UNIT 8: COST/BENEFIT ANALYSIS

UNIT STRUCTURE

8.1 Learning Objectives

8.2 Introduction

8.3 Data Analysis

8.4 Cost and Benefit Categories

8.5 Procedure for Cost/Benefit Determination

8.6 Let Us Sum Up

8.7 Further Readings

8.8 Answers to Check Your Progress

8.9 Model Questions

8.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- get an idea about data analysis
- learn about data analysis process
- acquire the knowledge of cost and benefit analysis
- learn about various categories of cost/benefit
- learn the procedure for cost/benefit determination

8.2 INTRODUCTION

In the previous unit, we discussed about the feasibility study. Once the feasibility study for a system is completed and finalised, a feasibility report is generated. It is to be followed by data analysis process. Data gathering is only one part of systems analysis. Subsequently, cost and benefit analysis are processed. The costs and benefits of each alternative guide the selection of the best system for the job. Cost/ benefit analysis is a procedure that gives a picture of the various costs, benefits and rules associated with a system. The determination of costs and benefits is based on some steps. This unit covers data analysis and its different phases. Also, it describes the cost/benefit analysis and various categories of cost/

benefit. The next unit will explore the topics related to the process and stages of system design.

8.3 DATA ANALYSIS

Data analysis is a prerequisite to cost/ benefit analysis. System investigation and data gathering lead to an assessment of current findings. From the data analysis, the design requirements for a system can be identified which include better customer service, faster information retrieval, quicker report generation, minimal time consumption, accuracy, reduce data redundancy, improved staff efficiency and lower processing & operating costs.

Data analysis is defined as a process of cleaning, transforming, and modelling data to discover useful information for business decision-making. The purpose of data analysis is to extract useful information from data and taking the decision based upon the data analysis.

To achieve these design objectives, several alternatives must be evaluated. The analyst selects those that are feasible economically, technically and operationally. Each approach has its benefits and drawbacks. An analysis of the costs & benefits of each alternative guides the selection process.

Data Analysis Process

The data analysis is nothing but a process of gathering information by using a proper application or tool which allows us to explore the data and find a pattern in it. Based on that information and data, we can either take decisions or be able to get ultimate conclusions.

Data Analysis process consists of the following phases:

- Data Requirement Gathering
- Data Collection
- Data Cleaning
- Data Analysis
- Data Interpretation
- Data Visualization

Data Requirement Gathering

Initially, we have to think about the reason or purpose of doing this data analysis. Then we have to decide what type of data analysis we require to do. Finally, in this phase, we have to focus on what to analyze, how to measure it, and what measures we have to use to do this analysis.

Data Collection

After requirement gathering, a clear idea about what things we have to measure and what should be the findings can be achieved. Based on requirements, it is time to collect data from various sources by maintaining a log with a collection date and source of the data. This collected data must be processed or organized for analysis.

Data Cleaning

It is to be noted that collected data may contain duplicate records, white spaces or errors which may not be useful or irrelevant to analysis. Hence, that form of data should be cleaned and made error free. This phase must be done before analysis because based on data cleaning, output of analysis will be closer to our expected outcome.

Data Analysis

Once the data is collected, cleaned, and processed, it is in ready form for analysis. While manipulating data, we may be experienced either with the exact information we need, or we might need to collect more data. During this phase, various data analysis tools and software can be used which will help us to understand, interpret, and derive conclusions based on the requirements.

Data Interpretation

After analysing data, it is finally turn to interpret the results. We can choose the way to express our data analysis either in textual form or may be in a table or a chart form. From these, we can decide our best course of action.

Data Visualization

Data visualization is very important step by which data can be presented in the form of charts and graphs so that it will be easier for the human brain to understand and process it. Data visualization often used to discover unknown facts and trends. By observing relationships and comparing datasets, we can find a way to find out meaningful information.

Summary

- Data analysis refers to a process of cleaning, transforming and modelling data in order to discover useful information for business decision-making.
- Data Analysis consists of Data Requirement Gathering, Data Collection, Data Cleaning, Data Analysis, Data Interpretation and Data Visualization phases.
- Types of Data Analysis are Text, Statistical, Diagnostic, Predictive, Prescriptive Analysis

Cost/ benefit analysis

It is a procedure that gives a picture of the various costs, benefits and rules associated with a system. In performing Cost benefit analysis (CBA), it is important to identify different factors associated with cost and benefit. Developing an IT application is considered an investment for an organization. However, after developing that application, it provides the organization with profits. Profits can be monetary or in the form of an improved working environment. However, it carries risks, because in some cases an estimate can be wrong. Cost benefit analysis helps to give management a picture of the costs, benefits and risks. It usually involves comparing alternate investments.

8.4 COST AND BENEFIT CATEGORIES

Cost/benefit determines the benefits and savings that are expected from the system and compares them with the expected costs. The cost of an information system involves the development cost and maintenance cost. The development costs are one-time investment whereas maintenance costs are recurring. The development cost is basically the costs incurred during the various stages of the system development.

In developing cost estimates, there are various cost factors/elements which include hardware, personnel, facility, operating, and supply costs. Hardware costs relate to the procurement of computer and peripherals. Personal costs relate to staff salaries and benefits as well as for those involved in developing the system. Operating costs are associated with the

day-to-day operation of the system: the amount relates to the number of shifts and efficiency of the operating staff. Supply costs are variable costs because these may increase with the increase use of paper, ribbons, disks, etc.

A system is expected to provide benefits to the stack holders. Benefit can be computed by subtracting costs from the total income. Benefit is accumulated either by increasing income, or decreasing costs, or by both. The system will provide some benefits also. Benefits can be tangible or intangible, direct or indirect. In cost benefit analysis, the first task is to identify each benefit and assign a monetary value to it. The two main benefits are improved performance and minimized processing costs.



CHECK YOUR PROGRESS

Q1: Choose the correct options from the followings:

i) Data Analysis is a process of _____.

A. cleaning

B. modelling

C. transforming

D. All of the above

ii) Data analysis is accomplished by _____ phases.

A. 5

B. 6

C. 7

D. 8

iii) Which of the following is called variable cost?

A. Personnel cost

B. Operating cost

C. Supply cost

D. None of these

8.5 PROCEDURE FOR COST/BENEFIT DETERMINATION

There exists a difference between expenditure and investment. We usually spend to get something, but we invest to realize a return on the investment. Designing a computer-based system is considered an investment. Costs are incurred throughout its system development life cycle. In contrast, benefits are observed in the form of reduced operating costs, improved corporate image, staff efficiency, or revenues. The basic function of cost / benefit analysis is to what extent benefits outweigh costs.

Cost/benefit analysis gives a picture of the various costs, benefits and rules associated with a system. The costs and benefits can be determined by maintaining the following steps:

1. By Identifying the costs and benefits pertaining to given project.
2. Classifying the various costs and benefits for analysis.
3. Selecting a method of evaluation.
4. Interpreting the results of the analysis.
5. Taking an action.

Costs and benefits identification

There are two types of identification: easily identifiable and non-identifiable. In easily identifiable, the costs and benefits can easily be identified, such as direct cost towards the price of a hard disk can be identified from company invoice payments. Again, direct benefits often associate to direct costs, especially savings from decreasing costs in questioning activity. In the second type, the costs and benefits are not easily identifiable. An example of this type is the opportunity costs and opportunity benefits. These are the costs or benefits forgone by selectingone alternative over another one. They are not visible in the organization accounts and therefore they are not easy to identify.

Classifications of Costs and Benefits

Costs and benefits can be categorized as:

Tangible or Intangible Costs and Benefits

The cost and benefits which can be measured are called tangible. An outlay of cash for a specific item is known as a tangible cost. Hardware costs, salaries for human resources, software cost are all tangible costs. They are readily identified and measured. The purchase of hardware or software, personnel training, and employee salaries are example of tangible costs. On the other hand, costs which are known to exist but whose value cannot be measured are referred to as intangible costs. The cost of breakdown of an online system during banking hours is an example of intangible cost. Employee morale issues caused by a new system are called intangible costs.

Benefits may also be categorised into tangible or intangible classes. Like costs, they are often difficult to specify accurately. Completing jobs in less hours, producing reports without any errors are examples of tangible benefits which are quantifiable. More customer satisfaction, improved company status, etc are all intangible benefits. Both tangible and intangible costs and benefits should be considered in the evaluation process. Although management often tends to ignore intangibles, but according to Oxenfeldt, considering a zero value on intangible benefits is wrong and outcome of the evaluation may be quite different.

Direct or Indirect Costs and Benefits

From the cost accounting point of view, the costs are treated as either direct or indirect. Direct costs are having money value associated with a project. They are directly applied to the operation. Purchasing of a box of DVDs for Rs. 200/- is considered direct costs. Direct benefits are also attributable to a given project. For example, if the proposed system that can handle more transactions say 25% more than the present system then it is direct benefit.

Indirect costs are attributed from the operations that are not directly associated with a given system or activity. Insurance, maintenance, heat, light, air conditioning are all indirect costs. These are also termed as overhead. Indirect benefits are realised as a by-product of another activity or system. Direct and indirect costs/benefits can be identified for tangible costs and benefits respectively.

Fixed or Variable Costs and Benefits

Some costs and benefits are fixed regardless of how well a system is used. Fixed costs which are also known as sunk costs don't change. Once encountered, they will not recur. Depreciation of hardware, Insurance, etc. are all fixed costs. In contrast, variable costs are incurred on regular basis (weekly or monthly). Recurring period may be weekly or monthly depending upon the system. They are proportional to the work volume and continue as long as system is in operation.

Fixed benefits are also constant. For example, a decrease in the number of human resources by say 25% by virtue of the use of computer system may incur a fixed benefit. On the other hand, variable benefits are realized on a regular basis. Benefits are improved performance and minimized processing costs.



CHECK YOUR PROGRESS

Q2: Choose the correct options from the followings:

i) Benefits are realised in the form of _____.

A. operating costs

B. improved corporate image

C. staff efficiency

D. All of the above

ii) Which of the following is an example of tangible cost?

A. Employees morale issues

B. Employees salary

C. Depreciation of hardware

D. None of these

Select evaluation method

Once all financial data are identified and broken down into cost categories, the system analyst must proceed to the next step to select a method of evaluation. Although many evaluation methods are proposed, but the commonly employed methods are:

i. Net benefit analysis
ii. Present value analysis
iii. Net present value
iv. Payback analysis
v. Break-even analysis
vi. Cash flow analysis

Net benefit analysis:

Net benefit analysis process subtracts total costs from total benefits and hence, it is quite easy to compute, easy to interpret and easy to present. The main disadvantage in this analysis is that it does not account for the time value of the money and does not discount future cash flow. However, the time value of money is most important in evaluation process which is usually expressed in the form of interest on the funds to obtain the future value.

Present value analysis:

Present value analysis calculates the costs and benefits of the system in terms of today's value of the investment and compares with all alternatives. A critical factor to consider in computing present value is a discount rate equivalent to the forgone amount that the money could earn if it were invested in a different project.

The formula for finding present value analysis is

$$P=F/(1+i)^n$$

Where P indicates present value of investment, F refers to future value of an investment , i implies interest rate and n indicates number of years.

Net present value:

The net present value is computed by subtracting discounted costs from discounted benefits. This value is relatively easy to calculate and accounts for the time value of money. This value is expressed as a percentage of the investment.

Payback analysis

The payback analysis is an ideal measure of the relative time value of a project. It determines the time it takes for the accumulated benefits equal to the initial investment. It is obviously clear that a profit can be realized sooner

when the payback period will be shorter. This method is easy to compute and allows two or more activities. This is calculated by dividing Overall cost outlay by annual cash return.

Break-even analysis

Break-even is the point where the cost of the candidate system and that of the current one are equal. This method compares costs of the current system and costs of the candidate systems. If the cost of candidate system exceeds the costs of existing system then it is called investment period. If the cost of candidate system is less than the existing system then it is called return period.

Cash flow analysis

Cash flow analysis is done to keep track of accumulated costs and revenues on a regular basis i.e. it collects detailed information about the flow of cash in the system. Some projects such as those carried out by computer and word processing applications produce revenues from an investment in computerised system.

Interpret results of the analysis and final action

When the evaluation of the project is complete, the results are to be interpreted. This introduces in comparing actual results against a standard or the result of an alternative investment. The interpretation stage and subsequent decision stage are subjective which needs judgment and intuition. After comparing, the final action is to select the best candidate system.

In summary, cost/benefit analysis is an important tool for evaluating projects rather than a replacement of the decision maker. However, this tool has some problems:

i. **Valuation problems.** Intangible costs and benefits are difficult to quantify and tangible costs are generally more noticeable than tangible benefits.
ii. **Distortion problems**. There are two ways of distorting the results of cost/benefit analysis. First one is the intentional favouritism of an alternative for political reasons and the second one is when data are incomplete or missing from the analysis.

iii. **Completeness problems.** Occasionally, an alternative is ignored that compromises the quality of the final choice. Moreover, the costs related tocost/benefit analysis may be on the high side or not enough costs may beconsidered to do a complete analysis.



CHECK YOUR PROGRESS

Q3: Fill up the blanks:

i) _____ compares costs of the current system and costs of the candidate systems.

ii) The main drawback in -_____ analysis is that it does not account for the time value of the money and does not discount future cash flow.

iii) _____ analysis keeps track of accumulated costs and revenues on a regular basis.



8.6 LET US SUM UP

- Data analysis is a prerequisite to cost/benefit analysis.
- From the analysis, the requirements for system design are identified and subsequently, alternative systems are evaluated.
- To estimate cost of a system, it is required to consider hardware, personnel, operating and supply costs.
- Cost/benefit analysis gives a picture of the various costs, benefits, and rules associated with each alternative system.
- Costs and benefits are classified as tangible or intangible, direct or indirect, fixed or variable.
- Tangible costs and benefits can be readily identified and measured. Costs which are known to exist but whose monetary value cannot be measured accurately.
- Once all financial data are identified, the analyst must select a method of evaluation.
- When the evaluation of the project is complete, the results are to be interpreted. This introduces in comparing actual results against a standard or the result of an alternative investment.

- The decision to adopt an alternative system can be highly subjective depending on the analyst's or user's confidence in the estimated cost and benefit values.



8.7 FURTHER READINGS

1. System analysis and Design – Elias M. Awad - Galgotia Publications Pvt Ltd
2. System analysis, design and development – Charles S. Wasson-Wiley InterScience
3. Analysis and Design of Information Systems- Rajaraman-PHI.



8.8 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q No 1: i. D ii. B iii. C

Ans. to Q No 2: i. D ii. B

Ans. to Q No 3: i. Break-even analysis
ii. Net benefit
iii. Cash flow



8.9 MODEL QUESTIONS

Q1. What cost elements are measured in cost/benefit analysis? Which element is found to be most difficult to estimate and why?

Q2. Describe the procedure for cost/benefit determination.

Q3. How are tangible costs different from direct costs?

Q4. Why is it necessary to conduct cost/benefit analysis?

Q5. Compare the various evaluation methods.

Q6. What are the various cost benefit categories?

Q7. How is it easy to identify the costs and benefits of a system?

*** ***** ***

UNIT 9 : THE PROCESS AND STAGES OF SYSTEM DESIGN

UNIT STRUCTURE

9.1 Learning Objectives

9.2 Introduction

9.3 System Design

9.4 Types of System Design

- 9.4.1 Logical Design
- 9.4.2 Physical Design

9.5 Design Methodologies

- 9.5.1 Structured Design
- 9.5.2 Functional Decomposition
- 9.5.3 Factors Effecting System Complexity

9.6 Form-driven Methodology

- 9.6.1 HIPO and IPO Chart

9.7 Let Us Sum Up

9.8 Further Readings

9.9 Answers to Check Your Progress

9.10 Model Questions

9.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- learn the different types of system design
- design the structured chart
- discuss about functional dependencies
- understand about coupling and cohesion
- design HIPO/IPO charts

9.2 INTRODUCTION

We have already been acquainted with the various phases of a system development life cycle. In the previous chapters, we have learnt

about the system analyst, initial investigation and how to do the planning. The various types of feasibility that need to be considered are already explained. Also, we have got the concept up to the analysis phase of a system development life cycle.

In this unit, we will learn about the system design process in a system development life cycle. The most creative and challenging phase of system development life cycle is the system design. The next unit will explore the topics related to input/output and form design.

9.3 SYSTEM DESIGN

System design is the process of defining and developing a system in order to satisfy the specified requirements of the user. It defines the components, architecture, modules, interfaces and data for a system. It describes the approach in order to create a new system. The methods that are needed to implement the system given in the feasibility study are provided by system design. It tries to transform the requirements of the user to design specifications. The system design phase describes both the final system and the process by which the system is developed.

9.4 TYPES OF SYSTEM DESIGN

The system design gives emphasis on the detailed implementation of the recommended system in the feasibility study. It helps in the transition from a user-oriented document to a new document that is oriented to the programmers.

There are two types of system design

- Logical design
- Physical design

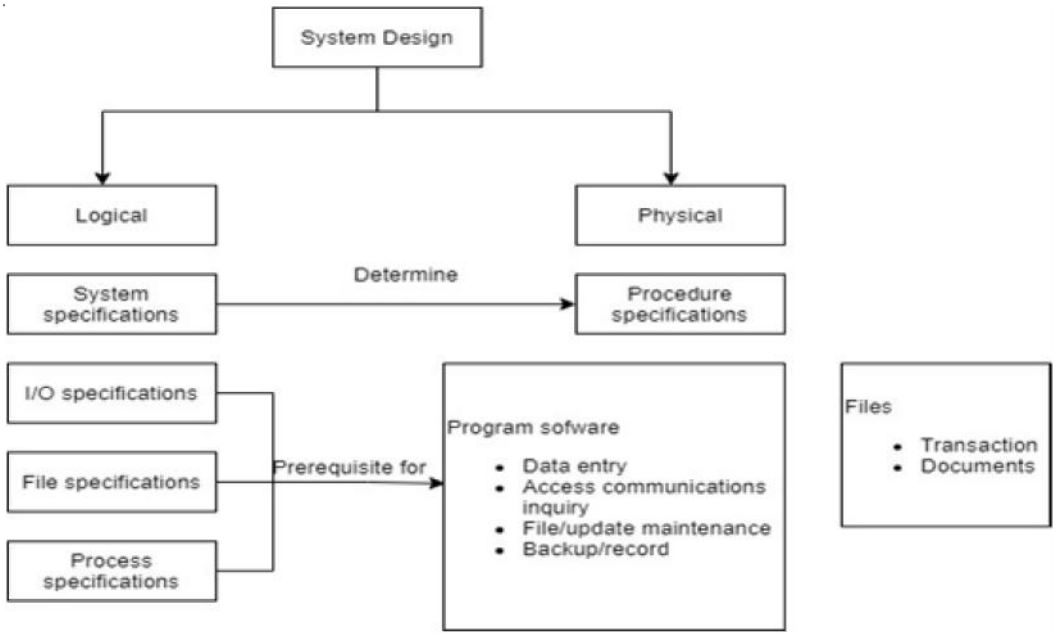


Figure 9.1: Overview of System Design

9.4.1 Logical Design

Logical design shows the logical representation of data flow, input and output of the system. This is achieved by conducting modeling using over-abstract and sometimes graphical model of the actual system. The logical design of a system can be represented by using different diagrams like Entity-Relationship Diagram.

The logical system design represents the user needs at a very detailed level that clearly shows the required information flow into and out of the system. It specifies the required data resources.

The logical design of any system should cover the following:

1. Review the data flow, file content, volumes, frequencies, etc of the current physical system.
2. Prepare the output specifications. The format, content, frequency of reports including terminal specifications and locations should be determined.
3. Prepare the input specifications. The format, content and the input data function, flow of document from the input data source to the actual input location should be determined.

4. Prepare edit, security and control specifications.
5. Specify the implementation plan.
6. Prepare a logical design that specifies the flow of information, output, input, controls and implementation plan.
7. Review the benefits, costs, target dates and system constraints.

9.4.2 Physical Design

Logical design is followed by physical design. Here the working system is produced by defining design specifications that tells the programmer exactly what the system should do. In turn, the programmers write the necessary programs or modify the software package so that the system can accept inputs from the user. The system performs the necessary operations on the existing file or database. The output is a report which is produced as a hard copy or displayed on the screen. The system maintains an updated database all times.

Physical design of any system consists of the following steps:

1. Design the physical system
   a. Specification of input and output media.
   b. Design of the database and specification of the backup procedures.
   c. Design of the physical information flow through the system and a physical design walkthrough.
2. Plan system implementation
   a. Preparation of conversion schedule and a target date.
   b. Determination of the training procedure, courses and timetable.
3. Devise a test and implementation plan. Specify any hardware or software.
4. Update costs, benefits, system constraints and conversion dates.

9.5 DESIGN METHODOLOGIES

For developing a system, the logical design must be specified clearly so that the user requirements are met. It needs to lead to technologies and methodologies that do the following functions:

1. Productivity of analyst and programmer should be improved.
2. Documentation and subsequent maintenance and enhancement should be improved.
3. Cost overruns and delays should be reduced.
4. Communication among the users, analyst, designer and programmer should be improved.
5. Approach to analysis and design should be standardized.
6. Design by segmentation should be simplified.

9.5.1 Structured Design

It is a data flow-based methodology. It allows identifying the input and output of the developing device. The main goal of structured design is to minimize the complexity of the system and increase the modularity. The structured design also describes the useful factors of the system.

The first step is to give a system specification that identifies inputs, outputs and describes the functional aspects of the system. Then the graphical representation will be done which is based on the system specification. The graphical representation can be done using Data Flow Diagram (DFD). From the DFD, the definition of the modules and their relationship to one another is formed using a structured chart. The structured chart is formed using a data dictionary and other structured tools. The flow of structured design is depicted in figure 9.2.

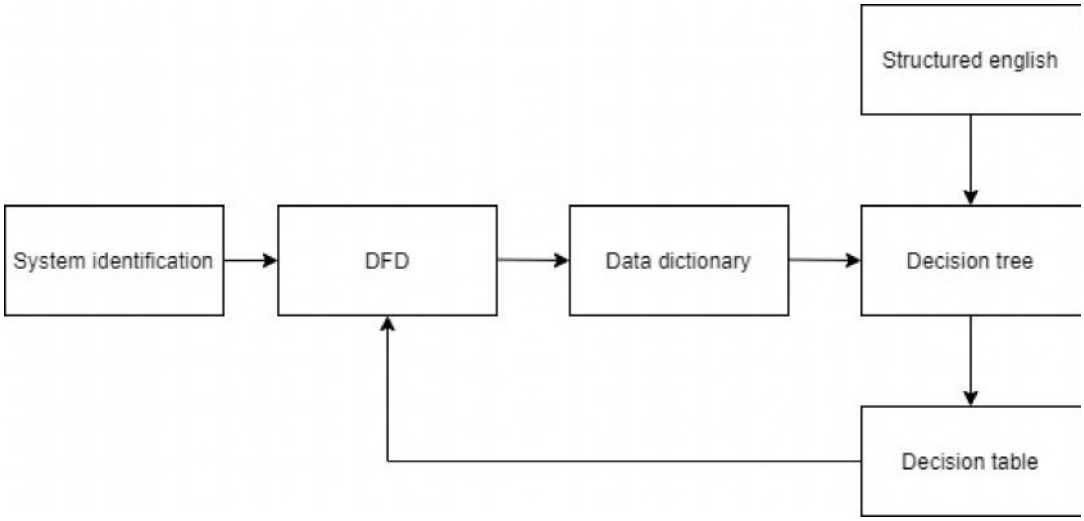


Figure 9.2: Structured design

Structured design partitions a program into small, independent modules. They are arranged in a top-down hierarchy. This method is useful in minimizing complexity and making a problem manageable by subdividing it into smaller segments. This approach is known as modularization or decomposition. Design is said to be top down if it consists of hierarchy of modules each having a single entry and single exit subroutine.

