic planning, risk management, and contingency planning.

Spreadsheet-Based DSS

Spreadsheets application scenarios

- Data visualization: Spreadsheets can be used to create charts, graphs, and other visualizations that helps managers to understand complex data and trends. This information can help managers to make decisions about resource allocation, marketing strategies, and other business decisions.

Spreadsheet-Based DSS

Steps involved in using Spreadsheets for decision making

- The following steps are involved when using spreadsheets to make an informed decision:
  - Gather data
  - Create a Spreadsheet
  - Conduct sensitivity analysis
  - Conduct scenario analysis
  - Evaluate results

How do companies use spreadsheets?

- Many companies use spreadsheets to store information, but they also have other valuable functions.
- Companies use spreadsheets to model and manipulate data sets, create graphical visualizations, and inform future planning, and decision making.
- With popular spreadsheet programs such as Microsoft Excel and alternatives from LibreOffice and GoogleDocs, businesses of any size can put spreadsheets to good use.

How do companies use spreadsheets?

- **Modeling of Information**
  - The main purpose of a spreadsheet is storage and modeling of a data set.
  - Whatever business sector we are in, we will likely have a store of information that is potentially of great value to you.
  - Spreadsheets model financial, statistical and any other numerical data within systems of rows and columns.
  - Each data item is contained within a single cell within these rows and columns.
  - Cells can also contain formulas and references to other cells, so a spreadsheet can include calculations that are automatically processed when the values in the cells are edited.
  - In this sense, spreadsheets continually update when new data is added.

How do companies use spreadsheets?

- **Manipulate and Analyze Data**
  - In addition to storing and modeling data, spreadsheets can manipulate and analyze data sets.
  - Most spreadsheet programs allow users to enter custom formulas, with a range of commonly used preset functions also available.
  - These functions allow us to sort our datasets on particular values, or to filter it, letting us get the particular analysis we need.
  - Spreadsheets also provide conditional processing, where the value in one cell depends on the result of a conditional test on the data.

How do companies use spreadsheets?

- **Visualize Data Graphically**
  - Spreadsheet programs include tools for data visualization.
  - When we have a data set stored within a spreadsheet, we can use this data as the basis for graphical displays such as bar charts, graphs, and pie charts, with a range of charting options we can edit to suit our own needs.
  - These charts can be used within management teams to gain an insight into the data, as well as to communicate this data as part of presentation activities in corporate contexts.

How do companies use spreadsheets?

- **Inform the Decision Making Process**
  - The ability to analyze and present data sets means that spreadsheets are used by many companies to inform the decision-making process.
  - Any effective future planning activity naturally needs to be informed by a clear picture of the present and past, a task spreadsheets are ideally placed to assist.
  - Spreadsheets allow us to carry out performance measurement.
  - For example, within a retail or service organization, we could use spreadsheet data transformed into a bar chart to see how well our business performs at different times of the year.
  - We can also carry out speculative analyses, for example, estimating sales projections and calculating the effect that these would have on the business as a whole.