

ALAGAPPA UNIVERSITY

[Accredited with 'A+' Grade by NAAC (CGPA:3.64) in the Third Cycle and Graded as Category–I University by MHRD-UGC]

(A State University Established by the Government of Tamil Nadu)

KARAIKUDI – 630 003



Directorate of Distance Education

B.Sc. (Information Technology)

VI - Semester

129 62

SYSTEM ANALYSIS AND DESIGN

Author:

Prof. (Dr.) Girish Kathuria, Director, Amity Global Business School, Amity University, Noida
Units: (1, 2, 4.0-4.2.1, 5.4, 6.0-6.2)

Rohit Khurana, CEO, ITL Education Solutions Ltd., New Delhi
Units: (7.6, 12.5, 14.0-14.2)

Manas Ghosh, Lecturer - RCC Institute of Information Technology, Kolkata
Sudipta Pathak, System Consultant, Cognizant, Kolkata
Unit: (11)

Vikas Publishing House, Units: (3, 4.2.2-4.7, 5.0-5.3, 5.4.1-5.9, 6.3-6.8, 7.0-7.5, 7.7-7.11, 8, 9, 10, 12.0-12.4, 12.6-12.14, 13, 14.3-14.9)

"The copyright shall be vested with Alagappa University"

All rights reserved. No part of this publication which is material protected by this copyright notice may be reproduced or transmitted or utilized or stored in any form or by any means now known or hereinafter invented, electronic, digital or mechanical, including photocopying, scanning, recording or by any information storage or retrieval system, without prior written permission from the Alagappa University, Karaikudi, Tamil Nadu.

Information contained in this book has been published by VIKAS® Publishing House Pvt. Ltd. and has been obtained by its Authors from sources believed to be reliable and are correct to the best of their knowledge. However, the Alagappa University, Publisher and its Authors shall in no event be liable for any errors, omissions or damages arising out of use of this information and specifically disclaim any implied warranties or merchantability or fitness for any particular use.



Vikas® is the registered trademark of Vikas® Publishing House Pvt. Ltd.

VIKAS® PUBLISHING HOUSE PVT. LTD.
E-28, Sector-8, Noida - 201301 (UP)
Phone: 0120-4078900 • Fax: 0120-4078999
Regd. Office: A-27, 2nd Floor, Mohan Co-operative Industrial Estate, New Delhi 1100 44
• Website: www.vikaspublishing.com • Email: helpline@vikaspublishing.com

Work Order No.AU/DDE/DE12-27/Preparation and Printing of Course Materials/2020 Dated 12.08.2020 Copies - 500

SYLLABI-BOOK MAPPING TABLE

System Analysis and Design

Syllabi	Mapping in Book
BLOCK I: SYSTEM CONCEPTS AND SYSTEM DEVELOPMENT LIFE CYCLE **UNIT 1:** System Concepts - Characteristics - Elements of a System - Types of Systems: Abstract, Physical, Open, Closed and Man-made Information System - Computer Based Information Systems: MIS, DSS, TPS and OAS **UNIT 2:** System Development Life Cycle - Problem Definition - Feasibility Study - Analysis - Design - Development - Implementation - Post Implementation and Maintenance **UNIT 3:** System Analyst: Interpersonal Skills - Technical Skill - Communication Skills - Role of Systems Analyst.	**Unit 1:** System Concepts **(Pages 1-19);** **Unit 2:** System Development Life Cycle **(Pages 20-34);** **Unit 3:** System Analyst **(Pages 35-61)**
BLOCK II: SYSTEM ANALYSIS **UNIT 4:** System Analysis: Bases for Planning in System Analysis - Preliminary Investigation - Determining the User's Information Requirements, Case Scenario, Problem Definition and Project Initiation, Background Analysis **UNIT 5:** Fact Finding Techniques: Interview - Questionnaire - Record Review - Observation. Systems Analysis: Analysing Systems Data - Feasibility Study: Technical, Economical and Operational - Steps in Feasibility Analysis, Feasibility Report, Oral Presentation **UNIT 6:** Systems Costs & Benefits: Categories of Cost - Benefits - Cost Benefit Analysis: Break Even, Present Value, Pay Back and Cash Flow. Analysis Tools: Data Flow Concept - Data Flow Diagram - Data Dictionary - Decision Table - Decision Tree - Structured English.	**Unit 4:** System Analysis **(Pages 62-75);** **Unit 5:** Fact Finding Techniques **(Pages 76-99);** **Unit 6:** System Cost and Benefits **(Pages 100-145)**
BLOCK III: SYSTEM DESIGN **UNIT 7:** System Design: Process and Stages of System Design: Logical and Physical Design. Design Methodologies: Structured design - Form Driven Methodology - Major Development Activities **UNIT 8:** Input Output and Form Design: Input Design: Capturing Data for Input - Input Validation - Input Design of On-Line Systems. Output Design - Printed, Display and Audio. **UNIT 9:** Forms Design: Definition - Classification of Forms, Requirements of Forms Design - Types of Forms - Forms Control.	**Unit 7:** System Design: An Introduction **(Pages 146-210);** **Unit 8:** Input Output and Form Design **(Pages 211-228);** **Unit 9:** Forms Design **(Pages 229-246)**
BLOCK IV: FILE AND DATABASE DESIGN **UNIT 10:** File Concepts - Types of Files - Methods of File Organization - Sequential - Direct - Indexed - Database Design: Database Concept **UNIT 11:** Types of Databases: Hierarchical, Network and Relational. **UNIT 12:** System Development: Software Design - Top Down Approach - Flow Chart: System Flow Chart - Program Flow Chart - HIPO - IPO - VTOC - Warnier Orr Diagram - Structured Walkthrough - Quality Assurance - Levels of Assurance - System Testing - Special Systems Tests	**Unit 10:** File Concepts **(Pages 247-264);** **Unit 11:** Types of Databases **(Pages 265-284);** **Unit 12:** System Development **(Pages 285-317)**
BLOCK V: SYSTEM EVALUATION, IMPLEMENTATION AND MAINTENANCE **UNIT 13:** System Evaluation and Implementation Training Personnel - Training Methods - Conversion: Conversion Methods - Parallel, Direct, Pilot and Phase-In. Conversion Plan - Site Preparation - Data and File Preparation - Post Implementation Review **UNIT 14:** System Maintenance: Corrective - Adaptive - Hardware and Software Selection : Computer Industry - Software Industry - Procedure of Hardware and Software Selection: Major Phases in Hardware and Software Selection - Evaluation Process - Financial Considerations.	**Unit 13:** System Evaluation and Implementation **(Pages 318-343);** **Unit 14:** System Maintenance **(Pages 344-396)**

CONTENTS

BLOCK I: SYSTEM CONCEPTS AND SYSTEM DEVELOPMENT LIFE CYCLE

UNIT 1 SYSTEM CONCEPTS 1-19

1.0 Introduction
1.1 Objectives
1.2 System Concepts: An Introduction
1.3 Elements of a System
1.4 Types of Systems
1.5 Computer Based Information Systems
1.5.1 Types of Information Systems
1.6 Answers to Check Your Progress Questions
1.7 Summary
1.8 Key Words
1.9 Self Assessment Questions and Exercises
1.10 Further Readings

UNIT 2 SYSTEM DEVELOPMENT LIFE CYCLE 20-34

2.0 Introduction
2.1 Objectives
2.2 System Development Life Cycle
2.3 Answers to Check Your Progress Questions
2.4 Summary
2.5 Key Words
2.6 Self Assessment Questions and Exercises
2.7 Further Readings

UNIT 3 SYSTEM ANALYST 35-61

3.0 Introduction
3.1 Objectives
3.2 System Analyst
3.2.1 Interpersonal Skills; 3.2.2 Analytical Skills
3.2.3 Technical Skills; 3.2.4 Management Skills
3.3 Role of System Analyst: System Planning and Initial Investigation
3.4 Answers to Check Your Progress Questions
3.5 Summary
3.6 Key Words
3.7 Self Assessment Questions and Exercises
3.8 Further Readings

BLOCK II: SYSTEM ANALYSIS

UNIT 4 SYSTEM ANALYSIS 62-75

4.0 Introduction
4.1 Objectives
4.2 System Analysis
4.2.1 Preliminary Investigation
4.2.2 Determining the User's Information Requirements
4.2.3 Case Scenario; 4.2.4 Problem Definition and Initiation; 4.2.5 Background Analysis

4.3 Answers to Check Your Progress Questions
4.4 Summary
4.5 Key Words
4.6 Self Assessment Questions and Exercises
4.7 Further Readings

UNIT 5 FACT FINDING TECHNIQUES 76-99

5.0 Introduction
5.1 Objectives
5.2 Fact Finding Techniques
5.2.1 Interview; 5.2.2 Questionnaire; 5.2.3 Record Review; 5.2.4 Observation
5.3 Analysing Systems Data
5.4 Feasibility Study
5.4.1 Steps Involved in Feasibility Analysis; 5.4.2 Study Phase Report; 5.4.3 Oral Presentation
5.5 Answers to Check Your Progress Questions
5.6 Summary
5.7 Key Words
5.8 Self Assessment Questions and Exercises
5.9 Further Readings

UNIT 6 SYSTEM COST AND BENEFITS 100-145

6.0 Introduction
6.1 Objectives
6.2 Cost-Benefit Analysis
6.2.1 Principles of Cost-Benefit Analysis
6.2.2 Categories of Costs and Benefits
6.2.3 Cost and Benefit Evaluation Methods
6.3 Tools for Structured Analysis
6.3.1 Types of Basic Charts; 6.3.2 Decision Tables
6.3.3 Decision Trees; 6.3.4 Structured English; 6.3.5 Data Flow Diagrams
6.4 Answers to Check Your Progress Questions
6.5 Summary
6.6 Key Words
6.7 Self Assessment Questions and Exercises
6.8 Further Readings

BLOCK III: SYSTEM DESIGN

UNIT 7 SYSTEM DESIGN: AN INTRODUCTION 146-210

7.0 Introduction
7.1 Objectives
7.2 The Process and Stages of System Design
7.2.1 Logical and Physical Design
7.3 Design Methodologies
7.4 Structured Design
7.5 Form Driven Methodology
7.6 Major Development Activities
7.7 Answers to Check Your Progress Questions
7.8 Summary
7.9 Key Words
7.10 Self Assessment Questions and Exercises
7.11 Further Readings

UNIT 8 INPUT OUTPUT AND FORM DESIGN **211-228**

8.0 Introduction
8.1 Objectives
8.2 Input Design
8.3 Output Design
8.4 Answers to Check Your Progress Questions
8.5 Summary
8.6 Key Words
8.7 Self Assessment Questions and Exercises
8.8 Further Readings

UNIT 9 FORMS DESIGN **229-246**

9.0 Introduction
9.1 Objectives
9.2 Forms Design
9.2.1 Classification of Forms; 9.2.2 Requirements of Form Design
9.2.3 Styles and Types of Form; 9.2.4 Forms Control
9.3 Answers to Check Your Progress Questions
9.4 Summary
9.5 Key Words
9.6 Self Assessment Questions and Exercises
9.7 Further Readings

BLOCK IV: FILE AND DATABASE DESIGN

UNIT 10 FILE CONCEPTS **247-264**

10.0 Introduction
10.1 Objectives
10.2 Introduction to File Organization
10.2.1 File Organization
10.3 Types of Files
10.4 Database Design
10.5 Answers to Check Your Progress Questions
10.6 Summary
10.7 Key Words
10.8 Self Assessment Questions and Exercises
10.9 Further Readings

UNIT 11 TYPES OF DATABASES **265-284**

11.0 Introduction
11.1 Objectives
11.2 Types of Databases
11.3 Hierarchical Model
11.4 Network Model
11.5 Relational Model
11.6 Answers to Check Your Progress Questions
11.7 Summary
11.8 Key Words
11.9 Self Assessment Questions and Exercises
11.10 Further Readings

UNIT 12 SYSTEM DEVELOPMENT **285-317**

12.0 Introduction
12.1 Objectives
12.2 Software Design
12.3 System Flowcharts
12.4 Program FlowChart
12.5 Hipo Charts
12.6 Warnier-Orr Diagram
12.7 Design Methodologies
12.7.1 Structured Design; 12.7.2 Structured Walkthrough
12.8 Overview of System Control And Quality
12.8.1 Levels of Quality Assurance (QA)
12.9 System Testing
12.9.1 Special Systems Tests
12.10 Answers to Check Your Progress Questions
12.11 Summary
12.12 Key Words
12.13 Self Assessment Questions and Exercises
12.14 Further Readings

BLOCK V: SYSTEM EVALUATION, IMPLEMENTATION AND MAINTENANCE

UNIT 13 SYSTEM EVALUATION AND IMPLEMENTATION **318-343**

13.0 Introduction
13.1 Objectives
13.2 Training
13.2.1 Training Strategies; 13.2.2 Training Personnel
13.3 Conversion
13.3.1 Conversion Plan; 13.3.2 Operating Plan
13.4 Post Implementation Review
13.5 Answers to Check Your Progress Questions
13.6 Summary
13.7 Key Words
13.8 Self Assessment Questions and Exercises
13.9 Further Readings

UNIT 14 SYSTEM MAINTENANCE **344-396**

14.0 Introduction
14.1 Objectives
14.2 System Maintenance
14.3 Hardware and Software Selection
14.3.1 Computer Hardware; 14.3.2 Computer Software
14.4 Procedure for Hardware and Software Selection
14.5 Answers to Check Your Progress Questions
14.6 Summary
14.7 Key Words
14.8 Self Assessment Questions and Exercises
14.9 Further Readings

INTRODUCTION

NOTES

Computers have brought about major changes, not only in all spheres of life but also in the way we communicate and work. Today, it is extremely difficult to imagine a world without computers. Since computers are used in every possible field today, it is important for us to understand their functioning in an effective way. In the System Analysis and Design (SAD) process, System Development Life Cycle (SDLC) means a combination of various activities, and deals mainly with the software development life cycle. The key person in the SDLC is the systems analyst who analyses the situation, identifies opportunities for improvements and designs an information system to implement them. The SDLC has a similar set of four fundamental phases known as planning, analysis, design and implementation. Different projects emphasize different parts of the SDLC but all the projects have elements of these four phases. A design methodology is a formalized approach to implementing the SDLC providing a list of steps and deliverables. The entire process of designing the software system is an exercise that specifies the 'What' and 'How' of the system. The modular design of the software consists of a number of files in which creation of file follows a predefined file structure.

This book, *System Analysis And Design*, is divided into five blocks, which are further subdivided into fourteen units. This book provides a basic understanding of the subject and helps to grasp its fundamentals. In a nutshell, it explains various aspects, such as system concepts and system development life cycle, types of systems, computer based information systems (MIS, DSS, TPS and OAS), feasibility study, system analyst, system analysis, determining the user's information requirements, fact finding techniques, systems costs and benefits analysis, analysis tools, data flow concept, data flow diagram, data dictionary, decision table, decision tree, system design, logical and physical design. design methodologies, input output design, forms design, file concepts, methods of file organization, database design, types of databases (hierarchical, network and relational), system top down approach, system flowchart, program flow chart (HIPO, IPO, VTOC), Warnier-Orr diagram, quality assurance, system testing, system evaluation, implementation and maintenance.

The book follows the Self-Instructional Mode (SIM) wherein each unit begins with an 'Introduction' to the topic. The 'Objectives' are then outlined before going on to the presentation of the detailed content in a simple and structured format. 'Check Your Progress' questions are provided at regular intervals to test the student's understanding of the subject. 'Answers to Check Your Progress Questions', a 'Summary', a list of 'Key Words', and a set of 'Self-Assessment Questions and Exercises' are provided at the end of each unit for effective recapitulation. This book provides a good learning platform to the people who need to be skilled in the area of operating system functions. Logically arranged topics, relevant examples and illustrations have been included for better understanding of the topics and for effective recapitulation.

BLOCK - I
SYSTEM CONCEPTS AND SYSTEM DEVELOPMENT LIFE CYCLE

NOTES

UNIT 1 SYSTEM CONCEPTS

Structure

1.0 Introduction
1.1 Objectives
1.2 System Concepts: An Introduction
1.3 Elements of a System
1.4 Types of Systems
1.5 Computer Based Information Systems
    1.5.1 Types of Information Systems
1.6 Answers to Check Your Progress Questions
1.7 Summary
1.8 Key Words
1.9 Self Assessment Questions and Exercises
1.10 Further Readings

1.0 INTRODUCTION

In today's fast changing world, every task and business activity is becoming complex day-by-day with interdependence on many different aspects. Therefore, the concept and understanding of systems analysis has become crucial. Systems analysis is especially important when we talk about informational technology and software projects, as all aspects of our life are directly or indirectly linked with computers and information technology.

The term system is derived from the Greek word '*systema'*, which means an 'organized relationship among functioning units or components'. A system exists because it is designed to achieve one or more objectives. System definition suggests some characteristics that are present in all systems: organization (order), interaction, interdependence, integration and a central objective. In most cases, systems analysts operate in a dynamic environment where change is a way of life. The environment may be a business firm, a business application, or a computer system.

Systems have been classified in different ways, such as physical or abstract information systems, open or closed information systems, and man-made information systems, etc. A system may be schematic, static or dynamic. An information system is an open system that allows inputs and facilitates interaction with the user. The main characteristic of an open system are input from outside, processing, output, and operation in cycles through feedback, differentiation, and

NOTES

equifinality. Physical systems are tangible entities that may be static or dynamic in operation. Another classification of systems is based on their degree of independence. Ideally, the true information reduces uncertainty about a state or event. Systems analysts relies on different types of computer information systems for problem solving. Hence, the systems analyst must be expert of the technology for providing up-to-date and authentic information.

In this unit, you will study about characteristics, elements of a system, types of systems, abstract, physical, open, closed and man-made information system, computer based information systems, MIS (Management Information System), DSS (Decision Support System), TPS (Transaction Process System) and OAS (Office Automation System).

1.1 OBJECTIVES

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

- Understand the concept of systems
- Elaborate on the elements of a system
- Define the types of system
- Explain the different types of information systems
- Discuss about the importance of information system for business

1.2 SYSTEM CONCEPTS: AN INTRODUCTION

Broadly speaking, a system is a collection or arrangement of different components which not only work in conjunction and are interrelated in one way or the other, but also collectively perform a function. There are systems all around us. Our body itself is a collection of so many systems — our digestive system contains so many components or parts which perform their individual functions but when all these parts work together to accomplish the task of digestion it becomes the digestive system. People from different functional areas will perceive the word 'system' differently. For example, software engineers see a system as a collection of programs performing a task; for an electrical engineer, the supply of electricity is a system; for the city's bus service, transportation takes place in a system. The system can be an integrated set of different entities that fulfilling a defined objective.

A metro railway works as a system, that is, the control room, the stations, the trains, the ticketing, automation, the employees, all work together to function as a system. A mobile service provider has components like mobile towers, control rooms, mobile sets and so many other complex components which function together as a system to facilitate communication.

NOTES

Characteristics of a System

As discussed, a system is made up of several components which integrate together to function and fulfil a task. The characteristics of a system can be generalized as follows:

1. **Core objective**: Core objective system has a core objective or purpose which needs to be fulfilled. It is on the core objective that the entire system is built or designed. The objectives of the system are well planned in advance and well stated. For example, the objective of the INSAT satellite system is to help in telecommunication, TV broadcasting and weather forecasting.
2. **Interdependence**: All the components or subsystems of a system are interdependent. The system can only function successfully when all their individual components function properly and provide input or output to each other. The coordination of these components or subsystems makes the system work.
3. **Integration**: The entire system as a whole is weaved together as one for the procurement of desired output. This is because all the subsystems are integrated together.
4. **Interaction**: All the subsystems in a system have a mutual interdependence on each other. Therefore, they need to interact with each other, as the change or process in one will affect the other subsystem also.

   For example, in a business system, the marketing department creates a demand for a product and accordingly the production department fulfills that demand.
5. **System Organization**: The system is laid out according to the manner in which the work flows. It also facilitates communication flow and command chain conduciveness. The attainment of objectives in an efficient manner is ensured.

   Some examples of systems are as follows:

   - CBSE Examination System
   - Railway Reservation System
   - Traffic Control System
   - Electricity Supply System
   - Judiciary System

What is a Business System?

Like so many other systems, business is also a system. The components of business are production, marketing, sales, inventory, shipping, distribution, research, accounting and human resource functions. All these function as a system so as to provide products and services to the consumer, profit to the owners and share

NOTES

holders and salaries to the employees. If you look closely, the business system is itself a collection of so many different subsystems, as production itself is a complex system and may contain many systems in it.

The common factor in all types of systems is that all systems have inputs. They go through a process (system) and they give an output. You are using the required resources and there are also constraints to overcome.

1.3 ELEMENTS OF A SYSTEM

The following elements are essentially required to build a system:

(1) Inputs

(2) Processor

(3) Control

(4) Environment

(5) Output

(6) Feedback

(7) Boundaries/ Interface

Inputs

Any system, in order to operate and function, needs inputs. Inputs are what a system processes for outputs.

Inputs should have the following characteristics:

(i) **Accuracy:** The data to be input should be accurate as erroneous data will result in wrong output, i.e., GIGO (Garbage In Garbage Out)

(ii) **Timeliness:** The data should be provided to the system at the required time, as only timely data input will only give the desired output.

(iii) **Relevance:** The data should be relevant as irrelevant data would not provide the desired information.

(iv) **Reliability:** The reliability of the data needs to be ensured as wrong input will result in wrong output.

(v) **Cost Effectiveness:** The input data procurement should be cost effective and at the lowest possible price.

(vi) **Quantity:** The quantity of the input data has to be sufficient so as to solve the purpose of the system.

NOTES

Processor

Processor is that part of the system which processes or manipulates the data or input into output. The processing is done through various programs/ software wherein data is transformed or modified from input to output as per the desired output.

Control

For any system to operate efficiently and effectively, control is required in a system to control all inputs, processing and outputs and other activities.

Environment

The system has to operate within an environment, all the components which affect the performance of the system constitute the environment of the system. All the external compounds that effect the system and guide how a system should function are referred to as environment of the system. The system should to able to customize as per the changing environment.

Output

Output is the element for which the entire system is built; Output is what the user desires it is the end result. The output may be a product, a service or information. The most valuable thing for the user is output.

Feedback

When the output is obtained, it is compared with the expectation of the desired result, and the information which is received is called feedback. Feedback is essential for the betterment of the system and for the accuracy of the output. The actual result is compared with the standard result and the information received is termed as feedback. Feedback may be positive or negative. Positive feedback means the system is functioning right and negative feedback initiates action for improving the system.

Boundaries/Interface

Every system has certain limitations and it has to work under those limitations or defined boundaries. The limitations of the system help the interfacing of the system with another system, if there is a requirement of integration of two or more systems.

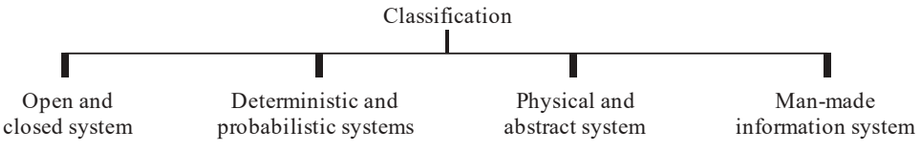
Check Your Progress

1. What is a system?
2. Define the term core objective for a system.
3. Explain the term integration.
4. Elaborate on the term business system.
5. Define the essential elements of a system.

NOTES

1.4 TYPES OF SYSTEMS

Systems can be classified into the following:



(a) Open and Closed Systems

Open systems are those systems which can interact with the outer environment. The main characteristics of open systems are as follows:

(i) They take or give inputs or outputs to the external environment

(ii) They can change as per the change in environment

(iii) They can function effectively

Closed systems, on the other hand, do not interact with the external environment.

The main characteristics of closed systems are as follows:

(i) Usually, they only exist as a concept.

(ii) They are very rare and not operatable.

(iii) They have a very short survival time.

(iv) They do not take/give input/output from the external environment.

(b) Physical and Abstract Systems

Physical Systems: Physical Systems are the systems which you can physically see. You can see its components, parts, etc., which can also be touched.

The main characteristics of physical systems are as follows:

- They are measurable entities which may be static or dynamic.
- They can be seen and felt.

Example: Refrigeration systems, which maintain a particular temperature and also adjust automatically with the external environment.

Abstract Systems

Abstract systems do not exist physically and are only conceptual in nature.

The main characteristics of abstract systems are as follows:

- These are used to understand physical concepts.
- Conceptual abstract models are used to understand physical systems and their interrelationships, etc.
- These are usually theories, principles and concepts,

For example: Einstein's theory of relativity.

(c) Deterministic and Probabilistic Systems

NOTES

Deterministic System

The systems whose outcomes are certain and are based on a predetermined set of rules are called deterministic systems.