Advantages of modularization:

1. Tests critical interfaces first.
2. Provides abstraction.
3. Allows multiple programmers to work simultaneously.
4. Allows code reuse.
5. Provides control and improves morale.
6. Makes identifying structures easier.

9.5.2 Functional Decomposition

Functional decomposition is a fundamental technical analysis. It breaks down a complex problem into successive layers that are manageable and comprehensive. In functional decomposition, the lower-level functions completely describe the parent. A function or process at lower level cannot exist unless it is included within the parent function or process.

Advantages of a good functional decomposition:

1. It helps in understanding the breakdown of processes or functions.
2. Specifying the requirements and features of each function becomes easier.
3. It helps in localizing errors and minimizing system faults.
4. It allows customers to view and discuss the system in way that can easily deal with.

Disadvantages of poor functional decomposition:

1. Hinders understanding of the system when functions and processes overlap.
2. Creates unnecessarily numerous complex interfaces.
3. Requires extensive rework of other process modeling methods such as Data Flow Diagrams.

Structured chart

The documentation tool for structured design is called the structured chart. It is a recommended tool to design a modular, top-down structure. It helps to define the various modules of the system development and helps to establish relationship between every module. It is a graphical tool for representing hierarchy. The symbols that are used in the construction of structured chart are explained below.

Module:

The process or task of the system is represented by a module. There are three types of modules which are shown in figure 9.3.

- Control module: It has branches to more than one sub module.
- Sub module: It is a child of another module.
- Library module: They are reusable and can be invoked from any module.

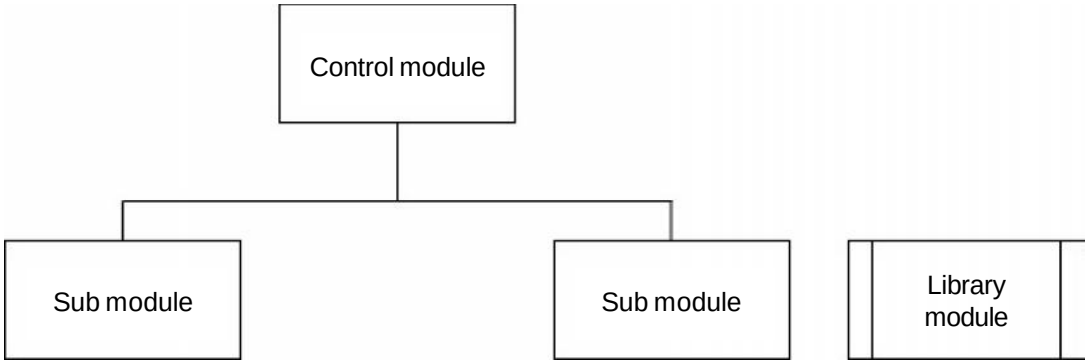


Figure 9.3: Modules

Conditional call:

It represents that based on some condition control module can select any of the sub module. It is presented in figure 9.4.

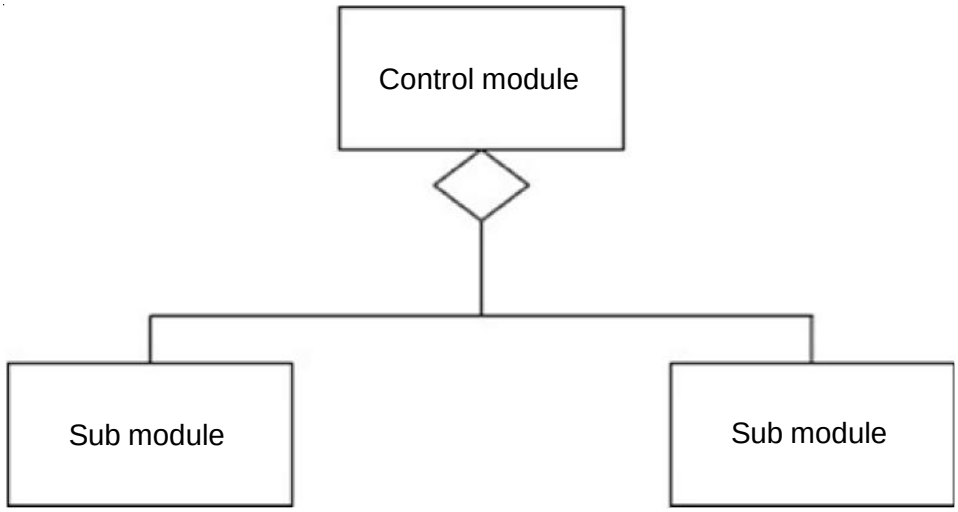


Figure 9.4: Conditional call

Loop

It represents a module to be executed repeatedly by the sub module till some condition becomes false. The loop in the module is represented by a curved arrow as depicted in figure 9.5.

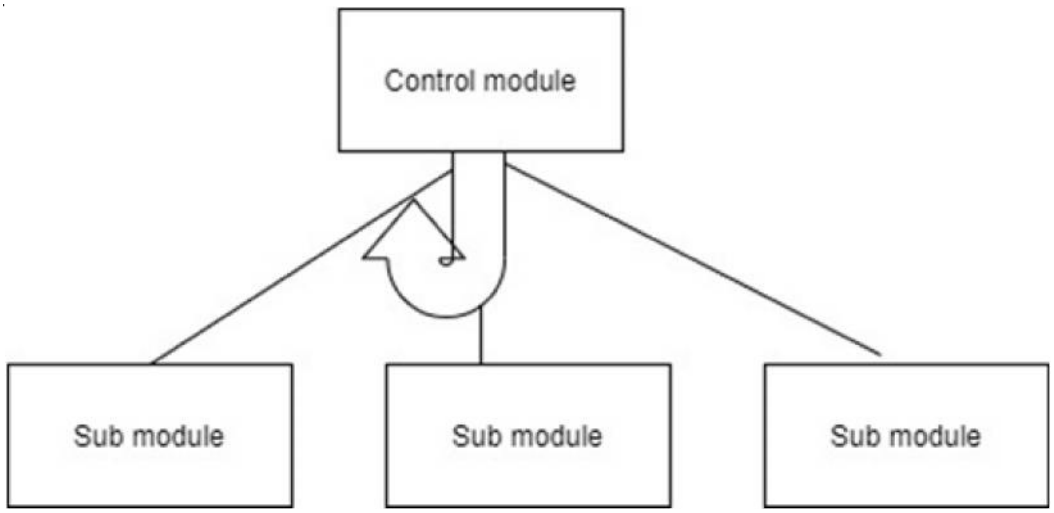


Figure 9.5: Module loop

Data flow

It represents the data flow between the modules. It is represented by an arrow with an empty circle at the end. Figure 9.6 shows module data flow.

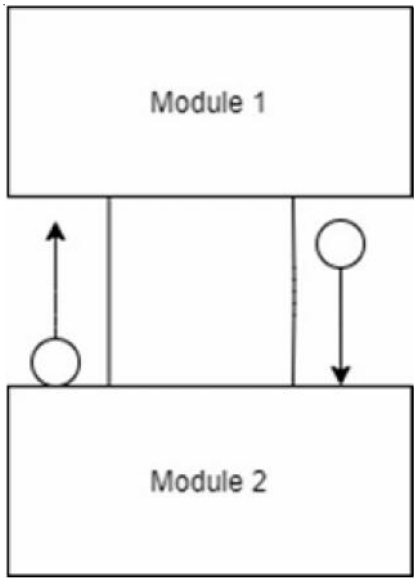


Figure 9.6: Module data flow

Control flow

It represents the flow control between the modules. It is represented by an arrow with a filled circle at the end as presented in figure 9.7.

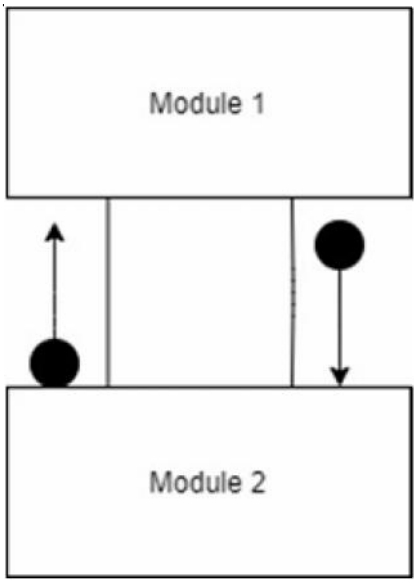


Figure 9.7: Module control flow

Physical storage

It represents storage of all information in a place as shown in figure 9.8.

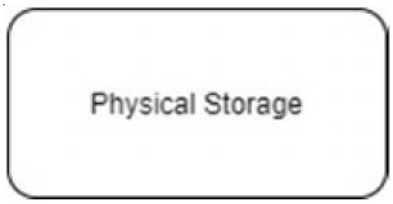


Figure 9.8: Physical storage

Let us draw the structure chart to calculate the average of numbers. The structure chart in figure 9.9 shows the data flow between the modules. “Calculate average” is the control module. “Input numbers, Average and Output average” are the sub modules.

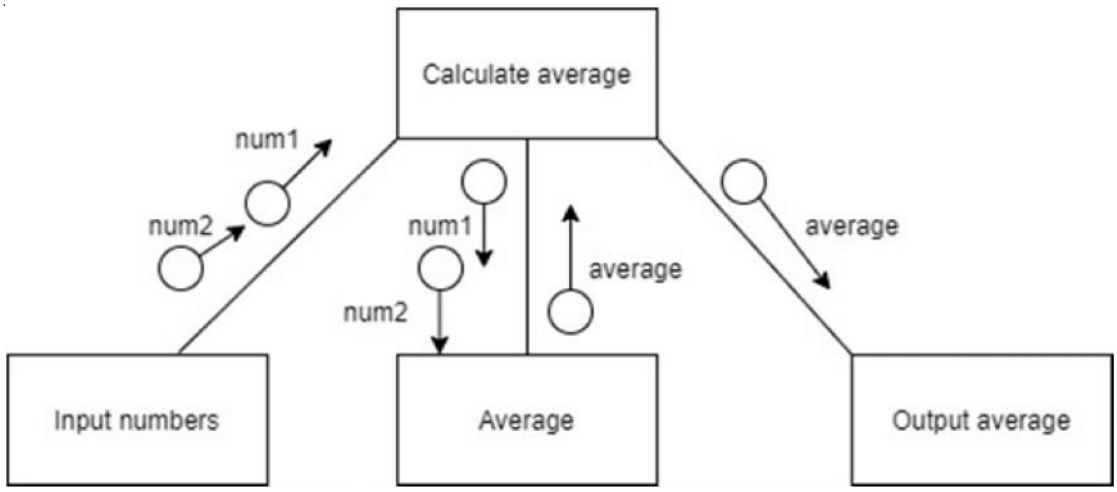


Figure 9.9: Structure chart to calculate average

Objectives of using structured charts are:

1. Encourage top-down design.
2. Support concept of modules and identify appropriate modules.
3. Show size and complexity of the system.
4. To identify number of readily identifiable functions and modules within each function.

9.5.3 Factors Effecting System Complexity

In functional decomposition approach to structured design, a system is partitioned into smaller modules so that each module is made manageable. In order to create a proper quality of system software, it is very important to develop a good design. A system is considered to be of good design if the complexity and cost expenditure in software development is minimum. In the evaluation of the modules, two criteria are considered: Coupling and cohesion. They help in determining the complexity of the system.

Coupling:

The module's connection to other module is called coupling. It defines the degree of dependency of each module on another. Stronger the coupling between modules, the more difficult it is to implement and maintain the system. In high coupling systems as

depicted in figure 9.10, modules are highly dependent on one another. Changes to one subsystem may lead to high impact on the other subsystems.

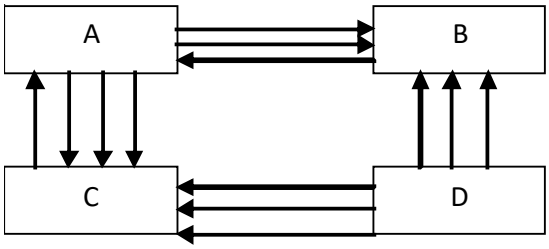


Figure 9.10: High Coupling

In low coupling systems (depicted in figure 9.11), modules are less dependent on each other or almost independent. A change in one system doesn't affect the other subsystems.

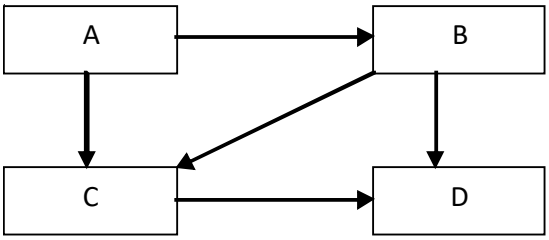


Figure 9.11: Low Coupling

Coupling measures:

1. Content coupling: Here one module actually modifies another module. The modified componentbecomes completely dependent on the modifying component.
2. Common coupling: Here amount of coupling is reduced by organizing the system design so that the data are accessible from a common data store.
3. Control coupling: Here one module passes parameter to control the activity of another module.
4. Stamp coupling: Here data structures are used to pass information from one module to another.
5. Data coupling: Here only data is passed between modules.

Cohesion

The inter-module strength is known as cohesion. It is the measure of the closeness of the relationship between its components. The amount of dependency of the components of a module with one another is defined by cohesion. Functionally cohesive modules are the best modules. Coincidentally cohesive modules are the worst modules. Functional cohesion is said to occur when all the components of the module are grouped together for a single, well-defined purpose. Coincidental cohesion is said to occur when components of modules are grouped together arbitrarily without any meaningful relationships. The system designer must ensure the following in order to achieve good cohesion—

- Essential processes should not be split into fragmented modules.
- Unrelated processes represented as procedures on data flow diagram should not be gathered together into meaningless modules.

Different types of cohesion are—

- Logical cohesion: It occurs when many logically related data elements grouped together in the same component.
- Temporal cohesion: It occurs when a component that is used to initialize a system or a set of data elements perform several functions in sequence but the functions are related to each other in terms of time.
- Procedurally cohesion: It occurs when components are grouped together to maintain order or procedure.
- Sequential cohesion: It occurs when the output of one component is the input of another component.



CHECK YOUR PROGRESS

Q1. Mention the main differences between logical and physical design?

Q2. Draw the structure chart of a food ordering system.

Q3. What is the difference between coupling and cohesion?

9.6 FORM-DRIVEN METHODOLOGY

In structured design, a hierarchy chart represents a good program design if it meets the criteria of cohesion and coupling. Each module should perform a single function (cohesion) and should be independent of the rest of the program (coupling). Each criterion calls for more details. This prompts the analyst to develop input/process/output (IPO) charts for each module in the hierarchy chart.

9.6.1 HIPO And IPO Chart

HIPO is a form-driven technique. Here standard documents are used to document the information. It consists of a hierarchy chart and a set of associated input/process/output chart. HIPO follows top-down decomposition. It describes clearly the data input and output from processes. It also defines the data flow composition. HIPO/ IPO charts weredeveloped by IBM. Figure 9.12 presents Hierarchical plus Input-Process-Output (HIPO) technique and Input-Process-Output (IPO) diagrams

Purpose of HIPO/IPO:

1. To understand structure and functions of systems.
2. To provide easily understandable description of various parts of the system.
3. State the function to be performed by the programmer rather than specifying program statement to be used to perform the function.
4. Provide the visual description of input to be used and output to be produced for each level of the diagram.

5. HIPO makes the transformation of input to output visible.

There are two parts of a complete HIPO package. First the hierarchy chart which represents the top-down structure of the program. For each and every module that is shown in the hierarchy chart, an input-process-output (IPO) chart is used that describes the input, output and the processes performed by the modules.

The HIPO/IPO package format consists of the following:

1. **Hierarchy chart**: Visual table of contents that shows the structure of the diagram and the relationship of the function in a hierarchical manner. It summarizes the primary tasks to be performed by the system. Each box represents a module. The modules call its subordinates and returns control to its parents.
2. **Visual table of contents**: It is a more formal name for the hierarchy chart.
3. **IPO chart**: It is prepared to document each module present in the hierarchy chart.
4. **Overview diagram**: It describes the major functions. It references the detailed diagram needed to expand the functions adequately. It is a high level IPO chart. It summarizes the inputs, outputs and processes to be performed by the modules. Overview diagram acts as planning tools. They usually don't appear in the complete documentation process.
5. **Detail diagram**: It contains an extended description section that amplifies the process steps and references the code associated with each process step. It is a low-level IPO chart. It shows how particular input and output data elements are linked to particular processes.

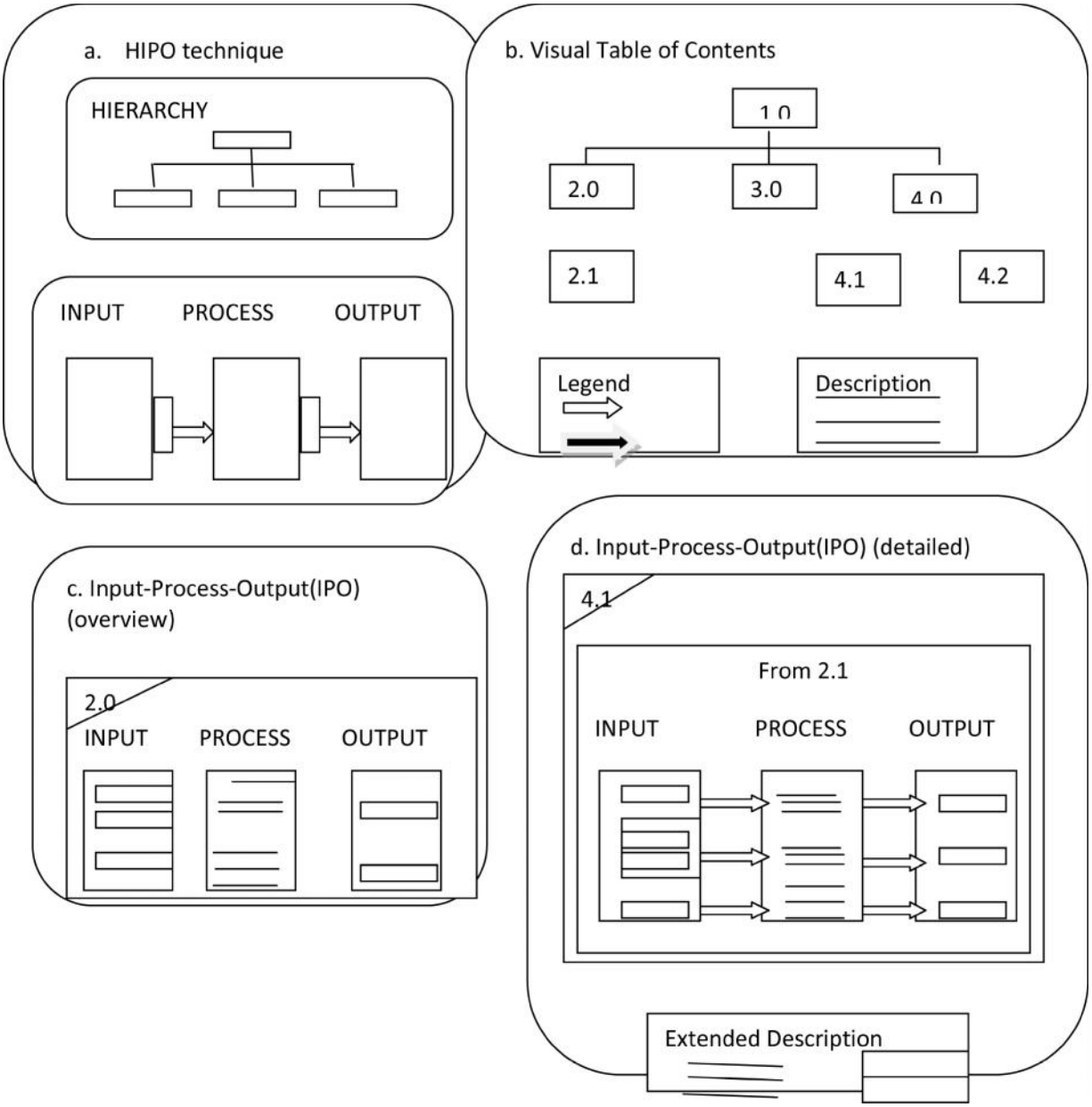


Figure 9.12: Hierarchical plus Input-Process-Output (HIPO) technique and Input-Process-Output (IPO) diagrams



CHECK YOUR PROGRESS

Q4. Draw the hierarchy chart of inventory system?

Q5. Name all the components present in the HIPO/IPO package.

Q6. Write four purposes of HIPO/IPO chart.



9.7 LET US SUM UP

- System design is the process of defining a developing a system in order to satisfy the specified requirements of the user. It defines the components, architecture, modules, interfaces and data for a system.
- There are two types of system design: logical and physical.
- Logical design shows the logical representation of data flow, input and output of the system. The logical design of a system can be represented by using different diagrams like Entity-Relationship Diagram.
- Logical design is followed by physical design. Here the working system is produced by defining design specifications that tells the programmer exactly what the system should do.
- Physical design of any system consists of steps: Design the physical system, plan system implementation, devise a test and implementation plan, specify any hardware or software and update costs, benefits, system constraints and conversion dates
- It is a data flow-based methodology. It allows identifying the input and output of the developing device. The main goal of structured design is to minimize the complexity of the system and increase the modularity.
- Structured design partitions a program into small, independent modules. This method is useful in minimizing complexity and making a problem manageable by subdividing it into smaller segments. This approach is known as modularization or decomposition.
- Functional decomposition is a fundamental technical analysis. It breaks down a complex problem into successive layers that are manageable and comprehensive.
- The documentation tool for structured design is called the structured chart. It is a graphical tool for representing hierarchy

- In the evaluation of the modules two criteria are considered. Coupling and cohesion. They help in determining the complexity of the system.
- The module's connection to other module is called coupling. It defines the degree of dependency of each module on another.
- The inter-module strength is known as cohesion. It is the measure of the closeness of the relationship between its components.
- HIPO is a form-driven technique. It describes clearly the data input and output from processes.
- The HIPO/IPO package format consists of the following: Hierarchy chart, Visual table of contents, IPO chart, Overview diagram and Detail diagram



9.8 FURTHER READINGS

1. Kenneth E.Kendall& Julie E.Kendall,9th Edition 2015, System Analysis and Design, Pearson
2. Goyal, A. (2011). *Systems Analysis and Design*. PHI Learning.



9.9 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q No 1: The main differences between logical and physical design are—

a. Logical design shows the logical representation of data flow, input and output of the system. In physical design, the working system is produced by defining design specifications that tells the programmer exactly what the system should do.
b. Logical design comes first. Then comes the physical design.
c. The logical design of a system can be represented by using different diagrams like Entity-Relationship Diagram. In physical design the programmers write the necessary programs or modify the software package so that the system can accept inputs from the user.

d. System specifications are mentioned in logical design and procedure specifications are mentioned in physical design.

Ans. to Q No 2: The structure chart of food ordering system is—

Ans. to Q No 3: The difference between cohesion and coupling are—

a. Cohesion follows the concept of intra module. Coupling follows the concept of inter module.

b. Increase in cohesion for good for a software system. Increase in coupling is not good for a software system.

c. Cohesion represents the functional strength of the modules. Coupling represents the independence among modules.

d. In cohesion, focus is given on a single module. In coupling, modules are connected on other modules.

Ans. to Q No 4: Hierarchy chart of online sales is—

Ans. to Q No 5: The components present in HIPO/IPO system are—

a. Hierarchy chart

b. Visual table of contents

c. IPO chart

d. Overview diagram

e. Detail diagram

Ans. to Q No 6: The four purposes of HIPO/IPO chart are—

1. To understand structure and functions of systems.
2. To provide easily understandable description of various parts of the system.
3. State the function to be performed by the programmer rather than specifying program statement to be used to perform the function.
4. Provide the visual description of input to be used and output to be produced for each level of the diagram.



9.10 MODEL QUESTIONS

Q1. What is system design?

Q2. Write notes on logical design and physical design.

Q3. Explain the steps involved in the process of physical design.

Q4. What is structured design?

Q5. What is modularization? Mention its advantages.

Q6. What is functional decomposition? Mention its advantages and disadvantages.

Q7. Draw the structured chart to find the area of a circle.

Q8. Write notes on coupling.

Q9. What are the different types of cohesion?

Q10. What is form-driven methodology?

Q11. Draw the hierarchy chart of a hotel management system.

*** ***** ***

UNIT 10: INPUT/OUTPUT AND FORM DESIGN

UNIT STRUCTURE

10.1 Learning Objectives

10.2 Introduction

10.3 Input design

- 10.3.1 Data Capture Guidelines

10.4 Input data

- 10.4.1 Source Data
- 10.4.2 Design of Source Document
- 10.4.3 Source Data Capture Methods

10.5 Input Media and Devices

10.6 Online Data Entry

10.7 Screen Design

10.8 Output Design

- 10.8.1 Basic Principles
- 10.8.2 Types of Outputs

10.9 Form Design

- 10.9.1 Requirement of Form Design
- 10.9.2 Types of Forms

10.10 Let Us Sum Up

10.11 Further Readings

10.12 Answers to Check Your Progress

10.13 Model Questions

10.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- learn about input design
- explain input data, source data and input media and devices
- design online data entry
- learn about output design
- discuss about form design

10.2 INTRODUCTION

In the previous unit, we are already acquainted with the types of system design. The logical and physical process of system design is explained in the last unit. Under design methodologies, structured design is explained clearly. We now know how to design structure chart. We also know how to design the HIPO/IPO chart under form driven methodology.

In this Unit, we will see the input data types and input design. The various types of input will be discussed. We will know what type of data should be entered and what type of data should be avoided. Online data entry will be discussed. We will know how to design online data entry by using menus, formatted form and prompts. The output design will be explained. The design of forms, its requirements and types of forms will also be discussed. The next unit will explore the topics related to database design.

10.3 INPUT DESIGN

In an information system, raw data is the input that is processed to produce the output. The most common cause of errors in data processing is inaccurate input data. Input design can control the errors entered by data entry operators. The process of converting user originated inputs to a computer based format is known as input design. In the system design phase, logical data flows, sources and destinations and data stores are identified in the data flow diagram. Database transaction files and computer programs are specified in a system flow chart. Input data from various sources are collected and similar input data are organized into groups. After that, for processing, appropriate input media is selected.

The input design is the connection that joins information system of the world to the users. The manner in which data enters into the system is described in the input specifications. If the input design features are found proper, it can produce reliable system otherwise it may result in the production of faulty system.

Developing specifications and steps of data preparation are included in input design. Data preparation steps are very important because it helps to put transaction data into a usable form for data entry processing. The activity of putting data into the computer for processing is known as data entry processing.

The objectives of input design are to:

1. **Keep the process simple**: The analyst should try to achieve all the objectives in the simplest manner as possible. If simple alternatives are available, complexities should be avoided.
2. **Reduce the input volume**: The input design should control the quantity of input data. Data reduction can lower costs by reducing labor expense. The analyst can also speed up the entire process from capturing the data to providing the results to the user.
3. **Avoid delay**: A delay in processing as a result of data preparation or data entry operation is known as bottleneck. While designing the input, analyst should always try to avoid bottleneck.
4. **Avoid errors in data**: The input data design should be such that the errors in data are as minimal as possible. Quantity of data plays a major role in deciding the rate at which errors occur. Smaller the amount of input data, fewer the opportunities of errors.
5. **Avoid extra steps**: Sometimes the transactions volume and amount of data preparation cannot be controlled. It becomes very important that the analyst must ensure that the process is as efficient as possible.

10.3.1 Data Capture Guidelines

It is very important to know what data is important in a transaction and what data should be collected for input and processing. It is the responsibility of the analyst to capture only those items that should be considered as input.

There are two types of data that must be input when transactions are processed. They are—

1. Variable data: The data items that keeps on changing for each transaction handled or if any decision is made.
2. Identification data: The data that uniquely identifies the item being processed. In each transaction record, the identifying data is known as record key.

It is very important to know what type of data is entered by user at the time of the input. This is determined by the analyst. The following types of data should not be entered as input—

1. Constant data: Data is same for every record or entry.
2. Details that system can retrieve: Here the stored data can be quickly retrieved from the system.
3. Details that the system can produce: Here system can produce results using the combination of stored data and entered data.

10.4 INPUT DATA

The design of appropriate data input methods is very important. It helps to prevent errors while entering data. The data input method to be used depends on whether the data is directly entered by users on PCs or it is entered by customers in forms manually and later entered into the system with the help of data entry operators. The goal of designing input data is to make the process of data entry logical, easy and error free. While doing data entry, operators need to consider the following:

1. The allocation space for each field.
2. The sequence of the fields must match that given in the source document.
3. The data fields format. For e.g. for entering date whether mm/dd/yy or dd/mm/yy is used.

Some of the popular data input methods are—

1. Offline data input method or batch input method.
2. Online data entry.
3. Computer readable forms.
4. Interactive data input.

Input integrity controls

Input integrity controls are very important in order to design proper input. It includes a number of methods that helps in the elimination of common input errors committed by end users. Checking the values of the individual fields is also included. Transaction logs are used to create audit trails for data entry and other system operations. Audit trails give a record of all changes made in the database. Audit trails helps to provide security and means of recovery in case of any failure.

Once input data design is completed and input data is identified, the source document should be designed. The source document captures the data and then the media can be selected to enter the data into computer.



CHECK YOUR PROGRESS

Q1. Why the volume of input data should be reduced in input design?

Q2. What type of data should not be entered as input?

Q3. Mention few popular data input methods.

10.4.1 Source Data

Source data is the form in which data are initially captured. For example, a cheque drawn for a bank account is a source document. After reaching the back, special MICR (Magnetic Ink Character Recognition) is used to encode it so that it can be processed by the information system of the bank. Therefore it can be said that source documents initiates a processing cycle as soon as they are entered into the system. Source document may be entered into a system using various methods like punch cards, from disks or even directly using keyboard. The candidate system may or may not retain the source document.

It is very important to evaluate the source document. The terms on which each source document is evaluated are—

1. Continuation of usage in candidate system.
2. Extend of modification for candidate system.

A source document should be logical. It should be easy to understand. User should be clearly able to identify each area in the form. It should clearly specify the user what to write and where to write. For example, consider the field “date of birth”. It can be written in different ways. Few of the ways are—

1. 21st August 2021
2. Aug, 21st 2021
3. 21/08/2021
4. 08/21/2021

Unless it is clear how the “date of birth” field should be written, responses can be expected having the above type of combinations.

10.4.2 Design of Source Document

In order to design the source document, first the analyst must decide what data to be captured. The layout of the document can be developed by the analyst stating what items should be included and where they should be placed. The document should also include information guiding the user how to fill the form and what information to provide. Following things should be considered while designing the source document—

Layout:

The layout organizes the whole document. It places the information in such a way so that it will be noticed. It helps to establish the appropriate sequence of items. For example, most people fill up a document from top to bottom and from left to right. The document should be designed in such a manner only. The document should be designed in a logical manner. In a well-designed form, data items should be asked only once as much as possible. Forms should have enough space so that users can provide all information in the space given. The document layout exactly shows the position of each data items with all the headings and instructions to the users.

Captions:

Captions on source document tell the user the type of data that should be entered and when they should be entered. The caption should be brief and easy to understand. The captions should be the standard terms that all the users use. It is a good idea to avoid abbreviations. Simple examples should be included that tells the users how to feed the data.

A well designed source document is easily completed. It speeds up the process of recording data. If checkmarks and boxes are sufficient, they should be provided to capture the data instead of asking the users to write long responses.

10.4.3 Source Data Capture Methods

There are three different methods to capture data. Each method uses specific data preparation and input equipments. The methods are—

1. **Source data capture with key punching**: This method is very old and seldom used now a day. First data is written on the source document. If necessary, data from the source document is coded into a form that is acceptable for computer processing. Data is key punched into the cards. The punch cards are then verified by rekeying the data on a verify machine that compares previously punched data with re-punched data. The cards are then batched together and read into the computer for processing. Data is validated as they are read into the computer for processing. At last the data is processed.
2. **Source data capture with key to storage**: First data is written on the source document. If necessary, data from the source document is coded into a form that is acceptable for computer processing. The tape or disk of data is processed directly. No extra steps are needed to enter data into the computer. The data is validated as it is read into the computer for processing. The data is finally processed.

3. **Direct entry through intelligent terminal**: Intelligent terminals are very similar to cathode ray terminals. They have built in processing capabilities. The need of source document can be eliminated by the use of intelligent terminal for data capture unless a paper record of the transaction is required. The moment a transaction takes place, the operator keys the data directly through the intelligent terminals. As the data is entered, the processor validates the data. One advantage of intelligent terminal is that it can directly interact with the computer.

10.5 INPUT MEDIA AND DEVICES

Source data can be put into the system using a variety of ways. Various media and devices are used for this purpose. Few of the important media and devices are listed below—

1. **Punch cards**: The width of punch cards is either 80 or 96 columns. Data is arranged in a sequential and logical order. It is arranged in such a way so that it becomes easy to put the data into the system. Key punch is used to copy data from source documents into cards. It is very important that while designing both source document and cards should be considered simultaneously.
2. **Key to diskette**: It is modeled after the key punch process. The card is replaced by a diskette and it can store up to 3,25,000 characters. It is equivalent to data stored in 4050 punch cards. Data is stored in diskette in sequential and logical manner in batches.
3. **MICR**: Magnetic Ink Character Recognition (MICR) is a technology that translates the special fonts printed in magnetic ink on cheques into directed computer input. This technology enables a machine to read, process and record information. MICR numbers are designed in such a way so that it can be read by both user and the system. The MICR numbers cannot be faked or copied due to the use of magnetic ink and unique fonts.

4. **Mark-sensing readers:** Mark sensing readers (MSR) is also known as optical mark reader (OMR). It can recognize marks made by a pencil or pen on a specially designed form or card. These cards and forms are pre-printed for special uses. A mark in a particular position in the form or card represents certain meaning. MSR are mostly used in tests, surveys, questionnaires, etc where answers are recorded as dark pencil marks in boxes or circles.
5. **Optical Character Recognition (OCR) readers**: It recognizes all kinds of different characters. It recognizes both hand-written and printed characters and converts it into a machine-readable, digital data format. There are various applications of OCR technology. Image based PDF, TIFF, or JPG can be automatically converted into a text-based machine readable file. Digital files processed with OCR can be searched, viewed, edited and sent to other systems.
6. **Optical bar code readers**: Barcodes consists of bars and spaces that represent characters, numbers, etc. Optical recognition devices called barcode readers or barcode scanners are used to read the barcodes. Optical barcode readers detect any combination of marks that represents data. The most widely used optical bar code reader is Universal Product Code (UPC). UPC is widely used in retail stores.
7. **Cathode Ray Tube**: CRT is mostly used for online data entry. CRT screen has the ability to display 20, 40 or 80 characters simultaneously on a screen. They can display as many as 24 lines of data.

10.6 ONLINE DATA ENTRY

The most widely used technology for online data entry is Cathode Ray Tube (CRT) screens. As computers are commonly used and its price also declining, entering data directly into the computers using keyboards has become very popular. For example, Automated Teller Machine (ATM) in banking sector, Point Of Sale (POS) in retailing, etc.

Online data entry methods uses a processor that accepts data and commands from operator through keyboard, voice input or a touch sensitive screen. The processor analyzes the input and then accepts or rejects accordingly. The three major approaches for entering data into the system or computer are:

- Menu
- Formatted forms
- Prompts

Menu

It is a selection list that helps to simplify computerized data access or entry. There is no need for the user to remember what to enter. The user can simply choose from a list of options and type the option letter associated with it. In a menu, the number of options from which an user can choose is limited. But it reduces the chances of error in data entry.

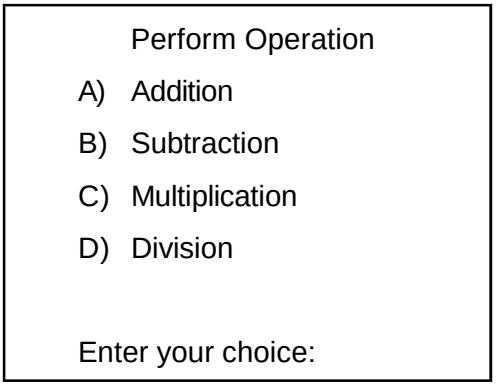


Figure 10.1: Menu

In the example given in Figure 10.1, user can select any option to perform the respective operation.

Formatted form

A formatted form is a pre-printed form or a template. It requests the user to enter data in the appropriate locations. It looks similar to a “Fill in the blanks” type of form. The form is displayed on the screen as a unit. The cursor is usually positioned at the first entry of the form. After the user starts responding by filling up the appropriate data, the cursor automatically keeps moving to the next entry and so on till the form is fully filled up. During the

entire process, the user has the liberty to move the cursor up, down, left, right to various locations to enter data into the form.

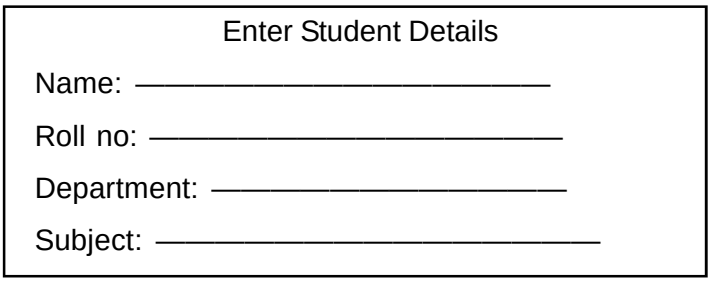


Figure 10.2: Formatted form

In the example given in Figure 10.2, shows a sample of how student details can be added in a formatted form.

The prompt

In prompt, the data to be entered gets displayed one by one. One data enquiry will be displayed; the system will wait for the user response. Once the user responds, the next data entry gets displayed and so on. The next data entry doesn't get displayed unless user enters the data.

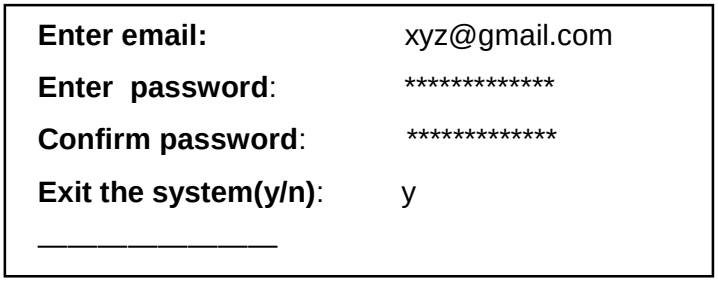


Figure 10.3: Prompt

In the above example in Figure 10.3, the bold letters indicates messages generated from the system and the other data is entered by the user. The system generates the message "**Enter email**". Once the email is entered by the user, the system generates the next message "**Enter password**" and so on.



CHECK YOUR PROGRESS

Q4. How the captions of a source document should be?

Q5. ———————— is one of the traditional input devices. (Fill in the blank)

Q6. Design the formatted form of “Employee Details”.

10.7 SCREEN DESIGN

Form filling is the most effective method of online data entry. The main objectives of screen display design are simplicity, accuracy and quick data capture or data entry. The screen display design should be simple so that the user can understand easily. The design should be accurate and no unnecessary things should be included. Guidelines should be framed very carefully to design data entry screens.

The below mentioned guidelines will help to design data entry screen—

1. Same format should be used throughout the project.
2. Ample of space should be allowed for the data. If there is too much overcrowd of data, it may cause eye strain.
3. Tutorial or help should be provided for technical terms or procedures.

User access to screen locations should be restricted only where data is entered.Descriptive caption for every field should be provided. The user should be shown where to enter the data and the maximum field size should be mentioned.A sample format must be displayed if the user needs to enter values in a field in a specific manner.Entry stroke for every field is required.Users don’t require to type leading zeroes for numeric fields.Users don’t require to type trailing zeroes for numeric fields that includes decimals.Default values should be displayed so that the operator can press the ENTER key to accept the suggested values.A default value should be used when a field value is constant for successive records or throughout the data entry session.A list of acceptable values for each field should be displayed. Meaningful error messages should be provided for the user.A

way should be provided to leave the data entry screen at any time without entering the current record.Users should be provided with opportunities to confirm the accuracy of data before entering it.Users should be able to move among fields in the form in an efficient and easy manner.The screen form layout design should match the layout of the source document.Users should have the capability to add, change, delete and view records.A method should be provided that allows users to search for specific information.

CRT screen is one of the most popular online data entry devices. CRT screens provide instant visual verification of input data. The operator has the capability to make any changes before the data goes into the system for processing. CRT screen performs as a display screen that has a buffer for data storing. It has a common display screen consisting of 24 rows of 80 characters each or 1920 characters.

There are two methods for entering data into a CRT screen. They are—

1. Manual method: This method uses worksheet which is very similar to a printout chart. Reserved areas are used to block out the data or menu to be displayed. For example, in dBase, if we want to print "COMPUTER" in the 12th row and 12th column, the below command can be used.

   @ 10,10 SAY "COMPUTER"
2. Software utility method: Software utility is used to design the screen layout. It is faster and more efficient than the manual method.

10.8 OUTPUT DESIGN

The most important task of any system is the design of the output. Outputs are the reports that the system produces. It can be produced on the screen, on paper, or in media such as the web. The most visible part of a system is the output.

The objectives of the output design are—

1. The output design should be developed in a way that serves intended purpose and eliminates the production of any unwanted output.

2. The output design should be developed in a manner that meets the end user requirements.
3. Appropriate quantity of output should be developed.
4. Output should be formed in appropriate format and it should be directed to the right person.
5. The output should be made available on time for making good decisions.

10.8.1 Basic Principles

The main goal of the output design is to present the information to the users in such a manner so that it can be easily and accurately understood with least effort. The fundamental principles for output design helps to show how outputs are used and the ways in which they can be made simpler so that users can understand them.

Understand Report Usage

While designing reports, the first principle is to understand how they are used. Reports are used for various purposes. In some cases, they are read from cover to cover because all information is needed. In many cases they are used as references to identify specific information. Here, it is critical to sort the items or group them under categories. For example, electronic or web-based reports.

In the design and distribution of reports, the frequency of reports plays an important role. Real-time reports produce accurate data the moment they are produced. For e.g. stock market report. Batch reports produce historical information that may be hours, months or years old. They often provide additional information beyond the reported information.

Manage Information Load

Most managers get lots of information. The goal of a well designed report is to provide the lot of information in such a way that it covers all the important points in a summarized way. In some case, different reports may be produced for the same topic for the same users because the reports are used in different ways.

In western countries, the most important information is produced first in the top left corner of the report. The report should be produced in such a format so that users need not do any modification, resort the report's information, try to find critical information or perform any additional calculations. Report should provide all the important information to the user.

Minimize Bias

The analyst should design a report with minimum bias. The problem with biased report is that it may be produced by the analyst unintentionally. One common way of introducing bias in reports is by displaying list of sorted data. The data that comes first may appear to be more useful than data that comes later in the list. If data are sorted in alphabetical order, entries with letter 'A' are more prominent. There is no easy solution to this problem. Analyst may try to present the information in an order in which information is used. Data may be presented in chronological order where more emphasis is given on older data. Data may also be sorted by numerical values in situations where emphasis is given more on higher or lower numerical values. E.g. Monthly sale report by state.

Challenging design issues may be presented by graphic display and reports. Bias can be introduced by the scale on the axes in the graphs. In most cases, the scale should always start at zero otherwise comparison among values may be misleading. For example, consider Figure11.4, have sales increased very much since 2010. The values in both the charts are same but visual images that the two charts produce is quite different. Figure 11.4a would give an impression of minor changes while Figure 11.4b would give an impression of significant increases. In both charts, sales have increased by 15% over five years and approximately by 3% yearly. Figure 11.4a gives a more accurate picture. Figure 11.4b is biased because the scale starts very close to the lowest value in the graph.



Figure 11.4a: Unbiased Graph starting at 0

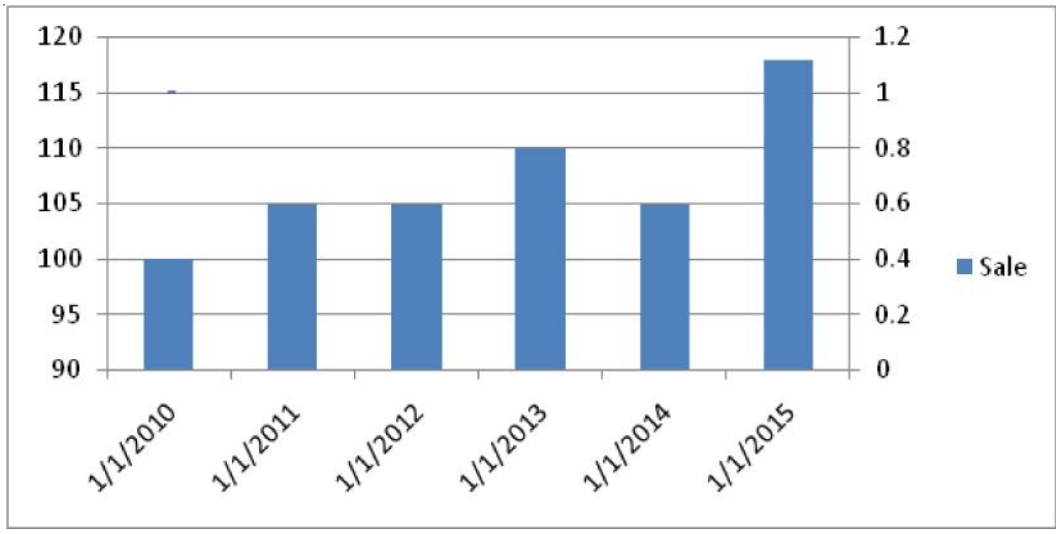


Figure 11.4b: Biased Graph starting at 90

10.8.2 Types of Outputs

The various types of outputs are: external outputs, internal outputs and media.

External outputs

Manufacturers are responsible for creating and designing external outputs for printers. External outputs enable the system to leave the trigger actions for the recipients. They are designed as turnaround outputs. They are implemented as a form.

Internal outputs

Internal outputs are present inside the system. They are used by end-users and managers. They help in decision making and reporting. Management information produces three types of reports—

1. Detailed reports

   They contain all the information with almost no filtering or restriction generated to assist management planning and control.

2. Summary reports

   They contain trends and potential problems. They are categorized and summarized so that they can be generated for managers who don't want details.

3. Exception reports

   They don't contain detailed data. They contain exceptions and data filtered to some conditions or standards before they are presented to the manager as information.