The main characteristics of deterministic systems are as follows:

- Their outputs are predictable
- Their behaviour is totally known
- Known outputs for known inputs.
- Interrelationships and interaction with other systems is certain.

Probabilistic Systems

The systems that have uncertain outcomes not based on predetermined sets of rules are called probabilistic systems.

The main characteristics of probabilistic systems are as follows:

- Their outputs are unpredictable.
- Their interrelationship with other systems is uncertain.
- They can only provide probable estimates.
- These systems are controlled by chance events.

(d) Man-Made Information System

Man-made information system is an interconnected set of information resources to manage data for particular organization under Direct Management Control (DMC). This system includes hardware, software, communication, data, and application for producing information according to the need of an organization.

Man-made information systems are divided into three types –

- **Formal Information System:** It is based on the flow of information in the form of memos, instructions, etc., from top level to lower levels of management.
- **Informal Information System:** This is employee based system which solves the day to day work related problems.
- **Computer Based System:** Computer based system system is directly dependent on the computer for managing business applications. For example, automatic library system, railway reservation system, banking system, etc.

Ideally, information reduces uncertainty about a state or event. For example, information that the wind is calm reduces the uncertainty that the boat trip will be pleasant. An information system is the basis for interaction between the user and the analyst. It provides instruction, commands and feedback. It determines the nature of the relationships among decision-makers. In fact, it may be viewed as a

NOTES

decision centre for personnel at all levels. From this basis, an information system may be defined as a set of devices, procedures and operating systems designed around user based criteria to produce information and communicate it to the user for planning, control and performance. In systems analysis, it is important to keep in mind that considering an alternative system means improving one or more of these criteria.

Many practitioners fail to recognize that a business has several information systems; each is designed for a purpose and works to accommodate data flow, communications, decision making, control and effectiveness.

1.5 COMPUTER BASED INFORMATION SYSTEMS

According to Whitten and Bentlay, 'An information system is an arrangement of people, data, processes, interfaces and geography that are integrated for the purposes of supporting and improving the day to day operations in a business as well fulfilling the problem solving and decision making information needs of business managers.'

An information system can be computerized or manual.

The major information systems are as follows:

Formal Information System

The organization chart of organizations depicts the formal structure of positions, the authority and its relationships. The formal information system is entirely based on the organization and it shows the pattern of flow of communication, functions and authority.

Informal Information System

The informal information system is a need based information system and is build on the basis of work related problems to accomplish vocational and personnel needs. The information in this information system is transmitted through indirect and unconventional channels. The informal information system provides information which is not possibly available through formal IS (Information System) and it proves to be very useful in business environment.

Computer Based Information System

As the name suggests, computer based information system are based on computers and the data is stored in computers and processing is done on computers to obtain information.

The Computers Based Information Systems (CBIS) are categorized on the basis of type of information processing and usage by the different levels of management hierarchy.

They are classified as under:

- Invoicing
- Accounts receivable and accounts payable system
- Purchase order processing
- Sales order processing
- Payroll preparation
- Inventory control system

NOTES

Importance of Information Systems for Business

All the business have systems according to which they operate. Broadly speaking you can categorize the business functions into.

Business Operations

The business operations are supported by the, OAS, TPS and Communication Systems.

Managerial Decision Making

The Managerial Decision Making is supported by MIS, DSS, and Expert Systems.

Strategy/Business Policy Making

Strategy/Business Policy Making is being supported by executive support systems.

The information systems so designed helps the business at all the above levels.

1.5.1 Types of Information Systems

Information system has been categorized into several types based on their processes and functions. The most common information systems are:

Transaction Processing System (TPS)

The most common examples of Transaction Processing Systems (TPS) are payroll and inventory. The TPS information system deals with a large amount of data and processes it in a routine business activity. A transaction processing system will involve:

(1) Data capture

(2) Date verification

(3) Date acceptance/rejection

(4) Data validation

(5) Date storage

(6) Reports

The primary objective of TPS is to increase the speed and efficiency of transaction processing in a business activity. The most common example is banking which involves transactions such as deposits, withdrawals and interests on balances.

NOTES

Office Automation System (OAS)

Office Automation Systems (OAS) are systems which help in automating the daily office activities. The most common functions which are covered under this aspect are word processing, spreadsheets, communication through e-mail, voicemail, teleconferencing, videoconferencing, desktop publishing, e-scheduling, record arrangements, etc.

Today's offices cannot be imagined without office automation systems. The basic activities of an OAS are:

(1) Data entry and storage

(2) Data processing (converting into information)

(3) Data exchange

(4) Data management

All these activities are fulfilled using certain hardware and software.

Examples of OAS

- Mailing
- Filing and retailing of documents
- Fixing appointments and scheduling
- Corresponding, involving word processing
- Making worksheets using spread sheets

Management Information System (MIS)

As the name suggests, Management Information System or MIS systems are for the management to analyse the information and take decisions. In MIS, data is taken as an input from transaction processing systems or other information systems and is processed (converted) into meaningful information. A good MIS system is designed keeping in mind the information that a manager needs for decision making and how the manager uses that information for his/her job. Data for developing the MIS system is collected from various transaction processing systems like order processing, purchasing system, inventory processing, etc. The data is collected from various subject areas for development of a comprehensive and accurate model for an efficient MIS system.

Many organizational tasks are supported by the MIS system including decision making and decision analysis. MIS systems also help compile the information from various information functions and the output can be used for effective decision making.

Decision Support System (DSS)

A DSS or Decision Support System is a computer-based information system that helps a user to take decisions in semi-structured situations.

NOTES

Keen and Scott-Martin define DSS as 'Decision Support Systems couple the intellectual resources of individuals with the capabilities of the computer to improve the quality of decisions. It is a computer-based support system for management decision makers who deal with semi-structured problems.'

The main functions of DSS systems are:

(1) Providing solutions and assisting the users in semi-structure problems.

(2) Short listing and identifying actions and plans for problems.

(3) Ranking all the alternatives to be identified accordingly for the solutions.

Expert Systems (ES)

Expert systems are special information systems which are designed to manipulate knowledge rather than information. Expert systems are developed by utilizing the knowledge and expertise of a specialist (expert) in a particular domain.

The user interacts with the expert system through an interactive dialogue system. The queries are put by the expert system to the user; the user provides the answers to the expert system and the expert system uses the responses to provide possible recommendations to the users. There are certain rules and constraints which also apply. Software engineers, called knowledge engineers here, acquire the knowledge of the experts in the field and design the system. Artificial intelligence technology is utilized for creating expert systems which simulate as an expert. The ES uses the same reasoning and logic as an expert in that particular domain. Therefore, you can say that the expert system is like an electronic expert that gives recommendations and advice as a domain expert will provide in the given circumstances. For example, the MYCIN Expert System is used for diagnosing bacterial infections.

Check Your Progress

6. Explain the main characteristics of open system.
7. What is man-made information system?
8. Give the definition of information systems according to Whitten and Bentlay.
9. What is computer based information systems?
10. Explain the term management information system.
11. What is Decision Support System (DSS)? Give the Keen and Scott-Martin definition for DSS.
12. Define the concept of expert system.

1.6 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

NOTES

1. A system is a collection or arrangement of different components which not only work in conjunction and are interrelated in one way or the other, but also collectively perform a function.
2. Core objective system has a core objective or purpose which needs to be fulfilled. It is on the core objective that the entire system is built or designed. The objectives of the system are well planned in advance and well stated.
3. The entire system as a whole is weaved together as one for the procurement of desired output. This is because all the subsystems are integrated together.
4. Like so many other systems, business is also a system. The components of business are production, marketing, sales, inventory, shipping, distribution, research, accounting and human resource functions. All these function as a system so as to provide products and services to the consumer, profit to the owners and share holders and salaries to the employees. If you look closely, the business system is itself a collection of so many different subsystems, as production itself is a complex system and may contain many systems in it.
5. The following elements are essentially required to build a system:
   (1) Inputs
   (2) Processor
   (3) Control
   (4) Environment
   (5) Output
   (6) Feedback
   (7) Boundaries/ Interface
6. The main characteristics of open systems are as follows:
   (i) They take or give inputs or outputs to the external environment
   (ii) They can change as per the change in environment
   (iii) They can function effectively
7. Man-made information system is an interconnected set of information resources to manage data for particular organization under Direct Management Control (DMC). This system includes hardware, software, communication, data, and application for producing information according to the need of an organization.
8. According to Whitten and Bentlay, 'An information system is an arrangement of people, data, processes, interfaces and geography that are integrated for

the purposes of supporting and improving the day to day operations in a business as well fulfilling the problem solving and decision making information needs of business managers.'

An information system can be computerized or manual.

9. As the name suggests, computer based information system are based on computers and the data is stored in computers and processing is done on computers to obtain information.

   The Computers Based Information Systems (CBIS) are categorized on the basis of type of information processing and usage by the different levels of management hierarchy.

10. Management Information System or MIS systems are for the management to analyse the information and take decisions. In MIS, data is taken as an input from transaction processing systems or other information systems and is processed (converted) into meaningful information. A good MIS system is designed keeping in mind the information that a manager needs for decision making and how the manager uses that information for his/her job. Data for developing the MIS system is collected from various transaction processing systems like order processing, purchasing system, inventory processing, etc. The data is collected from various subject areas for development of a comprehensive and accurate model for an efficient MIS system.

11. A DSS or Decision Support System is a computer-based information system that helps a user to take decisions in semi-structured situations.

    Keen and Scott-Martin define DSS as 'Decision Support Systems couple the intellectual resources of individuals with the capabilities of the computer to improve the quality of decisions. It is a computer-based support system for management decision makers who deal with semi-structured problems.'

12. Expert systems are special information systems which are designed to manipulate knowledge rather than information. Expert systems are developed by utilizing the knowledge and expertise of a specialist (expert) in a particular domain.

    The user interacts with the expert system through an interactive dialogue system. The queries are put by the expert system to the user; the user provides the answers to the expert system and the expert system uses the responses to provide possible recommendations to the users.

1.7 SUMMARY

- A system is a collection or arrangement of different components which not only work in conjunction and are interrelated in one way or the other, but also collectively perform a function.

NOTES

- Our digestive system contains so many components or parts which perform their individual functions but when all these parts work together to accomplish the task of digestion it becomes the digestive system.

  A system is made up of several components which integrate together to function and fulfil a task.
- Core objective system has a core objective or purpose which needs to be fulfilled. It is on the core objective that the entire system is built or designed. The objectives of the system are well planned in advance and well stated.
- All the components or subsystems of a system are interdependent.
- The system can only function successfully when all their individual components function properly and provide input or output to each other.
- The entire system as a whole is weaved together as one for the procurement of desired output. This is because all the subsystems are integrated together.
- All the subsystems in a system have a mutual interdependence on each other.
- The system is laid out according to the manner in which the work flows. It also facilitates communication flow and command chain conduciveness.
- Like so many other systems, business is also a system. The components of business are production, marketing, sales, inventory, shipping, distribution, research, accounting and human resource functions. All these function as a system so as to provide products and services to the consumer, profit to the owners and share holders and salaries to the employees. If you look closely, the business system is itself a collection of so many different subsystems, as production itself is a complex system and may contain many systems in it.
- The common factor in all types of systems is that all systems have inputs.
- Any system, in order to operate and function, needs inputs.
- The data to be input should be accurate as erroneous data will result in wrong output, i.e., GIGO (Garbage In Garbage Out)
- The data should be provided to the system at the required time, as only timely data input will only give the desired output.
- The data should be relevant as irrelevant data would not provide the desired information.
- The reliability of the data needs to be ensured as wrong input will result in wrong output.
- The input data procurement should be cost effective and at the lowest possible price.

NOTES

- The quantity of the input data has to be sufficient so as to solve the purpose of the system.
- Processor is that part of the system which processes or manipulates the data or input into output. The processing is done through various programs/ software wherein data is transformed or modified from input to output as per the desired output.
- For any system to operate efficiently and effectively, control is required in a system to control all inputs, processing and outputs and other activities.
- The system has to operate within an environment, all the components which affect the performance of the system constitute the environment of the system.
- All the external compounds that effect the system and guide how a system should function are referred to as environment of the system. The system should to able to customize as per the changing environment.
- Output is the element for which the entire system is built; Output is what the user desires it is the end result.
- The output may be a product, a service or information. The most valuable thing for the user is output.
- When the output is obtained, it is compared with the expectation of the desired result, and the information which is received is called feedback.
- Feedback is essential for the betterment of the system and for the accuracy of the output.
- Feedback may be positive or negative.
- Positive feedback means the system is functioning right and negative feedback initiates action for improving the system.
- Every system has certain limitations and it has to work under those limitations or defined boundaries.
- The limitations of the system help the interfacing of the system with another system, if there is a requirement of integration of two or more systems.
- Open systems are those systems which can interact with the outer environment.
- Closed systems, on the other hand, do not interact with the external environment.
- Physical Systems are the systems which you can physically see. You can see its components, parts, etc., which can also be touched.
- Abstract systems do not exist physically and are only conceptual in nature.
- The systems whose outcomes are certain and are based on a predetermined set of rules are called deterministic systems.

NOTES

- The systems that have uncertain outcomes not based on predetermined sets of rules are called probabilistic systems.
- This system includes hardware, software, communication, data, and application for producing information according to the need of an organization.
- It is based on the flow of information in the form of memos, instructions, etc., from top level to lower levels of management.
- Computer based system system is directly dependent on the computer for managing business applications.
- An information system is the basis for interaction between the user and the analyst. It provides instruction, commands and feedback.
- Many practitioners fail to recognize that a business has several information systems; each is designed for a purpose and works to accommodate data flow, communications, decision making, control and effectiveness.
- According to Whitten and Bentlay, 'An information system is an arrangement of people, data, processes, interfaces and geography that are integrated for the purposes of supporting and improving the day to day operations in a business as well fulfilling the problem solving and decision making information needs of business managers.'

  An information system can be computerized or manual.
- The informal information system is a need based information system and is build on the basis of work related problems to accomplish vocational and personnel needs.
- All the business have systems according to which they operate.
- The business operations are supported by the, OAS, TPS and Communication Systems.
- The Managerial Decision Making is supported by MIS, DSS, and Expert Systems.
- Strategy/Business Policy Making is being supported by executive support systems.
- Information system has been categorized into several types based on their processes and functions.
- The most common examples of Transaction Processing Systems (TPS) are payroll and inventory.
- The TPS information system deals with a large amount of data and processes it in a routine business activity.
- The primary objective of TPS is to increase the speed and efficiency of transaction processing in a business activity.

NOTES

- Office Automation Systems (OAS) are systems which help in automating the daily office activities. The most common functions which are covered under this aspect are word processing, spreadsheets, communication through e-mail, voicemail, teleconferencing, videoconferencing, desktop publishing, e-scheduling, record arrangements, etc.
- Management Information System (MIS) systems are for the management to analyse the information and take decisions. In MIS, data is taken as an input from transaction processing systems or other information systems and is processed (converted) into meaningful information. A good MIS system is designed keeping in mind the information that a manager needs for decision making and how the manager uses that information for his/her job. Data for developing the MIS system is collected from various transaction processing systems like order processing, purchasing system, inventory processing, etc. The data is collected from various subject areas for development of a comprehensive and accurate model for an efficient MIS system.
- Many organizational tasks are supported by the MIS system including decision making and decision analysis. MIS systems also help compile the information from various information functions and the output can be used for effective decision making.
- A DSS or Decision Support System is a computer-based information system that helps a user to take decisions in semi-structured situations.
- Expert systems are special information systems which are designed to manipulate knowledge rather than information.

1.8 KEY WORDS

- **Information system**: It is a system where there is an interaction between the users (people), subsystems (processes) and technology.
- **Integration**: The entire system as a whole is weaved together as one for the procurement of desired output.
- **Inputs**: In order to operate and function, needs input. Inputs are what a system processes for output.
- **Output**: Output is what the user desired it is the end result.
- **Open system:** Open system are those systems which can interact with the outer environment.
- **Physical system:** These are systems which you can physically see.
- **Man-made information system:** Man-made information system is an interconnected set of information resources to manage data for particular organization, under Direct Management Control (DMC).

NOTES

- **Information system** : An information system is an arrangement of people, data, processes, interfaces and geography that are integrated for the purposes of supporting and improving the day to day operations in a business as well fulfilling the problem solving and decision making information needs of business managers.
- **Computer based information system:** Computer based information system are based on computers and the data is stored in computers and processing is done on computers to obtain information.

1.9 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. Define the term system concept.
2. Explain the term interdependence.
3. Elaborate on system organization giving an example.
4. Explain the elements of a system.
5. Differentiate between physical and abstract systems.
6. State about deterministic and probabilistic system.
7. Differentiate between formal and informal system.
8. State about the man-made information system.
9. Define the term business operations.
10. Explain the term transaction processing system.
11. What is office automation system?

Long Answer Question

1. Discuss the concept of system and its characteristics giving appropriate examples.
2. Briefly explain about the various elements of a system with the help of examples.
3. Give a detailed analysis on the types of systems.
4. Explain briefly about the computer-based information systems giving definition of each type.
5. Discuss about the various types of information systems giving appropriate examples.
6. Explain the characteristics features of the following information systems:
   (i) Business System

(ii) Deterministic System

(iii) Man-Made Information Problem

(iv) Managerial Decision Making

(v) Transaction Processing System

(vi) Office Automation System

(vii) Management Information System

(viii) Decision Support System

(ix) Expert Systems

NOTES

1.10 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

NOTES

UNIT 2 SYSTEM DEVELOPMENT LIFE CYCLE

Structure

2.0 Introduction
2.1 Objectives
2.2 System Development Life Cycle
2.3 Answers to Check Your Progress Questions
2.4 Summary
2.5 Key Words
2.6 Self Assessment Questions and Exercises
2.7 Further Readings

2.0 INTRODUCTION

The system analyst gives a system development project meaning and direction. A candidate system is approached after the analyst has a thorough understanding of user needs and problems. A possible solution is worked out and then communicates the same. Candidate systems often cut across the boundaries of users in the organization.

The system development life cycle method is typically thought of as the set of activities that analysts, designers and users carry out to develop and implement an information system. The various stages in the business are closely related to each other, even the order of the steps in these activities is hard to determine.

System analysis and design are keyed to the System Development Life Cycle (SDLC). The stages are project selection, feasibility, analysis, Design, implementation, and post implementation stages. The idea for the project is originates in the environment or from within the organization. Once the problem is verified an initial investigation is conducted to determine whether change is feasible. If the answer is yes, a feasibility study is conducted. Analysis is a detailed study of the various operation performed by a system. System design refer to the technical specifications that will be applied in implementing the candidate system. Implementation is concerned with details of the candidate system. After implementation, maintenance begins includes enhancements, modifications, or any changes from the original specifications. To ensure the success of the system, careful and often extensive planning is required. The overall management process is crucial to the successful completion of system.

In this unit, you will study about problem definition, feasibility study, analysis, design, development, implementation, post implementation and maintenance.

2.1 OBJECTIVES

NOTES

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

- Describe the phases in the System Development Life Cycle (SDLC)
- Define the preliminary investigation
- Elaborate on the various aspects of feasibility study
- Discuss about the system design and development
- Know about the different testing techniques
- Elaborate on the implementation, post implementation and maintenance of the system

2.2 SYSTEM DEVELOPMENT LIFE CYCLE

System Development Life Cycle (SDLC) is a systematic and sequential approach to solving and developing a system. In other words it refers to the various phases which occur in a sequence to develop a system. The SDLC gives you a process or suggests steps to convert the goals/subjects into requirements and the requirements into design. Further that design can be developed into a system and then the system can be tested and maintained.

A system is needed because of the following reasons:

(i) To make the operations more efficient (i.e., to make it more productive and economical.

(ii) To cope up with and adapt to new technologies.

(iii) To overcome flaws in the present business functions.

(iv) To cope up with the changes in the industry/business.

(v) To cope up with the increasing volume of business (i.e., more data volume).

Phases of the System Development Life Cycle

The phases of SDLC can be categorized as under:

(i) Initial or preliminary investigation

(ii) Feasibility study

(iii) System requirement determination

(iv) System design

(v) Software development

(vi) System/software testing

(vii) System implementation

(viii) Post-implementation and maintence

NOTES

Phase I: Preliminary Investigation

The first phase of the system development life cycle is preliminary investigation. As the name suggests, it is the first and initial investigation, which deals with finding out 'what' the problem is, recognizing that the problem is the most important and foremost part of SDLC. To find out the correct problem or flaw for a new system or an existing system is a very important aspect. Infact, it forms the basis of the entire SDLC. The initial investigation may involve the following:

(i) Computerizing the process which is currently manual in nature.

(ii) Improving upon or enhancing the existing computer system.

The preliminary investigation process will involve the following important aspects:

(i) Studying the current system thoroughly in detail.

(ii) Finding the problems/flaws in the current system.

(iii) Conducting a complete investigation of the problems faced in the current system.

(iv) Investigating and analyse the requirement of the new system.

(v) Finding out the goals and objectives of the new system.

(vi) Checking and seeing if the requirements match with the objectives of the system.

(vii) Checking if any constraints or limitations are there to develop the system.

(viii) Finding if the new requirements are necessary for the system and will improve the business organization /system.

The above process requires a lot of deliberation, time, personal skills and interaction of the systems analyst with the users, to arrive at the right conclusion.

The system or process you want to develop may be as small as developing a work scheduler on your personal computer or as complex as putting a satellite on the moon. You may develop a Web portal, or build a computerized payroll system or inventory control system or computerize the entire business organization. The requirement analysis phase plays an important role in the SDLC.

Phase II: Feasibility Study

Based on the conclusions drawn in the preliminary investigation, a more detailed feasibility study is carried out. The feasibility study encompasses judging whether the system to be developed would be feasible, i.e. whether it will prove to be beneficial to the organization, whether it will be workable in various aspects, whether it will be acceptable, whether it will be feasible in various aspects such as:

(i) Economic Feasibility/cost benefit analysis

(ii) Technical feasibility

(iii) Human/ behavioral feasibility

NOTES

Besides these above major aspects, feasibility in terms of time, operations, maintainability, legality, etc. is also studied to completely see the practicality of the project. Let us now look at each aspect in a little more detail.

(a) Economic Feasibility

It deals with the cost benefit analysis of the proposed system. Economic feasibility determines the total cost of development of the system, the operating cost, the cost of the software and hardware, the maintenance cost, etc. If the benefits of the proposed system outweigh the cost of developing and maintaining the system, then a go-ahead is given to develop the new system.

(b) Technical Feasibility

Technical feasibility involves the technology aspect of the feasibility study. It will analyse the current technologies as well the emerging technologies. The study will also involve the software, hardware and upgrade requirements. The study will attempt to answer questions like the following:

- Is the current software and hardware suitable?
- What new software and hardware will be required?
- What will be the capability of software and hardware?
- How long will the system work without upgradation?
- Is the system upgradable?
- What type of inputs would be required?
- How will the aspects of consistency, security, user-friendliness, etc., be handled?

(c) Human Feasibility

Human feasibility deals with the people of the organization. It studies the acceptability level of the employees of the organization. It is human tendency to resist change. So, for a new system to be implemented, it is much better if you study in advance, how the people in the organization are going to react and what would be the strength of their reaction so as to assess the situation in advance. According to the feasibility report, the pace of implementation has to be adjusted for the new system.

Phase III: System Requirement Determination

The systems analyst/business analyst identifies the requirement of the system and finds and understands the complete details of the system. This includes the functional and operational processes of existing systems, the flaws of the existing system as well as the expectations from the new system.

The purpose of the system requirement determination analysis is to gather all data and information to understand user requirements.

NOTES

This phase results in a Software Requirement Specification (SRS) document. SRS is an agreement between the analyst and the client, which mentions the requirement from the new system including all performance and operational requirements.

The SRS will also describe the inputs and outputs of the system. The better the SRS, the better the software will meet its expectations. The SRS document helps the system developers and software engineers to design and develop the system with the least possibility of errors.

A good SRS will have the following features:

- **Completeness:** The SRS document should contain all the system requirements
- **Understandability:** The requirements should be completely understood by the clients, user's analysts and developers.
- **Consistency:** The requirements should not be contradicting in nature.
- **Correctness:** The requirements mentioned are the SRS should be same as the user desires.
- **Clarity:** The requirements mentioned in SRS should mean the same to all.
- **Modification capability:** The SRS document should be made in such a manner that if there in change is the user requirement, it would allow the changes without affecting the entire system.
- **Traceability:** Each requirement in the SRS should be able to match with the ongoing software development. It helps in verification and validation.
- **Verifiability:** The requirements of the SRS should be verifiable at all stages of the development of the software(structure of SRS).

Phase IV: System Design

System design is the most important phase before system development. It is the blue print of the requirements that will help in developing the software. All the technical specifications are finalized for the development of the system at this stage. The design phase mentions the type of data to be input, the processes used, the output format, etc.

The major steps involved in system design are:

(i) **Output Design:** Output design describes the format of the output as per the requirement.

(ii) **Input Design:** It describes the type of data requirements for the output design format.

(iii) **File Design:** Files are made up of data. The file design mentions the process of file creation, data storage in files and data retrieval.

NOTES

(iv) **Processing Design:** This step describes how the processing of data takes place. It describes how the processing will take place; the programs to be used, and the order of the programs to be executed. The operations to be carried out on the computer and the operations to be carried out manually are also mentioned.

(v) **Control Design:** This step deals with the aspect of quality of development. It keeps a check on the correctness of the input data, processing correctness, output accuracy and project timeliness.

Phase V: Software Development

At this stage, the following activities take place:

- The design gets converted into the actual system. The blue prints are translated into software, (i.e. programs).
- The entire project according to the size is broken into modules and each module is assigned to separate teams as per the expertise of programmers in the respective teams.
- The modules developed are made independent so that they can be executed and tested individually.
- The design specifications are converted into codes (programs) by the programmers.
- As per the design requirement, it is decided to develop new programs or modify the existing programs.
- The procurement of new software or hardware is also assessed and purchases are made accordingly.
- Sometimes, the entire system development is outsourced to an external organization.

The entire system development process from input to output stage is documented, as proper documentation of the development process is very useful in future upgradation and maintenance of the system.

Phase VI: System Testing

To check and see if the system is performing as per the expectations, system testing is done. Software testing finds the errors in software, software design, etc. During software testing, bugs and errors in the software are detected and removed.

Testing is a quality control system technique in software development. As per IEEE (Institute of Electrical and Electronic Engineers), testing is the process of evaluating a system and its components, using manual or automated techniques. This is done to verify that the software satisfies the specified requirements and determine the difference between expected and actual results.

There is a proper Test-Plan which identifies the type of test to be performed, the schedules of tests, guidelines for testing and the resources to be allocated.

NOTES

Testing identifies the errors and the errors are removed using debugging techniques. Testing also ensures performance issues, safety and capacity of the system.

Testing Techniques

There are two types of testing techniques:

1. Black Box Testing

This type of testing involves testing the software for functionality. This will involve looking for errors in data structure, faulty functions, interface errors, software initialization and termination errors, etc.

2. White Box Testing

The internal structure of the software program is tested, and errors are detected during white box testing. It involves testing of all logic of the program, testing of loops, conditional testing and data flow-based testing.

Phase VII: System Implementation

Putting the system into operation is what known as system implementation. The system implementation phase involves installation of the system, training of the users, creation of computer files and installation of all the hardware and software components.

The main motive of the system implementation phase is to:

(i) Make a proper working system

(ii) Install the system

(iii) Replace the old system

(iv) Finalize user and system documentation

(v) Train the users

(vi) Provide support systems for users

There are three types of implementations:

(i) **Direct conversion:** It involves implementing the new system directly in place of the old system (Manual or Existing System).

(ii) **Phased conversion:** As the name suggests, the conversion is carried out in phases, that is, a module by module approach is adopted.

(iii) **Parallel conversion:** Both the old and new system run together in parallel, after verification of the results, parallel processing is stopped.

Phase VIII: Post-Implementation and Maintenance

This is the last phase of Software Development Life Cycle (SDLC) and it is an ongoing phase. This is because it deals with fixing the problems of the changing business environment as well enhancing the system. The maintenance involves activities across all the phases of the system development life cycle. Correcting

code, updating documentation, correcting design errors, making the system more user friendly etc. are some of the activities involved in system maintenance.

Maintenance activity is the most time consuming and costly activity of SDLC.

Maintenance has been classified into four types:

(a) **Corrective maintenance:** It means correcting or removing bugs in the system or making changes in the programs, designs etc.

(b) **Adaptive maintenance:** When the software is put into a new environment (new hardware, etc.), the maintenance done on it is called as adaptive maintenance.

(c) **Perfective maintenance:** When the system is enhanced in terms of performance or new needs of the user, it is knows as perfective maintenance.

(d) **Preventive maintenance:** When a particular bug or error is anticipated in advance and changes are done in the system to overcome the error before it actually effects the functioning of the system, is called as preventive maintenance.

NOTES

Check Your Progress

1. What is a system development life cycle?
2. Write the various phases of system development life cycle.
3. What is human feasibility?
4. Explain the purpose of system requirement determination.
5. Write the five features of Software Requirements Specification (SRS).
6. Explain the concept of system design.
7. Name the types of testing techniques.
8. What is role of system implementation?
9. Write the types of system maintenance.

2.3 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. System Development Life Cycle (SDLC) is a systematic and sequential approach to solving and developing a system. In other words it refers to the various phases which occur in a sequence to develop a system. The SDLC gives you a process or suggests steps to convert the goals/subjects into requirements and the requirements into design. Further that design can be developed into a system and then the system can be tested and maintained.

NOTES

2. The phases of SDLC can be categorized as under:
   (i) Initial or preliminary investigation
   (ii) Feasibility study
   (iii) System requirement determination
   (iv) System design
   (v) Software development
   (vi) System/software testing
   (vii) System implementation
   (viii) Post-implementation and maintence
3. Human feasibility deals with the people of the organization. It studies the acceptability level of the employees of the organization. It is human tendency to resist change. So, for a new system to be implemented, it is much better if you study in advance, how the people in the organization are going to react and what would be the strength of their reaction so as to assess the situation in advance. According to the feasibility report, the pace of implementation has to be adjusted for the new system.
4. Software Requirement Specification (SRS) document. SRS is an agreement between the analyst and the client, which mentions the requirement from the new system including all performance and operational requirements.

   The SRS will also describe the inputs and outputs of the system. The better the SRS, the better the software will meet its expectations. The SRS document helps the system developers and software engineers to design and develop the system with the least possibility of errors.
5. Completeness: The SRS document should contain all the system requirements.

   Understandability: The requirements should be completely understood by the clients, user's analysts and developers.

   Consistency: The requirements should not be contradicting in nature.

   Correctness: The requirements mentioned are the SRS should be same as the user desires.

   Clarity: The requirements mentioned in SRS should mean the same to all.

   Modification capability: The SRS document should be made in such a manner that if there in change is the user requirement, it would allow the changes without affecting the entire system.

   Traceability: Each requirement in the SRS should be able to match with the ongoing software development. It helps in verification and validation.

   Verifiability: The requirements of the SRS should be verifiable at all stages of the development of the software(structure of SRS).

6. System design is the most important phase before system development. It is the blue print of the requirements that will help in developing the software. All the technical specifications are finalized for the development of the system at this stage. The design phase mentions the type of data to be input, the processes used, the output format, etc.
7. There are two types of testing techniques:

   - **Black Box Testing**

   Black box testing type of testing involves testing the software for functionality. This will involve looking for errors in data structure, faulty functions, interface errors, software initialization and termination errors, etc.

   - **White Box Testing**

   The internal structure of the software program is tested, and errors are detected during white box testing. It involves testing of all logic of the program, testing of loops, conditional testing and data flow-based testing.
8. Putting the system into operation is what known as system implementation. The system implementation phase involves installation of the system, training of the users, creation of computer files and installation of all the hardware and software components.
9. Maintenance has been classified into four types:
   - Corrective maintenance: It means correcting or removing bugs in the system or making changes in the programs, designs etc.
   - Adaptive maintenance: When the software is put into a new environment (new hardware, etc.), the maintenance done on it is called as adaptive maintenance.
   - Perfective maintenance: When the system is enhanced in terms of performance or new needs of the user, it is knows as perfective maintenance.
   - Preventive maintenance: When a particular bug or error is anticipated in advance and changes are done in the system to overcome the error before it actually effects the functioning of the system, is called as preventive maintenance.

2.4 SUMMARY

- System Development Life Cycle (SDLC) is a systematic and sequential approach to solving and developing a system. In other words it refers to the various phases which occur in a sequence to develop a system.
- The SDLC gives you a process or suggests steps to convert the goals/ subjects into requirements and the requirements into design. Further that

NOTES

design can be developed into a system and then the system can be tested and maintained.

- The first phase of the system development life cycle is preliminary investigation.
- To find out the correct problem or flaw for a new system or an existing system is a very important aspect. Infact, it forms the basis of the entire SDLC.
- The system or process you want to develop may be as small as developing a work scheduler on your personal computer or as complex as putting a satellite on the moon.
- The feasibility study encompasses judging whether the system to be developed would be feasible, i.e. whether it will prove to be beneficial to the organization, whether it will be workable in various aspects, whether it will be acceptable.
- Feasibility in terms of time, operations, maintainability, legality, etc. is also studied to completely see the practicality of the project.
- Economic feasibility determines the total cost of development of the system, the operating cost, the cost of the software and hardware, the maintenance cost, etc.
- If the benefits of the proposed system outweigh the cost of developing and maintaining the system, then a go-ahead is given to develop the new system.
- Technical feasibility involves the technology aspect of the feasibility study. It will analyse the current technologies as well the emerging technologies.
- The study will also involve the software, hardware and upgrade requirements.
- Human feasibility deals with the people of the organization. It studies the acceptability level of the employees of the organization. It is human tendency to resist change. So, for a new system to be implemented, it is much better if you study in advance, how the people in the organization are going to react and what would be the strength of their reaction so as to assess the situation in advance. According to the feasibility report, the pace of implementation has to be adjusted for the new system.
- The purpose of the system requirement determination analysis is to gather all data and information to understand user requirements.
- The SRS will also describe the inputs and outputs of the system. The better the SRS, the better the software will meet its expectations.
- The SRS document helps the system developers and software engineers to design and develop the system with the least possibility of errors.
- The SRS document should contain all the system requirements.

NOTES

- The requirements should be completely understood by the clients, user's analysts and developers.
- The requirements should not be contradicting in nature.
- The requirements mentioned are the SRS should be same as the user desires.
- The requirements mentioned in SRS should mean the same to all.
- The SRS document should be made in such a manner that if there in change is the user requirement, it would allow the changes without affecting the entire system.
- Each requirement in the SRS should be able to match with the ongoing software development. It helps in verification and validation.
- The requirements of the SRS should be verifiable at all stages of the development of the software(structure of SRS).
- System design is the most important phase before system development. It is the blue print of the requirements that will help in developing the software. All the technical specifications are finalized for the development of the system at this stage. The design phase mentions the type of data to be input, the processes used, the output format, etc.
- Output design describes the format of the output as per the requirement.
- Files are made up of data. The file design mentions the process of file creation, data storage in files and data retrieval.
- The operations to be carried out on the computer and the operations to be carried out manually are also mentioned.
- The design gets converted into the actual system. The blue prints are translated into software, (i.e. programs).
- The entire project according to the size is broken into modules and each module is assigned to separate teams as per the expertise of programmers in the respective teams.
- The modules developed are made independent so that they can be executed and tested individually.
- The design specifications are converted into codes (programs) by the programmers.
- As per the design requirement, it is decided to develop new programs or modify the existing programs.
- The procurement of new software or hardware is also assessed and purchases are made accordingly.
- Sometimes, the entire system development is outsourced to an external organization.

NOTES

- The entire system development process from input to output stage is documented, as proper documentation of the development process is very useful in future upgradation and maintenance of the system.
- To check and see if the system is performing as per the expectations, system testing is done.
- Testing is a quality control system technique in software development.
- Testing identifies the errors and the errors are removed using debugging techniques.
- Testing also ensures performance issues, safety and capacity of the system.
- Black box testing type of testing involves testing the software for functionality. This will involve looking for errors in data structure, faulty functions, interface errors, software initialization and termination errors, etc.
- Putting the system into operation is what known as system implementation. The system implementation phase involves installation of the system, training of the users, creation of computer files and installation of all the hardware and software components.
- The maintenance involves activities across all the phases of the system development life cycle.
- When the software is put into a new environment (new hardware, etc.), the maintenance done on it is called as adaptive maintenance.
- When the system is enhanced in terms of performance or new needs of the user, it is knows as perfective maintenance.
- When a particular bug or error is anticipated in advance and changes are done in the system to overcome the error before it actually effects the functioning of the system, is called as preventive maintenance.

2.5 KEY WORDS

- **System Development Life Cycle (SDLC):** SDLC is a systematic and sequential approach to solving and developing a system.
- **Feasibility study:** The feasibility study encompasses judging whether the system to be developed would be feasible, i.e. whether it will prove to be beneficial to the organization, whether it will be workable in various aspects, whether it will be acceptable.
- **Economic feasibility:** It determines the total cost of development of the system, the operating cost, the cost of the software and hardware, the maintenance cost, etc.
- **Human feasibility:** It deals with the people of the organization.

NOTES

- **Completeness:** The SRS document should contain all the system requirements.
- **Understand ability**: The requirements should be completely understood by the clients, user's analysts and developers.
- **Consistency**: The requirements should not be contradicting in nature.

2.6 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. What is system development life cycle?
2. Define the term preliminary investigation.
3. Differentiate between technical and human feasibility.
4. Explain the term system requirement determination.
5. State about the system design.
6. What is software development?
7. Explain the term system testing.
8. Write the types of testing techniques.
9. State about the system implementation.
10. Write the types of system maintenance.

Long- Answer Question

1. Describe the phases of the system development life cycle.
2. Briefly explain the feasibility study and the various types of feasibility giving details.
3. Differentiate between system design and software development.
4. Describe the various testing techniques giving appropriate examples.
5. Briefly explain the system implementation giving details of phases and its various types.
6. Discuss about the system maintenance and explain the types in details.

2.7 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

NOTES

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 3 SYSTEM ANALYST

NOTES

Structure

3.0 Introduction
3.1 Objectives
3.2 System Analyst
  3.2.1 Interpersonal Skills
  3.2.2 Analytical Skills
  3.2.3 Technical Skills
  3.2.4 Management Skills
3.3 Role of System Analyst: System Planning and Initial Investigation
3.4 Answers to Check Your Progress Questions
3.5 Summary
3.6 Key Words
3.7 Self Assessment Questions and Exercises
3.8 Further Readings

3.0 INTRODUCTION

System analysts are IT professionals who act as an intermediary between users and technical team. They are responsible for integrating business requirements into technology and ensure smooth functioning of the business operations. They use both business and technical knowledge for analysing business processes, computer systems, and infrastructure to develop effective strategies that can help in accomplishing daily needs of the organization. To perform their job, system analysts have to be proficient in programming language, the configuration of systems, and multiple operating systems.

The person who plays a major role in the analysis, design and development of the system is termed as a system analyst. The role of a system analyst has been emerging with the changing technology. The main responsibility of the system analyst is to bridge the gap between the user and the software developer. He understands both the business and the computing. An analyst must possess various skills to effectively carry out his responsibilities. These skills are basically divided into four categories: analytical skills, technical skills, management skills and interpersonal skills.

System analysts bridge between customers, IT persons, and stakeholders to develop information systems capable of delivering business requirements. The integration of technology into business requirements has to be futuristic. It means systems analysts have to develop information systems that are easy to upgrade in the future if the need arises. They have to design an information system architecture according to the user's requirements which acts as a blueprint for the programmers. For that, they need to know exactly what users want and also have to build good

NOTES

relationships and rapport with them to understand their requirements as well as convey correct and complete information to the development team.

In this unit, you will study about the interpersonal skills, technical skill, communication skills, and role of systems analyst.

3.1 OBJECTIVES

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

- Know about the system analyst
- Discuss about the different types of skills
- Understand the role of system analyst

3.2 SYSTEM ANALYST

The person who plays a major role in the analysis, design and development of the system is termed as a system analyst. The role of a system analyst has been emerging with the changing technology. The main responsibility of the system analyst is to bridge the gap between the user and the software developer. He understands both the business and the computing. An analyst must possess various skills to effectively carry out his responsibilities. These skills are basically divided into four categories: analytical skills, technical skills, management skills and interpersonal skills.

3.2.1 Interpersonal Skills

To perform systems analysis, the system analysts must have different skills to perform their tasks. These skills are mainly categorized into two types namely, interpersonal skills and technical skills that are needed for developing and analysing a system respectively. The interpersonal skills enable the system analyst to handle the relationships with their surrounding business people. The interpersonal skills help system analysts to build trust and solve conflicts with the users. The various interpersonal skills of system analyst includes the following:

- **Communication**: It enables the system analyst to communicate with the users in terms of speaking, listening, feeling or reacting to maintain harmony and coordination.
- **Understanding:** It enables the system analysts in attaining a strong understanding of the company goals and objectives that helps in identifying problems which may arise during the system development process. It also enables the analyst to assess the remedies for the identified problems.
- **Selling:** It enables the system analyst to contribute in selling ideas and provide innovative solutions to solve problems using computer systems.

NOTES

The technical skills of the system analyst lay stress on the methods, techniques and operations involved in system analysis. Some of the technical skills of the system analyst are:

- **Creativity:** It helps the system analyst to convert the ideas of system development into planning for system development.
- **Problem Solving:** It enables the system analyst to reduce the number of problems at the elemental level of system analysis, so that if there is a large number of problems, alternative solutions can be considered.
- **Project Management:** It enables the system analyst to manage the project handling of the development system that may include preparing schedules, coordinating the people involved in the project and managing costs and expenditure.
- **Dynamic Interface:** Dynamic interface helps the system analyst to consider both technical and non-technical requirements in optimum combination for the development of the system.

System analysis is a process of analysing the existing system in order to gather the data that determines the scope, functionality and focus of the proposed system. The purpose of system analysis is to transform the input, such as system charts and user policies into structured specification. System analysis involves representing the user environment with system analysis tools, such as Entity-Relationship (E-R) diagrams, functioning diagrams and data diagrams. It is the responsibility of the systems analyst to carry out all the activities of the system analysis phase which includes the following functions:

- Identifying and understanding the various operations handled by the existing system.
- Understanding the user requirements.
- Verifying the efficiency of the working methods applied to the participants, such as users, analysts, designers and programmers of system development.
- Verifying the changes that are implemented in the process of system development.
- Determining the tools required for system analysis.
- Verifying that the workload allocation is logical and whether or not the users have to take part in both the systems, existing and new, simultaneously.
- Verifying that whether the documents are easily understandable by the user or not.
- Deciding the scope and focus of the new system.
- Deciding the functions of the new system.

The role of the system analysts has been emerging with changing technology and nowadays they have to perform a different number of roles according to the changes in their responsibilities for system analysis. The various roles of the system analyst includes the following:

- **Change Agent:** In this role, the system analyst is viewed as an agent of change where the analyst introduces the changes in the system to the users.

NOTES

- **Investigator and Monitor:** As an investigator, the analyst defines a problem by analysing the available system information to deduce the errors or loop holes in the system and provides the corrective changes for the system. Again, as a monitor, the analyst is continuously involved in checking the corrected system for further errors or drawbacks.
- **Architect:** This role involves creating a detailed physical design of the system on the basis of the user requirements.
- **Psychologist:** This role enables the analyst to assess the perception of the users about the system and helps evaluate their responses and feedback.
- **Motivator:** This role enables an analyst to provide sufficient motivation to the new users that help them work on a newly implemented system.

3.2.2 Analytical Skills

An analyst must have the ability to see things as systems. He should have the ability to identify and analyse the problem in an effective way so that an optimal solution of the problem can be provided for a specific organization. Analytical skills include the following:

- **Systems Thinking:** An analyst must have the ability to determine the components, boundaries, purpose, environment, interfaces, input, output and constraints of the system to be built.
- **Organizational Knowledge:** An analyst should have the understanding of the working of the organization. He should also have the knowledge of specific functions, internal policies, strategies and tactics of the organization.
- **Problem Identification:** An analyst must be able to identify the problem by comparing the existing situation and the desired situation.
- **Problem Analysing and Solving:** An analyst should be able to analyse the problem by collecting all relevant information. He should be familiar with numerous design approaches required during the development of software so that various design alternatives can be formulated. At the same time, he should also be able to translate user's ambiguous requirements into precise specifications. He should be able to choose the best solution among the various alternative solutions and finally, he should be able to put the chosen solution into practice.

3.2.3 Technical Skills

An analyst must have the ability to understand how computers, networks, databases, operating systems, etc., work together as a system. He should also know their potentials and limitations. He must have a good understanding of a wide variety of hardware and software technologies including:

- Computers, such as Personal Computers (PCs), microcomputers, workstations, minicomputers and mainframe computers.

- Computer networks, such as Local Area Network (LAN), Wide Area Network (WAN), Virtual Private Network (VPNs), administration and security.
- Operating systems, such as UNIX, Mac/OS and Windows.
- Data communication protocols and standards, such as File Transfer Protocol (FTP) and HyperText Transfer Protocol (HTTP).
- Programming and database languages, such as C++, Java, eXtensible Markup Language (XML), Structured Query Language (SQL), and so on.
- Software applications, such as Microsoft Office.
- Information systems, such as databases, Management Information System (MIS), decision support systems, and so on.
- System development tools and environments, such as report generators, office automation tools, and so on.

3.2.4 Management Skills

Management skills include project management, resource management, risk management and change management.

- **Project Management:** A system analyst should be able to determine the tasks and resources needed for a project and how these tasks and resources are related to each other. He should assist management in keeping track of progress of the project to meet due date and budget requirement.
- **Resource Management:** A system analyst should be able to effectively manage the resources required for the project, including time, equipment, hardware, software, people, money, etc.
- **Risk Management:** A system analyst should be able to identify and minimize the risks involved in the project.
- **Change Management:** A system analyst should be able to manage the transition of the system from one state to another.

Check Your Progress

1. Who is system analyst? Define its role.
2. How the systems analysis task is performed?
3. Define the concept of dynamic interface.

3.3 ROLE OF SYSTEM ANALYST: SYSTEM PLANNING AND INITIAL INVESTIGATION

The system planning phase usually begins with a formal request known as system request which describes problems or desired changes in an information system or a business process. In many organizations, system planning is an integral part of overall

NOTES

system development life cycle. A system request comes from a top manager, a planning team or a department head. The purpose of this phase is to perform a preliminary investigation to evaluate business opportunities or problem. The preliminary or initial investigation is a critical step because the outcome will affect the entire development process. A key feature of preliminary investigation is a feasibility study that reviews anticipated costs and benefits and recommends the actions based on operational, technical, economic and time factors. Suppose you are a system analyst and you receive a request for system change or improvement. Then the first step of system analyst is to determine whether it launches a preliminary investigation at all. After an investigation, you might find that the information system functions properly but users need more training to be made aware of the systems functioning. But, the complete scale system review is necessary.

Preliminary investigation is the first step in SDLC. It begins once a project request is made. The purpose of this preliminary investigation is to evaluate the project request. However, this investigation does not completely leads to the description of the business system. Such investigation only helps in the collection of information for committee members to evaluate the merits of the project requests leading to an informed decision about the feasibility of the proposed project.

Systems analysts carry on the work of preliminary investigation with an aim to achieve the following objectives:

- Clarification and understanding of the project request.
- Determination of the size of the project.
- Cost assessment and decision on benefits of alternative approaches.
- Determination of the technical and operational feasibility of alternative approaches.
- Report of the investigation to the management.
- Recommendations on the acceptance or rejection of the proposal.

During the preliminary investigation, a systems analyst gathers data through the following three primary methods:

- Reviewing organization documents.
- Onsite observations.
- Conducting interviews.

Reviewing Organization Documents

This is the first approach that a systems analyst adopts. He first learns about the organization involved in the project. For example, while reviewing an inventory system proposal he gathers information to know about the working of the existing system in the department and about the people directly associated with the inventory system. The analyst obtains details by examining the organization's charts and also carries out studies of written operating procedures. These procedures define important steps involved in receiving, managing and dispensing stock. After gathering information at the first step, the analyst moves on to actual locations to collect data by onsite observation. Thus, the analyst or a group of analysts directly observe the activities of the system. The purpose of this onsite observation is to get a close feel

of the real system being studied. During this phase of observation, the analyst is able to observe the office environment, the workload of the system, the users involved in the operation, the methodology adopted for work and the facilities provided by the organization to the users.

NOTES

Conducting Interviews

The two techniques of investigation discussed till now reveals to the analyst the way a system operates but these may not provide enough details to take a decision on the merits of a system's proposal. Also, they do not reveal the users' views on the current operations. These details can be gathered only by personal interaction with the users. The analyst then resorts to interviews to gather these details. Interviews give more information to the analyst about the nature of the project request whether it is technically, economically and operationally correct.

When a request for an information system arrives, the preliminary investigation, which is the first activity, begins. This comprises of following three parts:

- Clarification of request.
- Feasibility study.
- Approval of request.

Clarification of Request: Requests that come from employees and users in the organizations are not clear and it is difficult to decide the scope of work. This warrants the need to examine and clarify properly before carrying out system investigation.