Media

Different types of media are available to produce reports. Now a days, the most dominant media are paper and electronic media. Paper is the traditional medium. The advantage is that it is relatively easy to use, permanent, highly portable and accessible in most cases. It has several significant drawbacks. Once report is printed, it cannot be changed. It is inflexible. They are expensive and hard to move to very long distance. All the drawbacks in paper are solved in electronic media. The users can easily access them. It is relatively inexpensive, can be modified if needed.

10.9 FORM DESIGN

A form identifies the data that we want to collect. It is a powerful tool that allows entering data into the database, displaying it to review or do modification if required and also printing it for distribution. Various examples of forms are business forms, ATM transaction layout, electronic

spreadsheets, etc. Forms are the product of input and output design. They are business document consisting of specified data.

There are some useful questions that should be considered while creating the forms such as “who, what, how, where, when”. This will help to design effective forms.

WHO	It will help to understand who the actual users are.
WHAT	It will help to understand what will be the purpose of the form, what task will be performed and what information is required.
WHEN	It will help to know when the form will be needed.
WHERE	It will help to know where the users will be using the form.
HOW	It will help to know how many users will be using the form.

The objectives of a good form design are—

- Screen should be kept simple by providing proper sequence, information and clear captions.
- Intended purpose should be met by using appropriate forms.
- Completion of the form with accuracy should be ensured.
- The form should be kept attractive using icons, blinking cursors, inverse videos, etc.
- Navigation should be facilitated.

There are three basic classification of forms based on what a form does in the system. They are—

1. **Action**: This is a very basic type of form that asks the users to do something. E.g. purchase orders.
2. **Memory**: This type of form is a record that stores historical data. It remains in a file and later can be used as a reference. E.g. inventory records.
3. **Report**: This type of form is very helpful in guiding supervisors and other administrators in the activities. It provides the important and necessary data on a project or a job. E.g. sales analysis reports.

10.9.1 Requirement of Form Design

The purpose of a form is to communicate effectively to the user through effective forms design. There are several major requirements of form design—

1. **Identification and wording**: The form title is very important. It must clearly identify its purpose. To avoid confusion, columns and rows should be labeled. Form name or code should be used to identify the form so that it becomes easy to reorder.
2. **Maximum readability and use**: The form should be designed in a way that is easy to use and fill out. It should be intelligible, legible and uncomplicated. It should have ample writing space for inserting data.
3. **Physical factors**: In order to achieve easy reading, the form composition, color, layout and paper stock should be taken into consideration. Numbering of pages is very important when multiple reports need to be generated for the users.
4. **Order of data items**: A logical sequence should be maintained while requesting data from the users. Related data should be placed in adjacent positions. The sequence of data in the form and the source documents should be same.
5. **Ease of data entry**: If the form is used for data entry purpose, it should have field positions indicated under each data column. If numerical values are involved, there should be indication of where decimal points are.
6. **Size and arrangement**: The form should be stored easily and filed. Signature facility should be provided. Important items should be placed in a highlighted position in the form.
7. **Use of instructions**: A form should be accompanied by instructions that clearly shows how it should be used and handled.

8. **Efficiency consideration**: Cost is a major factor while designing the form. The form should be cost effective. Unnecessary data should be eliminated from the form.
9. **Type of report**: While designing the form, the purpose of the form should be considered. It should be known whether the form is for executive summary, intermediate managerial information or supporting data. The requirements of the users define the final form design.

10.9.2 Types of Forms

Various types of forms are available. They are—

Flat forms

It is a single copy form. It is prepared manually or by a machine and printed on paper. If additional copies of the original form are needed, carbon papers are inserted in between the copies. The advantage of flat form is that it is simple, inexpensive to design, print and reproduce.

Unit Set/Snap out forms

These forms are either hand written or prepared by machines. These are forms with one time carbon interleaved into unit sets. Carbons may be either blue or black in color with standard grade medium intensity. Generally it's a good practice to use blue carbon for hand written forms and black carbon for machine forms.

Continuous Strip/Fanfold forms

They are multiple unit forms. They are joined in a continuous strip with perforations between each pair of forms. It is comparatively less expensive method for large volume of data.

No Carbon Required (NCR) paper

These forms are carbonless. They have two chemical coatings (capsules), one on the front and other on the back of a sheet of paper. When pressure is applied, the two capsules come in contact and an image is created. It is the cleanest and costliest type of form.



10.10 LET US SUM UP

- The process of converting user originated inputs to a computer based format is known as input design.
- The input design is the connection that joins information system of the world to the users.
- There are two types of data that must be inputted when transactions are processed: Variable data and Identification data
- The design of appropriate data input methods is very important. It helps to prevent errors while entering data.
- Source data is the form in which data are initially captured.
- A source document should be logical. It should be easy to understand
- Following things should be considered while designing the source document: Layout and Caption.
- The three major approaches for entering data into the system or computer are: Menus, formatted forms and prompts.
- The main objective of screen display design is simplicity, accuracy and quick data capture or data entry. The screen display design should be simple so that the user can understand easily.
- The most important task of any system is the design of the output. Outputs are the reports that the system produces.
- The basic principles of output design are: Understand Report Usage, manage Information Load and minimize Bias
- The various types of outputs are: external outputs, internal outputs and media.
- A form identifies the data that we want to collect. It is a powerful tool that allows entering data into the database, displaying it to review or do modification if required and also printing it for distribution.
- There are some useful questions that should be considered while creating the forms such as “who, what, how, where, when”. This will help to design effective forms.

- There are three basic classification of forms based on what a form does in the system: Action, Memory and report
- Various types of forms are available are: Flat forms, Unit Set/Snap out forms, Continuous Strip/Fanfold forms and No Carbon Required (NCR) paper



10.11 FURTHER READINGS

1. Howryskiewycz, System Analysis and Design, PHI
2. Rajaraman, Analysis and Design of Information Systems, PHI.



10.12 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q No 1: The volume of input data should be reduced because data reduction can lower costs by reducing labor expense. It also helps to speed up the entire process from capturing the data to providing the results to the user. It helps to minimize errors, avoid delays, keep the process simple and avoid extra steps.

Ans. to Q No 2: The types of data that should not be entered as inputs are—

1. Constant data: Data is same for every record or entry.
2. Details that system can retrieve: Here the stored data can be quickly retrieved from the system.
3. Details that the system can retrieve: Here system can produce results using the combination of stored data and entered data.

Ans. to Q No 3: Few popular data input methods are:

1. Offline data input method or batch input method.
2. Online data entry.
3. Computer readable forms.
4. Interactive data input.

Ans. to Q No 4: The captions of a source document should be simple. Captions on source document tell the user the type of data that should

be entered and when they should be entered. The caption should be brief and easy to understand. The captions should be the standard terms that all the users use. It is a good idea to avoid abbreviations. Simple examples should be included that tells the users how to feed the data.

Ans. to QNo 5: Punch cards is one of the traditional input devices.

Ans. to QNo 6. The formatted form for "employee details"

Enter Employee Details

Employee Name: _____

Employee ID: _____

Designation: _____

Salary: _____



10.13 MODEL QUESTIONS

Q1. What is input design? What are its objectives?

Q2. What type of data should be entered as input and what type of data should not be entered as input?

Q3. What is input integrity control?

Q4. Explain how a source document should be designed.

Q5. Explain the source data capture methods.

Q6. Write notes on any one input media and device.

Q7. Design a prompt for "students details system".

Q8. Write short notes on screen design.

Q9. What are the objectives of a good form design?

Q10. What are the requirements of a form design?

*** ***** ***

UNIT 11: DATABASE DESIGN

UNIT STRUCTURE

11.1 Learning Objectives

11.2 Introduction

11.3 Database Concept

11.4 Objectives and Characteristics of the Database

11.5 Key Terms in a Database

11.6 Logical and Physical Views of Data

11.7 Schema and Subschema

11.8 Types of Relationships in the Database

11.9 Types of Database

11.10 Role of Database Administrators

11.11 Normalization

11.12 Let Us Sum Up

11.13 Further Readings

11.14 Answer to Check Your Progress

11.15 Model Questions

11.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- understand about the database and its types, characteristics
- learn logical and physical views of data
- describe about schema and subschema
- understand about the role of a database administrator
- learn about normalization

11.2 INTRODUCTION

In the previous unit, we discussed about input, output and form design. In this unit, we will discuss about the database, the physical and logical view of data, different types of relationships in the database. The various types of databases along with the role of database administrator

will also be discussed in this unit. In the next unit, we will discuss about system testing and the parameters related to it.

11.3 DATABASE CONCEPT

A database is an organized collection of information or data. This information and data are stored in a computer system and can access digitally using another computer system. It is a method of storing, managing, and retrieving information. Before the discussion of the database, first, you should know about the data. A data is a fact that relates to an object, such as name, age, father name, semester, branch, etc. The data may be numeric, text, image, video, etc. It may be in any format. The storing, managing, and retrieving of data is known as a database. It is the collection of inter-related data for efficient retrieval, insertion, and deletion to and from a database.The data of a database are organized in the form of tables, views, schemas, reports, etc. For Example, the KKHSOU database organizes the data or information about students, faculty, and admin staff, etc.

11.4 OBJECTIVES AND CHARACTERISTICS OF THE DATABASE

A database should have the following objectives:

a) **Data Availability:** It defines the degree or limit or extent up to which data is readable. A data along with the management procedure, technologies, etc. should be available for access in the database. Data acquisition is also referred to as data availability. Data acquisition means the gathering of information from various sources, in the forms of text, image, video, etc.The data availability ensures that data continues to be available at a required level. The data may recover, repair, but the process ensures the availability of the data upon request.

b) **Data Integrity:** The correctness, completeness, accuracy, and reliability of data define data integrity. It defines the authenticity and consistency of the data. The data integrity test is usually performed

to check the quality of the data in the database, and also to verify the data, in the database.

c) **Data independence:** Data independence defines the change of database schema at one level without hampering the other level or another schema. In the database, there are two types of data independence, a) Physical data independence b) Logical data independence. Physical data independence separatesthe conceptual levels of the database from the physical level of the database. The logical data independence changes the conceptual scheme without changing the external views or API.

The followings are the characteristics of a database.

1. It providessecurity for databy applying cryptography techniques.
2. The redundancy is reduced by a database.
3. It acts as insulation between programs and data abstraction.
4. Multiple views of the data are possible in the database.
5. Sharing of data and multi transaction processing are possible in the database.
6. A database accepts the ACID property, Atomicity, Consistency, Isolation, and Durability.
7. DBMS supports a multi-user environment that allows users to access and manipulate data in parallel.

11.5 KEY TERMS IN THE DATABASE

The following are the key terms that are related to a database.

i) **Table:**A table is a group of related records or information or data in a database. It consists of rows and columns.

ii) **Record:** A group of related fields in a table of a database is known as a record. Every record has the same fields.

iii) **Field:** The smallest unit of data or information of a database is known as field. For example, the name of the person contains the first, middle, and last name. So, the first, middle, and last names are the field of data.

iv) **Field Name:** Each field in the table has some name. For example, the name is a field that contains some values or data.

v) **Field Value:** Every field has a value. For example, the university name is a field in a database, and that contains a value, KKHSOU. So, the KKHSOU is a field value.

vi) **Query:** Query is a request for data from a table of a database. One can use this query, to select the records from a table. The query may execute based on the conditions or may not.

vii) **Form:** It is used to view the data of a record.

viii) **Report:** A document or a report is produced from the data or information in a database.



CHECK YOUR PROGRESS

Q 1: What do you mean by a database?

Q 2: What is data?

Q 3: What is data integrity?

Q 4: What is a table in the database?

Q 5: The information about data in a database is called _____. (Fill in the blank)

11.6 LOGICAL AND PHYSICAL VIEWS OF DATA

The data of the database are presented in terms of logical and physical views. The logical views define the top-level design and are created at the requirements gathering, and system analysis. The logical view of data is what data look like. The physical view of a database is the way data or information exists in the physical storage of a database. It deals with how data are stored in the database, data accessed, or related to other data in storage.

To handle logical views of data, a logical data model defines the data element and its relationship. The data architect and business analytics are also defined by the logical data model. It produces a technical map of rules for the data and its structures. Like, the physical data model, defines

the actual database design and development. The database administrator and database developers create the physical views of a database.

11.7 SCHEMA AND SUBSCHEMA

A database schema is the frame structure that denotes the logical view of the database. It represents how the data is stored and organized in a database. It also defines how the relations among dataare associated. A database schema representsthe entities and the relationship among the entities. The database designers design the schema to help database programmers to understand the database and make it useful. The schema is the complete description of a database that includes the field names and descriptions of all records, and elements.

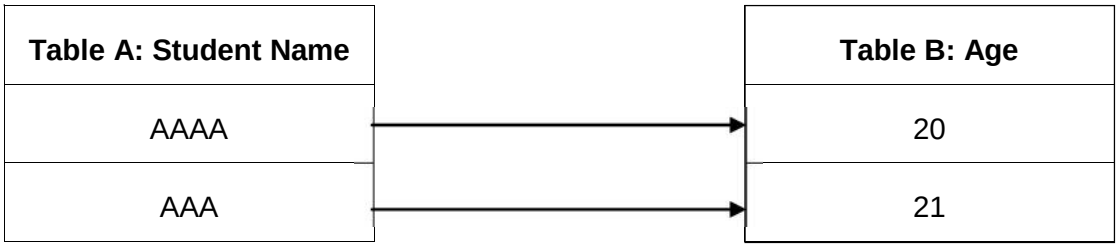
A subschema provides a view of the database as seen by an application program. This view is often a subset of the complete schema definition. A subschema is used at runtime to provide the DBMS with a description of those portions of the database that are accessible to the application program.The subschema can restrict access to the database by identifying:

- Areas, records, elements, and sets that are accessible
- The Data Manipulation Language (DML) functions that can be performed

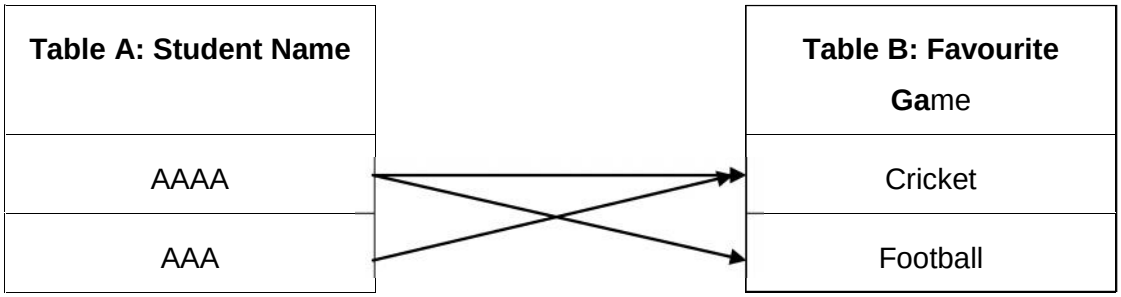
11.8 TYPES OF RELATIONSHIPS IN THE DATABASE

Three types of relations are present in the database:

- **one-to-one (1:1):**The one-to-one relation is used to create a relationship between two tables. Here, a single row of the first table can be related to one and only one record of a second table. Similarly, the row of a second table can also be related to anyone row of the first table. Let's, have two tables, table A and table B. Table A contains the First name of the student of KKHSOU and Table B contains the age of the students, now the one-to-one (1:1) relationship is presented as follows.



- **one-to-many (1:M):**It is used to create a relationship between two tables. Any single row of the first table can be related to one or more rows of the second table. But, the rows of the second table can only relate to the only row in the first table. It is also known as a **many-to-one** relationship. Let's, have two tables, table A and table B. Table A contains the First name of the student of KKHSOU and Table B contains the name of the favorite game of the students, now one-to-many (1:M) relationship is presented as follows.



- **Many-to-many (M:M):**In many to many relationships,each record of the first table can relate to any records (or no records) in the second table. Similarly, each record of the second table can also relate to more than one record of the first table. It is also represented an **N:N** relationship.Let's, have two tables, table A and table B. Table A contains the First name of the student of KKHSOU and Table B contains the name of the favorite game of the students, now one-to-many (1:M) relationship is presented as follows.



CHECK YOUR PROGRESS

Q 6: What is difference between the logical and physical views of data?

Q 7: What is a schema in database?

Q 8: What is subschema in database?

Q 9: What is a one-to-many relationship of tables in database?

Q 10: A relational database consists of a collection of _____.

Q 11: A _____ in a table represents a relationship among a set of values.

11.9 TYPES OF DATABASES

Databases can be organized into the following categories:

a. **Centralised Database:** Centralized database stores data or information at a centralized database system or server. Users can access the stored data from different locations through several applications. The authentication process ensures the users access data securely. For example, Central Library. It decreases the risk of data management, i.e., manipulation of data will not affect the core data. Data consistency is properly maintained. The main issue of a centralized database is the larger size of the centralized database which increases the response time for fetching the data.If any central server failure occurs, entire data will be lost.

b. **Distributed Database:** Unlike a centralized database system, in a distributed database, the information or data is distributed among different database systems. These database systems are connected via communication links. For Example, Apache, Ignite, etc. A distributed database system may be either a homogeneous or heterogeneous distributed database. The homogeneous database systems execute on the same operating system, use the same application process, and carry the same hardware devices to execute the user queries. The heterogeneous database systems execute on different operating systems with different application procedures and different hardware devices. Failures of the server will not affect the entire data set.

c. **Relational Database:** The relational databases are based on the relational data model. In the relational data model, the data are stored in the form of rows(tuple) and columns(attributes), and these rows and columns are from a table(relation). A SQL language is used for storing, manipulating, as well as maintaining the data. For Example, MySQL, Microsoft SQL Server, Oracle, etc, are relational databases. In this type of database, the ACID properties are maintained where ACID stands for atomicity, consistency, isolations, and durability.

d. **NoSQL Database:** Non-SQL/Not Only SQL is a type of database that is used for storing a wide range of data sets. Apart from the tabular format, it stores data in several different ways such as key-value storage, documented oriented database, graph database, and wide column database, and that is the reason that it is not a relational database. It is applicable for building modern applications database systems.

e. **Cloud Database:** When data are stored in a virtual environment and executing over the cloud computing platform, then these types of databases are known as cloud databases. It provides different cloud computing services, such as SaaS, PaaS, IaaS, etc. for accessing the data from the database. Following are the different cloud platforms.
   - Amazon Web Services (AWS)
   - Microsoft Azure
   - Google Cloud SQL, etc.

f. **Object-oriented Databases:** In object-oriented databases, the object-oriented models are used to storethe data in the database system. The data present in the object-oriented databases are known as objects.

g. **Hierarchical Databases:** It is the type of database that stores data in the form of Hierarchical order or parent-children relationship nodes. Here, it organizes data in a tree-like structure.Data are stored in the tree nodes, which are connected via links in the Hierarchical model.

h. Network Databases: It is the database that is based on the network data model. Here, the data is in the form of nodes that are connected via links between them. Unlike the hierarchical database, it allows each record to have multiple children and parent nodes to form a generalized graph structure.

11.10 ROLE OF A DATABASE ADMINISTRATOR

A Database Administrator (DBA) takes care of the design, organize, update, etc. of a computer database so that a user gets the necessary information from the database at the time of the request. The followings are the responsibilities of a DBA.

i) A DBA works with a database and finds the best possible way to store, organize, and manage data in the database.
ii) A DBA handles the database troubleshooting.
iii) A DBA keeps the database update.
iv) A DBA manages the database access.
v) A DBA ensures to meet the user demands during the query processing.
vi) A DBA backup the data of a database.
vii) A DBA writes the reports, documentation, and operation manuals of a database.
viii) A DBA implements the security measure of a database.
ix) A DBA achieves the data of a database.

11.11 NORMALIZATION

A database may have redundant data. Normalization is a process of minimizing the redundancy from a database by minimizing the redundancy from the relations. Redundant data may cause insertion, deletion, and update anomalies. To understand the data redundancy, let's take an example of the student table of KKHSOU.

Rollno	Name	Program
10401	A	Computer Science
10402	B	Computer Science
10403	C	Computer Science
10404	D	Computer Science

In the table above, you have data of 3 Computer Science students. As you can see, data for the field Program is repeated for the students who are in the same programComputer Science in KKHSOU, this is Data Redundancy.To minimize the anomalies, different normalization techniques are used in the database. They are presented below.

i) 1 NF: A relation is said to be in 1NF if it contains an atomic value. If a relation contains more than one value or composite or multi-valued attribute, it violates the first normal form. A relation is in first normal form if the relation has a single-valued attribute. The following table is in 1 NF.

Rollno	Name	Program
10401	A	Computer Science
10402	B	Computer Science
10403	C	Computer Science
10404	D	Computer Science

But, the following table is not in 1-NF.

Rollno	Name	Phone No
10401	A	111,222
10402	B	333
10403	C	444
10404	D	444

But, you can decompose the table into 1NF and it has been shown below:

Rollno	Name	Phone No
10401	A	111
10401	A	222
10402	B	333
10403	C	444
10404	D	444

ii) 2 NF: A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key. Let's understated the 2NF using the following examples.

Rollno	Name	Age
10401	A	18
10401	A	19
10402	B	18
10403	C	20
10404	D	19

In the above table, the non-prime attribute age is fully functionally dependent on the primary key (**Rollno**). It is a proper subset of candidate keys. That's why, it violates the rule for 2NF. You can convert the given table into 2NF by decomposing the above table into two tables:

Rollno	Name
10401	A
10401	A
10402	B
10403	C
10404	D

Rollno	Age
10401	18
10401	19
10402	18
10403	20
10404	19

iii) 3 NF: Arelation will be in 3NF if it is in 2NF and does not contain any transitive partial dependency. It is used to reduce data duplication and achieve data integrity. A relation must be in third normal form if there is no transitive dependency for non-prime attributes.

A relation is said to be in third normal form if it holds atleast one of the following conditions for every non-trivial function dependency $X \rightarrow Y$.

- X is a super key.

- Y is a prime attribute, i.e., each element of Y is part of some candidate key.

Let's understand the 3NF using the following examples.

Rollno	Std_Name	Std_Age
10401	A	18
10402	B	19
10403	C	18
10404	D	20
10405	E	19

The Candidate key of the above table is {Rollno}. The non-prime attributes are the attributes except the Rollno. Here, Std_age is dependent on Std_name. The non-prime attributes (Std_Name and Std_Age) are transitively dependent on the super key (Rollno). It violates the rule of the third normal form. That's why we need to decompose the table as follows.

Rollno	Std_Name
10401	A
10402	B
10403	C
10404	D
10405	E

Std_Name	Std_Age
A	18
B	19
C	18
D	20
E	19

iv) Boyce Codd normal form (BCNF): The BCNF is the advanced version of 3NF. A relation is in BCNF if every functional dependency X '! Y, X is the super key of the table. For a BCNF, the relation should be in 3NF, and for every functional dependency, the Left-Hand Side should be super key. Let's understand the BCNF with the help of the following table where one student can take the different subjects of the different department

Stdid	StdPlace	StdDept	Stddepttype	Stddeptid
10401	A	CS	UG	CSUG1
10401	A	IT	UG	ITUG1
10402	B	CS	PG	CSPG1
10402	B	IT	PG	CSPG 1

In the above table, Functional dependencies are as follows.

1. Stdid → StdPlace
2. StdDept → {Stddepttype, Stddeptid}

And the Candidate key is {Stdid, StdDept}. The table is not in BCNF because neither StdDept nor Stdid alone are keys. To convert the given table into BCNF, we decompose it into three tables:

a) Student Place table

Stdid	StdPlace
10401	A
10401	A
10402	B
10402	B

b) Student Department table

StdDept	Stddepttype	Stddeptid
CS	UG	CSUG1
IT	UG	ITUG1
CS	PG	CSPG1
IT	PG	CSPG 1

c) Student department mapping table.

Stdid	StdDept
10401	CS
10401	IT
10402	CS
10402	IT

In the above table, the Functional dependencies are as follows:

- Stdid → StdPlace
- Stddept → {Stddepttype, Stddeptid}

And the Candidate key is {Stdid, StdDept}

v) 4NF: A relation is said to be in 4NF if it is in BCNF and has no multi-valued dependency. For a dependency X '! Y, if for a single value of X, multiple values of Y exist, then the relationship will be a multi-valued dependency. Let's consider the following table STUDENT.

Stdid	StdDept	Hobby
10401	CS	UG
10402	IT	UG
10403	CS	PG
10404	IT	PG

The given STUDENT table is in 3NF, but the StdDept and Hobby are two independent entities. Hence, there is no relationship between StdDept and Hobby. So to make the above table into 4NF, we can decompose it into two tables:

Stdid	StdDept
10401	CS
10402	IT
10403	CS
10404	IT

Stdid	Hobby
10401	UG
10402	UG
10403	PG
10404	PG



CHECK YOUR PROGRESS

Q 12: What is data normalization?

Q 13: What is the 1 NF?

Q 14: State the definition of 2NF and 3NF.



11.12 LET US SUM UP

- A database is an organized collection of information or data. This information and data are stored in a computer system and can access digitally using another computer system. It is a method of storing, managing, and retrieving information.
- A database should have the following objectives.
  a. Data Availability
  b. Data Integrity
  c. Data independence
- A table in a database is a group of related records or information or data in a database. It consists of rows and columns.
- Record: A group of related fields in a table of a database is known as a record. Every record has the same fields.
- The logical views define the top-level design and are created at the requirements gathering, and system analysis. The logical view of data is what ta looks like.
- The physical view of a database is the way data or information exists in the physical storage of a database.
- A database schema is the frame structure that denotes the logical view of the database. It represents how the data is stored and organized in a database. It also defines how the relations among data are associated.
- A subschema provides a view of the database as seen by an application program. This view is often a subset of the complete schema definition.
- Three types of relations are present in the database:
  a. One to one
  b. One to many
  c. Many to many
- The followings are the types of databases.

a. Centralized Database
b. Distributed Database
c. Relational Database
d. NoSQL Database
e. Cloud Database
f. Object-oriented Databases
g. Hierarchical Databases
h. Network Databases

- The followings are the responsibilities of a DBA.
  a. A DBA works with a database and finds the best possible way to store, organize, and manage data in the database.
  b. A DBA handles the database troubleshooting.
  c. A DBA keeps the database update.
  d. A DBA manages the database access.
  e. A DBA ensures to meet the user demands during the query processing.
  f. A DBA backup the data of the database.
  g. A DBA writes the reports, documentation, and operation manuals of a database.
  h. A DBA implements the security measure of a database.
  i. A DBA achieves the data of a database.
- Normalization is a process of minimizing the redundancy from a database by minimizing the redundancy from the relations.
- A relation is said to be in 1NF if it contains an atomic value. If a relation contains more than one value or composite or multi-valued attribute, it violates the first normal form.
- A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.
- Arelation will be in 3NF if it is in 2NF and does not contain any transitive partial dependency. It is used to reduce data duplication and achieve data integrity. A relation.
- The BCNF is the advanced version of 3NF. A relation is in BCNF if every functional dependency $X \rightarrow Y$, X is the super key of the table.

For a BCNF, the relation should be in 3NF, and for every functional dependency, the Left-Hand Side should be super key.

- A relation is said to be in 4NF if it is in BCNF and has no multi-valued dependency. For a dependency X → Y, if for a single value of X, multiple values of Y exist, then the relationship will be a multi-valued dependency.



11.13 FURTHER READINGS

1. Fundamentals of Database Systems - RamezElmasri



11.14 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q No 1: A database is an organized collection of information or data. This information and data are stored in a computer system and can access digitally using another computer system. It is a method of storing, managing, and retrieving information.

Ans. to Q No 2: A data is a fact that relates to an object, such as name, age, father name, semester, branch, etc. The data may be numeric, text, image, video, etc. It may be in any format.

Ans. to Q No 3: The correctness, completeness, accuracy, and reliability of data define data integrity. It defines the authenticity and consistency of the data. The data integrity test is usually performed to check the quality of the data in the database, and also to verify the data, in the database.

Ans. to Q No 4: A table is a group of related records or information or data in a database. It consists of rows and columns.

Ans. to Q No 5: Metadata.

Ans. to Q No 6: The logical views define the top-level design and are created at the requirements gathering, and system analysis. The logical view of data is what data look like. The physical view of a database is the way data or information exists in the physical storage of a database.

It deals with how data are stored in the database, data accessed, or related to other data in storage.

Ans. to Q No 7: A database schema is the frame structure that denotes the logical view of the database. It represents how the data is stored and organized in a database. It also defines how the relations among dataare associated.

Ans. to Q No 8: A subschema provides a view of the database as seen by an application program. This view is often a subset of the complete schema definition. A subschema is used at runtime to provide the DBMS with a description of those portions of the database that are accessible to the application program.

Ans. to Q No 9: The one-to-many (1:M) is used to create a relationship between two tables. Any single row of the first table can be related to one or more rows of the second table. But, the rows of the second table can only relate to the only row in the first table. It is also known as a many-to-one relationship.

Ans. to Q No 10: Tables

Ans. to Q No 11: Rows.

Ans. to Q No 12: Normalization is a process of minimizing the redundancy from a database by minimizing the redundancy from the relations. Redundant data may cause insertion, deletion, and update anomalies.

Ans. to Q No 13: A relation is said to be in 1NF if it contains an atomic value. If a relation contains more than one value or composite or multi-valued attribute, it violates the first normal form. A relation is in first normal form if the relation has a single-valued attribute.

Ans. to Q No 14: A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.Arelation will be in 3NF if it is in 2NF and does not contain any transitive partial dependency. It is used to reduce data duplication and achieve data integrity. A relation must be in third normal form if there is no transitive dependency for non-prime attributes.



11.15 MODEL QUESTIONS

Q1. What is a database system?

Q2. What is data? Give examples.

Q3. What are the objectives of a database?

Q4. What are the characteristics of a database?

Q5. Explain different normal forms with examples.

Q6. Explain the role of a database administrator.

Q7. What are the different types of databases? Explain it.

*** ***** ***

UNIT 12: SYSTEM TESTING

UNIT STRUCTURE

12.1 Learning Objectives

12.2 Introduction

12.3 Need for System Testing

12.3.1 Process of System Testing

12.4 Nature of Test Data

12.5 Test Plan

12.5.1 Importance of Test Plan

12.5.2 Components of a Test Plan

12.5.3 Types of Test Plan

12.5.4 Creation of a Test Plan

12.6 Activity Network for System Testing

12.7 Types of System Testing

12.8 Let Us Sum Up

12.9 Further Readings

12.10 Answers to Check Your Progress

12.11 Model Questions

12.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- learn about system testing and its process
- understand the nature of test data
- explain test plan
- discuss activity network for system testing
- discuss types of system testing

12.2 INTRODUCTION

In the previous unit, we learned about the database design. We are acquainted with the objective of database and the various terms that are

involved with database design. Data can be viewed in terms of physical and logical view. Physical view, logical view, schemas and sub schemas were discussed. We learned about the various types of relationships. Data structure plays an important role in the mathematical organization of data inside a system. We discussed about three types of data structures, namely, hierarchical, network and relational data structure. Role of entity and attributes were seen properly. Database administrator plays a vital role in database design. The role of database administrator was discussed. Finally, we have discussed how to apply normalization.

In this unit, we will learn about system testing and its process. We will see why system testing is so important. The nature of test data will be discussed. We will see the importance of test plan, components of a test plan, types of test plan and creation of a test plan. Discussion will be done on activity network for system testing. Finally, the types of system test will be discussed. In the next unit, we will discuss about the quality assurance of a system.

12.3 NEED FOR SYSTEM TESTING

System testing is a level of testing that helps to validate a complete and fully-integrated software product. System testing is very important in the process of development of a software product. Its purpose is to evaluate the end-to-end system specifications. System testing is viewed as a series of different tests whose main goal is to exercise the full computer-based system.

System testing evaluates the compliance of the system with the corresponding system requirements. The output of the system testing is the observed behavior of individual components or the whole system when it is tested. System testing is done on the whole system in the context of either system requirement specifications or functional requirement specifications or both. It tests the design and behavior of the system so that it meets the expectations of the customers. It is performed to test the system beyond the requirements mentioned in the Software Requirements Specifications (SRS).

System testing is a level of testing performed by a testing team. It is independent of the development team and helps in testing the quality of the system impartial. It does both functional and non-functional testing. It is a black box testing method. Black box testing method does not require internal knowledge of the code. Input is provided to the system and the output is verified. The knowledge of the internal system is not required. System testing is performed after the integration testing step and before the acceptance testing step. Figure 12.1 presents an overview of levels of testing.

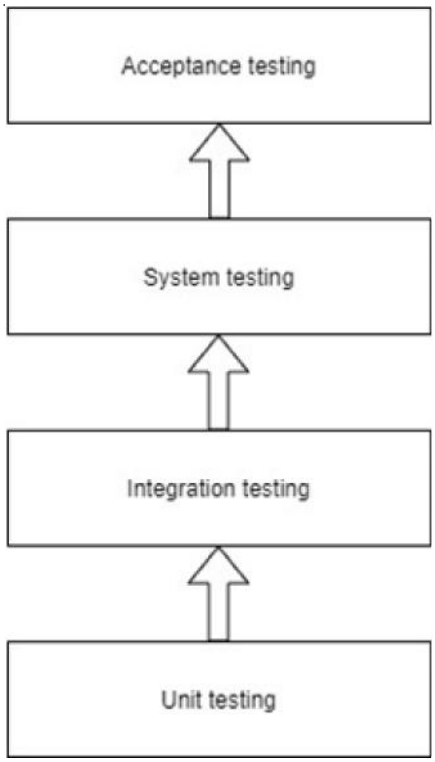


Figure 12.1: Levels of Testing

In brief, system testing is important due to the following reasons—

- It is the first level of software testing. It tests the software/application as a whole.
- It verifies and validates the business, technical, functional and non-functional requirements of the software.
- It verifies and validates the software application architecture.
- It is done on staging environment. It closely resembles the production environment where the final software will be developed.

There are some criteria that must be satisfied in order to apply system testing. Theseinclude the following.

- Unit testing must be completed.
- Integration testing must be completed.
- The development of the whole software system should be complete.
- A software testing environment that closely resembles the production environment must be available. This is called staging environment.

12.3.1 Process of System Testing

System testing is a very important step in the process of software testing life cycle. (STLC). It is very important to define the process of system testing in order to ensure that its procedure is executed accurately. It helps to ensure that no critical step or detail is missed out. The steps involved in the process of system testing can vary from project to project or from organization to organization.

System testing process has the following steps—

- **Create a test plan**: Test plan is the initial step in the process of system testing. The leader or test manager defines the scope and objective of testing. It determines the strategies, makes the decision between manual and automated testing. It defines the entry and exit criteria, assigns roles and responsibilities, etc.
- **Test case creation**: It is the step from which the process of testing is initiated by the team. Test cases are prepared on the basis of use cases, testing requirements, user/client requirements such as user interface, technical requirements, functional and non-functional requirements, performance, etc.
- **Select test data**: Once the team develops the test cases, they work together in the process of selection and creation of required test data. Test data plays an important role in test execution. Test data are the inputs that help the team in getting expected results.

- **Test case execution**: The test cases that were created earlier will be executed by the team. The team constantly monitors the process and records any problems or issues or discrepancies encountered by the team during the process. The output of the testing is also recorded.
- **Bug reporting and fixation**: It is the step in which the team reports all the recorded bugs and issues if it exists to the concerned member of the team. Once the bugs and issues are reported, the programmer or the developer works with the testing team to fix and resolve the issues.
- **Repeat the test cycle (if required)**: Once all the issues and bugs are resolved and fixed, the team repeats the entire test cycle till the expected results are obtained.

The process of system testing is depicted in figure 12.2.

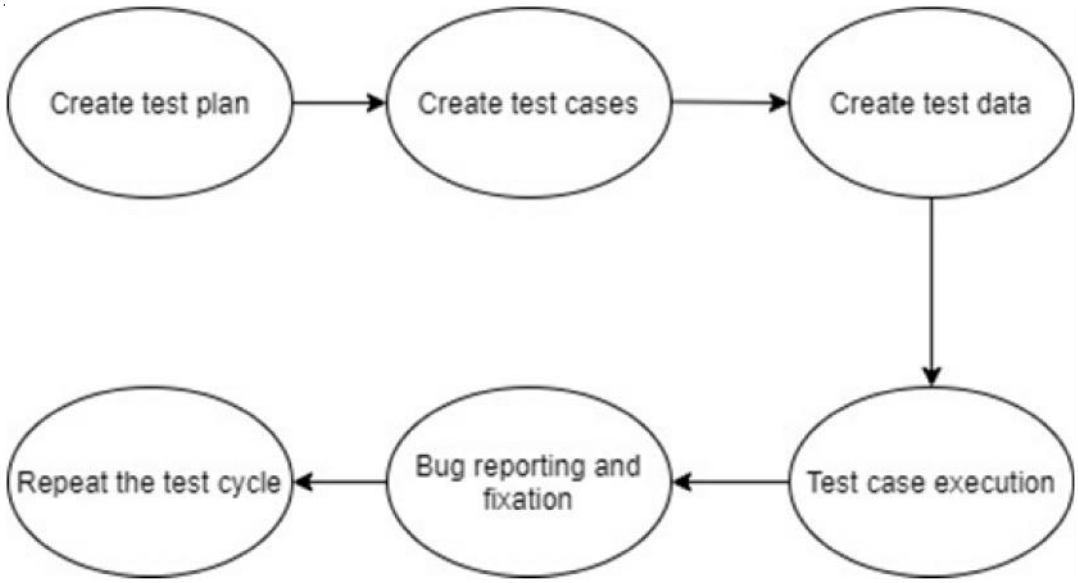


Figure 12.2: Process of System Testing

12.4 NATURE OF TEST DATA

The proper choice of test data is important. It is as important as the test itself. If the test data to be used as input are not valid or does not match the representative of the data to be provided by the user, then the reliability of the output is susceptible.

Test data may be artificially taken or taken from live data. It means test data can be created solely for test purpose or may be taken from user's actual files. Artificial data should be properly created. It should provide all combination of values or formats. It should be made possible to test all logic and transaction path subroutines. Live data are biased towards typical values. Artificial data provides extreme testing conditions. It provides extreme values for testing the boundaries of the candidate system.

For large and complex systems, dedicated computer programs are there to generate necessary test data. Substantial amount of time is saved by data generating programs both for the programmer and the test. In order to write an effective data-generating program, a familiarity with the system files and parameters is important.



CHECK YOUR PROGRESS

Q1. System testing follows white box testing. (State True/False).

Q2. System testing is performed before———————————— and after ————————————. (Fill up the blank)

Q3. How is test data taken?

12.5 TEST PLAN

A test plan is a detailed document that has a catalog containing the test strategy, objectives, schedule, estimations, deadlines and resources that are required for completing the particular project. It acts as a blueprint for running the tests in order to ensure that the software works properly. This process is controlled by the test managers. A good test plan should be dynamic in nature. It should be able to change according to progressions in the project and should stay current at all times. It is the point of reference based on which all the testing activities are executed and coordinated among a quality analysis team.

The test plan is shared with everyone associated with the project like business analyst, project managers, development team, etc. Doing

this offers transparency into the quality analysis activities so that all stakeholders get the idea of how the software will be tested. Quality Assurance (QA) managers or leaders build the test plan based on inputs from QA and non-QA team members. For its creation, it should not take more than one-third of the time allocated for the entire project.

12.5.1 Importance of Test Plan

The test plan is important due to the following reasons:

- It helps individuals who are not part of the QA team like developers, business managers, etc. to understand how exactly the software will be tested.
- It offers a guide or blue print for QA engineers to conduct testing activities.
- The details of aspects such as test scope, estimation, strategy, etc. are clearly written. Providing all these information into a single document makes it easier to be reviewed by the management personnel or to be reused in other projects.

12.5.2 Components of A Test Plan

The components of a test plan are—

- **Scope**: The details about the objectives of a particular project are written. It gives the details of the user scenarios to be used in the test. If necessary, the scope can also specify the scenarios or issues that will not be covered in the project.
- **Schedule**: It gives details of the start dates and deadlines for testers so that results are delivered on time.
- **Resource allocation**: It gives details about which tester will work on which test.
- **Environment**: It provides details of the test environment which will include nature, configuration and availability.
- **Tools**: It gives details about the tools that will be used for testing, bug reporting and other relevant activities.

- **Defect management**: It gives details about the bugs. It specifies how bugs will be reported; to whom and what each bug report needs to be accompanied by. For example, on occurrence of a bug, should it be reported with text logs, screenshots or videos of the code where it has occurred?
- **Risk management**: It gives details about risks that may occur during software testing and what risks the software may suffer from if sufficient testing is not done before the release of the software.
- **Exit parameters**: It gives details of when to stop the testing activities. It gives the description of the results that are expected from the QA operations. It gives a benchmark to the testers so that they can compare with the actual results.

12.5.3 Types of Test Plan

There are three types of test plan—

- Master test plan
- Phase test plan
- Specific test plan

Master test plan: It is a test plan where multiple levels of testing are involved. It includes the complete test strategy.

Phase test plan: It is a test plan where only one phase of the testing strategy is involved. For example, a list of test cases, a list of tools, etc.

Specific test plan: It is a test plan that addresses specific or major types of testing like load testing, security testing, performance testing, etc. A specific test plan is usually designed for non-functional testing.

12.5.4 Creation of A Test Plan

The process of creating a test plan involves the following steps:

Product Analysis

It is very important to analyze a product thoroughly before testing it. The clients and the end users should be researched thoroughly in

order to know their needs and expectations from the product. The answers for the following questions should be ready—

- Who will use the product?
- What the product is used for?
- How the product will work?
- What software/hardware the product uses?

The following approach given in Figure 12.3 can be used to analyze the product.

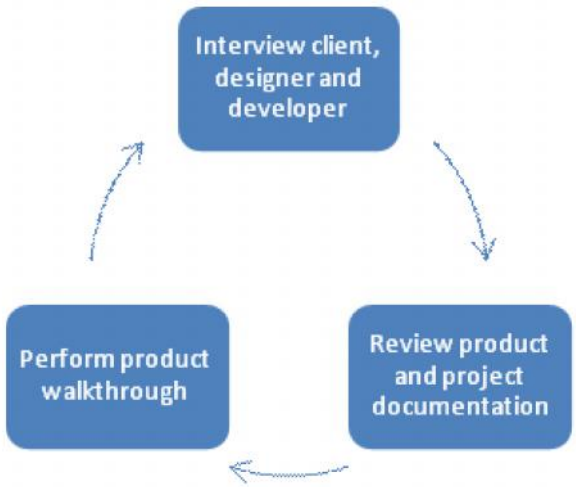


Figure 12.3: Approach for Product Analysis

Develop test strategy

It is a very critical step in making a test plan. A test strategy document is a high level document. It is usually developed by Test Manager. This document defines the objectives of the project's testing and how it will be achieved. It determines the testing efforts and costs. A test strategy involves the following steps shown in Figure 12.4.



Figure 12.4: Process of Test Strategy

- **Define scope of testing:** The “in scope” and “out scope” of the testing should be clearly defined. The components of the system that will be tested are known as “in scope” and those components that won’t be tested are known as “out scope”. A precise scope will give everyone confidence and accurate information of the testing that is being done. All the project members will understand what is being tested and what not.
- **Identify testing type:** It describes the type of tests to be used in the project. Each testing type identifies a special type of product bug. The common goal of all testing types is early detection of all defects before the product is released to the customer. The commonly used testing types are unit test, API testing, integration test, system test, install/uninstall testing and agile testing.
- **Document risks and issues:** It describes all possible risks that may occur during testing phase like insufficient management, tight deadlines, inadequate or erroneous budget estimates. It also describes the effects of these risks on the product.
- **Create test logistics:** In test logistics, the Test Manager mentions the names of the testers and the tests to be run by them and when the test will be run. A tester having the following skills is considered to be most ideal for performing software testing:
  - Understanding customer point of view.
  - Strong desire for quality.
  - Good cooperation.
  - Giving attention to details.

Define test objectives

This phase defines the ultimate goals and the results that is expected from the test execution. The main goal of test objectives is to find as many software bugs as possible so that after release the product is bugs free. Following steps should be considered to define the test objectives—

- All the software features that may be tested should be listed.
- Target or goal based on the above features should be defined.

Define test criteria

The standard or rule on which a test procedure can be based is known as test criteria. There are two types of test criteria—

- **Suspension criteria**: The critical suspension criteria for a test should be specified. During testing if suspension criteria are met, the active test cycle should be suspended until the criteria are resolved.
- **Exit criteria**: The successful completion of a test phase is denoted by the exit criteria. It is the targeted result of the test and is necessary before proceeding to the next development phase. For example, 95% of all test cases must pass.

Resource planning

The detailed summary of all the types of resources required to complete the project task is known as resource planning. Resources can be human, equipment, materials, etc. needed to complete a project. Resource planning helps to determine the number of resources to be used in the project. It helps the Test Manager to make correct schedule and estimation of the project. Human resource can be test manager, tester, developer in test, test administrator, etc. System resource can be server, test tool, network, computer, etc.

Plan test environment

Set up of the software and hardware on which the testing team is going to execute test cases is called the test environment. For example, business environment, user environment, physical environments such as server, front and running environment. To set up the test environment, a strong cooperation is needed between the test team and the development team.

Schedule and estimation

For test estimation, the project is broken down into smaller tasks and time and effort required for each task are allocated. Then, a schedule is created to complete these tasks in the designated time

and using specified amount of effort. Inputs from multiple perspectives are required in order to create the schedule. For example,

- Number of working days, project deadlines, employee availability, daily resource availability.
- Risks associated with the project.

Establish test deliverables

The list of all documents, tools and other components that needs to be developed and maintained in order to conduct the testing is known as test deliverables. A different set of test deliverables is required before, during and after testing.

Test deliverables required before testing are—

- Test plan documents.
- Test cases documents.
- Test design specifications.

Test deliverables required during testing are—

- Test data.
- Simulators.
- Test scripts.
- Error logs and execution logs.
- Test traceability matrix.

Test deliverables required after testing are—

- Defect reports.
- Release notes.
- Test results/reports.
- Installation/test procedures guidelines.

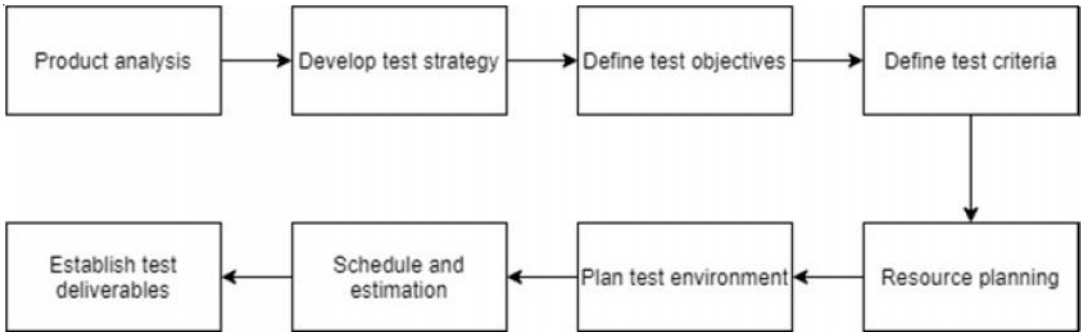


Figure 12.5: Process of Creation of Test Plan

The process of creating a test plan is depicted in figure 12.5.