Feasibility Study: It is an important outcome of the preliminary investigation which determines whether the system, for which the request has been made, is feasible or not. Feasibility study has three aspects as follows:

- ***Technical Feasibility:*** This is related to the technicality of the project. This ponders over the question whether the work for the project can be done with the existing equipment, the current software package and the available human resources. In case, a new technology is required, it is essential to know whether it is possible to develop it further.

  Technical issues, which are generally pointed out during the feasibility stage, are as follows:
  - Availability of necessary technology.
  - Technical capability to procure data and to use a new system equipment.
  - Possibility of future upgradation if developed.
  - Technical guarantees for accuracy and reliability over and above the factors like ease of access and security of data.
- ***Economic Feasibility:*** Every organization wants to develop a system that gives it some benefit in terms of financial return and that too at a reasonable cost. So economic feasibility addresses issues related to the creation of the system to make the costs affordable and acceptable. Another important question is whether the cost of not creating the system is so great that it is advisable to undertake the project.

NOTES

A system developed technically and installed properly must be profitable for the organization. Financial benefits should be equal to or more than the cost of the system. The analysts ponder over various queries of financial and economic nature during the preliminary investigation to estimate the following:

- The cost of full systems investigation.
- The cost of hardware and software.
- The benefits in the form of reduced costs of operation or fewer costly errors.
- The cost if the proposed system is not developed.

- ***Operational Feasibility:*** This feature of feasibility study is related to its operational aspect wherein the working of the hardware, the software and the human resource is to be taken into account. So, the question of whether the system will work if it is developed and implemented comes naturally. Another aspect related to human resource is whether there will be resistance from users who may not agree to accept the changes.

  Proposed projects can be beneficial if turned into information systems meeting the requirements of operation for the organization. Important aspects of assessing the operating feasibility are as follows:

  - Enough support for the project from the management as well as users. There may be resistance if people are accustomed to the present system and are not ready to think of any alternative system and resist a change.
  - If an existing business methodology does not suit the users, a change takes place. This brings about a better and useful system in operation.
  - Users' involvement in the planning and development of the new system as a project is very much required. Involvement from the early stages, i.e., from the start of the project and during the development stage too, reduces the chances of resistance to a great extent.

  If users feel that the proposed new system is not in their interest or may not produce the desired results, or is feared that it will have an adverse effect on the overall working personnel, then, there will be a sharp reaction and resistance to the installation of a new system.

  Issues that appear minor in the beginning may grow and assume major proportions later on when the system is implemented. This warrants a careful consideration of the system's operational aspects.

  Feasibility studies are carried out by a selected group of people having expertise in information systems techniques. They have the ability to understand the parts of the business or organization involved or affected by the project, and have enough skill in the systems analysis and design process.

  To be judged feasible, a proposal for the specific project must pass all these tests. Otherwise, it is not considered a feasible project.

- **Approval of Request:** Teams carrying out feasibility studies may not find all requested projects feasible. Out of so many project requests, only a few

may be found worth pursuing. Projects found feasible and considered as desirable get listed in a schedule. Development work, in some cases, may start immediately if systems staff members are not busy with other ongoing projects. In case systems personnel are not available and are busy with other projects, management decides on setting priorities and then projects are scheduled accordingly. The estimation of cost, priority, completion time and personnel requirements is made post approval. These projects are queued up and later on when the other projects are completed, the proposed application development which is next in queue, is initiated.

Information Gathering

A systems analyst must obtain a clear understanding of an existing system in order to decide how much improvement can be made with the new system in the context of fact finding techniques. Many people understand the specific operations in a system but may not understand the entire system or vice versa. It is the responsibility of the analyst to understand the existing system before making any changes. This will help the analyst to evaluate the performance of the new system compared with the previous system. Figure 3.1 shows what the systems analyst has to decide.

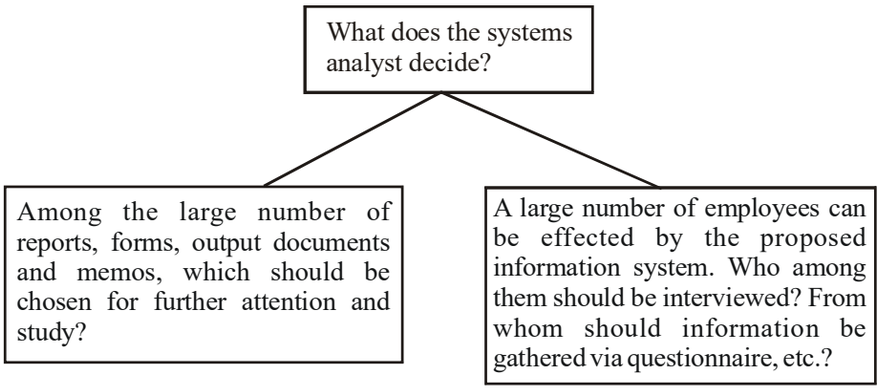


Fig. 3.1 *Work Area of Systems Analyst*

Various fact finding methods are available that help the systems analysts to collect information about the existing system. For example, the systems analyst can review documents, conduct interviews and questionnaires and perform onsite observations and Joint Application Development (JAD) operations. Systems analysts usually apply more than one of these techniques to ensure an accurate and comprehensive investigation of the existing system. Another important technique that helps in collecting information related to a software system is sampling. Sampling is a technique that helps collect information related to a large software system by investigating smaller and randomly selected items of the software. While examining these selected elements, it is assumed that the analysis will reveal useful information about the existing software system as a whole. Sampling reduces costs, speeds up the data gathering process and improves effectiveness. Figure 3.2 shows information gathering methods.

NOTES

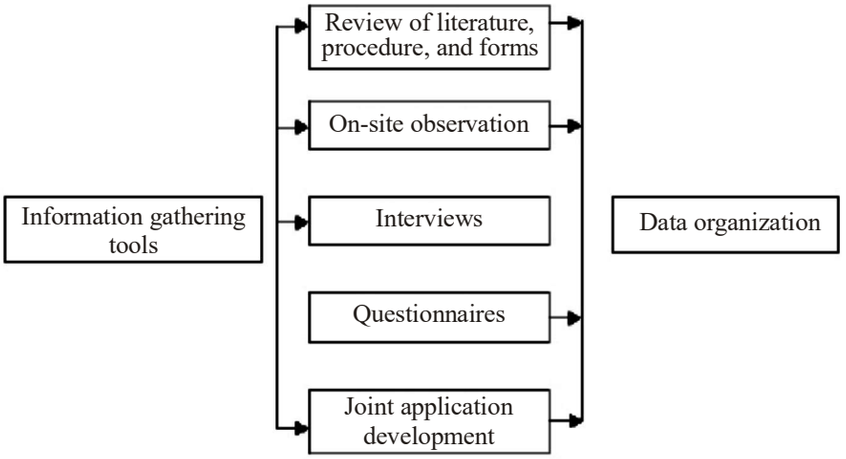


Fig. 3.2 *Information Gathering Techniques*

While collecting information about the existing system, the systems analyst should determine how that system came into existence. After conducting the preliminary investigation, the analyst must be in a position to answer the following questions:

- How was the system designed and installed?
- Who implemented it?
- Will anyone try to justify or defend the existing system and thus slow up the change to a new system?

Information Gathering Tools

Various kinds of reports and records that store information related to the existing system are maintained in an organization. These reports and records can provide valuable information regarding the organization, its operations and activities, to the systems analysts. A review of available documentation is a logical starting point when seeking insight into a system. It helps people gain some knowledge about the organization or operations by themselves. Reference to inputs, outputs and files implemented in any given procedure will be helpful in documenting current operations.

Procedure manuals and forms are excellent sources of information because they describe the format and functions of the present system. Update manuals can provide a lot of information, which saves a lot of time. Recent developments in the area of Information Technology (IT) have made it possible to avail all the updated documentations about most of the systems within a short span of time. The existing forms can provide a clear understanding about the transactions that are handled in the organization. An input form can be used to identify the various data items captured for processing. An output form or report can be used to evaluate the performance of the process or to notice the different information products that are being generated from the system. When the analyst analyses the existing forms, the following question must be pointed out:

- Who are the users of the forms?

- How important are these forms to the users?
- Do the forms contain all the required information?
- Is it required to add or delete any item?
- What is the frequency of use of each form?
- How many users receive the forms?
- Is it readable or ambiguous?
- Does the information in the forms help the users to take decisions?
- How does it help the user in decision-making?

NOTES

Analysing Quantitative Documents

Each of the quantitative documents has a specific purpose and an audience for which they are targeted. Reports used for decision-making are sales reports, production reports and inventory reports, etc. Sales reports might help the decision-maker to readily spot the trend. These reports supply strategic overviews of the organizational plans. Beyond these key reports, decision-makers use many summary reports to provide background information and spot exceptions to normal circumstances.

Performance Reports

Performance reports provide the information related to the performance of the system. It helps in analysing the varying trends in the performance and in implementing the required changes. The main functions of the performance reports are as follows:

- They report the actual versus the intended performance.
- They help the systems analyst to access the size of the gap between the actual and intended performance.
- They indicate whether this gap is widening or narrowing.
- These performance reports show goals.

Records

Records are maintained to help the systems analyst determine the functioning of the system. The records are maintained daily, monthly or yearly based on the requirements and the time period specified for analysing the reports. The main functions of records are as follows:

- They provide a periodic update of what is occurring in the organization if updated in a regular manner.
- They help the analyst to check for errors in amounts and totals.
- They help in observing the number and type of transactions.
- By inspecting the records, the systems analyst can think about improving or changing the recording form design.

- It helps find instances when computers can simplify the work further, for example calculations and data manipulations.

Data Capturing Forms

NOTES

Data capturing forms help in collecting specific information regarding the requirement of the system. The factors that should be considered while using data capturing forms is as follows:

- The organizational system is to be understood before changing the information flow in the organization.
- Blank forms, along with their instructions for completion and distribution, can be compared to filled-in forms for analysing the collected information. Also, one needs to verify whether the people, who are supposed to receive the forms, actually do get them and if they follow standard procedures for using, storing and discarding them.

Analysing Qualitative Documents

Analysing qualitative documents helps you to understand how the organizational members are engaged in the process of organizing. The various types of qualitative documents include memos and signs on bulletin boards and in work areas, manuals and policy handbooks.

Memos

Memos are qualitative documents that help in providing details regarding the daily activities of the organization. Maintenance of memos on daily basis helps in understanding the environment of the organization. The functions of a memo are as follows:

- Memos provide an insight into the metaphors that guide the organization's thinking. While analysing memos, it should be determined who sends the memos and who receives them.
- Memos in print or on e-mail present a lively, continuing dialog in the organization.
- Memos represent the clear picture of the values, thoughts and beliefs of the employees of an organization.

Manuals

Manuals are qualitative documents that pertain to the working area of any system. These manuals can be based on the functioning of an organization or the system and these manuals help in analysing their performance. Some facts related to manuals are as follows:

- Systems analysts should examine the organizational manuals, computer system manuals and different kinds of online manuals.

- Manuals present the ideal way in which machines and people are expected to behave.
- Examining the current manuals helps determine the way things ought to happen.

NOTES

Policy Handbooks

Policy handbooks are documents that provide information regarding the policies and rules of the organization or the system. The following are the facts related to the policy handbook:

- They indicate the company's ideal way of doing things and achieving goals.
- Some computer policies may appear on the screen whenever a particular program is run.
- Sometimes, the members are not aware of some particular policies.
- Policies are sometimes purposely sidestepped in the name of efficiency or simplicity.

Unfortunately, documentation seldom describes a system completely and is often not up-to-date. The current operation may differ significantly from the one described. Complete information can be obtained by searching such literature as professional references, user manuals, textbooks and company or government publications.

Sometimes, it is difficult to get certain reports and publications. Sometimes, it may be too expensive and time consuming. Records can be inspected at the beginning of a study. This can serve as an introduction which can help the analyst to understand the history of the system. This can also help later on during the study.

Onsite Observation

Observation allows analysts to gain information they cannot obtain by any other fact finding methods. With thorough observation, you can gain first hand information whereas by reviewing the documents, you get only secondary information. This method is really useful when analysts want to observe how transactions are handled, when and where the documents are captured, how the data flows or how activities are carried out, and so on. According to Harper Boyd and Ralph Westfall, 'Observation is the process of recognizing and noting people, objects and occurrences to obtain information.'

The objective of an onsite observation is to get as close as possible to the actual system being studied. As observers, analysts follow certain rules. While making observations, they are more likely to listen than talk with a sympathetic and genuine interest when information is conveyed. Care is taken not to argue with the persons being observed. Observation provides a more tangible perspective of what is described in the documentation. It also brings out the aspects of the documentation that are incomplete or outdated.

NOTES

Steps for Observation

The required steps are as follows:

- Decide what is to be observed.
- Decide the level of activities.
- Create categories for capturing key activities.
- Prepare materials for observations.
- Decide the time for observation.

All the same one must note that the observation method. It is subject to what is known as the Hawthorne effect. The Hawthorne effect suggest that when people are aware that they are being watched, they tend to act differently. The systems analyst who is unaware of this fact may draw erroneous conclusions.

Onsite Observation Methods

When human observers are used, the specific alternative observation methods are considered. These are as follows:

(i) **Natural Observation:** It takes place in a setting, such as at an employee's workplace.

(ii) **Contrived Observation:** It takes place in a setup by the observer, such as laboratories. For example, checking bugs in an individual system and checking the progress of trainees in a training program.

(iii) **Obtrusive Observation:** It takes place with the knowledge of the respondent. For example, visiting of external auditor in the organization comes under obtrusive observation.

(iv) **Unobtrusive Observation:** It is held without the knowledge of the respondent. For example, surprise internal audit comes under unobtrusive observation.

(v) **Direct Observation:** It takes place when the observer observes the system at work by physically being present at the actual workplace.

(vi) **Indirect Observation:** It makes use of secondary devices, such as video cameras to capture information. For example, daily proceedings of the bank may be observed with the help of a video camera.

(vii) **Structured Observation:** It takes place in a formal way in which each activity in the observation process is predefined. For example, tracing the route of the sales invoice in a system.

(viii) **Unstructured Observation:** The observer observes whatever might be pertinent at the time. For example, if the observer wants to create a list of activities of the production supervisor by observing him from a remote location.

Time and Event Sampling

Time sampling is also an information collection technique that helps the systems analyst maintain a definite time interval between two subsequent observations of

various activities, managers and decision-makers. For example, observing a decision-maker during five randomly chosen ten minute intervals throughout a week. Time sampling cuts down the bias that might affect that observation which made just any time. This sampling gives a representative view of the activities that occur frequently. Event sampling involves sampling of entire events, such as meetings and conferences. This sampling provides observation of an integral behaviour in its natural context.

Time Sampling

Advantages

The advantages of time sampling are as follows:

- It reduces the chances of faults.
- It provides a descriptive view of activities.

Disadvantages

The disadvantages of time sampling are as follows:

- It gathers data in a fragmented form.
- It overlooks uncommon but important decisions.

Event Sampling

Advantages

The advantages of event sampling are as follows:

- It examines the behaviour of various events of a software system.
- It examines the important event of a software system more clearly.

Disadvantages

The disadvantages of event sampling are as follows:

- It is time consuming.
- It does not provide the descriptive sample of frequent decisions.

Adequate training and preparation is required for an onsite observation because intruding into the user affected areas often results in adverse reactions by the staff. Observations are subject to error because sometimes the observers may be biased or they may also unknowingly misinterpret facts. Another disadvantage of this method is that it consumes a lot of time. Too many hours or days are often spent in an attempt to observe specific, one time activities. Observation is then a basic method of gathering information for the systems analyst.

Examining the Behaviour of Decision-Makers

To obtain information about how tasks related to a software system are done, you need to observe each task carefully. It must be structured in order to be interpretable. Reasons for observation are to gain information, which cannot be acquired by using any other method and to confirm or negate what has been found through other

NOTES

methods. By observing the activities of the decision-makers, the analyst can gain an insight into what is to be actually done. It is important to observe the relationship between the decision- makers and the other organizational members. The analyst examines the physical elements of the decision-maker's workspace for their influence on the decision-making behaviour.

Through observation of the physical elements, over which the decision-maker has control, it can be understood what messages he/she is sending. Through the method of observation, it is important to comprehend the influence of the decision-maker on others in the organization. Observations must be structured and systematic in order to correctly interpret the findings. The analyst must know what is being observed. He/she must know who will be observed and when, where, why and how the observation will take place.

Physical Environment Observation

Generally, this means systematically examining the offices of decision-makers since offices are their primary workplace. Decision-makers influence and in turn are influenced by their physical environment. The method for the STRuctured OBservation of the Environment (STROBE) provides a standard methodology and classification for the analysis of those organizational elements that influence decision-making. STROBE allows other systems analysts to apply the same analytic framework to the same organization. A few key points are as follows:

Office Location

The following points need to be kept in mind regarding office location:

Accessible: It tends to increase interaction frequency and informal messages.

Inaccessible: It decreases the interaction frequency and increases task-orientated messages.

Stationery Office Equipment

- No file cabinets or bookshelves imply that the decision-maker keeps very little personal information.

Properties

Office related properties are as follows:

- Small equipment, for example Personal Computers (PCs) or calculators.
- Trade journals and newspapers.

Source of information frequently used can be,

- External, for example journals or newspapers.
- Internal, for example reports or policy handbooks.

Office lighting and color can be,

- Incandescent lighting allows easy personal communication.
- Bright lighting allows formal communication.
- Clothing worn by decision-makers.

Formal or informal clothes,

- Formal clothes imply maximum authority.
- Casual clothes imply more participative decision-making.

Analysts can choose among many application strategies while using the STROBE approach.

NOTES

Photograph Analysis

Photograph analysis includes the activity of photographing the environments of decision- makers and then analysing the photographs. The following are the advantages:

- Photographs can be referred to repeatedly.
- A photograph can focus specifically on some element of STROBE.
- A photograph can supply details that are easily overlooked during personal contact.
- All members of the systems analyst's team can use a photograph for the purpose of analysis.

Interviews

Your personal ability to talk with other people and listen to them is probably the most important asset you have for getting along in the business world. As a systems analyst, it is one of the most important tools for gathering primary data for a personal interview. It is an excellent way to understand what is happening in a system by interviewing and talking directly with responsible individuals who perform various activities in an organization.

Interviewing is a common method used by systems analysts for collecting data on information requirements. An interview is a directed conversation with a specific purpose that uses a 'Question and Answer' format. Analysts use interviews to collect information from individuals or from groups. The respondents can be internal users as well as external users. That includes Chief Executive Officer (CEOs), Vice Presidents, Chief Technical Officers (CTOs), managers, supervisors, other technical staffs, remote users and mobile users, etc.

A real world situation is sometimes different from the case studies utilized to teach interview methods as the analyst develops an interviewing approach that best fits a situation.

Jerry Fitzgerald and Ardra Fitzgerald describe four ways of interviewing. This is referred to as 'FLIP' which represents the Formal, Legalistic, Informal and Political approaches.

The formal approach is purely businesslike which follows a systematically logical plan with no idle talk or undue friendship. The legalistic approach follows a 'Thumb Rule' approach. The interviewer always goes 'By The Book' or always follows rules. The political approach is based on doing favours for people, performing only the jobs that make you look good and acting in a prudent, shrewd and diplomatic

NOTES

way. The informal or 'Play It By Ear' approach follows a plan but the interviewer is able to deviate as required without losing control of the situation. The informal approach may appear to the interviewee that the analyst is unstructured and disorganized.

The best approach is the one in which the interviewer is able to combine the above four approaches. The analyst can be formal when necessary, legalistic when challenged, play politics in necessary situations and be informal in unknown situations. An interview has the following advantages as compared with other techniques:

(i) It is flexible. The analyst can go for an informal interview where the subject/ area is not much known to him; where he is does not know what questions to ask and how to organize questions.

(ii) It validates information. The analyst can observe not only what subjects people talk but also how they express it. Information can be verified.

(iii) It can handle complex subjects. An interview is the most effective technique for developing opinions about complex subjects and probing the sentiments underlying expressed opinions.

(iv) Many people enjoy it. Usually people enjoy being interviewed when all they have to do is talk, especially expressing their opinions, if the time permits.

(v) By seeking opinions, it is easier to discover a key problem.

(vi) Facts obtained from hard data explain past performances but goals project the organizational future.

(vii) Expressed feelings help the analyst to understand emotions and attitudes (may be you can get a hint whether the project will succeed or not).

Systems interviewing refers to just talking with people in order to find out how the system is functioning and how they expect their future systems to run. Interviewing must be adopted throughout the system study because it can produce up-to-date information whereas the organization charts and other documents are at least six months behind the times. During interviews, an analyst can observe how the system runs as well as the personality and attitude of the employee which is very important in many situations.

Interviewing is an art. The initial requirements for a successful interview are to develop a cordial atmosphere and put the respondent in a relaxed mood. Then, the interviewer proceeds with asking questions properly, obtaining valuable and reliable responses and recording them accurately and completely. Any approach can be adopted to conduct an interview.

Questionnaires

Interviews are one of the best tools to collect information from a limited number of people. However, sometimes the systems analyst needs to gather information from a large number of people regarding certain issues. In such situations, the most efficient

way to obtain information is through questionnaires. A questionnaire asks people to respond to various written questions which are presented to them in a formal document.

The use of standardized questions in a questionnaire can help capture more data that is reliable. By using questionnaires, which is an information gathering technique, the systems analyst can collect valuable information from the people in the organization who may be affected by the current and proposed systems. Various tasks that are performed in the questionnaire technique are as follows:

- Acquiring information before conducting the interview.
- Gaining information in order to prove facts found in the interview.
- Acquiring information on how users feel about the current system.
- Checking for problems that remain unsolved.
- Studying the expectations of people from a new or modifed system.

Advantages of Questionnaires

The following are some of the benefits provided by questionnaires:

Well designed and interpreted, questionnaires are very effective in surveying people's interests, attitudes, feelings and beliefs.

Open-ended questions can be used by the analyst to learn more about feelings, opinions and general experiences of the respondent. A closed-end questionnaire controls the frame of reference by presenting respondents with specific responses from which they can choose.

Questionnaires help in the collection of a huge amount of data about what problems they are experiencing with their work and about what they expect from a new or modified system. Developing a useful questionnaire takes quite some time, but in doing so, you can gather lots of information without spending time in face-to-face interviews. For example, a questionnaire would be the right technique if you want to know what percentage of users prefers 'online manuals' for learning about a new software package before using it.

However, in certain situations, such as doing an in-depth analysis of a manager's decision-making process, an interview is a better choice than a questionnaire. The following are the situations in which questionnaires should be used:

- The people who are to be questioned are widely dispersed, for example in different departments or branches of the same organization.
- The project involves a large number of people and you want to know what proportion of a given group approves or disapproves of a particular feature of the proposed system.
- The analyst wants to determine the overall opinion before the systems project is given any specific direction.

NOTES

Types of Questions

The following are the types of questions:

(i) **Open-Ended:** These are questions that can be easily and correctly interpreted. They are more objective and less leading and mean to encourage a full answwer.

(ii) **Closed-Ended Questions:** These are the questions that are used when the systems analyst is able to effectively list all possible responses to the question. All possible responses of the closed questions should be mutually exclusive.

Check Your Progress

4. Write the purpose of preliminary investigation.
5. Explain the term feasibility study.
6. Write the features of operational feasibility.
7. Why are procedure manuals and forms considered as excellent sources of information?
8. Define the term memos.
9. What are policy handbooks?
10. Explain the significance of time sampling.
11. What does method of STROBE provide?
12. Explain about the types of questions that are included in the questionnaires.

3.4 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. The person who plays a major role in the analysis, design and development of the system is termed as a system analyst. The role of a system analyst has been emerging with the changing technology. The main responsibility of the system analyst is to bridge the gap between the user and the software developer. He understands both the business and the computing. An analyst must possess various skills to effectively carry out his responsibilities. These skills are basically divided into four categories: analytical skills, technical skills, management skills and interpersonal skills.
2. To perform systems analysis, the system analysts must have different skills to perform their tasks. These skills are mainly categorized into two types namely, interpersonal skills and technical skills that are needed for developing and analysing a system respectively. The interpersonal skills enable the system analyst to handle the relationships with their surrounding business people. The interpersonal skills help system analysts to build trust and solve conflicts with the users.