12.6 ACTIVITY NETWORK FOR SYSTEM TESTING

Activity network for system testing consists of the following activities—

Prepare test plan

Following items are included in a test plan—

- Expected outputs from the system.
- Criteria for evaluating outputs.
- Volume of test data.
- Procedure for using test data.
- Personnel and training requirements.

Specify conditions for user acceptance testing

In order to plan for user acceptance testing, the user and the analyst must agree on the conditions for the test. Most of these conditions can be derived from the test plan. There should be agreement on the test schedules, test durations and the persons designated for the test. It is very important to specify in advance the starting dates and the termination dates for the tests.

Prepare test data for program testing

As the coding of the program begins, the preparation of the test data and documentation is done to ensure that all aspects of the program are properly tested. The data is filled up for future reference.

Prepare test data for transaction path testing

This activity is responsible for the development of data required for testing every condition and transactions that are to be introduced into the system. In order to obtain reliable results, the paths for each transaction from origin to destination should be carefully tested.

Plan user training

The design of the user planning is done to prepare the user for testing and converting the system. User involvement and training is done parallel with programming for the following reasons—

- System group has time for training.
- System gets a clearer image of the user's interest in the new system.
- Trained user participates more effectively in system testing.

Preparation of a checklist is important for user training. The checklist includes provisions for developing training materials like training manuals and other documents that are necessary to complete the training activity. It is important to specify and document the facility requirements and the necessary hardware. Supervisors are trained using a common procedure because of the following reasons—

- User supervisors have knowledge about the capabilities of their staff.
- Staff members respond and accept instructions better from supervisors as compared to outsiders.
- Users are better candidates for handling user training than the system analyst because the users are more familiar with their particular problems.

Compile and assemble programs

All the programs must be compiled and assembled for testing. Before doing this, a complete program description should be available. The description should include purpose of the program, its use, the programmers who wrote the programs and the amount of time taken by the computer to run it. For further reference, program and system flowcharts of the project should be available. Three types of activities are done in this phase—

- **Desk checking**: Programming errors are uncovered in the source code.
- **A run order schedule**: The transactions to be tested and the order in which they should be tested are specified.
- **A test scheme**: How the program should be debugged is specified in the test scheme. It follows two approaches—
  - Bottom-up approach which is responsible for testing small-scale program modules which are linked to a higher-level module. This is done until the program is completed.
  - Top-down approach in which the general program is tested first. It is followed by adding program modules one level at a time to the lowest level.

Prepare job performance aids

In this activity, specification and schedule of the materials to be used by the personnel to run the system are done. This includes activities like display of materials such as program codes, a list of input codes attached to CRT terminals and an instruction schedule posted to load the disk drive. These aids help in the reduction of training and employment of personnel at lower positions.

Prepare operational documents

During the test plan stage, it is important to finalize all the operational documents. Documentation is also done on the experience, training and educational qualifications of personnel in order to ensure proper operation of the system.

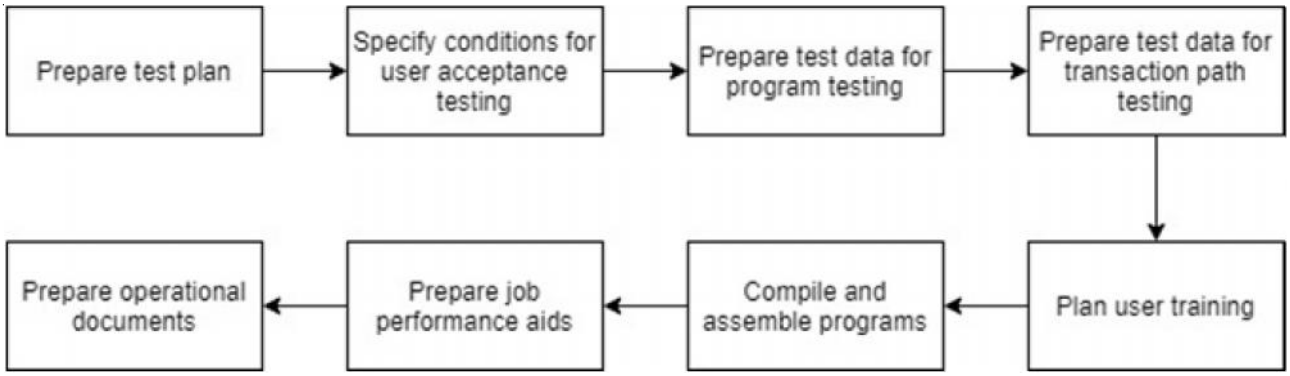


Figure 12.6: Process of Activity Network for System Testing

All the steps involved in activity network for system testing are shown in figure 12.6.

12.7 TYPES OF SYSTEM TESTING

The different types of system testing which are presented in figure 12.7 are explained in the following.

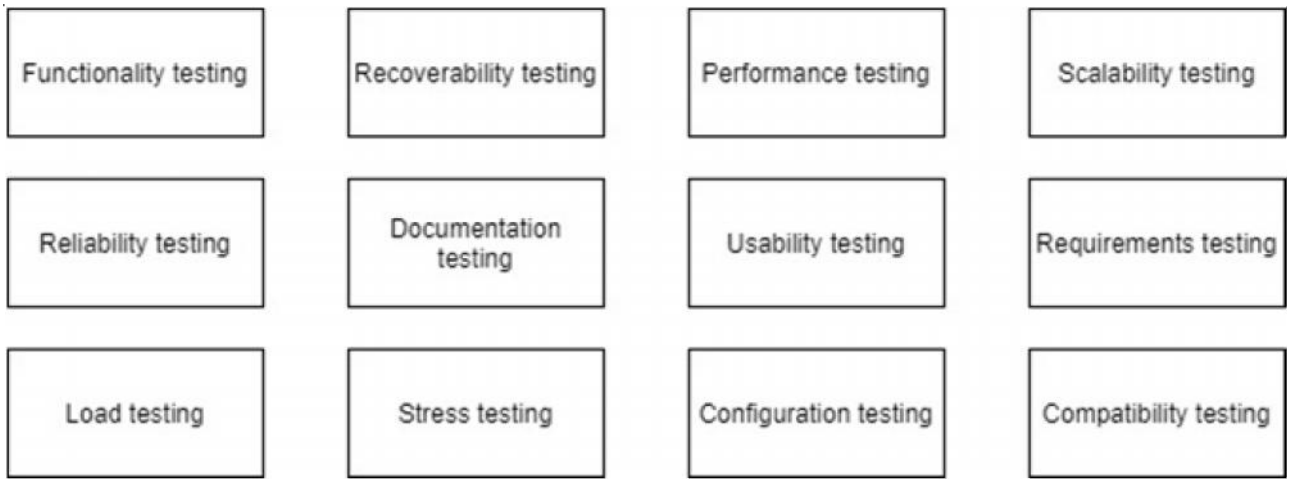


Figure 12.7: Types of System Testing

Functionality testing

This testing helps to test whether the functionality of a product is working as per the requirements specifications and it should be within the capabilities of the system. Functionality testing can be done manually or with automated tools.

Recoverability testing

This type of testing helps to determine whether operations can be continued after a disaster or after the integrity of the system is lost. For example, we are downloading a file and suddenly connection goes off. After connection resumes, downloading starts from where we have left. It does not start from the beginning.

Performance testing

This type of testing is done to test the performance of the system under various conditions in terms of performance characteristics. This testing is also known as compliance testing with respect to performance. This testing ensures that the system requirements are met. Performance testing is done based on three categories: speed, scalability and stability.

Scalability testing

This type of testing is done to test the system's scaling abilities in various terms such as geographic scaling, user scaling and resource scaling.

Reliability testing

This testing is done in order to make the system bug free. This type of testing is done whether a system can operate for a longer duration without developing failures.

Documentation testing

This type of testing is done to test whether the system's user guide and other help topics documents are correct and usable.

Security testing

This type of testing is done to ensure that the programs can be accessed by authorized personnel and that the authorized personnel can access the functions available to their security levels. The system should not allow unauthorized access to data and resources. The goal of security testing is to test how well a system protects against unauthorized internal or external access or damage. Security testing can be done in areas like authentication, authorization, data validation, transport security, data protection, session management, etc.

Usability testing

This type of testing is done to test whether the system is easy to use, learn and operate.

Requirements testing

This type of testing is done to check how the real user will work in the environment. Actual users try to test goals and tasks associated with a product under controlled conditions. It is mainly used for the design of applications. This testing is done by direct observations of people using the system, usability surveys, etc. User tests under this testing is also known as beta testing.

Load testing

This type of testing is done to test how the application behaves when multiple users try to access the system simultaneously across multiple locations. This testing ensures if the performance of the system is acceptable at a pre-determined load level. It tests the normal and predefined conditions of the system.

Stress testing

It is also known as endurance testing. This type of testing is done to test if the system is going to continue working it encounters large volume of data than expected. It may contain internal tables, input transactions, disk space, communication channels, etc. It tests the system under extreme conditions.

Configuration testing

This type of testing is done to test compatibility issue. This testing helps to determine minimal and optimal hardware and software configuration. It shows the effects of adding or modifying resources such as disk space, memory, CPU, network card, etc.

Compatibility testing

This type of testing is done to test whether the system is capable of running on different operating system, hardware, network environments, applications, etc. It is similar to multi-platform testing. It is more useful in web-based applications where we can test if the application can be accessed from every browser.



CHECK YOUR PROGRESS

Q3: What are the types of test plans?

Q4: Write the steps involved in the approach for product analysis.

Q6: What is security testing?



12.8 LET US SUM UP

- System testing is a level of testing that helps to validate a complete and fully-integrated software product. Its purpose is to evaluate the end-to-end system specifications.
- It is independent of the development team and helps in testing the quality of the system impartial.
- System testing is performed after the integration testing step and before the acceptance testing step.

- System testing process has the following steps: create a test plan, test case creation, select test data, test case execution, bug reporting and fixation and repeat the test cycle (if required).
- Test data may be artificially taken or taken from live data.
- A test plan is a detailed document that has a catalog containing the test strategy, objectives, schedule, estimations, deadlines and resources that are required for completing the particular project.
- The components of a test plan are: scope, schedule, resource allocation, environment, tools, defect management, risk management and exit parameters.
- There are three types of test plan: master test plan, phase test plan and specific test plan.
- The process of creation of a test plan involves the following steps: product analysis, develop test strategy, define test objectives, define test criteria, resource planning, plan test environment, schedule and estimation and establish test deliverables.
- Activity network for system testing consist of the following activities: prepare test plan, specify conditions for user acceptance testing, prepare test data for program testing, prepare test data for transaction path testing, plan user training, compile and assemble programs, prepare job performance aids and prepare operational documents
- The different types of system testing are functionality testing, recoverability testing, performance testing, scalability testing, reliability testing, documentation testing, security testing, usability testing, requirements testing, load testing, stress testing, configuration testing and compatibility testing



12.9 FURTHER READINGS

1. Glenford J. Myers, Corey Sandler, Tom Badgett, The Art of Software Testing, 3rd Edition
2. Ron Patton, Software Testing, 2nd Edition, 2005



12.10 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q No 1: System testing follows white box testing – False.

Ans. to Q No 2: System testing is performed before integration testing and after acceptance testing.

Ans. to Q No 3: Test data may be artificially taken or taken from live data. It means test data can be created solely for test purpose or may be taken from user's actual files. Artificial data should be properly created. It should provide all combination of values or formats. It should be made possible to test all logic and transaction path subroutines. Live data are biased towards typical values. Artificial data provides extreme testing conditions. It provides extreme values for testing the boundaries of the candidate system.

Ans. to Q No 4: There are three types of test plan—

- Master test plan
- Phase test plan
- Specific test plan

Ans. to Q No 5: The steps involved in the approach for product analysis: Perform product walkthrough, interview client, designer and developer and review product and project documentation in loops

Ans. to Q No 6: Security testing is a type of testing is done to ensure that the programs can be accessed by authorized personnel and that the authorized personnel can access the functions available to their security levels. The system should not allow unauthorized access to data and resources. The goal of security testing is to test how well a system protects against unauthorized internal or external access or damage. Security testing can be done in areas like authentication, authorization, data validation, transport security, data protection, session management, etc.



12.11 MODEL QUESTIONS

Q1. What is system testing?

Q2. What is the need of system testing?

Q3. Explain the components of a test plan.

Q4. How a test strategy is developed in the creation of a test plan?

Q5. What are test criteria? Explain its types.

Q6. Explain the test deliverables required before, during and after testing.

Q7. Write notes on activity network for system testing.

Q8. What is the difference between configuration testing and compatibility testing?

Q9. What is test plan?

Q10. What are the approaches for a test scheme?

*** ***** ***

UNIT 13: QUALITY ASSURANCE

UNIT STRUCTURE

13.1 Learning Objectives

13.2 Introduction

13.3 Software Quality

13.3.1 Functional and Non-functional Requirements

13.4 Levels of Quality Assurance

13.5 Quality Factor Specification

13.5.1 McCall's Factor Model

13.6 Software Requirement Specification

13.6.1 Structure of Software Requirement Specification

13.6.2 Characteristics of a Good Software Requirement Specification

13.7 Software Design Specification

13.7.1 Characteristics of a Good Software Design Specification

13.7.2 Software Design Principles

13.8 Let Us Sum Up

13.9 Further Readings

13.10 Answers To Check Your Progress

13.11 Model Questions

13.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- understand the essence of quality and the concept of software quality as a whole
- learn about the issues and factors that affect the quality of software
- learn about the significance of Software Requirement Specification and how to formulate it
- understand principles of Software Design Specification

13.2 INTRODUCTION

In the previous unit, we learned about the significance of System testing, its different types, cases and the process of performing testing. In this unit, we will understand the concept of software quality, the factor and level of quality assurance, and finally how quality is associated with software requirement specification and design specification. In the next unit, we will discuss about the implementation of a system.

13.3 SOFTWARE QUALITY

When we say quality of a product, different people may visualize it differently. It depends on the individual's need, interest, opinion and point of view. There can even be multiple levels in an individual's interpretation of quality. For example, a person with low income may regard a simpleMaruti-Suzuki car as having great quality, whereas a person with very high income may find the lowest model of Mercedes is of low quality. Another example is that in some manufactured products like say box-cover, if there are some products that miss the approved standards by a very slight margin, still they may be considered good because there is a high probability that customers might never notice it or even ignore such small defects. But same cannot be said in case of other products like eye glasses, medicines, etc. where even a small defect may not be acceptable. In some cases, people associate expensive and high price with good quality, and lesser price with lower quality. For example, an expensive item like bag or watch of international brands may be considered very good quality and the same item of lesser price may be considered having lesser quality.

To quantify quality in industry, two definitions of quality exist: "conformation to requirement" and "fitness for use". Conformation to requirement means that the requirements and standards for a product are clearly predefined and the product has to satisfy all these set requirements in order to be considered of good quality. For example, a hatchback car has to have certain requirements like a rear flip-up hatch door, front-wheel drive-train, foldable rear seats, etc. A car has to satisfy all the essential necessities of hatchback car to be considered a quality hatchback car.

Fitness for use means that not only the essential requirements but the requirements of the customers are also taken into consideration. For example, customers may desire a certain product with some additional requirements, like a hatchback car with added space, comfort, performance, etc. As different customer may have different requirements, so there may be different models of the same products.

13.3.1 Functional and Non-functional requirements

For a product to be successful it should be able to meet all its goals and requirements. In case of software, there are two types of requirements which are described as below:

- **Functional requirements**: These are the fundamental criteria that the software should essentially satisfy and are usually stated by the users / customers. Some of the examples of functional requirements are that the system should be able to accept and process valid input, produce the anticipated output, should be free of errors and bugs, etc. Here the user is mostly able to visualize them in the software.
- **Non-functional requirements**: These are the requirements that are mostly specified by the software developer or project leader based on the requirements and restrictions of the system as a whole. Although it consists of many attributes like Complexity, Efficiency, Interoperability, Learnability, Modularity, Portability, etc. (as shown in table 13.1), but not all of these attributes are obligatory to be fulfilled.

Sl. no.	Attribute	Significance
1	Complexity	Software complexity describes a specific set of characteristics that focus on how our piece of code or software interacts with other pieces of code.
2	Efficiency	Software Efficiency determines the amount of code and testing resources required by a program to perform a particular function.
3	Interoperability	It is the ability of two or more systems to communicate or exchange data easily and to use the data that has been exchanged.
4	Learnability	Learnability is a quality of software products and interfaces that allows users to quickly become familiar with them and able to make good use of all their features and capabilities
5	Modularity	Modularity is the degree to which a system's components may be separated and recombined easily.
6	Portability	It is the ability of a software application to run on numerous platforms such as data portability, hosting, viewing, etc.
7	Reliability	It relates to the software's ability to continue operating, under given environmental conditions, for a particular amount of time.
8	Resilience	Software resilience refers to the ability to absorb the impact of disruption in one or more parts of a system, while still continuing to provide an acceptable level of service.
9	Reusability	Reusability is the likelihood that a segment of source code can be used again to add new functionalities with slight or no modification.
10	Robustness	Robustness is the ability of the system to handle with abnormalities like errors during execution, erroneous input, sudden changes in hardware or software environment, etc.
11	Scalability	It is the ability of a system to handle the demand for stress caused by increased usage without decreasing performance.
12	Security	It is the ability of a system to resist or block malicious or unauthorized attempts that destroy the system and at the same time provide access to legitimate users.

13	Supportability	It is the ability of a system that satisfies necessary requirements and needs to identifying and solving problems.
14	Testability	It shows how well the system or component facilitates to perform tests to determine whether the predefined test criteria have been met.
15	Usability	It is described as how the user is utilizing a system effectively and the ease of which users can learn to operate or control the system. One of the well-known principles of usability is to make it very user-friendly and keep it as simple as possible.

Table 13.1: Non-functional Requirements

In software industry, the essential conformation to requirement is that software should be able to accept and process valid input, produce the anticipated output, should be free of errors and bugs. The assessment of bugs and errors can be measured in two ways: defect rate (i.e., number of defects per million lines of source code, per function point, or other unit) and reliability (i.e., number of failures per n hours of operation, mean time to failure, or the probability of failure-free operation in a specified time). But software quality also depends on other non-functional attributes like Complexity, Efficiency, Interoperability, Learnability, Modularity, etc. Particular software may not be able to satisfy all the above mentioned attributes — for example, if we want to improve robustness then this might lead to loss of efficiency. Therefore, the quality management team should try to find out what are the major criteria for the system being developed. Say, for example, if efficiency is very important then there is reason why other criteria like scalability or usability may be relaxed. This should be stated in the quality plan so that the engineers working on the software development can cooperate to achieve this.

13.4 LEVELS OF QUALITY ASSURANCE

As we have discussed earlier that software quality depends not only on functional attributes but also on non-functional attributes. Thus Software Quality Assurance (SQA) is a method to assure that the best product or

service is being delivered the customer by ensuring that all the essential functional as well as non-functional attributes are satisfied. Software quality assurance gives importance not only to the quality of the software but also to the various process that were used to develop, test and release the software.

Thus the purpose of Software Quality Assurance (SQA) is to ensure that the software meets the required specifications and standards. It provides a clear picture about the performance of the software, information of resource allocation, existence of risk, deficiencies, etc. To assure software quality, many reviews and audits are performed throughout the life cycle of system development process.

There are several levels of quality assurance which are described as below:

- Level 1 " Code Walk-through: Here the software is checked to see if the official coding rules are followed or not. In code walk-through, the documentation and level of in-code comments are thoroughly examined.
- Level 2 " Compilation and Linking: Here the software is checked to see if it can compile and link in all official platforms and operating systems.
- Level 3 " Routine Running: When placed under different constraints and conditions, it is checked to see if the software can run still properly.
- Level 4 " Performance test: At this final level, it is checked to see whether the performance of the software satisfies the specified performance level.



EXERCISE

1. In industry, two definitions of quality exist: "" and "". (Fill in the blank)
2. Complexity, Efficiency, Interoperability, Learnability, Modularity, etc. are the functional requirements of software. (True or False)
3. Why software quality cannot be determined only by satisfying functional requirements?



CHECK YOUR PROGRESS

Q1: In terms of quality, what does "conformation to requirement" and "fitness for use" mean?

Q2: What are the functional requirements and non-functional requirements of software?

Q3: What is the purpose of Software Quality Assurance?

13.5 QUALITY FACTOR SPECIFICATION

Software may successfully fulfill the basic requirement of essentially having no inherent bugs or errors, but it may not stand strong on other non-measurable attributes like Reusability, Robustness, etc.As a result, the overall quality of the software may be hampered. Thus, software quality is assessed not only by measureable attributes like bugs, logical errors, etc., but also depends on other non-functional attributes like Interoperability, Learnability, Maintainability, etc. So, we need an all-inclusive method which will cover all the attributes and aspects of software quality.

13.5.1 McCall's Factor Model

The McCall's Factor Model classifies all software requirements into eleven software quality factors. The 11 factors are grouped into three categories – *product operation*, *product revision*, and *product transition factors*.*Product operation*has five factors " Correctness, Reliability, Efficiency, Integrity, Usability.*Product revision*has three factors" Maintainability, Flexibility, Testability. And *Product transition*also has three factors " Portability, Reusability, Interoperability.

(i) Product operation factors: It has five software quality factors. It relates to the requirements that directly affect the daily operation of the software. Each one of them are described as given below:

- Correctness: It says how accurately and correctly the software can do its predefined task and provide the required output. It has many elements like:

- Output mission.
- The required accuracy of output that can be negatively affected by inaccurate data or inaccurate calculations.
- The completeness of the output information, which can be affected by incomplete data.
- The up-to-dateness of the information defined as the time between the event and the response by the software system.
- The availability of the information.
- The standards for coding and documenting the software system.

- Reliability: It relates to the extent the software can keep functioning in the event of environmental failures. They determine the maximum allowed failure rate of the software system, and can refer to the entire system or to one or more of its separate functions.
- Efficiency: It is about the hardware resources required for the software to run and provide the required services. It includes processing capabilities (in MHz), storage capacity (in MB or GB), data communication capability (in MBPS or GBPS) and the time required between recharging of the system's portable units, such as, information system units located in portable computers, or meteorological units placed outdoors.
- Integrity: It says how the software provides security by giving valid access to authorized person or group of persons, the level of access (like read/ write permission, etc.) and firmly blocking people with unauthorized access.
- Usability: Usability requirement relates to the extent of effort required for a user to learn, operate, interpret and understand the functions of the software.

(ii) Product revision factors: It has three factors which are as follows:

- Maintainability: This factor specifies the effort required to identify the software failure, to find the reason for the failure, to correct the failure, and to finally verify its correctness.
- Flexibility: In the event of accommodating additional features and requirements, whether the software has the ability to adapt and how much effort is required to do so without bringing major changes to the software.
- Testability: It deals with testing all the components and functions of the software and obtaining a report about the detected faults. To find whether all the components are working fine predefined intermediate results and log files are used; as well as automatic diagnostics are performed by the software prior to starting the system.

(iii) Product transition factors: There are three factors which are as follows:

- Portability: Portability requirement depicts how well the software is able to adapt in different hardware and operating systems and still perform its required functions.
- Reusability: This factor deals with the ability to reuse the various modules of the software in order to augment new features and capabilities in the existing system.
- Interoperability: This factor focus on the ability to integrate and interact with other software systems or other equipment firmware. For example, the firmware of the production machinery and testing equipment interfaces with the production control software.