NOTES

3. Dynamic interface helps the system analyst to consider both technical and non-technical requirements in optimum combination for the development of the system.
4. Preliminary investigation is the first step in SDLC. It begins once a project request is made. The purpose of this preliminary investigation is to evaluate the project request. However, this investigation does not completely leads to the description of the business system.
5. Feasibility study is an important outcome of the preliminary investigation which determines whether the system, for which the request has been made, is feasible or not.
6. This feature of feasibility study is related to its operational aspect wherein the working of the hardware, the software and the human resource is to be taken into account. So, the question of whether the system will work if it is developed and implemented comes naturally. Another aspect related to human resource is whether there will be resistance from users who may not agree to accept the changes.
7. Procedure manuals and forms are excellent sources of information because they describe the format and functions of the present system. Update manuals can provide a lot of information, which saves a lot of time. Recent developments in the area of Information Technology (IT) have made it possible to avail all the updated documentations about most of the systems within a short span of time. The existing forms can provide a clear understanding about the transactions that are handled in the organization.
8. Memos are qualitative documents that help in providing details regarding the daily activities of the organization. Maintenance of memos on daily basis helps in understanding the environment of the organization.
9. Policy handbooks are documents that provide information regarding the policies and rules of the organization or the system.
10. Time sampling is also an information collection technique that helps the systems analyst maintain a definite time interval between two subsequent observations of various activities, managers and decision-makers. For example, observing a decision-maker during five randomly chosen ten minute intervals throughout a week. Time sampling cuts down the bias that might affect that observation which made just any time.
11. The method for the STRuctured OBservation of the Environment (STROBE) provides a standard methodology and classification for the analysis of those organizational elements that influence decision-making. STROBE allows other systems analysts to apply the same analytic framework to the same organization.
12. The following are the types of questions:
    (i) Open-Ended
    (ii) Closed-Ended Questions

NOTES

3.5 SUMMARY

- The person who plays a major role in the analysis, design and development of the system is termed as a system analyst.
- The role of a system analyst has been emerging with the changing technology. The main responsibility of the system analyst is to bridge the gap between the user and the software developer.
- An analyst must possess various skills to effectively carry out his responsibilities.
- To perform systems analysis, the system analysts must have different skills to perform their tasks.
- These skills are mainly categorized into two types namely, interpersonal skills and technical skills that are needed for developing and analysing a system respectively.
- The interpersonal skills help system analysts to build trust and solve conflicts with the users.
- The technical skills of the system analyst lay stress on the methods, techniques and operations involved in system analysis.
- Dynamic interface helps the system analyst to consider both technical and non-technical requirements in optimum combination for the development of the system.
- System analysis is a process of analysing the existing system in order to gather the data that determines the scope, functionality and focus of the proposed system.
- System analysis involves representing the user environment with system analysis tools, such as Entity-Relationship (E-R) diagrams, functioning diagrams and data diagrams.
- Identifying and understanding the various operations handled by the existing system.
- Verifying the efficiency of the working methods applied to the participants, such as users, analysts, designers and programmers of system development.
- The role of the system analysts has been emerging with changing technology and nowadays they have to perform a different number of roles according to the changes in their responsibilities for system analysis.
- An analyst must have the ability to see things as systems. He should have the ability to identify and analyse the problem in an effective way so that an optimal solution of the problem can be provided for a specific organization.
- An analyst must have the ability to determine the components, boundaries, purpose, environment, interfaces, input, output and constraints of the system to be built.

NOTES

- An analyst should have the understanding of the working of the organization. He should also have the knowledge of specific functions, internal policies, strategies and tactics of the organization.
- An analyst must be able to identify the problem by comparing the existing situation and the desired situation.
- An analyst should be able to analyse the problem by collecting all relevant information.
- An analyst must have the ability to understand how computers, networks, databases, operating systems, etc., work together as a system.
- Computer networks, such as Local Area Network (LAN), Wide Area Network (WAN), Virtual Private Network (VPNs), administration and security.
- Information systems, such as databases, Management Information System (MIS), decision support systems, and so on.
- A system analyst should be able to determine the tasks and resources needed for a project and how these tasks and resources are related to each other.
- A system analyst should be able to identify and minimize the risks involved in the project.
- A system analyst should be able to manage the transition of the system from one state to another.
- The system planning phase usually begins with a formal request known as system request which describes problems or desired changes in an information system or a business process.
- Preliminary investigation is the first step in SDLC. It begins once a project request is made. The purpose of this preliminary investigation is to evaluate the project request. However, this investigation does not completely leads to the description of the business system.
- Requests that come from employees and users in the organizations are not clear and it is difficult to decide the scope of work. This warrants the need to examine and clarify properly before carrying out system investigation.
- Feasibility Study is an important outcome of the preliminary investigation which determines whether the system, for which the request has been made, is feasible or not.
- This feature of feasibility study is related to its operational aspect wherein the working of the hardware, the software and the human resource is to be taken into account. So, the question of whether the system will work if it is developed and implemented comes naturally. Another aspect related to human resource is whether there will be resistance from users who may not agree to accept the changes.

NOTES

- Issues that appear minor in the beginning may grow and assume major proportions later on when the system is implemented. This warrants a careful consideration of the system's operational aspects.
- Teams carrying out feasibility studies may not find all requested projects feasible. Out of so many project requests, only a few may be found worth pursuing. Projects found feasible and considered as desirable get listed in a schedule.
- A systems analyst must obtain a clear understanding of an existing system in order to decide how much improvement can be made with the new system in the context of fact finding techniques.
- Various fact finding methods are available that help the systems analysts to collect information about the existing system.
- Procedure manuals and forms are excellent sources of information because they describe the format and functions of the present system. Update manuals can provide a lot of information, which saves a lot of time. Recent developments in the area of Information Technology (IT) have made it possible to avail all the updated documentations about most of the systems within a short span of time. The existing forms can provide a clear understanding about the transactions that are handled in the organization.
- Performance reports provide the information related to the performance of the system.
- Records are maintained to help the systems analyst determine the functioning of the system.
- Memos are qualitative documents that help in providing details regarding the daily activities of the organization. Maintenance of memos on daily basis helps in understanding the environment of the organization.
- Manuals are qualitative documents that pertain to the working area of any system.
- Policy handbooks are documents that provide information regarding the policies and rules of the organization or the system.
- Observation allows analysts to gain information they cannot obtain by any other fact finding methods.
- The objective of an onsite observation is to get as close as possible to the actual system being studied. As observers, analysts follow certain rules.
- Time sampling is also an information collection technique that helps the systems analyst maintain a definite time interval between two subsequent observations of various activities, managers and decision-makers. For example, observing a decision-maker during five randomly chosen ten minute intervals throughout a week. Time sampling cuts down the bias that might affect that observation which made just any time.

NOTES

- The method for the STRuctured OBservation of the Environment (STROBE) provides a standard methodology and classification for the analysis of those organizational elements that influence decision-making. STROBE allows other systems analysts to apply the same analytic framework to the same organization.
- Photograph analysis includes the activity of photographing the environments of decision-makers and then analysing the photographs.
- The formal approach is purely businesslike which follows a systematically logical plan with no idle talk or undue friendship. The legalistic approach follows a 'Thumb Rule' approach.
- Questionnaires help in the collection of a huge amount of data about what problems they are experiencing with their work and about what they expect from a new or modified system.
- The project involves a large number of people and you want to know what proportion of a given group approves or disapproves of a particular feature of the proposed system.
- The analyst wants to determine the overall opinion before the systems project is given any specific direction.

3.6 KEY WORDS

- **System analysis:** A process in which data are collected and then interpreted to identify the problems within the system.
- **Project management:** It enables the system analyst to manage the project handling of the development system that may include preparing schedules, coordinating the people involved in the project and managing costs and expenditure.
- **Dynamic interface:** It helps the system analyst to consider both technical and non-technical requirements in optimum combination for the development of the system.
- **Interviewing:** A common method used by system analysts for collecting data on information requirements.
- **Memos:** Memos are qualitative documents that help in providing details regarding the daily activities of the organization.
- **Policy handbooks:** Policy handbooks are documents that provide information regarding the policies and rules of the organization or the system.

3.7 SELF ASSESSMENT QUESTIONS AND EXERCISES

NOTES

Short-Answer Questions

1. Who are system analysts?
2. Explain the various roles of the system analyst.
3. Define the concept of analytical skills.
4. State the technical skills of the system analyst.
5. Explain the significance of management skills.
6. State about the interpersonal skills.
7. Why are procedure manuals and forms considered as excellent sources of information?
8. Explain the functions of memos.
9. Write the disadvantages of time and event sampling.
10. What is the advantage of interviews?

Long-Answer Questions

1. Explain the term 'System Analyst' with the help of appropriate examples.
2. Discuss about the types of a system giving examples and significance of each type.
3. Describe analytical skills, technical skills, management skills and interpersonal skills of a system analyst with the help of examples.
4. Discuss briefly about the interview process giving relevant examples.
5. Briefly explain about the information gathering tools required for developing a system.
6. Describe the onsite observation method giving relevant examples.
7. Elaborate on the time and event sampling giving the advantages and disadvantages.
8. Briefly explain about the physical environment observation and photography analysis.
9. Discuss briefly about the interviews method giving its advantages.
10. Briefly explain the questionnaires method giving its techniques and advantages.

3.8 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

NOTES

BLOCK - II
SYSTEM ANALYSIS

NOTES

UNIT 4 SYSTEM ANALYSIS

Structure

4.0 Introduction
4.1 Objectives
4.2 System Analysis
  4.2.1 Preliminary Investigation
  4.2.2 Determining the User's Information Requirements
  4.2.3 Case Scenario
  4.2.4 Problem Definition and Initiation
  4.2.5 Background Analysis
4.3 Answers to Check Your Progress Questions
4.4 Summary
4.5 Key Words
4.6 Self Assessment Questions and Exercises
4.7 Further Readings

4.0 INTRODUCTION

Whether a system will be developed by means of the Systems Development Life Cycle method (SDLC) prototyping strategy, or the structured analysis method, or a combination of these methods, a project request should first be reviewed. The choice of development strategy is secondary to whether a request merits the investment of organization's resources in an information system project.

It is advisable for all proposals to be submitted to the selection committee for evaluation to identify those projects that are most beneficial to the organization. The preliminary investigation is then carried out by systems analysts, working under the direction of the selection committee.

Shared, complete and accurate information requirements are essential in building computer – based information systems. Unfortunately, determining the information each user needs is particularly difficult task. In fact, it is recognized as one of the most difficult tasks in system development. The Association for Computing Machinery (ACM) Curriculum Committee on Computing Education for Management recognized this by suggesting two distinct job titles for systems developments: "Information Analyst" and "Systems Designer" rather than the more general term "Systems Analyst".

A problem definition usually contains some sort of problem statement, summarized in a paragraph or two. This is followed by a series of issues, or major,

independent pieces of the problem. The issues are followed by a series of objectives, or goals that match the issues point by point. Issues are the current situation; objectives are the desired situation. The objectives may be very specific or worded using a general statement.

Background analysis that clarifies the elements of the working environment is necessary before entering into the planning stage of a project. It is a process of gathering necessary information for a sustainable and suitable working approach in a specific environment. When a preliminary project idea is found, it is normally time for a proper background analysis. It is advisable to study the preliminary idea in relation to the national development plans or the local on-going development processes.

In this unit, you will study about the bases for planning in system analysis, preliminary investigation, determining the user's information requirements, case scenario, problem definition and project initiation and background analysis .

4.1 OBJECTIVES

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

- Explain about the preliminary investigation
- Elaborate on the user's information requirements
- Define the case scenario
- Discuss about the problem definition and project initiation
- Understand about the background analysis

4.2 SYSTEM ANALYSIS

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

The system planning phase usually begins with a formal request known as system request which describes problems or desired changes in an information system or a business process. In many organizations, system planning is an integral part of overall System Development Life Cycle (SDLC). A system request comes from a top manager, a planning team or a department head. The purpose of this phase is to perform a preliminary investigation to evaluate business opportunities or problem. The preliminary or initial investigation is a critical step because the

NOTES

outcome will affect the entire development process. A key feature of preliminary investigation is a feasibility study that reviews anticipated costs and benefits and recommends the actions based on operational, technical, economic and time factors. Suppose you are a system analyst and you receive a request for system change or improvement. Then the first step of system analyst is to determine whether it launches a preliminary investigation at all. After an investigation, you might find that the information system functions properly but users need more training to make aware of the system. But, the complete scale system review is necessary.

4.2.1 Preliminary Investigation

Preliminary investigation is the first phase of the systems development life cycle. As you know, there are limited resources in an organization and only those projects that are critical to its mission, goals and objectives can be undertaken. Hence, the goal of preliminary investigation is simply to identify and select a project for development among all the proposals that are under consideration. Organizations may differ in how they identify and select projects for development. Many organizations have special committees to handle the task of preliminary investigation. Such a committee or taskforce identifies and evaluates the proposals that the organization should consider for development. Several organizations function in other ways to identify and select the promising projects. The objective of the systems investigation phase is to answer the following questions:

- What is a business problem?
- What are the major causes of the problem?
- Is a new information system needed?
- Is it a problem or an opportunity?
- Is this a feasible system solution to this problem?
- Can the problem be solved by improving the current information system?

The preliminary investigation phase sets the stage for gathering information about the current problem and the existing information system. This information is then used in studying the feasibility of the possible information systems solutions. It is essential that the source of the project has a great deal to do with its scope and content.

Many different criteria can be used within an organization for classifying and ranking potential projects. The systems analyst—with the help of the stakeholders of the proposed project—collects information about the project. The collected information has a wide scope and helps understand the size of the project, cost involved and the intended benefits. After the information is analysed, a report is prepared to compare and review all the proposed projects. All the proposed project proposals are analysed and assessed on various criteria and feasibility studies carried out.

NOTES

The activities involved in preliminary investigation are as follows:

(i) Authorization to proceed: Proper authorization is taken from the relevant authorities to go ahead and investigate as the results will highlight all the details of the organization.

(ii) Understanding the objectives of the request: It is essential for the analyst conducting the investigation to clearly understand the objectives of the project, the benefit of the project, its future scope and its importance to the organization.

(iii) Investigation: It entails searching for information and analysing the collected information keeping the objectives of the organization and the aims of the request in mind. It entails the following procedure:

(a) Review current system: The current system needs to be studied, the benefits and flaws are to be clearly understood and analysed.

(b) Identify needed information: The analyst needs to identify the information required for the investigation and for doing so the following activities will be carried out:

- Organization charts and documents will be obtain and created, based on which the analyst will study the following:
  - Policy manuals
  - User manuals
  - Organization charts
  - Data input forms
  - System documentation
  - Accounts information
  - System audit and review
  - System reports
  - Reports and documentation of the current system
- The analyst will conduct interviews with all the people involved in the project, the senior managerial level as well as the users. The interview process will help to bring out those issues which may not be available in documents and forms. By conducting interviews the analyst will be able to explore and identify the exact problems in the systems, the new requirements of the system and the expectations, opinions and insecurities will also surface during the process.
- Observing the current system in operation is the best way to fact finding. The analyst will be able to observe the procedures, workload on the current system, actual gravity of the problems arising while in operation and the environment in which the system is operating. The information gathered by the analyst by this method will allow for better understanding.

NOTES

(iv) **Project feasibility analysis:** It is an important outcome of the preliminary investigation which determines whether the system, for which the request has been made, is feasible or not. Feasibility study has three aspects as follows:

(a) Technical feasibility: This is related to the technicality of the project. This ponders over the question whether the work for the project can be done with the existing equipment, the current software package and the available human resources. In case, a new technology is required, it is essential to know whether it is possible to develop it further.

Technical issues, which are generally pointed out during the feasibility stage, are as follows:

- Availability of necessary technology
- Technical capability to procure data and to use new system equipment
- Possibility of future upgradation if developed
- Technical guarantee for accuracy and reliability over and above the factors like ease of access and security of data

(b) Economic feasibility: Every organization wants to develop a system that gives it some benefit in terms of financial return and that too at a reasonable cost. So economic feasibility addresses issues related to the creation of the system to make the costs affordable and acceptable. Another important question is whether the cost of not creating the system is so great that it is advisable to undertake the project.

A system developed technically and installed properly must be profitable for the organization. Financial benefits should be equal to or more than the cost of the system. The analysts ponder over various queries of financial and economic nature during the preliminary investigation to estimate the following:

- Cost of full systems investigation
- Cost of hardware and software
- Benefits in the form of reduced costs of operation or fewer costly errors
- Cost if the proposed system is not developed

(c) Operational feasibility: This feature of feasibility study is related to its operational aspect wherein the working of the hardware, the software and the human resource is to be taken into account. So, the question of whether the system will work if it is developed and implemented comes naturally. Another aspect related to human resource is whether there will be resistance from users who may not agree to accept the changes.

Proposed projects can be beneficial if turned into information systems meeting the requirements of operation for the organization. Important aspects of assessing the operating feasibility are as follows:

- Enough support for the project from the management as well as users. There may be resistance if people are accustomed to the present system and are not ready to think of any alternative system and resist a change.
- If an existing business methodology does not suit the users, a change takes place. This brings about a better and useful system in operation.
- Users' involvement in the planning and development of the new system as a project is very much required. Involvement from the early stages, i.e., from the start of the project and during the development stage too, reduces the chances of resistance to a great extent.

Feasibility studies are carried out by a selected group of people having expertise in IT techniques. They have the ability to understand the parts of the organization involved or affected by the project and are skilled in the systems analysis and design process. To be judged feasible, a proposal for the specific project must pass all these tests. Otherwise, it is not considered to be feasible.

Preliminary Investigation Report

Finally, a report of the preliminary investigation is prepared which contains the following elements:

- **Introduction:** Here the request and the report are introduced.
- **Summary of request:** The original system request is summarized so that there is an understanding of the reason why the investigation is being carried out.
- **Findings of the investigation:** The findings will include a description of the real nature of problem(s), scope and constraints of the proposed project.
- **Recommendations:** It will include what and why further action is required. Senior executives will make final decisions keeping in mind the recommendations.
- **Cost and time estimates:** Further action includes an estimate of cost and time. The management should be aware of all costs whenever a new system is installed or changes are made to the existing system.
- **Expected benefits:** The benefits of implementing the new system are highlighted.
- **Appendix:** It includes supporting data and information, list of interviews, documents or any other sources of information.

NOTES

4.2.2 Determining the User's Information Requirements

Shared, complete and accurate information requirements are essential in building computer – based information systems. Unfortunately, determining the information each user needs is particularly difficult task. In fact, it is recognized as one of the most difficult tasks in system development. The Association for Computing Machinery (ACM) Curriculum Committee on Computing Education for Management recognized this by suggesting two distinct job titles for systems developments: "Information Analyst" and "Systems Designer" rather than the more general term "Systems Analyst". The information analyst determines the needs of the user and the information flow that will satisfy those needs. The usual approach is to ask the user what information is currently available and what other information is required. Interaction between the analyst and the user usually leads to an agreement about what information will be provided by the candidate system.

There are several reasons why it is difficult to determine user requirements:

1. Systems requirements change and user requirements must be modified to account for those these changes.
2. The articulation of requirements is difficult, except for experienced users. Functions and processes are not easily described.
3. Heavy user involvement and motivation are difficult. Reinforcement for their work is usually not realized until the implementation phase – too long to wait.
4. The pattern of interaction between users and analysts in designing information requirements is complex.

Users and analysts traditionally do not share a common orientation toward problem definition. For example, in the analyst's view the problem definition must be translatable into a system design expressed quantitatively in terms of outputs, inputs, processes and data structures. This is the best of situations, and within time constraints. In contrast, the user seems to be satisfied with a qualitative definition that specifies the system in generalities. Flexibility is a key consideration. System specifications must change with their needs, as must the system after implementation.

Based on these contrasting views, users who try to define their information requirements with the analyst's views find themselves in a predicament. According to Scharer, they defend themselves by producing strategies that will satisfy the analyst.

1. In the kitchen sink strategy the user throws, everything into the – requirement definition- overstatement of needs such as an overabundance of reports, exception processing and the like. This approach usually reflects the user's lack of experience in the area.
2. The smoking strategy sets up a smoke screen by requesting several system features when only one or two are needed. The extra requests are used as bargaining power. This strategy usually reflects the user's

experience in knowing what he/ she wants. Requests have to be reduced to one that is realistic, manageable, and achievable.

3. The same thing strategy indicates the user's laziness, lack of knowledge, or both. "Give me the same thing but in a better format through the computer" is a typical statement. Here the analyst has little chance of succeeding because only the user can fully discover the real needs and problems.

4.2.3 Case Scenario

Once you have developed an initial set of functional requirements during the Requirements Gathering phase you will have a good understanding of the intended behaviour of the system. Understand what functionality is desired, what constraints are imposed, and what business objectives will be satisfied. However, one shortcoming of a traditional 'laundry-list' of requirements is that they are static and don't concern themselves with the different business processes that need be supported by one feature.

4.2.4 Problem Definition and Initiation

Whether using the classical Software Development Life Cycle (SDLC) or an object-oriented approach, the analyst first defines the problems and objectives of the system. These form the foundation of determining what needs to be accomplished by the system.

A problem definition usually contains some sort of problem statement, summarized in a paragraph or two. This is followed by a series of issues, or major, independent pieces of the problem. The issues are followed by a series of objectives, or goals that match the issues point by point. Issues are the current situation; objectives are the desired situation. The objectives may be very specific or worded using a general statement.

Here are some examples of business questions relating to business objectives:

- What are the purposes of the business?
- Is the business profit or non-profit?
- Does the company plan to grow or expand?
- What is the business's attitude (culture) about technology?
- What is the business's budget for IT?
- Does the business's staff have the expertise?

The last part of the problem definition contains requirements, the things that must be accomplished, along with the possible solutions and the constraints that limit the development of the system. The requirements section may include security, usability, government requirements, and, so on. Constraints often include the word not, indicating a limitation, and may contain budget restrictions or time limitations.

NOTES

The problem definition is produced after completing interviews, observations, and document analysis with the users. The result of gathering this information is a wealth of facts and important opinions in need of summary. The first step in producing the problem definition is to find a number of points that may be included in one issue. Major points can be identified in the interview in a number of ways:

1. Users may identify an issue, topic, or theme that is repeated several times, sometimes by different people in several interviews.
2. Users may communicate the same metaphors, such as saying the business is a journey, war, game, organism, machine, and so on.
3. Users may speak at length on a topic.
4. Users may tell you outright that "This is a major problem."
5. Users may communicate importance by body language or may speak emphatically on an issue.
6. The problem may be the first thing mentioned by the user.

4.2.5 Background Analysis

An analysis of the working environment reveals what questions, factors, problems and solutions to focus upon. The background analysis ensures that planning is based on current information and experience. Knowledge of the working environment is imperative to select suitable and sustainable methods of work.

Background analysis that clarifies the elements of the working environment is necessary before entering into the planning stage of a project. It is a process of gathering necessary information for a sustainable and suitable working approach in a specific environment. When a preliminary project idea is found, it is normally time for a proper background analysis. It is advisable to study the preliminary idea in relation to the national development plans or the local on-going development processes. The specification of the analysis varies from case to case but in general it is good to reserve sufficient time for the background work.

Check Your Progress

1. Give the definition of system planning phase.
2. What is preliminary investigation?
3. State about the technical issues which are generally pointed out during the feasibility stage.
4. Write down the elements of preliminary investigation.
5. State that why it is difficult to determine user requirements.
6. Explain about the Kitchen sink strategy.
7. Write down about the case scenario.
8. What is problem definition?

4.3 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

NOTES

1. The system planning phase usually begins with a formal request known as system request which describes problems or desired changes in an information system or a business process. In many organizations, system planning is an integral part of overall System Development Life Cycle (SDLC). A system request comes from a top manager, a planning team or a department head. The purpose of this phase is to perform a preliminary investigation to evaluate business opportunities or problem. The preliminary or initial investigation is a critical step because the outcome will affect the entire development process.
2. Preliminary investigation is the first phase of the systems development life cycle. As you know, there are limited resources in an organization and only those projects that are critical to its mission, goals and objectives can be undertaken. Hence, the goal of preliminary investigation is simply to identify and select a project for development among all the proposals that are under consideration.
3. Technical issues, which are generally pointed out during the feasibility stage, are as follows:
   - Availability of necessary technology
   - Technical capability to procure data and to use new system equipment
   - Possibility of future upgradation if developed
   - Technical guarantee for accuracy and reliability over and above the factors like ease of access and security of data
4. Following are the elements of preliminary investigation:
   - Introduction
   - Summary of request
   - Findings of the investigation
   - Recommendations
   - Cost and time estimates
   - Expected benefits
   - Appendix
5. There are several reasons why it is difficult to determine user requirements:
   - Systems requirements change and user requirements must be modified to account for those these changes.
   - The articulation of requirements is difficult, except for experienced users. Functions and processes are not easily described.

NOTES

- Heavy user involvement and motivation are difficult. Reinforcement for their work is usually not realized until the implementation phase – too long to wait.
- The pattern of interaction between users and analysts in designing information requirements is complex.

6. In the kitchen sink strategy the user throws, everything into the –requirement definition- overstatement of needs such as an overabundance of reports, exception processing and the like. This approach usually reflects the user's lack of experience in the area.
7. Once you have developed an initial set of functional requirements during the Requirements Gathering phase you will have a good understanding of the intended behaviour of the system. Understand what functionality is desired, what constraints are imposed, and what business objectives will be satisfied. However, one shortcoming of a traditional 'laundry-list' of requirements is that they are static and don't concern themselves with the different business processes that need be supported by one feature.
8. A problem definition usually contains some sort of problem statement, summarized in a paragraph or two. This is followed by a series of issues, or major, independent pieces of the problem. The issues are followed by a series of objectives, or goals that match the issues point by point. Issues are the current situation; objectives are the desired situation. The objectives may be very specific or worded using a general statement.

4.4 SUMMARY

- The system planning phase usually begins with a formal request known as system request which describes problems or desired changes in an information system or a business process.
- A system request comes from a top manager, a planning team or a department head.
- Preliminary investigation is the first phase of the systems development life cycle.
- Organizations may differ in how they identify and select projects for development. Many organizations have special committees to handle the task of preliminary investigation.
- The preliminary investigation phase sets the stage for gathering information about the current problem and the existing information system.
- Many different criteria can be used within an organization for classifying and ranking potential projects.
- The systems analyst—with the help of the stakeholders of the proposed project—collects information about the project. The collected information

has a wide scope and helps understand the size of the project, cost involved and the intended benefits. After the information is analysed, a report is prepared to compare and review all the proposed projects.

- Proper authorization is taken from the relevant authorities to go ahead and investigate as the results will highlight all the details of the organization.
- It is essential for the analyst conducting the investigation to clearly understand the objectives of the project, the benefit of the project, its future scope and its importance to the organization.
- The current system needs to be studied, the benefits and flaws are to be clearly understood and analysed.
- The analyst will conduct interviews with all the people involved in the project, the senior managerial level as well as the users.
- Observing the current system in operation is the best way to fact finding.
- Every organization wants to develop a system that gives it some benefit in terms of financial return and that too at a reasonable cost.
- Another important question is whether the cost of not creating the system is so great that it is advisable to undertake the project.
- A system developed technically and installed properly must be profitable for the organization.
- Financial benefits should be equal to or more than the cost of the system.
- Proposed projects can be beneficial if turned into information systems meeting the requirements of operation for the organization.
- Enough support for the project from the management as well as users. There may be resistance if people are accustomed to the present system and are not ready to think of any alternative system and resist a change.
- If an existing business methodology does not suit the users, a change takes place. This brings about a better and useful system in operation.
- Feasibility studies are carried out by a selected group of people having expertise in IT techniques.
- To be judged feasible, a proposal for the specific project must pass all these tests.
- Shared, complete and accurate information requirements are essential in building computer – based information systems.
- The Association for Computing Machinery (ACM) Curriculum Committee on Computing Education for Management recognized this by suggesting two distinct job titles for systems developments.
- Users and analysts traditionally do not share a common orientation toward problem definition.
- System specifications must change with their needs, as must the system after implementation.

NOTES

NOTES

- In the kitchen sink strategy the user throws, everything into the –requirement definition- overstatement of needs such as an overabundance of reports, exception processing and the like. This approach usually reflects the user's lack of experience in the area.
- The smoking strategy sets up a smoke screen by requesting several system features when only one or two are needed. The extra requests are used as bargaining power. This strategy usually reflects the user's experience in knowing what he/ she wants. Requests have to be reduced to one that is realistic, manageable, and achievable.
- A problem definition usually contains some sort of problem statement, summarized in a paragraph or two. This is followed by a series of issues, or major, independent pieces of the problem. The issues are followed by a series of objectives, or goals that match the issues point by point. Issues are the current situation; objectives are the desired situation. The objectives may be very specific or worded using a general statement.
- The problem definition is produced after completing interviews, observations, and document analysis with the users. The result of gathering this information is a wealth of facts and important opinions in need of summary.
- An analysis of the working environment reveals what questions, factors, problems and solutions to focus upon.
- Background analysis that clarifies the elements of the working environment is necessary before entering into the planning stage of a project.
- When a preliminary project idea is found, it is normally time for a proper background analysis.

4.5 KEY WORDS

- **Preliminary investigation:** It is the first phase of the systems development life cycle that identifies and selects a project for development among all the proposals that are under consideration.
- **Kitchen sink strategy**: In the kitchen sink strategy the user throws everything into the –requirement definition- overstatement of needs, such as an overabundance of reports, exception processing and the like. This approach usually reflects the user's lack of experience in the area.
- **Smoking strategy**: The smoking strategy sets up a smoke screen by requesting several system features when only one or two are needed. The extra requests are used as bargaining power. This strategy usually reflects the user's experience in knowing what he/ she wants. Requests have to be reduced to one that is realistic, manageable, and achievable.
- **Problem definition:** A problem definition usually contains some sort of problem statement, summarized in a paragraph or two.

4.6 SELF ASSESSMENT QUESTIONS AND EXERCISES

NOTES

Short-Answer Questions

1. Define the term preliminary investigation.
2. State about the term investigation.
3. Define the significance of preliminary investigation report.
4. Write about the appendix in preliminary investigation report.
5. Explain the concept of user's information requirements.
6. Elaborate on the kitchen sink strategy.
7. Write about the problem definition.

Long-Answer Questions

1. Describe the system analysis with the help of appropriate examples.
2. Briefly explain preliminary investigation and preliminary investigation report giving examples.
3. Discuss that how the user's information requirements are determined.
4. Elaborate briefly on the producing strategies according to Scharer.
5. Describe the problem definition and initiation with the help of suitable examples.
6. Briefly explain the background analysis with the help of examples.

4.7 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 5 FACT FINDING TECHNIQUES

NOTES

Structure

5.0 Introduction
5.1 Objectives
5.2 Fact Finding Techniques
  5.2.1 Interview
  5.2.2 Questionnaire
  5.2.3 Record Review
  5.2.4 Observation
5.3 Analysing Systems Data
5.4 Feasibility Study
  5.4.1 Steps Involved in Feasibility Analysis
  5.4.2 Study Phase Report
  5.4.3 Oral Presentation
5.5 Answers to Check Your Progress Questions
5.6 Summary
5.7 Key Words
5.8 Self Assessment Questions and Exercises
5.9 Further Readings

5.0 INTRODUCTION

The specific methods analysts use for collecting data about requirements are called fact finding techniques. These include the interview, questionnaire, record review and observation. System analysis is about understanding situations, not solving problems. Effective analysts therefore emphasize investigation and questioning to learn how a system currently operates and to identify the requirements users have for a new or modified one.

Only after analysts fully understand the systems are they able to analyse it and assemble recommendations for systems design.

A feasibility study is carried out to select the best system that meets performance requirements. It comprises of identification, description and evaluation of candidate system and finally selection of the best system for the job. The feasibility study recommends to the management either the most effective system or concludes that the system may not be evolved.

A feasibility study is conducted to select the best system that meets performance requirements. A system required performance is defined by statement of constraints, the identification of specific system objectives, and a description of outputs. The analyst is ready to evaluate the feasibility of the candidate systems to

produce these outputs. Three key considerations are involved in feasibility analysis are economic, technical, and behavioural feasibility.

In this unit, you will study about the interview, questionnaire, record review, observation, systems analysis, analysing systems data, feasibility study, technical, economical and operational, steps in feasibility analysis, feasibility report and oral presentation.

NOTES

5.1 OBJECTIVES

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

- Explain about the various fact finding techniques
- Discuss about the system analysis
- Understand the feasibility study and its types
- Describe the steps in feasibility analysis
- Define the feasibility report and oral presentation

5.2 FACT FINDING TECHNIQUES

The specific methods analysts use for collecting data about requirements are called fact finding techniques. These include the interview, questionnaire, record review and observation. Analysts usually employ more than one of these techniques to help ensure an accurate and comprehensive investigation.

5.2.1 Interview

Analysts use interviews to collect information from individuals or from groups.

The respondents are generally current users of the existing system or potential users of the proposed system. In some instances, the respondents may be managers or employees who provide data for the proposed system or who will be affected by it. Although some analysts prefer the interview to other fact finding techniques, it is not always the best source of application data. Because of the time required for interviewing, other methods must also be used to gather the information needed to conduct an investigation.

It is important to remember that respondents and analysts converse during an interview – the respondents are not being interrogated. Interviews provide analysts with opportunities for gathering information form respondents who have been chosen for their knowledge of the system under study. This method is frequently the best source of qualitative information (opinions, policies, and subjective descriptions of activities and problems). Other fact finding methods are likely to be more useful for collecting quantitative data (numbers, frequencies, and quantities).

NOTES

This method of fact finding can be especially helpful for gathering information from individuals who do not communicate effectively in writing or who may not have the time to complete questionnaires. Interviews allow analysts to discover areas of misunderstanding, unrealistic expectations, and even indications of resistance to the proposed system.

Interviews can be either structured or unstructured. Unstructured interviews, using a question and answer format, are appropriate when analysts want to acquire general information about a system. This format encourages respondents to share their feelings, ideas, and beliefs. Structured interviews use standardized questions in either an open – response or closed – response format. The former allows respondents to answer in their own words; the latter uses a set of prescribed answers. Each approach has advantages and disadvantages.

The success of an interview depends on the skill or the interviewer and on his or her preparation for the interview. Analysts also need to be sensitive to the kinds of difficulties that some respondents create during interviews and know how to deal with potential problems. They need to consider not only the information that is acquired during an interview, but also its significance. It is important to have adequate verification of data through other data collection methods.

5.2.2 Questionnaire

The use of questionnaires allows analysts to collect information about various aspects of a system from a large number of persons. The use of standardized question formats can yield more reliable data than other fact – finding techniques, and the wide distribution ensures greater anonymity for respondents, which can lead to more honest responses. However, this method does not allow analysts to observe the expressions or reactions or respondents. In addition, response may be limited, since completing questionnaires may not have high priority among the respondents.

Analysts often use open – ended questionnaires to learn about feeling, opinions, and general experiences or to explore a process or problem. Closed questionnaires control the frame of reference by presenting respondents with specific responses from which to choose. This format is appropriate for electing factual information.

The high cost of developing and distributing questionnaires demands that analysts carefully consider the objective of the questionnaire and determine what structure will be most useful to the study and most easily understood by the respondents. Questionnaires should also be tested and, if necessary, modified before being printed and distributed.

As with interviewees, recipients, of questionnaires would be selected for the information they can provide. The analysts should ensure that the respondents, background and experiences qualify them to answer the questions.

NOTES

5.2.3 Record Review

Many kinds of records and reports can provide analysts with valuable information about organizations and operations. In record reviews, analysts examine information that has been recorded about the system and user. Record inspection can be performed at the beginning of the study, as an introduction, or later in the study, as a basis for comparing, actual operations with the records indicate should be happening.

Records include written policy manuals, regulations and standard operating procedures used by most organizations and a guide for managers and employees. They do not show what activities are actually occurring, where the decision – making power lies, or how tasks are performed. However, they can help analysts understand the system by familiarizing them with what operations must be supported and with formal relations within the organization.

5.2.4 Observation

Observation allows analysts to gain information they cannot obtain by any other fact finding method. Through observation, analysts can obtain first hand information about how activities are carried out. This method is most useful when analysts need to actually observe how documents are handled, how processes are carried out, observers know what to look for and how to assess the significance of what they observe.

5.3 ANALYSING SYSTEMS DATA

The process of analysis involves a detailed study of the current system leading to specifications of a new system. The approach of analysis is a detailed study of various operations performed by a system and their relationships within and outside the system. During analysis, data are collected on the available files, decision points and transactions handled by the present system. Interviews, onsite observation and questionnaire are the tools used for system analysis. Using the following steps it becomes easy to draw the exact boundary of the new system under consideration:

- Keeping in view the problems and new requirements.
- Workout the pros and cons including new areas of the system.

All procedures, requirements must be analysed and documented in the form of detailed Data Flow Diagrams or DFDs, data dictionary, logical data structures and miniature specifications. System analysis also includes sub-dividing of complex process involving the entire system, identification of data store and manual processes. The features discussed in system analysis are as follows:

- Specification of what the new system is to accomplish based on the user requirements.

NOTES

- Functional hierarchy showing the functions to be performed by the new system and their relationship with each other.
- Function network which are similar to function hierarchy but they highlight those functions which are common to more than one procedure.
- List of attributes of the entities (data items) which need to be held about each entity, i.e., record.

The purpose of system analysis phase is to build a logical model of the new system. The first step is related to requirements modelling where you investigate business processes and documents what the new system must do to satisfy users. Requirements modelling continues the investigation which begins during the system planning phase. To understand the system, you perform information gathering, such as interviews, observations, surveys, document review, observation and sampling. You can use information gathering result produced by the tools to build business models, data and process models, and object models. The deliverable from the system analysis phase is the system requirement document. The system requirements document describes management and user requirements, costs and benefits, and outlines alternative development strategies. The system analysis phase placed in software development life cycle is shown in Figure 5.1.

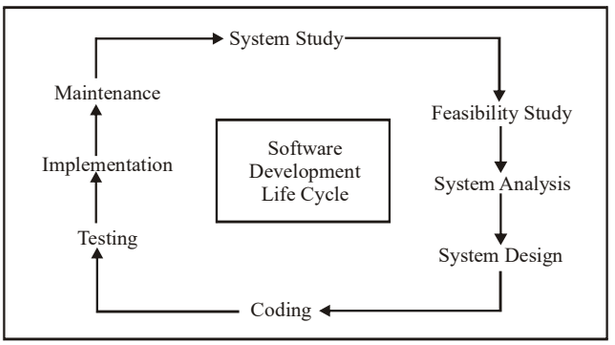


Fig. 5.1 *System Analysis Phase in Software Development Life Cycle*

System analysis is a process in which data are collected and then interpreted to identify the problems within the system. Therefore, the collected information can then be used to recommend improvements in a system. In other words, system analysis includes identification, understanding and examining a system so that the objectives of the system can be achieved. The predetermined objectives of the system analysis are as follows:

- It produces knowledge of the system operation.
- It identifies the user requirements for the proposed system.

The system analysis stage investigates the system operations and determines the solutions to solve the problem. Therefore, system analysis is considered as a logical process.

NOTES

System analysis is really vital in the completion of the development process of a system. It may be possible that a user is aware of the problem but may not have the solution to it. In such a scenario, a system analyst works with the user so that a logical model of the system can be developed. A system analyst of technical background may move too quickly to the program design stage to make a system more physical. This approach is not desirable and should be avoided because it can affect the ultimate success of the system. A system analyst can work with a user to obtain the complete knowledge of a system. For having the complete information about a system, a logical model of the system is developed based on the detailed study of the given system. The detailed study should be done using various modern tools and techniques which includes data flow diagrams, data dictionary and rough descriptions of the relevant algorithms. A set of system requirements for a proposed information system is the final product of the system analysis stage.

Check Your Progress

1. Give the definition of fact finding techniques.
2. What are structured and unstructured interviews?
3. Why the questionnaire is used?
4. State about the system analysis process.
5. Write the predetermined objective of the system analysis.

5.4 FEASIBILITY STUDY

A feasibility study is an evaluation of a proposal to determine the difficulty in carrying out a task. The major purpose of the feasibility study is to evaluate and consider alternative systems solutions, evaluate their feasibility and propose the best possible alternative to the organization. The following are the objectives of the feasibility study:

- To decide on the number and designation of the persons involved in operating the proposed system in the organization.
- To determine the probable achievements of the proposed system.
- To estimate the benefits of the new system.
- The organizational changes needed for its successful implementation.
- What will be the estimated cost of the system.

There are several components based on which a feasibility study is evaluated. These components or types are as follows:

- Economic
- Operational

NOTES

- Technical
- Marketing
- Cultural
- Legal
- Schedule

Economic Feasibility Study

For any project to go ahead, it is essential during the early development that the economic feasibility study is carried out. It forms the essential component of any system development process in any business. Before financial resources are made available to the proposed project, economic feasibility studies should be carried out to assess the cost, benefits and liabilities of the project. Economic feasibility enhances the stakeholder's confidence and credibility of the project.

The questions that a feasibility study should be able to answer are:

- In the given constraints, is the project possible?
- What are the benefits from the proposed system?
  - o What are the tangible and intangible benefits?
  - o What is the quantity of the benefits?
  - o Do the benefits outweigh the costs?
- What are the developmental and operational costs?

The economic feasibility study is followed by the detailed feasibility report, which contains the following items:

- An introduction with an outline of the proposal.
- A broad data flow diagram of the existing system.
- A modified data flow diagram.
- All possible alternative solutions.
- Merits and demerits of each solution.
- Any new equipment to be installed or any new resources required for proposed system.
- Expected benefits of the proposed system.
- Cost of the system.
- New procedures to be implemented and environment of the persons in operating new system.
- Any anticipation problem in implementation.

Based on the information in the economic feasibility report, a case is put before the stakeholders of the selected project to be implemented. This case will

provide the requisite details for the selection of a particular project over others. An Economic Feasibility Study (EFS) does the following things:

- Makes a business case.
- Prepares analytical worksheets and other necessary supporting documentation.
- Summarizes the strengths, weaknesses, opportunities and threats.
- Estimates the costs involved as well as the benefits of the proposed project.
- Estimates on the returns and costs in the long term as well as the short term.

NOTES

Factors like time value of money, quality of available data, investment and operating costs, suitability of assumptions, risk and uncertainty are essential to the Economic Feasibility Study (EFS). The EFS should be able to show the net benefit generated by result of the proposed system with direct and indirect benefits and costs to be stake holders.

Operational Feasibility Study

Operational feasibility study (OFS) determines how well the solution of problems or a specific solution will work in an organization and also how people feel about the solution provided. The objectives of the operational feasibility are as follows:

- It determines and explores the urgency of the problem and acceptability of the solution.
- The OFS will deal with social issues which will arise because of the new solution.
- It will also look into the following internal issues:
  - o Internal organizational conflicts and policies.
  - o Manpower problems.
  - o Objections by labour.
  - o Managerial resistance.
- The OFS will also study the following external issues:
  - o Legal aspects of implementing the new system.
  - o Government regulations that govern the new solution/system.
- OFS will also ensure the system is actually used, if developed.
- It will also deal with people-oriented issues.

The PIECES Framework: A framework called PIECES has been formulated for identifying operational problems and their urgency. It stands for the following:

- Performance: Does the current system provide adequate throughput and response time?

NOTES

- Information: Does the current system provide end users and managers with timely, pertinent, accurate, usefull and formatted information?
- Economy: Does the current system provide cost-effective information services to the business? Could there be a reduction in costs and/or an increase in benefits?
- Control: Does the current system offer effective controls to protect against fraud and guarantee accuracy and security of data and information?
- Efficiency: Does the current system make maximum use of available resources, including people, time and flow of forms?
- Services: Does the current system provide reliable service? Is it flexible and expandable?

Technical Feasibility Study

In technical feasibility, the following issues are identified and analysed:

- Can the current equipment be utilized for the proposed system?
- Is the project feasible within the limits of current technology?
- If the proposed system is developed, can it be further upgraded or expanded?
- Does the technology exist at all?
- Is it available within given resource constraints, i.e., budget, schedule, etc.?
- Is it also concerned with the aspects like software and hardware capabilities, the volume and speed of inputs and outputs?
- Are the system and data security issues also considered while doing technical feasibility?
- Are the user friendliness of the proposed system, information accuracy and reliability also studied during technical feasibility?

The answers of the preceding questions provide important inputs and specifications for the proposed system and are very useful to derive complete information for estimating the project cost and developing schedule.

Marketing Feasibility Study

The objectives of the marketing feasibility study are as follows:

- To identify and determine the suitability of the proposed system for profitable development.
- To define optimal products and amenities in accordance with projected market demand.
- To project sales absorption and annual revenues and savings from development of the proposed system.

NOTES

The marketing feasibility study basically analyses the market potential of the product or services, which will be produced as a result of implementation of the proposed system. Research findings are analysed with reference to market competition and preliminary development plans to define the marketing feasibility in terms of the following feasibility factors:

- Potential market share
- Marketing constraints
- Opportunities

These feasibility factors provide the grounds for definitive recommendations on system development strategy, cost effectiveness and sales strategy for the product/services, which are produced as a result of the implementation of the proposed project.

Cultural Feasibility Study

The cultural feasibility study deals with aspects like the impact of the proposed project on the local culture. It also determines if an alternative project would better suit the proposed project and the environmental factors and cultural traditions and values. This study checks whether the prevalent culture would accept or refuse the project. For example, before setting up a mining company that can damage the environment, all the cultural and environmental factors should be considered.

Legal Feasibility Study

The legal feasibility study determines if the proposed study is within the local and state legal frameworks. Care is taken that the proposed system is designed and developed in a manner that it follows and abides by the law of the land. Usually large organizations have legal advisors to carry out the legal feasibility study.

Schedule Feasibility Study

The schedule feasibility study broadly explores the following factors:

- Is it possible to build the solution in time to be beneficial?
- Are there constraints in the project schedule and can they be met?
- How much is the time available to build the new system?
- When can it be built?
- Does it interfere with normal business operation?
- How many resources are required and what are the factors on which it is dependent?
- Are the contingency and mitigation plans stated?

Steps in Feasibility Study

Preliminary investigations examine the feasibility of the software project under study. In the preliminary investigation and feasibility study, the analyst collects

NOTES

information about what the end user expects from the new system. The feasibility study provides enough information so that an alternative can be selected for the development of the new system. Feasibility study helps management to make a decision about whether the system under study is feasible. Since it is not possible to implement all proposed projects, feasibility study should be completed within a short period of time. Some possible objectives for requesting projects are as follows:

- Improving accuracy of data input (reducing errors).
- Eliminating unnecessary reports (duplicate reports, etc.).
- Integrating business subsystems.
- Upgrading customer services.
- Shortening data processing time.

The foundation for the project effort, in totality, is set by a project team at the very beginning of the Software Development Life Cycle (SDLC). While initiating a project, it is very important to justify it, i.e., to determine whether it should be built or not. However, the unfortunate part is that it is never justified properly. It will be beneficial if the non-feasible projects are discarded at this stage. This will prevent unnecessary investments from being made. The primary objective of the feasibility or justify stage is to define the best solution for implementing the project and to justify its need and appropriateness.

5.4.1 Steps Involved in Feasibility Analysis

Figure 5.2 represents the justify stage process pattern. There are several activities involved in justifying a project. Creation of a feasibility study involves the following steps:

- Determination of the alternatives of implementation
- Assessment of each alternative economic feasibility
- Assessment of each alternative technical feasibility
- Assessment of the operational feasibility of each option
- Selection of an alternative
- Identification of potential risks

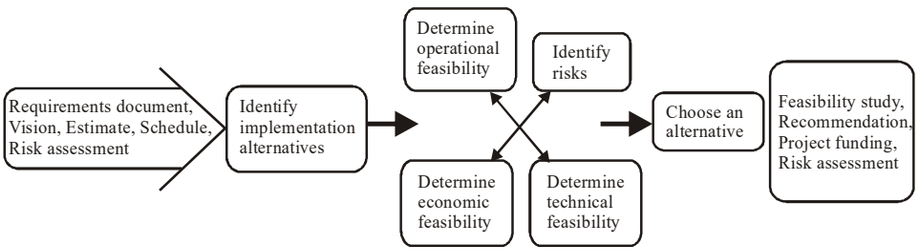


Fig. 5.2 *Justify Stage Process Pattern*

NOTES

The first phase of the feasibility study is to identify the possible implementation alternatives for a project. Contrary to what is popularly believed, there are many options available for implementation of a project, include the following:

- Doing nothing.
- Implementing using a variety of technologies.
- Buying a similar system or outsourcing its development.

What is important is to identify many alternatives for a project that are viable. This facilitates their easy comparison and assessment, which in turn, facilitates the selection of the best option for an organization.

Technical, economical and operational areas of the feasibility study must be addressed for the further development of the system. The following steps are involved in feasibility analysis:

- Decision regarding feasibility of the project should be made by the management and not the systems analyst.
- Feasibility data gathered by experts and professionals forms the basis of these decisions and it is presented by the analyst.
- The systems analyst must address the three areas technical, economic and operational feasibility in the preliminary study of the system under development.
- The requested system project must be studied quickly.
- The study should result in a reasonable information output.
- Following the preliminary study, the projects that successfully fulfil the three feasibility criteria should be selected for a detailed system study.
- At this juncture, the systems analyst suggests the most suitable system project to the management.
- At this point, a commitment from the management implies that only the systems studied can proceed.

Following steps are involved in conducting the feasibility analysis:

- **Step 1:** Form a project team and appoint a project leader.
- **Step 2:** Develop system flowcharts.
- **Step 3:** Enumerate candidate solutions.
- **Step 4:** Identify, analyse and describe the features of candidate systems.
- **Step 5:** Evaluate performance and cost effectiveness of each candidate solutions.
- **Step 6:** Choose the best solution.
- **Step 7:** Prepare a formal feasibility report and submit it to the management.