13.6 SOFTWARE REQUIREMENT SPECIFICATION

Let us say a client / customer has requested us to make a comprehensive School Management Software. Let us assume that you have experience in developing Employee Management System and you

think that apart from a few additional features, your previous knowledge is enough to design the new software. You have some verbal discussions with your team members as well as with the client and start making the software. You may end up with following scenarios: the end product may not be exactly as the client wanted and there may be some features missing.Say for example you did not know that the teachers can attend or take leave from the school in the middle of the day; or say you did not know that apart from attendance, the teachers are required to register the detail progress of the students. Another case may be that you have added a functionality which is not required by the client at all. For example, say you have given the provision that after the teacher has entered the marks, the students can view their marks immediately; but in order to prevent any confusion among the students, the school wants to release the marks only at a certain date after the marks of all the subjects have been entered,evaluated and properly checked.There is another scenario: while working on certain functionality, there is dispute and disagreement among your own team members. For example, some of your team members might say that the software should be fully accessible to the teachers even outside the school premises, whereas some might argue that outside the school premises the teachers should have limited accessibility. Thus you will eventually discover that even though you might have previous experiences in similar areas, but every organization has its own unique sets of requirements and constraints, which along with all the stakeholders, has to be clearly discussed, thoroughly examined and properly planned, otherwise there will be utter chaos, disarray, disorganization and disappointment.

With the software requirements specification (SRS), the developers, the product owners, and various other stakeholders will have to meet, discuss and formally write down about the system's detailed description and purpose, its functionalities, its interfaces, performance criteria, functional and non-functional requirements, quality standards, security issues, financial as well as time constraints,resource allocation, risk involved, deficiencies, etc., all has to be noted down. Here, the developers will get to know what goals and vision the clientexactly has in his or her mind, whereas the client

will get to know about the technology aspects.Once the SRS has been written, developers can refer it at any point of time to clear any ambiguity, managers can use it to plan the system development and maintenance process, users / clients can use it to understand the software, test engineers can use it to create test cases, develop validations sets, etc. Thus the SRS will provide the system with a concrete and well-defined structure to follow.

13.6.1 Structure of Software Requirement Specification

The structure of a software requirement specification (SRS) documenthas been described in detail, as given below:

(i) Introduction: The first section of SRS is the introduction part where we describe the overall SRS meaning, its scope, and its structure.It includes Purpose, Intended audience, Intended use, Scope and Definitions and acronyms.

- **Purpose**: It is a short section where we describe the purpose of the SRS documentation, its structure, type of requirements andthe different stakeholders who will work with the document.
- **Intended audience**: It is a detail description of what the stakeholders like product owners, investors, business analysts, developers, testers, etc will do and how they will participate in its creation.
- **Intended use**: It the information about the situations under which the stakeholders will use the SRS document. For example: designing and brainstorming new features, planning project duration, estimating costs, evaluating risks, monitoring team's success, clearing conflicting scenarios etc.
- **Scope**: Here we have to describe the different users who will use the software. We also have to briefly describe all essential parts of the architecture, how our solution is different from others and what the marketing strategies are.

- **Definitions and acronyms**: The developer and engineers may use certain technical acronyms whereas the clients / product owners may use certain financial acronyms. This will create confusion among both the parties. So, to remove such misunderstanding, the SRS should have all the acronyms, their definitions, their functions, affiliated stakeholder, etc.

(ii) **Overall description**: This section of the SRS includes user needs, assumptions and dependencies.

- **User needs**: Here we need to decide which user problems we want to solve and what solutions we want to give. If the users are very diverse, then we can also divide them into groups. We should also leave room for modifications, if in case a problem turns out to be more significant than we initially thought.
- **Assumptions and dependencies**: It describes the team's views and expectations about the product and its functionalities. For instance, if we are developing a mobile app to help drivers drive at night, then it will obviously be used mostly in the night. Though some may use it in the day, but our focus and goal should be to develop the features that will be more relevant for night time usage rather than for day time.

(iii) **System features and requirements**: The earlier sections give a broad idea about the product, but this section gives a more in-detail description about the software's different functionalities and its execution criteria. This the most important section of the SRS where we set up 'use cases' and distinctly define how a user will interact with the system. Here each user interaction can be a use case. This section of the SRS includes system requirements, functional and non- functional requirements, external interface requirements and other supplementary requirements.

- **Functional requirements**: Here the functional requirements of the software are clearly defined, that is, here the importance is given to the essential question: 'What has to be developed'? or 'What are supposed to be the essential functions of the system'? The internal logic like 'how to develop the functionalities' or 'when to develop them' is not concentrated right now. While describing the functional requirements of the system, the views of the users / customers / product owners are primarily taken into consideration.
- **External interface requirements**: Say, for a food delivery mobile app, 'search' will be an essential functional requirement, but the search option itself may have a list of other requirements like search by location, by restaurant, by food, by price, by user satisfaction, etc. Such requirements are called as external interface requirements.They include the elements like themes, designs, list of pages, etc., that will be visible to the client / user while using the system. Depending on the project, external interface requirements can consist of four types: user interface, software interface, hardware interface, communication interface.
- **System requirements**: System requirements describe the minimum as well as maximum hardware requirement for the software to run smoothly. This is especially important for the developers because they should not develop software that does not comply with the already stated hardware requirements.
- **Non-functional requirements**: As the name suggests, here the non-functional requirements like Efficiency, Interoperability, Modularity, Portability, Reliability, Robustness, etc. are elaborated. For functional requirement we deal with the questions like: 'What has to

be developed'? or 'What are supposed to be the essential functions of the system'?; for non-functional requirements we have handle the questions like: 'How each feature / function of the software should work? Given certain conditions and limitations, how it should behave? How to handle accessibility and security issues?' etc. Here we have to describe how the software has to work in terms of Efficiency, Interoperability, Modularity, Scalability, etc. As the non-functionalattributesarevery difficult to describe because different person will have a different point of view, so we can assign score or point to each attribute.

- **Other supplementary requirements**: Finally, at the end of the SRS we include additional details like appendixes, glossaries and references that might be of some help to the developers and engineers.

13.6.2 Characteristics of a Good Software Requirement Specification

The following are the characteristics of a good Software Requirement Specification (SRS):

(i) Correctness: A SRS should accurately describe all the required functionalities and specifications of the software that is to be built, without any conflicts. User / client review can be used to check the correctness of the SRS.

(ii) Completeness: A SRS should fully and completely describe all the functionalities and specifications of the software to be developed. It should clearly address various queries like 'What exactly the software is supposed to do?', 'Given different types of input, how the software is supposed to response?', 'How complex it should be?', 'What security privileges should be there?' etc. It is very difficult to determine completeness as it is very hard to spot the absence of specification. So, one has to vigorously and diligently look for all the different aspects and perspectives of the software.

(iii) Consistency: SRS is said to consistent if there is no conflict and disagreement between any requirements. There should not be any logical conflicts like for example say, one part of the SRS says the user can access its configuration page but another part in the SRS says only administrator can have access to the configuration page, or for example one part in the SRS says that the administrator can perform backup any time, but another part says backup is restricted during peak hours as it might slow the system. There should not even be conflicts in simple terminology definition, for example different requirements may use different terms to refer to the same objects.

(iv) Unambiguousness: A SRS is said to unambiguous when every requirement and every element of the software has been uniquely interpreted and distinctively explained. To avoid ambiguity, one should avoid using words, phrases or sentences which can have vague or loose translations, and pursue using methods like ER diagrams, reviews, inspections, etc.

(v) Ranking for importance and stability: SRS should clearly classify each functional requirement, feature, use case, or user story into its level of importance or significance to mark its rank or stability. Such ranking will come handy if there aresome budget cuts, schedule overruns, loss of resources or addition of new requirements. Here stability implies the probability of changes in the requirement in future.

(vi) Modifiability: A SRS is said to be modifiable if to some extent the SRS can be modified and changes can be incorporated, without affecting other characteristics like unambiguity, correctness, etc. Also it is necessary to maintain the history of the changes that has been made.

(vii)Verifiability: A SRS is verifiable if there exists some method to measure the extent to which each requirement has been met. To determine whether each requirement has been met, we can perform tests or should at least be able to use other verification approaches, such as inspection, reviews or demonstrations.

(viii) Traceable: The SRS is traceable if each requirement in the SRS is uniquely labeled and identified and that it is forward traceable as well as backward traceable. Forward traceable means we are able to trace a requirement to its design component and then to the code segment. Backward traceable means we are able to trace each requirement explicitly back to its source in the earlier documents.

(ix) Design Independence: The SRS should not include any implementation detail so that there are multiple design alternatives for the system.

(x) Testability: A SRS is testable if it is written in such a way that it is very easy to generate test cases and test plans from the report.

(xi) Understandable: The language used in writing the SRS should be simple and very clear; and the use of formal notations and symbols should be limited as much as possible so that the client / user / investors can easily understand it.

(xii) Right level of abstraction: The SRS should be written according to the appropriate level of abstraction. If the SRS is written for the requirements phase, the details should be explained explicitly, whereas, for a feasibility study, fewer details can be used.



EXERCISE

1. The McCall's Factor Model classifies all software requirements into _____software quality factors which are grouped into _____categories.
2. What are the factors included in each of the three categories of McCall's Factor Model.
3. What will happen if you do not prepare the SRS?
4. Name some stakeholders of software product.



CHECK YOUR PROGRESS

Q4: Why do we have to use McCall's Factor Model?

Q5: State the significance of Software Requirement Specification (SRS).

Q6: While constructing the Software Requirement Specification, how to construct the functional and non-functional requirements?

Q7: State the characteristics of a good SRS.

13.7 SOFTWARE DESIGN SPECIFICATION

We have understood that Software Requirement Specification (SRS) describes what the various stakeholders (like customers, users, project managers, software developers, engineers, investors, test engineers, etc.) want in terms of features, functions, performance, constraints, quality, security, etc. Software Design Specification (SDS) describes the technical details of how the features and functions have to be implemented. It will contain information about the software architecture and describe how each component will contribute to meeting the requirements. It will help the programmer during coding and implementation.

13.7.1 Characteristics of a Good Software Design Specification

A good design specification should have the following characteristics:

(i) **Correctness**: Software design should be correct as per requirement.

(ii) **Completeness**: The design should have all components like data structures, modules, and external interfaces, etc.

(iii) **Efficiency**: It should allocate resources very efficiently.

(iv) **Flexibility**: In case of occurrence of changes, it should be modifiable.

(v) **Consistency**: There should not be any inconsistency in the design.

(vi) **Maintainability**: The design should be as simple as possible so that it can be easily maintained by other designers.

13.7.2 Software Design Principles

Software design principles are applied to effectively handle the complexity of software design. By following the software design principles we can reduce the effort of design as well asreduce the occurrence of errors. Following are the principles of Software Design:

(i) **Problem Partitioning**: Though small projects may be easier to handle, the same cannot be said of large projects. Large projects cannot be handled all at once, so, we have to divide the large projects into smaller parts and deal with the smaller parts separately. Breaking it into smaller pieces makes the large project easier to understand, test, modify and maintain. Since the smaller parts are fragments of a large project, the smaller parts cannot be independent of each other and has to communicate with each other to complete the whole project.

(ii) **Abstraction**: In this method the designer is not concerned about the internal details about the implementation. There are two common abstraction mechanisms: Functional abstraction and Data abstraction. In functional abstraction,functions are used to specifythe job it is supposed to do. The details of the algorithm to accomplish the functions are not visible to the user of the function. It is done by using function oriented design approaches. In data abstraction, details of the data elements are not visible to the users of data. It is done by using object oriented design approaches.

(iii) **Modularity**: In modular design, the system is divided into modules such that each module is well-defined, more readable, less complex, eachmodule has specified objectives, can be separately compiled and saved in the library, can be separately written by different people, provides more checkpoints to measure progress, provides a framework for comprehensive testing. There are some disadvantages of modularity like longer compilation time, execution time, loading time, inter-module

communication time and storage size and more documentation has to be maintained.

(iv) Strategy of Design: To design a system, there are two possible approaches: Top-down approach and Bottom-up approach. In top-down approach we identify the main components and then decomposing them into their more detailed sub-components. In bottom-up approach we begin with the lower details and move up the hierarchy. This approach is suitable in the case of an existing system.



CHECKYOURPROGRESS

Q8: What is the difference between Software Requirement Specification and Software Design Specification?

Q9: Describe the characteristics of a good Software Design Specification.



13.8 LET US SUM UP

- Quality of a product depends on an individual person's need, interest, opinion and point of view. Different people may visualize it differently.
- To quantify quality in industry, two definitions of quality exist: "conformation to requirement" and "fitness for use".
- Software quality depends not only on functional attributes but also on other non-functional attributes like Complexity, Efficiency, Interoperability, Learnability, Modularity, Portability, Reliability, Resilience, Reusability, Robustness, Scalability, Security, Supportability, Testability and Usability.
- The quality management team should try to find out what are the major non-functional attributes for the system being developed.
- The purpose of Software Quality Assurance (SQA) is to ensure that the software meets the required specifications and standards.

- The McCall's Factor Model classifies all software requirements into eleven software quality factors which are grouped into three categories.
- Without software requirements specification (SRS), there will be utter chaos, disarray, disorganization and disappointment among all the relevant stakeholders.
- The characteristics of a good SRS are (i) Correctness, (ii) Completeness, (iii) Consistency, (iv) Unambiguousness, (v) Ranking for importance and stability, (vi) Modifiability, (vii) Verifiability, (viii) Traceable, (ix) Design Independence, (x) Testability, (xi) Understandable and (xii) Right level of abstraction.
- Software Design Specification (SDS) describes the technical details of how the features and functions have to be implemented. It will help the programmer during coding and implementation.



13.9 FURTHER READINGS

1. Wixom, B. H., Roth, R. M., (2008), Systems Analysis and Design, Wiley Publishing
2. Kendall, K. E., Kendall J. E., (2019), Systems Analysis and Design, Pearson



13.10 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q No1: Conformation to requirement means that the requirements and standards for a product are clearly predefined and the product has to satisfy all these set requirements in order to be considered of good quality. For example, a hatchback car has to have certain requirements like a rear flip-up hatch door, front-wheel drive-train, foldable rear seats, etc. A car has to satisfy all the essential necessities of hatchback car to be considered a quality hatchback car.

Fitness for use means that not only the essential requirements but the requirements of the customers are also taken into consideration. For example, customers may desire a certain product with some additional requirements, like a hatchback car with added space, comfort, performance, etc. As different customer may have different requirements, so there may be different models of the same products.

Ans. to QNo2:Functional requirements: These are the fundamental criteria that the software should essentially satisfy and are usually stated by the users / customers. Some of the examples of functional requirements are that the system should be able to accept and process valid input, produce the anticipated output, should be free of errors and bugs, etc. Here the user is mostly able to visualize them in the software.

Non-functional requirements: These are the requirements that are mostly specified by the software developer or project leader based on the requirements and restrictions of the system as a whole. Although it consists of many attributes like Complexity, Efficiency, Interoperability, Learnability, Modularity, Portability, Reliability, Resilience, Reusability, Robustness, Scalability, Security, Supportability, Testability and Usability, but not all of these attributes are obligatory to be fulfilled.

Ans. to QNo3: The purpose of Software Quality Assurance (SQA) is to ensure that the software meets the required specifications and standards. It provides a clear picture about the performance of the software, information of resource allocation, existence of risk, deficiencies, etc. To assure software quality, many reviews and audits are performed throughout the life cycle of system development process.

Ans. to Q No4: Software may successfully fulfill the basic requirement of essentially having no inherent bugs or errors, but it may not stand strong on other non-measurable attributes like Reusability, Robustness, etc. As a result, the overall quality of the software may be hampered. Thus software quality is assessed not only by

measureable attributes like bugs, logical errors, etc., but also depends on other non-functional attributes like Interoperability, Learnability, Maintainability, etc. McCall's Factor Model gives us an all-inclusive method which will cover all the attributes and aspects of software quality.

Ans. to Q No5: With the software requirements specification (SRS), the developers, the product owners, and various other stakeholders will have to meet, discuss and formally write down about the system's detailed description and purpose, its functionalities, its interfaces, performance criteria, functional and non-functional requirements, quality standards, security issues, financial as well as time constraints, resource allocation, risk involved, deficiencies, etc., all has to be noted down. Here, the developers will get to know what goals and vision the client exactly has in his or her mind, whereas the client will get to know about the technology aspects. Once the SRS has been written, developers can refer it at any point of time to clear any ambiguity, managers can use it to plan the system development and maintenance process, users / clients can use it to understand the software, test engineers can use it to create test cases, develop validations sets, etc. Thus the SRS will provide the system with a concrete and well-defined structure to follow.

Ans. to Q No6: While writing the Software Requirement Specification, the functional requirements of the software should be clearly defined, that is, here the following question should be asked: 'What has to be developed'? or 'What are supposed to be the essential functions of the system'? The internal logic like 'how to develop the functionalities' or 'when to develop them' are not dealt right now. Here primarily, the views of the users / customers / product owners are taken into consideration. For non-functional requirements we have handle the questions like: 'How each feature / function of the software should work? Given certain conditions and limitations, how it should behave? How to handle accessibility and security issues?' etc. Here we have to describe how the software has to work in terms of Efficiency,

Interoperability, Modularity, Scalability, etc. As the non-functional attributes are very difficult to describe because different person will have a different point of view, so we can assign score or point to each attribute.

Ans. to Q No7: The characteristics of a good SRS are (i) Correctness, (ii) Completeness, (iii) Consistency, (iv) Unambiguousness, (v) Ranking for importance and stability, (vi) Modifiability, (vii) Verifiability, (viii) Traceable, (ix) Design Independence, (x) Testability, (xi) Understandable and (xii) Right level of abstraction.

Ans. to Q No8: Software Requirement Specification (SRS) describes what the various stakeholders (like customers, users, project managers, software developers, engineers, invertors, test engineers, etc.) want in terms of features, functions, performance, constraints, quality, security, etc. Software Design Specification (SDS) describes the technical details of how the features and functions have to be implemented. It will contain information about the software architecture and describe how each component will contribute to meeting the requirements. It will help the programmer during coding and implementation.

Ans. to Q No9: A good design specification should have the following characteristics:

- Correctness: Software design should be correct as per requirement.
- Completeness: The design should have all components like data structures, modules, and external interfaces, etc.
- Efficiency: It should allocate resources very efficiently.
- Flexibility: In case of occurrence of changes, it should be modifiable.
- Consistency: There should not be any inconsistency in the design.
- Maintainability: The design should be as simple as possible so that it can be easily maintained by other designers.



13.11 MODEL QUESTIONS

Q1. "When we say quality of a product, different people may visualize it differently." - Elaborate. In software industry, describe the issues related to software quality and software quality assurance.

Q2. Explain in detail McCall's Factor Model.

Q3. Explain in detail the structure of a Software Requirement Specification (SRS).

Q4. Describe the characteristics of a good Software Requirement Specification?

Q5. Describe the Software design principles.

*** ***** ***

UNIT 14: IMPLEMENTATION

UNIT STRUCTURE

14.1 Learning Objectives

14.2 Introduction

14.3 Conversion

- 14.3.1 Conversion Plans
- 14.3.2 Conversion Methods

14.4 Activity Network for Conversion

- 14.4.1 Training
- 14.4.2 File Conversion
- 14.4.3 Forms and Displays Conversion
- 14.4.4 Conversion of Physical Facilities
- 14.4.5 Conversion of Administrative Procedures

14.5 Let Us Sum Up

14.6 Further Readings

14.7 Answers To Check Your Progress

14.8 ModelQuestions

14.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- learn about the conversion process of migrating from old to new system
- understand different conversion plans and methods
- learn about the different activities of the conversion process
- visualize that success of software depends on proper training of all the people who will be using the system

14.2 INTRODUCTION

In the previous unit, we learned about the various aspects, issues and factors of software quality. We also learned about the significance of

Software Requirement Specification and Software Design Specification; and also various methods to formulate good requirements and design specifications. In the unit, we will discuss about how to implement thesoftware that has been developed in the actual environment, what are the processes involved, how to train the users so that they can efficiently handle the system and how to plan for a smooth conversion. In the next unit, we will discuss about project management.

14.3 CONVERSION

Conversion is the part of the implementation plan where the process of transferring the softwarefrom the old system to the new one occurs. If we have a proper conversion plan then it will provide understandable and structured approach to improve the communication between management and project team.The users, project managers, software developers and other engineers meet and review the project plan, the system test documentation and the implementation plan. Once the conversion plan is approved, files are converted, parallel processing between the existing and the new systems is initialized and results of computer runs from the new system are saved. After conversion is completed, parallel processing is discontinued, implementation results are documented and the new system becomes operational.

14.3.1 Conversion Plans

Conversion plan contains description of all the activities that must occur during implementation of the new system. It anticipates possible problems and solutions to deal with them. It includes the following activities:

- Naming all files for conversions.
- Identifying the data requirements to develop new files during conversion.
- Listing all the new documents and procedures that are required.
- Identifying the controls to be used in each activity.
- Identifying the responsibility of person for each activity.

- Verifying conversion schedules.

14.3.2 Conversion Methods

There are four methods of conversion which are: Parallel Conversion, Direct Cutover Conversion, Pilot Approach and Phase-In Method.

Parallel Conversion: In Parallel conversion, both old and new systems are used simultaneously. If in case the new system fails then the old system will provide a fallback, thus providing a greater sense of security.Here the cost may increase as we have to keep on running both the systems.

Direct Cutover Conversion: In Direct cutover conversion, the old system is completely replaced by the new system. Since there is no fallback, it requires meticulous planning as we have to ensure that the new system works at all cost. The advantage is that the benefit and utilities of the new system can be realized immediately.

Pilot Approach: In Pilot approach, phased approach is used that gradually implements the system across all users. Since it is implemented in a phased manner, large risk that comes from implementing the whole system at once can be avoided. Also,installation and training of personals can also be easily implemented. But phase-wise implementation has a disadvantage that it might take very long time to completely install and implement the whole system.

Phase-In Method: This method is similar to Pilot approach in that the system is implemented in one part of the organization and feedback is collected from the users. Based on the feedback, the system can further be improved to give a better user experience and then the implementation moves on to the next part of the organization. The feedback given by the user is used to show that the old system had problems and hence had to be replaced. This will help in smooth transition of the new system to the rest of the organization.



EXERCISE

1. Explain the conversion process.
2. How many conversion methods are there?
3. Can you spot the major difference in each of the conversion methods?



CHECK YOUR PROGRESS

Q1: What is the significance of conversion?

Q2: What are the conversion plans?

Q3: Explain the difference between Parallel conversion and Direct cutover conversion.

14.4 ACTIVITY NETWORK FOR CONVERSION

The activity network for conversion consists of the following sections: Training, File Conversion, Forms and Displays conversion, Conversion of physical facilities and Conversion of Administrative Procedures. Each of theseis discussed below:

14.4.1 Training

Even if a software product is well designed and technically superior, but if the people who are supposed to use it do not know how to use it, then the software may turn out to be a failure. So, the intended people who will actually use the software should know how to use it, what their specific roles will be, and what the system will do or will not do under different conditions.

Training Systems Operators:

As software can have different constraints and limitations, so the systems operators must be trained properly so that they can graciously handle all kinds of outcomes and issues. The operators should be acquainted with all the common types of errors or anomalies that might occur, they should know how to recognize

such problems, and what steps to take when they come. They should be provided with troubleshooting lists of possible problems along with the solutions, and in case of extraordinarily difficult problems, they should be provided with names and telephone numbers of individuals to contact.