NOTES

Types of Feasibility

Feasibility study is an important outcome of the preliminary investigation which determines whether the system, for which the request has been made, is feasible or not. Feasibility study has three aspects as follows:

- ***Technical feasibility.*** This is related to the technicality of the project. This ponders over the question whether the work for the project can be done with the existing equipment, the current software package and the available human resources. In case, a new technology is required, it is essential to know whether it is possible to develop it further.
- ***Economic feasibility.*** Every organization wants to develop a system that gives it some benefit in terms of financial return and that too at a reasonable cost. So economic feasibility addresses issues related to the creation of the system to make the costs affordable and acceptable. Another important question is whether the cost of not creating the system is so great that it is advisable to undertake the project.

  A system developed technically and installed properly must be profitable for the organization. Financial benefits should be equal to or more than the cost of the system.
- ***Operational feasibility.*** This feature of feasibility study is related to its operational aspect wherein the working of the hardware, the software and the human resource is to be taken into account. So, the question of whether the system will work if it is developed and implemented comes naturally. Another aspect related to human resource is whether there will be resistance from users who may not agree to accept the changes.

5.4.2 Study Phase Report

Study phase report refers to the process of executing plan with the help of lower and upper level planning on the basis of report.

Once the project is under way, a micro level plan is required to be drawn up in greater detail for each activity as it nears the due date. Planning for each activity in detail is done during the later stages as we get more information about the activity when we near the start of the activity. In some cases, provisional plans are made earlier but they are refined just before the activity commences.

After the completion of software project planning, the task of the software project manager is to make a proper documentation of the software project planning results and this document is termed as Software Project Management Plan (SPMP). Institute of Electrical and Electronics Engineers (IEEE 1987) provides a framework (IEEE 1058.1) for the SPMP document which is outlined as follows:

1. Introduction
   - Project overview
   - Project deliverable

- Evolution of software project management plan
- Definitions and acronyms
- Reference materials

2. Organization of project
   - Process model
   - Organizational structure
   - Project responsibilities
   - Organizational boundaries and interfaces
3. Managerial process
   - Management objectives and priorities
   - Assumptions, dependencies and constraints
   - Risk management
   - Monitoring and controlling mechanisms
   - Staffing plan
4. Technical process
   - Methods, tools and techniques
   - Software documentation
   - Project support functions
5. Work packages, schedule and budget
   - Work packages
   - Dependencies
   - Resource requirements
   - Budget and resource allocation
   - Schedule
6. Additional components
7. Index
8. Appendices

5.4.3 Oral Presentation

The feasibility report is a good written presentation documenting the activities involving the candidate system. The pivotal step, however, is selling the proposed change. Invariably the project leader or analyst is expected to give an oral presentation to the end user. Although it is not as polished as the written report, the oral presentation has several important objectives. The most critical requirements for the analyst who gives the oral presentation are: (1) communication skills and knowledge about the candidate system that can be translated into language

NOTES

understandable to the user and (2) the ability to answer questions, clarify issues, maintain credibility and pick up on any new ideas or suggestions.

The substance and form of the presentation depend largely on the purposes sought. The presentation may aim at informing, confirming, or persuading.

1. **Informing:** This simply means communicating the decisions already reached on system recommendations and the resulting action plans to those who will participate in the implementation. No detailed findings or conclusions are included.
2. **Confirming:** A presentation with this purpose verifies facts and recommendations already discussed and agreed upon. Unlike the persuading approach, no supportive evidence is presented to sell the proposed change, nor is there elaborate reasoning behind recommendations and conclusions. Although the presentation is not detailed, it should be complete. Confirming is itself part of the process of securing approval. It should reaffirm the benefits of the candidate system and provide a clear statement of results to be achieved.

Oral Presentation – Suggested Outline

1. Introduction
   - Introduce self
   - Introduce topic.
   - Briefly describe current system.
     - o Explain why it is not solving the problem
     - o Highlight user dissatisfaction with it.
     - o Briefly describe scope, objectives and recommendation of the proposed system.
2. Body of presentation.
   - Highlight weaknesses of current system.
   - Describe proposed system. How is it going to solve the problem?
   - Sell proposed system.
     - o Specify savings and benefits, costs and expenses.
     - o Use visual aids to justify project and explain system.
   - Summarize implementation plan and schedule.
   - Review human resources requirements to install system.

3. Conclusion.
   - Summarize proposal
   - Restate recommendations and objectives of proposal.
   - Summarize benefits and savings.
   - Ask for top-level management support. Solicit go-ahead for project.
4. Discussion period- Answer questions convincingly.

3. Persuading: This is a presentation pitched toward selling ideas attempts to convince executives to take action on recommendations for implementing a candidate system.

NOTES

Regardless of the purpose sought, the effectiveness of the oral presentation depends on how successful the project team has been in gaining the confidence of frontline personnel during the initial investigation. How the recommendations are presented also has an impact. Here are some pointers on how to give an oral presentation:

1. Rehearse and test your ideas before the presentation. Show that you are in command. Appear relaxed.
2. Final recommendations are more easily accepted if they are presented as ideas for discussion, even though they seem to be settled and final.
3. The presentation should be brief, factual and interesting Clarity and persuasiveness are critical. Skill is needed to generate enthusiasm and interest throughout the presentation.
4. Use good organization. Distribute relevant material to the user and other parties in advance.
5. Visual aids (graphs, charts) are effective if they are simple, meaningful and imaginative. An effective graph should teach or tell what is to be communicated.
6. Most important, present the report in an appropriate physical environment where the acoustics, seating pattern, visual aid technology and refreshments are available.

The most important element to consider is the length of the presentation. The duration often depends on the complexity of the project, the interest of the user group and the competence of the project team. A study that has company wide applications and took months to complete would require hours or longer to present. The user group that was involved at the outset would likely permit a lengthy presentation, although familiarity with the project often dictates a brief presentation. Unfortunately, many oral presentations tend to be a rehash of the written document with little flare or excitement. Also, when the analyst or the project leader has a good reputation and success record from previous projects, the end user may request only a brief presentation.

NOTES

Check Your Progress

6. Why feasibility study is done?
7. Explain the objective the economic feasibility study.
8. Give the details for the term PIECES.
9. State about the cultural feasibility study.
10. Explain the steps in feasibility study.
11. What is SPMP?
12. Write the most critical requirements for the oral presentation.

5.5 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. The specific methods analysts use for collecting data about requirements are called fact finding techniques. These include the interview, questionnaire, record review and observation. Analysts usually employ more than one of these techniques to help ensure an accurate and comprehensive investigation.
2. Unstructured interviews, using a question and answer format, are appropriate when analysts want to acquire general information about a system. This format encourages respondents to share their feelings, ideas, and beliefs. Structured interviews use standardized questions in either an open – response or closed – response format. The former allows respondents to answer in their own words; the latter uses a set of prescribed answers.
3. The use of questionnaires allows analysts to collect information about various aspects of a system from a large number of persons. The use of standardized question formats can yield more reliable data than other fact – finding techniques, and the wide distribution ensures greater anonymity for respondents, which can lead to more honest responses.
4. The process of analysis involves a detailed study of the current system leading to specifications of a new system. The approach of analysis is a detailed study of various operations performed by a system and their relationships within and outside the system. During analysis, data are collected on the available files, decision points and transactions handled by the present system. Interviews, onsite observation and questionnaire are the tools used for system analysis.
5. The predetermined objectives of the system analysis are as follows:
   - It produces knowledge of the system operation.
   - It identifies the user requirements for the proposed system.

NOTES

6. A feasibility study is an evaluation of a proposal to determine the difficulty in carrying out a task. The major purpose of the feasibility study is to evaluate and consider alternative systems solutions, evaluate their feasibility and propose the best possible alternative to the organization.
7. For any project to go ahead, it is essential during the early development that the economic feasibility study is carried out. It forms the essential component of any system development process in any business.
8. The PIECES Framework: A framework called PIECES has been formulated for identifying operational problems and their urgency. It stands for the following:
   - Performance: Does the current system provide adequate throughput and response time?
   - Information: Does the current system provide end users and managers with timely, pertinent, accurate, usefull and formatted information?
   - Economy: Does the current system provide cost-effective information services to the business? Could there be a reduction in costs and/or an increase in benefits?
   - Control: Does the current system offer effective controls to protect against fraud and guarantee accuracy and security of data and information?
   - Efficiency: Does the current system make maximum use of available resources, including people, time and flow of forms?
   - Services: Does the current system provide reliable service? Is it flexible and expandable?
9. The cultural feasibility study deals with aspects like the impact of the proposed project on the local culture. It also determines if an alternative project would better suit the proposed project and the environmental factors and cultural traditions and values. This study checks whether the prevalent culture would accept or refuse the project.
10. Creation of a feasibility study involves the following steps:
    - Determination of the alternatives of implementation
    - Assessment of each alternative economic feasibility
    - Assessment of each alternative technical feasibility
    - Assessment of the operational feasibility of each option
    - Selection of an alternative
    - Identification of potential risks
11. After the completion of software project planning, the task of the software project manager is to make a proper documentation of the software project planning results and this document is termed as Software Project Management Plan (SPMP).

NOTES

12. The most critical requirements for the analyst who gives the oral presentation are:
    - communication skills and knowledge about the candidate system that can be translated into language understandable to the user and
    - the ability to answer questions, clarify issues, maintain credibility and pick up on any new ideas or suggestions.

5.6 SUMMARY

- The specific methods analysts use for collecting data about requirements are called fact finding techniques.
- These include the interview, questionnaire, record review and observation.
- Analysts usually employ more than one of these techniques to help ensure an accurate and comprehensive investigation.
- Analysts use interviews to collect information from individuals or from groups.
- The respondents are generally current users of the existing system or potential users of the proposed system.
- It is important to remember that respondents and analysts converse during an interview – the respondents are not being interrogated.
- Interviews provide analysts with opportunities for gathering information form respondents who have been chosen for their knowledge of the system under study.

  Interviews allow analysts to discover areas of misunderstanding, unrealistic expectations, and even indications of resistance to the proposed system.
- Unstructured interviews, using a question and answer format, are appropriate when analysts want to acquire general information about a system. This format encourages respondents to share their feelings, ideas, and beliefs. Structured interviews use standardized questions in either an open – response or closed – response format. The former allows respondents to answer in their own words; the latter uses a set of prescribed answers.
- The success of an interview depends on the skill or the interviewer and on his or her preparation for the interview.
- It is important to have adequate verification of data through other data collection methods.
- The use of questionnaires allows analysts to collect information about various aspects of a system from a large number of persons. The use of standardized question formats can yield more reliable data than other fact – finding techniques, and the wide distribution ensures greater anonymity for respondents, which can lead to more honest responses.

NOTES

- The high cost of developing and distributing questionnaires demands that analysts carefully consider the objective of the questionnaire and determine what structure will be most useful to the study and most easily understood by the respondents. Questionnaires should also be tested and, if necessary, modified before being printed and distributed.
- Many kinds of records and reports can provide analysts with valuable information about organizations and operations.
- In record reviews, analysts examine information that has been recorded about the system and user. Record inspection can be performed at the beginning of the study, as an introduction, or later in the study, as a basis for comparing, actual operations with the records indicate should be happening.
- Records include written policy manuals, regulations and standard operating procedures used by most organizations and a guide for managers and employees.
- Observation allows analysts to gain information they cannot obtain by any other fact finding method. Through observation, analysts can obtain first hand information about how activities are carried out.
- This method is most useful when analysts need to actually observe how documents are handled, how processes are carried out, observers know what to look for and how to assess the significance of what they observe.
- The process of analysis involves a detailed study of the current system leading to specifications of a new system. The approach of analysis is a detailed study of various operations performed by a system and their relationships within and outside the system. During analysis, data are collected on the available files, decision points and transactions handled by the present system. Interviews, onsite observation and questionnaire are the tools used for system analysis.
- All procedures, requirements must be analysed and documented in the form of detailed Data Flow Diagrams or DFDs, data dictionary, logical data structures and miniature specifications.
- System analysis also includes sub-dividing of complex process involving the entire system, identification of data store and manual processes.
- The purpose of system analysis phase is to build a logical model of the new system.
- The first step is related to requirements modelling where you investigate business processes and documents what the new system must do to satisfy users.
- To understand the system, you perform information gathering, such as interviews, observations, surveys, document review, observation and sampling. You can use information gathering result produced by the tools to build business models, data and process models, and object models.

NOTES

- The system requirements document describes management and user requirements, costs and benefits, and outlines alternative development strategies.
- System analysis is a process in which data are collected and then interpreted to identify the problems within the system.
- The system analysis stage investigates the system operations and determines the solutions to solve the problem.
- System analysis is really vital in the completion of the development process of a system.
- A system analyst can work with a user to obtain the complete knowledge of a system.
- A set of system requirements for a proposed information system is the final product of the system analysis stage.
- A feasibility study is an evaluation of a proposal to determine the difficulty in carrying out a task. The major purpose of the feasibility study is to evaluate and consider alternative systems solutions, evaluate their feasibility and propose the best possible alternative to the organization.
- For any project to go ahead, it is essential during the early development that the economic feasibility study is carried out. It forms the essential component of any system development process in any business.
- Based on the information in the economic feasibility report, a case is put before the stakeholders of the selected project to be implemented. This case will provide the requisite details for the selection of a particular project over others.
- Factors like time value of money, quality of available data, investment and operating costs, suitability of assumptions, risk and uncertainty are essential to the Economic Feasibility Study (EFS).
- Operational feasibility study (OFS) determines how well the solution of problems or a specific solution will work in an organization and also how people feel about the solution provided.
- The marketing feasibility study basically analyses the market potential of the product or services, which will be produced as a result of implementation of the proposed system.
- These feasibility factors provide the grounds for definitive recommendations on system development strategy, cost effectiveness and sales strategy for the product/services, which are produced as a result of the implementation of the proposed project.
- The cultural feasibility study deals with aspects like the impact of the proposed project on the local culture. It also determines if an alternative project would

better suit the proposed project and the environmental factors and cultural traditions and values. This study checks whether the prevalent culture would accept or refuse the project.

- The legal feasibility study determines if the proposed study is within the local and state legal frameworks.
- Preliminary investigations examine the feasibility of the software project under study. In the preliminary investigation and feasibility study, the analyst collects information about what the end user expects from the new system.
- Feasibility study helps management to make a decision about whether the system under study is feasible. Since it is not possible to implement all proposed projects, feasibility study should be completed within a short period of time.
- The foundation for the project effort, in totality, is set by a project team at the very beginning of the Software Development Life Cycle (SDLC). While initiating a project, it is very important to justify it, i.e., to determine whether it should be built or not.
- The first phase of the feasibility study is to identify the possible implementation alternatives for a project.
- Technical, economical and operational areas of the feasibility study must be addressed for the further development of the system.
- Decision regarding feasibility of the project should be made by the management and not the systems analyst.
- The systems analyst must address the three areas technical, economic and operational feasibility in the preliminary study of the system under development.
- The most critical requirements for the analyst who gives the oral presentation are: (1) communication skills and knowledge about the candidate system that can be translated into language understandable to the user and (2) the ability to answer questions, clarify issues, maintain credibility and pick up on any new ideas or suggestions.
- The substance and form of the presentation depend largely on the purposes sought. The presentation may aim at informing, confirming, or persuading.
- A presentation with this purpose verifies facts and recommendations already discussed and agreed upon. Unlike the persuading approach, no supportive evidence is presented to sell the proposed change, nor is there elaborate reasoning behind recommendations and conclusions.
- The presentation should be brief, factual and interesting Clarity and persuasiveness are critical. Skill is needed to generate enthusiasm and interest throughout the presentation.

NOTES

- The most important element to consider is the length of the presentation. The duration often depends on the complexity of the project, the interest of the user group and the competence of the project team.
- After the completion of software project planning, the task of the software project manager is to make a proper documentation of the software project planning results and this document is termed as Software Project Management Plan (SPMP).

5.7 KEY WORDS

- **Interview:** Analysts use interviews to collect information from individuals or from groups. The respondents are generally current users of the existing system or potential users of the proposed system.
- **Questionnaire:** The use of questionnaires allows analysts to collect information about various aspects of a system from a large number of persons.
- **Record review:** In record reviews, analysts examine information that has been recorded about the system and user.
- **Observation:** Observation allows analysts to gain information they cannot obtain by any other fact finding method.
- **Study phase report:** It refers to the process of executing plan with the help of lower and upper level planning on the basis of report.
- **Informing:** This simply means communicating the decisions already reached on system recommendations and the resulting action plans to those who will participate in the implementation.

5.8 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. Differentiate between the structured and unstructured interviews.
2. Define the term record review.
3. State about the observation.
4. Define the term analysing systems data.
5. Write the features of system analysis.
6. Explain the concept of operational feasibility study.
7. Elaborate on the term marketing feasibility study.
8. Give the definition of legal feasibility study.
9. Write the types of feasibility.

10. Name the organization which provides a framework (IEEE 1058.1) for the SPMP document.
11. Define the significant of persuading.

Long-Answer Questions

1. Describe the fact finding techniques giving examples.
2. Briefly explain the system analysis phase in software development life cycle with the help of diagram.
3. Discuss about the several components for feasibility study.
4. Describe the steps of feasibility analysis and justify stage process pattern with the help of diagram.
5. Briefly explain the types of feasibility.
6. Discuss the concept of framework for the SPMP document provided by IEEE.
7. Discuss about the oral presentation giving details.

NOTES

5.9 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 6 SYSTEM COST AND BENEFITS

NOTES

Structure

6.0 Introduction
6.1 Objectives
6.2 Cost-Benefit Analysis
  6.2.1 Principles of Cost-Benefit Analysis
  6.2.2 Categories of Costs and Benefits
  6.2.3 Cost and Benefit Evaluation Methods
6.3 Tools for Structured Analysis
  6.3.1 Types of Basic Charts
  6.3.2 Decision Tables
  6.3.3 Decision Trees
  6.3.4 Structured English
  6.3.5 Data Flow Diagrams
6.4 Answers to Check Your Progress Questions
6.5 Summary
6.6 Key Words
6.7 Self Assessment Questions and Exercises
6.8 Further Readings

6.0 INTRODUCTION

Each problem has generally more than one solution. Each such approach has costs & benefits that are compared with those of other approaches before a final recommendation is made. The result is a project proposal. The findings of the analysis are summarized and the design is recommended.

In developing cost estimates for a system, we need to consider several cost elements. Among them are hardware, personnel, facility, operating and supply costs. There is a difference between expenditure and investment. We spend to get what we need, but we invest to realize a return on the investment. Building a computer – based system is an investment. Costs are incurred throughout its life cycle. Benefits are realized in the form of reduced operating costs, improved corporate image, staff efficiency, or revenues. To what extent benefits outweigh costs is the function of cost /benefit analysis. Cost and benefit analysis is a procedure that gives a picture of the various costs, benefits and rules associated with a system.

Certain costs and benefits are more easily identifiable than others. For example, direct costs, such as the price of a hard disk, are easily identified form company invoice payments or cancelled checks. Direct benefits often relate one-to-one to direct costs, especially savings from reducing costs in the activity in question. Other direct costs and benefits, however, may not be well defined, since they represent estimated costs or benefits that have some uncertainty. An example

of such costs is reserve for bad debt. It is a discerned real cost, although its exact amount is not so immediate.

In this unit, you will study about the categories of cost, cost-benefit analysis, break even, present value, pay back and cash flow, analysis tools**,** data flow concept data flow diagram, data dictionary, decision table, decision tree, structured english.

NOTES

6.1 OBJECTIVES

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

- Know about the Cost-Benefits Analysis (CBA)
- Understand the principles of cost-benefit analysis
- Analyse and evaluate the process of cost-benefit analysis
- Learn about the cost-benefit analysis tools

6.2 COST-BENEFIT ANALYSIS

A Cost-Benefit Analysis (CBA) is done to determine how well or how poorly a planned action will turn out. Although a CBA can be used for almost anything, it is most commonly done on financial questions. Since the CBA relies on the addition of positive factors and the subtraction of negative ones to determine a net result, it is also known as running the numbers. It is a powerful, widely used and relatively easy tool for deciding whether to make a change.

The benefits of a CBA are it finds, quantifies and adds all the positive factors. Then, it identifies, quantifies and subtracts all the negatives—the costs. The difference between the two indicates whether the planned action is advisable. The real trick to doing a cost-benefit analysis well is making sure you include all the costs and all the benefits and properly quantify them. This is the technique business-savvy organizations use to identify the best choice. A credible CBA provides hard facts that answer three fundamental questions:

1. What must be accomplished to succeed?
2. What are the pros and cons of the viable choices?
3. Which choice is the best?

6.2.1 Principles of Cost-Benefit Analysis

In practical terms, problems arise in making a cost-benefit analysis due to the fact that quantification of various components of costs is not easy in all cases. Some underlying basic principles of CBA are as follows:

- **Need for a common unit**

All aspects of a project, whether positive and negative, should be expressed in a common unit to decide the viability of any project, and money is the most convenient

NOTES

common unit. Thus, costs and benefits are measured as equivalent money value. Something that gives benefits may not be directly converted as money. However, if money is received in place of benefits, it may be considered as a benefit. For example, if there is a project that provides a free monthly visit to a doctor in a particular region as a scheme for middle-aged persons or senior citizens, the value in lieu of that facility is the money value equivalent to that benefit.

Apart from expressing the benefits and costs of a project as equivalent money value, the time at which this money value is provided is also important. An amount five years from now is not the same as an amount available at the present moment. One rupee, if invested now, to earn an interest for five years, will be worth more than a rupee in five years.

When the monetary value of benefits at a particular date in the future is multiplied to the present value, this results in a discounted present value of that benefit. This is also applicable to costs.

- **Valuations in CBA**

Cost-benefit analysis valuation should represent the valuation of consumers or producers, as per their actual behaviour. Such valuations are required to be reflected in the preferences revealed by the choices made. For example, if frequent improvements are brought about in transportation, it will save time. Here, you may wonder as to how the money value of the time saved is to be measured. This money value, which is equivalent to the time saved, will be different for different people. The value is not just what transportation planners think about the worth of the time saved. It is not even what others say about its worth. The public reveals the value of their time, keeping in mind the choices involving the tradeoffs between time and money. This value is the money equivalent to the benefit. If a person is very busy, to the extent that he earns ₹ 10,000 in an hour, then this amount should be considered the value of his/her time. For example, take the case of a parking space. If people are given a place for parking close to their workplace and a nominal fee of ₹ 5 is charged, and if another choice is given for parking, for free, but at a greater distance in which one has to spend an additional 5 minutes in walking, then most people would agree to spend ₹ 5 and save time and effort. This is because, for them, time is more precious than Re 1 per minute. If they ignore other things and show indifference between the two alternatives, then the value of their time is exactly Re 1 per minute.

- **Benefits as per market choices**

While purchasing things at market price, consumers feel that the money they are spending is worth what they are spending on. Consumers have a tendency to increase consumption of a commodity till the marginal benefit it provides equals the marginal cost. The relationship between consumption and market price is called the demand schedule. The demand schedule provides information on the marginal benefit. This schedule also guides the monetary expenditure for an increased consumption of that commodity.

6.2.2 Categories of Costs and Benefits

NOTES

Expenditure made on installing a system refers to the costs associated with the system. Extra expenditure incurred due to the inefficient working of a system is also a cost. Thus, cost towards not developing an efficient system is also a cost. Benefits are advantages converted to money equivalent, gained by installing and using the new system. Such costs and benefits are of different kinds and many times the evaluation in terms of money becomes very difficult. This is one of the trickiest parts of a cost-benefit analysis.

Costs and benefits may be categorized as follows:

- Tangible or intangible
- Fixed or variable
- Direct or indirect

Tangible or Intangible Costs and Benefits

Tangible costs and benefits are those that can be measured easily. The amount of cash dispensed for any specific item or service is known as its tangible cost. Such costs are clearly known and are estimated accurately. The same applies to tangible benefits.

There are intangible costs that exist but are not visible immediately. If management is sure that certain activities, such as timely procurement of raw material, are essential and that avoiding these activities may result in losses, then these are called intangible costs. There are benefits that are known to exist, but quite difficult to exactly measure. These are referred to as intangible benefits. The estimate can only be approximated, it cannot be exact. For example, providing some welfare measures undertaken by a management to motivate and boost the morale of employees cannot be exactly measured in terms of money values. One can relate it to the increased output and enhanced quality of the product. Again, it becomes subjective and subject to variation from one management to another. This too is an intangible benefit.

Thus, benefits are more difficult to measure than costs. For example, the suppliers quote the cost of purchasing a terminal but they cannot tell them of the specific benefits or financial advantages they gain by using it in a system. Tangible benefits like doing jobs in fewer hours or producing error-free output data are quantifiable. Intangible benefits that result in the form of more satisfied customers or improvement in corporate image (as user of modern and efficient systems) cannot be easily quantified. However, in the evaluation process, both tangible and intangible costs and benefits have to be taken into consideration. If a project is evaluated only on an intangible basis, benefits are far more than costs and then such projects would be called cost effective. However, if intangible costs and benefits are included and the total costs (tangible plus intangible) are found to be more than the benefits, the project might be declared as an undesirable investment.