User Training:

End user training is a vital part of the implementation phase because they are the one who will be finally using the software. They will be the first one to face any kinds of problems, failures or malfunctions. So, it becomes utmost important to train them, make them aware of not only the easier and user-friendly portions but also of the different risks and pressure points of the systems. They should also be trained to perform basic or common troubleshooting.

Training Methods

Instructor-led training: To carry out instructor-led training, both trainers and trainees have to meet at the same time, but not necessarily at the same place. The training session could be one-on-one or collaborative. It is of two types:

- **Virtual Classroom**: In this training, trainers must meet the trainees at the same time, but they can be located at different places. The primary tools used here are: video conferencing, text based Internet relay chat tools, or virtual reality packages, etc.
- **Normal Classroom**: The trainers must meet the trainees at the same time and at the same place. The primary tools used here are blackboard, overhead projectors, LCD projector, etc.

Self-Paced Training: In this case, there is no requirement for the trainers and the trainees to meet at the same place or at the same time. The trainers will create some learning courses and provide them to the trainees who will study these courses by themselves at their own convenience. It is of two types:

- **Multimedia Training**: In this training, courses are presented in multimedia format and stored on CD-ROM.

- **Web-based Training**: In this training, courses are often presented in hyper media format and developed to support internet and intranet. It is more interactive then multimedia based training as it provides the necessary information only when the user wants it by clicking the appropriate hyperlink.

14.4.2 File Conversion

When the process of transferring the software from the old system to the new systemis carried on, files may be required to be converted one format into another. For example, file in Microsoft Access format may be required to be converted into SQL format. The main concern here is that when the old files are copied to the new system, the files should remain intact and should produce identical files without any loss of information. The files to be transferred are first determined, and then the required data is converted. A file comparison program may be used to verify the accuracy of the file. Sometimes, the vendor's staff or an outside agency is paid to perform this file conversion process.

Test Files creation

To test the accuracy of files being transferred to the new system, we create test files on the old system, which are then copied on to the new system and finally tested. The test file should offer the following:

- Predictable results
- Previously determined output results for checking
- Printed results in seconds
- Simplified error handling routines
- Ability to build from a small number of records.

Systems are prone to errors.So, to control frauds and detect mistakes, an audit trail can be established.

14.4.3 Forms and Displays conversion

During this activity, old forms and displays are withdrawn and new ones are instituted. Various controls are implemented to ensure the system's reliability, integrity and security.

14.4.4 Conversion of Physical Facilities

Along with the conversion of files and different forms, sometimes the physical facilities may also be required to be changed.For example, the new system might require the hardware to be kept under constant monitoring; so cameras, monitors and barriers might have to be installed in the existing physical facility.

14.4.5 Conversion of Administrative Procedures

This includes scheduling, determining job priorities on the system, and implementing personnel policies for managing the system. The administrative personal / supervisor are trained to handle various emergencies and procedures. They are taught how to collect information and present them to the management.



EXERCISE

1. Training users is a trivial part of Implementation process. (True or False)
2. Training of System operators but training of users is not required. (True or False)
3. In instructor-led training, it is necessary that both trainers and trainees should always be present at the same place. (True or False)
4. In _____, there is no requirement for the trainers and the trainees to meet at the same place or at the same time.



CHECK YOUR PROGRESS

Q4: Why is training the users of the system important?

Q5: How do you train Systems operators and end-users?

Q6: Why in virtual classroom, the trainers need not be present at the same place as the trainees?

Q7: Why self-paced training does not require the trainers and the trainees to meet at the same place or at the same time?

Q8: Why sometimes conversion of physical facilities might be required?



14.5 LET US SUM UP

- Conversion is the part of the implementation plan where the process of transferring the software from the old system to the new one occurs.
- If we have a proper conversion plan then it will provide understandable and structured approach to improve the communication between management and project team.
- Conversion plan contains description of all the activities that must occur during implementation of the new system.
- There are four methods of conversion: Parallel Conversion, Direct Cutover Conversion, Pilot Approach and Phase-In Method.
- The activity network for conversion consists of the following sections: Training, File Conversion, Forms and Displays conversion, Conversion of physical facilities and Conversion of Administrative Procedures.
- If proper training is not imparted to the people who are supposed to use it, then the software may become a failure.
- Systems operators and end-users should be trained properly so that they can graciously handle all kinds of outcomes and issues.

- There can be instructor-led as well as self-paced training methods.
- The main concern of file conversion is that when the old files are copied to the new system, the files should remain intact and should produce identical files without any loss of information.
- Along with the conversion of files and different forms, sometimes the physical facilities may also be required to be changed.



14.6 FURTHER READINGS

1. Wixom, B. H., Roth, R. M., (2008), Systems Analysis and Design, Wiley Publishing
2. Kendall, K. E., Kendall J. E., (2019), Systems Analysis and Design, Pearson



14.7 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q No1: Conversion is the portion of the implementation plan where the process of transferring the software from the old system to the new one occurs. If we have a proper conversion plan then it will provide understandable and structured approach to improve the communication between management and project team. The users, project managers, software developers and other engineers meet and review the project plan, the system test documentation and the implementation plan. Once the conversion plan is approved, files are converted, parallel processing between the existing and the new systems is initialized and results of computer runs from the new system are saved. After conversion is completed, parallel processing is discontinued, implementation results are documented and the new system becomes operational.

Ans. to Q No2: Conversion plansinclude the following activities:

- Naming all files for conversions.
- Identifying the data requirements to develop new files during conversion.

- Listing all the new documents and procedures that are required.
- Identifying the controls to be used in each activity.
- Identifying the responsibility of person for each activity.
- Verifying conversion schedules.

Ans. to Q No3: Parallel Conversion: In Parallel conversion, both old and new systems are used simultaneously. If in case the new system fails then the old system will provide a fallback, thus providing a greater sense of security. Here the cost may increase as we have to keep on running both the systems.

Direct Cutover Conversion: In Direct cutover conversion, the old system is completely replaced by the new system. Since there is no fallback, it requires meticulous planning as we have to ensure that the new system works at all cost. The advantage is that the benefit and utilities of the new system can be realized immediately.

Ans. to Q No4: Even if a software product is well designed and technically superior, but if the people who are supposed to use it do not know how to use it, then the software may turn out to be a failure. So, the intended people who will actually use the software should know how to use it, what their specific roles will be, and what the system will do or will not do under different conditions.

Ans. to Q No5: Training Systems Operators: As software can have different constraints and limitations, so the systems operators must be trained properly so that they can graciously handle all kinds of outcomes and issues. The operators should be acquainted with all the common types of errors or anomalies that might occur, they should know how to recognize such problems, and what steps to take when they come. They should be provided with troubleshooting lists of possible problems along with the solutions, and in case of extraordinarily difficult problems they should be provided with names and telephone numbers of individuals to contact.

User Training: End user training is a vital part of the implementation phase because they are the one who will be finally using the software.

They will be the first one to face any kinds of problems, failures or malfunctions. So it becomes utmost important to train them, make them aware of not only the easier and user-friendly portions but also of the different risks and pressure points of the systems. They should also be trained to perform basic or common troubleshooting.

Ans. to Q No6: In virtual classroom, the trainers need not be present at the same place as the trainees because the training can be performed by using tools like video conferencing, text based Internet relay chat tools, or virtual reality packages, etc.

Ans. to Q No7: Self-paced training does not require the trainers and the trainees to meet at the same place or at the same time because here the trainers can create learning courses and provide them to the trainees, who will study the courses by themselves at their own convenience. It is of two types: Multimedia Training and Web-based Training.

Ans. to Q No8: Along with the conversion of files and different forms, sometimes the physical facilities may also be required to be changed. For example, the new system might require the hardware to be kept under constant monitoring; so cameras, monitors and barriers might have to be installed in the existing physical-facility.



14.8 MODELQUESTIONS

Q1. Explain why conversion is important and what are the different conversion methods?

Q2. What is significance of Training in the Implementation process? Who are the people required to train and how you will train them?

Q3. What are the different training methods available? Briefly explain them.

Q4. Explain what is File conversion and Test Files creation?

Q5. Explain in detail the activity network for conversion.

*** ***** ***

UNIT 15: PROJECT SCHEDULING

UNIT STRUCTURE

15.1 Learning Objectives

15.2 Introduction

15.3 Reason for System Failure

15.4 Project Management

15.5 Project Planning tools

15.6 Let Us Sum UP

15.7 Further Readings

15.8 Answer to Check Your Progress

15.9 Model Questions

15.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- understand Project scheduling
- learn about system failure
- explain Project management chart
- develop Project management software

15.2 INTRODUCTION

In the previous unit, we discussed various implementation issues of a system. In this unit, we will discuss about project scheduling.

Project-task scheduling is a noteworthy project planning action. It comprises deciding which activity would be taken up when. A software project manager does the following to schedule a project. It is to identify all the functions required to complete a project. In the beginning, a project manager always defines the project-related activities. Let's say, if you are acting as a project manager for a software development project at KKHSOU, then you should define the project-related activities as follows:

i) Plan schedule management

ii) Defining the workflow diagram, activities, and sequence of activities

iii) Collect and estimate the activity resources
iv) Define the activity duration
v) Develop schedule

The activities of a software related project include the following:

i) Initial survey and data collection
ii) Analysis of the collected data
iii) Develop the initial requirement
iv) Develop software upgrading options
v) Create cost estimates
vi) Write report
vii) Professional review and revision

To schedule the project, a software project manager always does the following:

1. Identifying the functional requirement of the project.
2. Dividing the large modules into smaller ones or activities.
3. Defining the dependency among various activities.
4. Defining the time duration required to complete the activities.
5. Allocating resources to activities.
6. Initiating the Plan for beginning and ending dates for the activities.

15.3 REASON FOR SYSTEM FAILURE

A system failure can occur due to the hardware or software failure of the system. The reasons for the system failure are presented below.

1. **Voltage spike or hike:** The voltage spike is a transitory change in the electric power supply. A small power malfunction can smash up a computer and damage computer data. For example: If the computer screen goes blank during a thunderstorm, it may be due to the reason of voltage spike.
2. **Circuit board fails:** A defective circuit board is sometimes supposed to be the reason of circuit board fails. In this case, the board is reinstalled and the operation will resume.

3. **Failure to load software upgrades:** Due to the failure to upgrade the various bugs or vulnerabilities of a computer system, running the patch can software lead to corruption of data.
4. **Failure to keep virus protection up to date:** The malware protection packages available in the computer system must be maintained and updated because a malware-infected computer is slow in the data processing.
5. **Inadequate cooling:** Computers produce more heat, during more data processing. So adequate cooling is essential for reducing the heating.
6. **Loss of PDU fails:** Due to the Power Distribution Units (PDU's) failure, or power loss, sometimes the system may fail.
7. **UPS failure:** Uninterruptible Power Supply (UPS) batteries are used in the computer system for continuous power for a limited time. Depending on the design of the battery, the life span of a battery may vary. High room temperatures will curtail the life of any battery.
8. **Storage defective:** Data is stored on magnetic disks or tape. Both can be damaged during normal operation, so it's a good practice to back up stored data regularly. Failure of a tape or disk drive is often, but not always, proceeded by noise.



CHECK YOUR PROGRESS

Q1: What do you mean by project scheduling tasks?

Q2: What do you mean by project management?

Q3: What are the functionalities of a project manager to schedule a project?

Q4: What are the reasons for system failure?

Q5: What do you mean by storage defective?

15.4 PROJECT MANAGEMENT

Project Management is the process of assisting the project managers in achieving the definite project goals by keeping in mind the

project scope, time, quality, and cost. It is done by carefully organizing, scheming, and executing the project. Managing a project is not an easy task, no matter what the level and range are. From planning to delivering the product to the client, there is a lot that can go wrong. To overcome this issue, the project management processes are divided into manageable stages with their own goals and deliverables. These stages will help to control the project and the quality of the output. The 5 basic phases in the project management process are discussed in the following:

i) **Initiation:** The project initiation is the first phase of project management. Here, the project goal is defined. The Business cases are defined with the help of the project charter. It is an important document consisting of project constraints, goals, appointment of the project manager, budget, expected timeline, etc. Once you have the project goals and project scope, list down the project stakeholders along with their designation and requirement.

ii) **Planning:** After defining the scope of the project, the project is planned along with the budget baseline. The primary tasks are defined to identify the technical requirement of the project with a detailed schedule. The proper goal and objectives, the communication plan, and the role and responsibilities of the stakeholders are defined in this stage.

iii) **Execution:** Here the project resources are allocated to execute the project. The resources are well maintained to execute the project. Here, the project development team will do the actual work by maintaining collaboration between the stakeholders. Sometimes, the project manager takes the help of project collaboration and management tools to execute the project efficiently.

iv) **Monitoring and Controlling:** The project monitoring and controlling run simultaneously. The following steps are carried in this stage of project management.

   i) Track the effort and cost of the projects.
   ii) Monitor the project progress.
   iii) Prevent any chance of disruptions.

v) **Closing:** This is the final phase of the project management process. It indicates the end of the project after the final delivery. The external resources or talents which are hired for the project are terminated. Sometimes, the project manager organizes the reflection meeting after the completion of the project to contemplate their successes and failures during the project. A detailed report which covers every aspect of the project will be prepared and delivered.

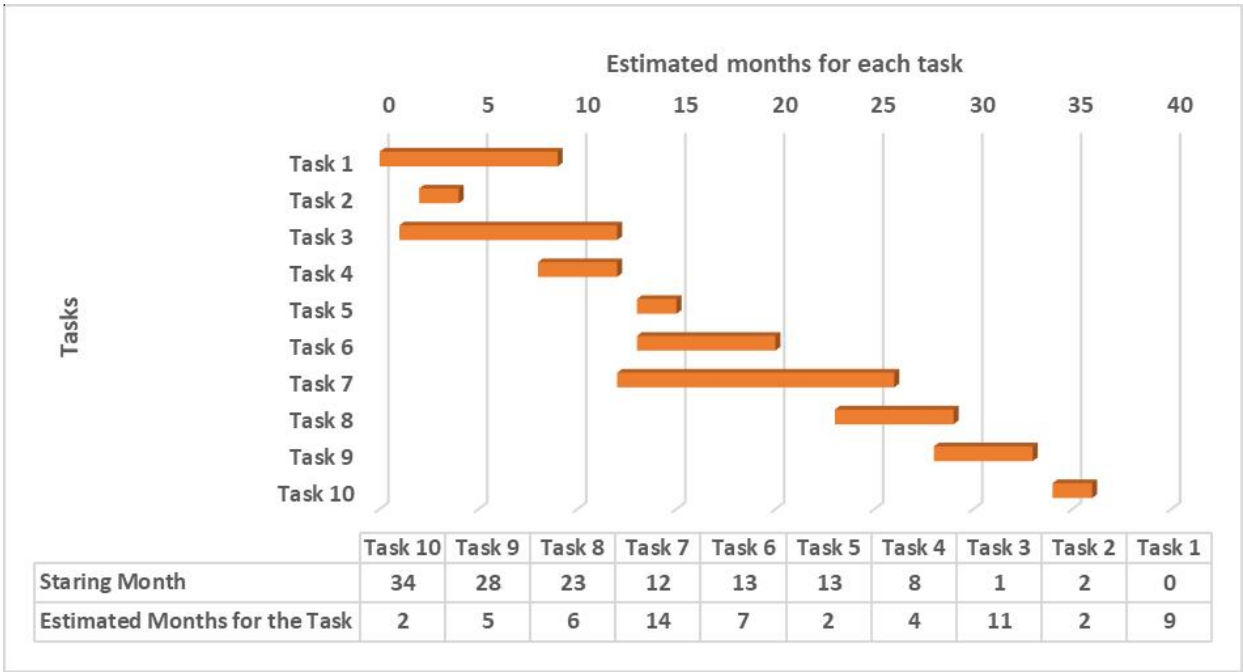
15.5 PROJECT PLANNING TOOLS

The project management tool is very much important to execute a project successfully. To execute a project, few tools are specially designed for project management work. These tools help the project manager to work in a standardized manner. Few tools are demonstrated here only.

i) **Milestone checklist:** This tool is used by the project manager to check the project progress in terms of its correctness. A project manager can create a milestone checklist with the help of a simple excel sheet without having any expensive software.

ii) **Gantt chart:** It is a chart used by the project manager to describe the interrelation between the activity of the project. It is the tool used by the project manager for any project development. It is more complex if the project has more activities. A project Gantt chart may be created in an excel sheet also. Nowadays many online Gantt chart editors are there to create a Gantt chart. Let's take an example with the following tasks for a project and the Gantt chart of the project is defined as follows.

Sl. No.	Work Plan	Staring Month	Ending Month	Estimated months for each Task
Task 1	Detailed background study along with field study	0	9	9
Task 2	Purchase of required equipment and setting up the recording requirement	2	4	2
Task 3	Data Collection and building of corpora	1	12	11
Task 4	Feature extraction process for ML	8	12	4
Task 5	Training and testing phase for ML	13	15	2
Task 6	Implementation DL Model	13	20	7
Task 7	Development of software Tool and digital preservation of traditional method	12	26	14
Task 8	Loading and testing of ML and DL Model into the developed system	23	29	6
Task 9	Implementation and Further improvement	28	33	5
Task 10	Preparation of the report and Plan for Post Project Activities	34	36	2



iii) **Pert Chart:** The pert chart means Program evaluation and review technique. It was developed in 1950 by U.S. Navy. It is presented as a network diagram where the nodes represent the events. The direction of the lines indicates the sequence of the task. In the example as shown in figure 15.1 given below, tasks between "Task 1 to Task 9" must complete, and these are known as dependent or serial tasks. Between Task 4 and 5, and Task 4 and 6, nodes are not dependent and can undertake simultaneously. These are known as Parallel or concurrent tasks. Without resource or completion time, the task must complete in the sequence which is considered as event dependency, and these are known as Dummy activity and represented by dotted lines.

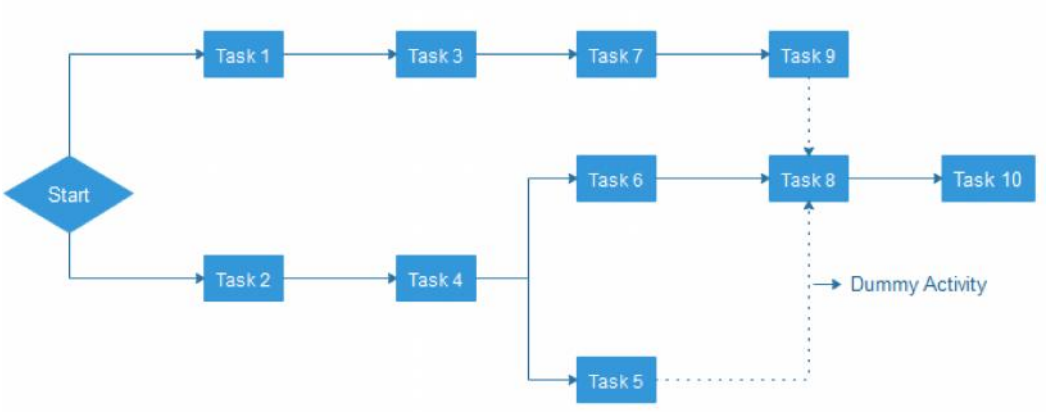


Figure 15.1: Network diagram for 10 tasks



CHECK YOUR PROGRESS

Q6: What are the phases of project execution?

Q7: What is a Gantt chart?

Q8: What do you mean by pert chart?

Q9: What does the direction of a line indicate in the pert chart?



15.6 LET US SUM UP

- Project-task scheduling is a noteworthy project planning action. It comprises deciding which activity would be taken up when.
- The project-related activities are as follows.
  - Plan schedule management
  - Defining the workflow diagram, activities, and sequence of activities
  - Collect and estimate the activity resources
  - Define the activity duration
  - Develop schedule
- To schedule the project, a software project manager always does the following:
  - Identifying the functional requirement of the project.
  - Dividing the large modules into smaller ones or activities.
  - Defining the dependency among various activities.
  - Defining the time duration required to complete the activities.
  - Allocating resources to activities.
  - Initiating the Plan for beginning and ending dates for the activities.
- System failure can occur due to the hardware or software failure of the system. The reasons for the system failure are presented below.
  - Voltage spike or hike
  - Circuit board fails
  - Failure to load software upgrades
  - Failure to keep virus protection up to date
  - Inadequate cooling
  - Loss of PDU fails
  - UPS
  - Storage defective
- The 5 basic phases in the project management process are
  - Initiation
  - Planning

- Execution
- Monitoring and Controlling
- Closing

- Gantt chart is used by the project manager to describe the interrelation between the activity of the project. It is the tool used by the project manager for any project development. It is more complex if the project has more activities.
- Pert Chart means program evaluation and review technique. It was developed in 1950 by U.S. Navy. It is presented as a network diagram where the nodes represent the events. The direction of the lines indicates the sequence of the task.



15.7 FURTHER READINGS

1. Wixom, B. H., Roth, R. M., (2008), Systems Analysis and Design, Wiley Publishing
2. Kendall, K. E., Kendall J. E., (2019), Systems Analysis and Design, Pearson



15.8 ANSWER TO CHECK YOUR PROGRESS

Ans. Q No 1: Project-task scheduling is a noteworthy project planning action. It comprises deciding which activity would be taken up when.

Ans. Q No 2: Project management is the application of processes, methods, skills, knowledge, and experience to achieve specific project objectives according to the project acceptance criteria within agreed parameters.

Ans. Q No 3: To schedule the project, a software project manager always does the following:

a. Identifying the functional requirement of the project.
b. Dividing the large modules into smaller ones or activities.

c. Defining the dependency among various activities.

d. Define the time duration required to complete the activities.

e. Allocate resources to activities.

f. Initiate the Plan for beginning and ending dates for the activities.

Ans. Q No 4: A system failure can occur due to the hardware or software failure of the system. The reasons for the system failure are presented below.

g. Voltage spike or hike

h. Circuit board fails

i. Failure to load software upgrades

j. Failure to keep virus protection up to date

k. Inadequate cooling

l. Loss of PDU fails

m. PS

n. Storage defective

Ans. Q No 5: Data is stored on magnetic disks or tape. Both can be damaged during normal operation, so it's a good practice to back up stored data regularly. Failure of a tape or disk drive is often, but not always, proceeded by noise.

Ans. Q No 6: The 5 basic phases in the project management process are.

- Initiation:
- Planning
- Execution
- Monitoring and Controlling
- Closing

Ans. Q No 7: Gantt chart is a chart used by the project manager to describe the interrelation between the activities of the project. It is the tool used by the project manager for any project development. It is more complex if the project has more activities. A project Gantt chart may be created in an excel sheet also. Nowadays many online Gantt chart editors are there to create a Gantt chart.

Ans. Q No 8: The pert chart means the program evaluation review technique. It was developed in 1950 by U.S. Navy. It is presented as a network diagram where the nodes represent the events. The direction of the lines indicates the sequence of the task.

Ans. Q No 9: The direction of the lines indicates the sequence of the task.



15.9 MODEL QUESTIONS

Q1: What is the role of the project manager and management in project development?

Q2: Does the project schedule identify all pre-construction activities and allocate adequate time for each?

Q3: Are all scheduled tasks of a PERT chart linked by predecessor and successor?

Q4: Is the amount of time allocated for each task in line with historic experience?

Q5: Represent a Gantt chart for software development of the KKHSOU website.

Q6: What is the difference between Gantt and Pert charts?

Q7: Can I modify my schedule in the Gantt chart?

Q8: How do you go about planning a schedule for a project?

Q9: What are the ways a project time schedule can be compressed?

*** ***** ***