Thus, systems projects should be evaluated by taking into consideration both tangible and intangible benefits.

Direct or Indirect Costs and Benefits

NOTES

Costs, directly associated with a system, are known as direct costs. For example, purchase of a pen drive for ₹ 500 is a direct cost as the association of the pen drive with the money spent is direct. Similarly, you can measure direct benefits.

Direct benefits can also be attributed to a given project. For example, if the installation of a new system is able to process 40 per cent more transactions per day, it can be taken as a direct benefit. Similarly, if opening a new counter increases sale by 25 per cent, this too is taken as a direct benefit.

Indirect costs, on the other hand, are not directly associated with a particular activity in the system. They are classified as overhead expenses. For example, to install a computer system, you need some space. Thus, the cost of space, the cost of maintenance of a computer centre, the cost of lighting and air conditioning are all tangible costs. But it is not easy to quantify the proportion attributable to any specific activity.

Indirect benefits are like by-products. For example, a system installed for tracking sales calls to customers is an indirect benefit in marketing as it gives additional information about the market and competition. Such information on competition is an indirect benefit but its evaluation in terms of money cannot be done exactly.

Fixed or Variable Costs and Benefits

Apart from the direct/indirect and tangible/intangible costs and benefits, there are some costs and benefits that remain constant, independent of how a system is used.

Fixed costs are sunk costs and are constant. They are one-time costs. For example, investment on installation of equipment for a computer system is a fixed cost, which is constant whether in use or not. Similarly, insurance and purchase of software are fixed costs. On the other hand, variable costs depend on the quantity of product and are incurred regularly; thus, these are proportional to the volume of work and continue along with the system in operation. For example, the cost of computerized forms varies in proportion to the amount of processing or the size of the reports.

In the same way, fixed benefits remain constant. When a new system is used and if 20 per cent of manpower is reduced, it is a fixed benefit. Such benefit of saving on personnel may occur every month. But variable benefits are realized on day-to-day operations and on a regular basis. For example, in a library information system, if saving is accomplished by 2 minutes in providing information to the borrower about a particular book, issued or not, as compared with the

manual system, it is a variable benefit. The amount of time saved depends on the information given to the number of borrowers.

Thus, cost-benefit analysis can be explained as follows:

- The method that helps to decide the value of the gross benefits when a new system is to be installed.
- The method that helps to decide the increased operating costs against gross benefits.
- The difference between the gross benefits achieved and operating costs to arrive at net benefits.
- The method that enables you to estimate the money value of the costs for development that produces benefits.
- The method that enables you to analyse a time-phased relationship between the costs towards development and net benefits derived.

NOTES

6.2.3 Cost and Benefit Evaluation Methods

The main cost-benefit evaluation methods are described here.

Net Benefit Analysis

Net benefit is total benefit net of total costs or total benefit minus total cost. The steps to calculate the net benefit are as follows:

1. Find out the components of benefit (due to an economic decision/economic project/investment).
2. Measure and value the benefit in different components and aggregate them (add them suitably because there could be benefits of different types/ components flowing over a period of time).
3. The same thing applies to all cost components: identify them, measure and value them and then aggregate them suitably.
4. Once the values of the total benefits and total costs are computed, it is simple to arrive at the net benefit as the difference between the two.

Therefore, the net benefit analysis is simply subtracting total costs from total benefits. The drawbacks of this method are as follows:

- It does not involve the time value of money.
- It does not discount future cash flow.

Break-Even Analysis

The break-even analysis is an analysis device used in business organizations. It involves the use of a chart to depict the overall volume of sales required to cover the costs (refer Figure 6.1). It is the point at which the cost and benefit of the organization are exactly equal, that is, it is the point where the organization neither earns profit and nor incurs loss.

NOTES

Break-even analysis can be used as a control device as well as an aid to decision making; its uses in the process of decision making are as follows:

- Finding the minimum sales required to meet the desired profit goals.
- Finding the minimum sales required for prevention of loss.
- Providing information and data that aid in decision making for adding or removing products.
- Providing information that is useful in making pricing decisions, i.e., to raise or lower the prices.

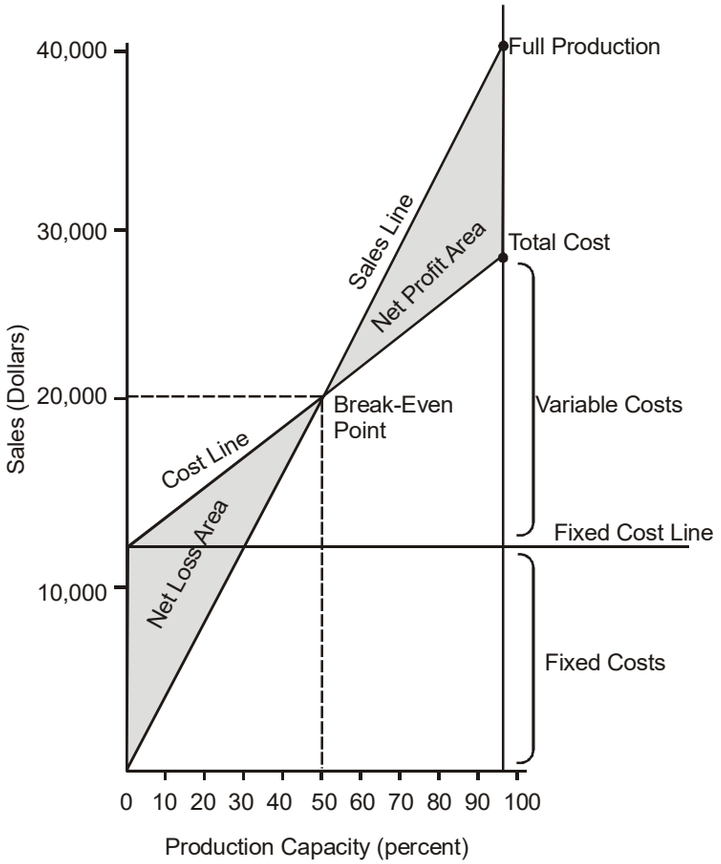


Fig. 6.1 *A Break-Even Chart*

Present Value Analysis

Usually in long-term projects, it is difficult to calculate and compare the cost incurred today with the complete value of tomorrow's benefits. Therefore, to overcome this problem, present value analysis calculates the costs and benefits in present day's terms, i.e. today's value of investment costs and then compares the cash flows at different times.

Present value is the value on a given date of a future payment or series of future payments, discounted to reflect the time value of money and other factors

such as investment risk. Present value calculations are widely used in business and economics to provide a means to compare cash flows at different times on a meaningful 'like to like' basis.

Present value analysis is the application of an appropriate discount rate to a stream of future cash flows. It allows differing payment streams to be compared.

The formula for calculating the present value is

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

Where F is future value [$F=P(1+i)^n$] and I is the interest at the end of the year (n).

Net Present Value Analysis

Net Present Value (NPV) is the difference between the present value of cash inflows and the present value of cash outflows. NPV is used in capital budgeting to analyse the profitability of an investment or project. In other words, NPV is calculated by summing the rupee-valued benefits and then subtracting all of the rupee-valued costs, with discounting applied to both the benefits and the costs as appropriate.

NPV analysis is sensitive to the reliability of future cash inflows that an investment or project will yield.

The formula for calculating the NPV as follows:

$$\mathbf{NPV} = \sum_{\mathbf{t=0}}^{\mathbf{n}} \frac{\mathbf{(Benefits - Costs)}}{\mathbf{(1 + r)^t}}$$

Where:

r = discound rate

t = year

n = analytic horizon (in years)

In addition to the formula, net present value can often be calculated using tables and spreadsheets such as Microsoft Excel.

The net present value is a measure of the value of a project: if the project NPV is positive, the project creates value and if the project NPV is negative, the project may not create value.

Cash Flow Analysis

Cash flow is the inflow and outflow of money from business to determine the business' solvency. Cash flow analysis is the study of the cycle of your business cash inflows and outflows, with the purpose of maintaining an adequate cash flow for your business and providing the basis for cash flow management.

Cash flow analysis involves examining the components of your business that affect cash flow, such as accounts receivable, inventory, accounts payable and

NOTES

credit terms. By performing a cash flow analysis on these separate components, you will be able to identify more easily cash flow problems and find ways to improve your cash flow.

A quick and easy way to perform a cash flow analysis is to compare the total unpaid purchases to the total sales due at the end of each month. If the total unpaid purchases are greater than the total sales due, you will need to spend more cash than you receive in the next month, indicating a potential cash flow problem.

Payback Analysis

Payback analysis is simply a calculation of how long it will take to recover your investment. Simple payback is a common economic analysis method and is understood by most business owners. Simple payback is the amount of time it will take to recover installation costs based on annual energy cost savings. The equation for simple payback is annual energy cost savings per year divided by the initial installation cost. The shorter the payback period, the sooner the organization will make profits and more attractive is the investment.

The payback period is calculated using the following formula:

$$\text{Payback Period (in years)} = \frac{\text{Initial Investment}}{\text{Annual Savings (Cash Flow)}}$$

Check Your Progress

1. Give the categories of costs and benefits.
2. What is a tangible cost?
3. Give the definition of direct cost.
4. Explain the concept of indirect benefits.
5. Define the significance of fixed cost and variable costs.
6. Elaborate the term net present value analysis.
7. What is determined by payback analysis?

6.3 TOOLS FOR STRUCTURED ANALYSIS

Structured analysis is a development method that analyses an existing system and directs it towards the development of specifications for the proposed system. This method permits the analyst to learn more about the system as well as the individual processes or activities in a logical manner. Figure 6.2 shows the layered model of a structured analysis.

NOTES

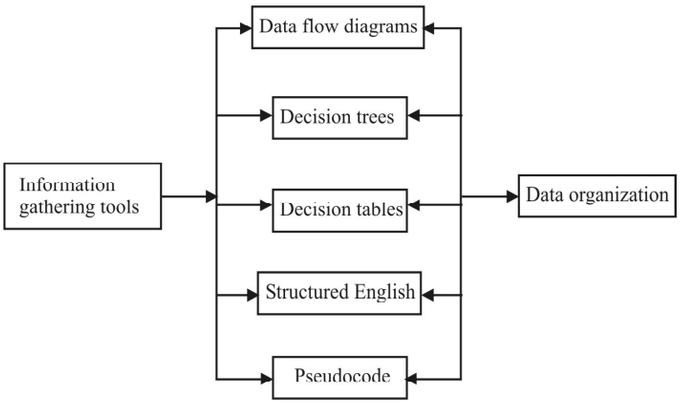


Fig. 6.2 *Layered Model of Structured Analysis*

Structured analysis contains various components, such as graphic symbols, data dictionary, procedure and process description and rules. Graphic symbols include icons and conventions that identify and describe the components of a system and the relationship between these components. Data dictionary describes the data items used in the system. The procedure and process description includes the formal statements using techniques and languages that assist the analyst to describe major activities in the system. Rules define the standards for describing and documenting the system. Diagrams and charts, decision tables, decision trees, structured English, data flow diagrams and data dictionary are some of the prime tools for structured analysis used in SDLC. They are discussed in subsequent sections:

6.3.1 Types of Basic Charts

Diagrams and charts are very useful tools for communicating how processes work for the proposed system. System designers and developers use many types of diagrams and charts as per system implementation requirements. Both tools make the system more interesting and informative because they illustrate boxes, texts as well as control flows which are easy and interactive to understand the system designers. Diagrams and charts are very useful because they communicate information visually. For this reason, the graphs and charts are used as phase wise report in the system implementation and design. The detail description of each type of charts is as follows:

Flowchart

Flowchart is known as oldest analysis tool to design and implement the system. It is used to depict the sequence of activities that a system must perform. The assumption in a flowchart is that activities occur in a linear sequence in which one event occurs first always followed by a specific second event, then a third, and so on. Programmers use flowcharts and decision tables as the basis through which

the logic of a computer program is derived. Figure 6.7 shows the flowchart of sample system having process boxes and decision box.

NOTES

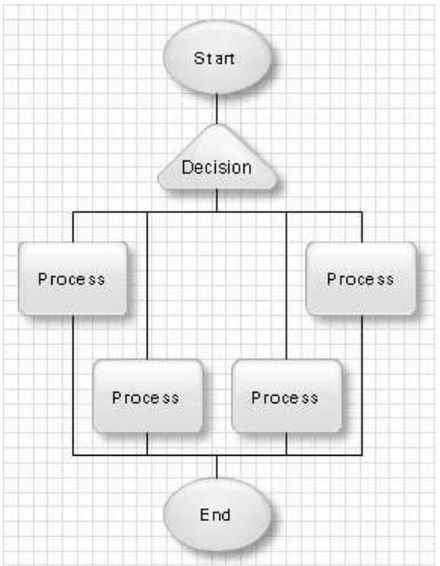


Fig. 6.3 *Flowchart of a Sample System*

Flowcharting Symbols

Flowchart helps all concerned persons, such as systems analysts and programmers in developing an intermediate interface for understanding the logic of a system using symbols. It provides an easy and effective approach for preparing a document and analysing the flow of data and control in a system. It helps in verifying and validating the developed program for appropriate debugging. Following are the benefits of flowchart symbols:

- ***Communication:*** Flowcharts are better way of communicating the logic of a system to all concerned.
- ***Effective Analysis:*** With the help of flowchart, problems can be analysed in more effective way.
- ***Proper Documentation:*** Program flowcharts serve as a good program documentation which is needed for various purposes.
- ***Efficient Coding:*** The flowcharts act as a guide or blueprint during the systems analysis and program development phase.
- ***Proper Debugging:*** The flowchart helps in debugging process.
- ***Efficient Program Maintenance:*** The maintenance of operating program becomes easy with the help of flowchart. It helps the programmer to put efforts more efficiently on that part.

A flowchart is a diagrammatic representation that illustrates the sequence of operations to be performed to get the solution of a problem. Flowcharts are

NOTES

generally drawn in the early stages of formulating computer solutions. They facilitate communication between programmers and business people. These flowcharts play a vital role in the programming of a problem and are quite helpful in understanding the logic of complicated and lengthy problems. Once the flowchart is drawn it becomes easy to write the program in any high level language. Flowcharts are helpful in explaining the program to others. Hence, it is correct to say that a flowchart is must for the better documentation of a complex program. Flowcharts are usually drawn using some standard symbols however some special symbols can also be developed when required. Table 6.1 lists the symbols that are used for creating a system flowchart.

Table 6.1 *Flowchart Symbols and Their Use*

Flowchart Symbol	Name of Symbols	Function
	Start or End	This symbol specifies the start and end points of a flowchart.
	Arrows	This symbol connects different flow chart symbols.
	Process	This symbol specifies the condition which is to be evaluated.
	Input or Output	This symbol specifies the input or the output of the process.
	Decision	This symbol specifies whether or not the specified condition is true
	Off page Connector	This symbol helps to connect the flowchart drawn on different pages.
	Stored Data	This symbol specifies the name of the file which is stored on the disk.
	Connector	This symbol is used to connect two parts of a program.

The flowchart is a way of visually presenting the flow of data through an information processing systems and the operations performed within the system and the sequence in which they are performed. Symbols used in flowcharting describe what operations and in what sequence are required to solve a given problem. The program flowchart is similar to the blueprint of a building. A designer draws a blueprint before starting construction on a building. Similarly, a programmer prefers to draw a flowchart prior to writing a computer program.

Control Chart

One of the major obstacles contributing to the cost, time and efficiency of improving the quality output of manufacturing systems is the propagation of defectives or errors through the system. Conventional individual control chart design does not address the problem of the interrelation of the processes adequately. Owing to the

NOTES

increasing complexity of manufacturing systems as well as the problems caused by the natural variability of the systems, trial and error methods are the most commonly used technique for implementing the control charts. Control charts refer to sample testing process which is used to monitor product quality. Figure 6.4 shows the control chart in which Center Line (CL) decides the median value of control points.

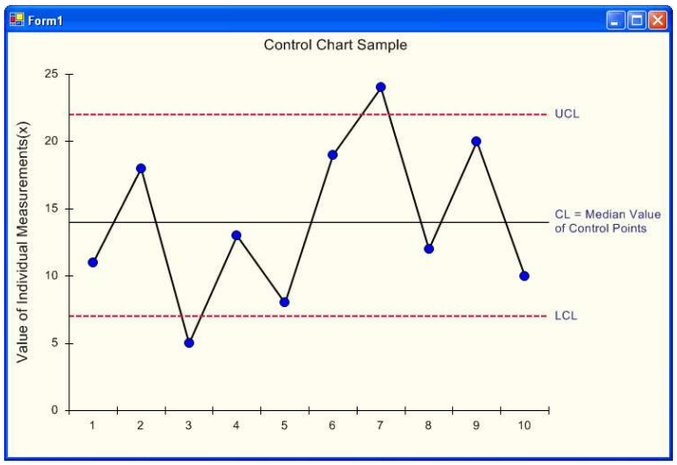


Fig. 6.4 *Control Chart*

Domain Chart

The domain chart is used to create partition for the system in terms of domains and sub-systems. Domains are autonomous entities logically grouped by their function. Domain chart depicts the system as distinct and independent domains that work together to provide the application's functionality. Figure 6.5 shows the domain chart of book store which maintains inventory, shipping, ordering, payment and database system.

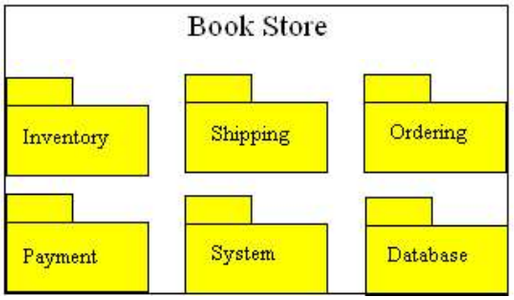


Fig. 6.5 *Domain Chart*

Gantt Chart

The Gantt chart is also known as business charts or time lines chart. This chart is a horizontal bar chart developed as a production control tool in 1917 by Henry L. Gantt who is an American engineer and social scientist. In project management, a

NOTES

Gantt chart provides a graphical illustration of a schedule that helps to plan, coordinate and track specific tasks given in a project. This chart is created using project management applications of system design and implementation in Microsoft Project or Microsoft Excel. A Gantt chart is constructed with a horizontal axis representing the total time span of the project which is broken down into increments, for example days, weeks or months and a vertical axis representing the tasks that make up the project. Horizontal bars of varying lengths represent the sequences, timing and time span for each task. A vertical line is used to represent the report date. Figure 6.6 shows the Gantt chart of project development schedule.

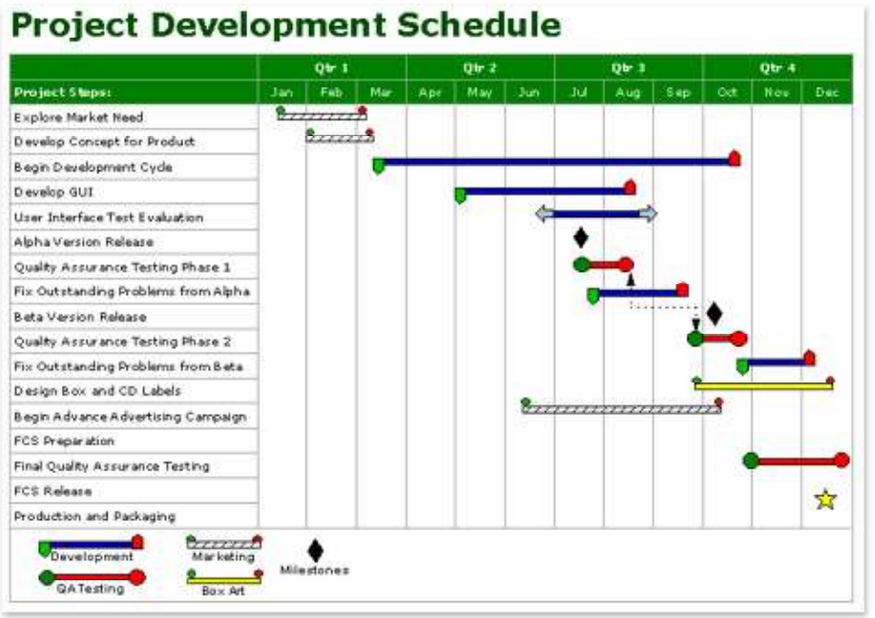


Fig. 6.6 *Gantt Chart*

Input/Output Chart

The Input/Output (I/O) chart is an important tool that is used to visualize the order flow during system implementation and designing. It is generally associated with funnel model which basically filters the output. Sometimes, system designers use this model with I/O chart to the system implementation processing as per defined time span and management capacity. The output is calculated in standard hours as per input parameters. This chart is an ideal tool to visualize dynamic processes along with leading time and due date deviation. Basically, the input and output slopes in the chart are plotted in accumulated curves. In this chart, each step indicates an operation finishing date in the system design whereas input curve represents the task which has to be processed during system implementation. There is no capacity loss, therefore, the output curve is defined as the potential capacity as per used system technology. The throughput time is equal to the horizontal distance between the average input and output slopes. An input chart is

NOTES

created for design phase tasks, deadlines, cost factors and management level people. The input chart helps to:

- Access information needs for planning, monitoring and evaluating the system.
- Decide the levels of information groups, information frequencies and content.
- Categorize data items to ensure system flexibility and adaptability by serving the lowest level of management.

6.3.2 Decision Tables

Decision table is an excellent tool for expressing complex logical relationships in a highly understandable and precise manner. A decision table is a matrix containing rows and columns which are used to define relationships. A decision table consists of two parts in which one is STUB and the other is ENTRY. The STUB is further divided into the upper and lower quadrant. The upper quadrant is known as condition STUB and the lower quadrant is known as action STUB.

The ENTRY part is also subdivided into upper quadrant and lower quadrant. The upper quadrant is known as condition entry and the lower quadrant is known as action entry. In a decision tree:

- Y represents the existence of a condition.
- N represents the condition which is evaluated and is not satisfied.
- A blank space represents the condition involved that has been tested.

There are three major components in a decision table. They are conditions, actions and decision rules. Conditions are events or facts that determine the course of action to be taken. Actions are the processes that get activated under certain conditions. Decision rules express the relationships between combination of conditions and courses of action.

Decision tables are useful in situations where the resulting actions depend on the occurrence of one or several combinations of independent conditions.

Four quadrants of a decision table are summarized in Table 6.2.

Table 6.2 *Quadrants of a Decision Table*

Conditions or causes	Condition rules (alternatives) represent the combinations of causes
Actions or effects	Action entries represent the combination of effects

Conditions are put in quadrants at the upper left corner of the table and condition rules (alternatives) are placed in the upper right corner. The corner at quadrant on the lower left has actions needed and at the lower right there are rules for action. Conditions are also known as causes and actions are also known as effects.

The manner in which these quadrants are filled is shown in subsequent tables. Following example shows factors, such as causes, values and combinations which are considered as prime factors to summarize the decision table.

NOTES

Example 6.1: In the following Table, there are 'causes' on the left upper corner and the number of causes are entered below it. Then below this head comes 'effects' and under it the possible effects, as relevant to the application are put. The Table also shows three causes (cause 1, cause 2 and cause 3). Under 'effect' only two effects are entered. Analyse it.

		Combinations							
Causes	**Values**	1	2	3	4	5	6	7	8
Cause 1	*Y, N*	Y	Y	Y	Y	N	N	N	N
Cause 2	*Y, N*	Y	Y	N	N	Y	Y	N	N
Cause 3	*Y, N*	Y	N	Y	N	Y	N	Y	N
Effects									
Effect 1		X			X				X
Effect 2			X				X		X

Solution: On the right side there are two heads under the combination head containing eight columns as shown in above Table. The number of columns depend on the number of causes. Combination will take value for a column in between these two values, Y or N. Significance of Y and N has already been mentioned earlier. Here, in this example, there are three causes. Hence, the header 'combination' will have $2^3 = 8$ columns. This shows all possible combinations of causes. If there are n causes, there will be 2^n combinations of causes. Number of combinations for effects or actions will depend on the number of actions. If number of actions is m, there will be m/n possible actions. Here, in this Table there are two effects (actions) and so, there will be $2 \times 8 = 16$ possible actions, all of which may not be relevant to the application. Here, combination that is relevant for action is only 6. The action has been shown as cross symbol.

***Note*:** Cause = Condition and Effect = Action = Expected results

The algorithm to create decision table is as follows:

- First divide all possible combinations by number of possible values. Here, for the first row, possible combinations are 8. Dividing 8 by 2, gives 4. So, RF for 1st row is 4.
- Enter first value, RF times and then second value also RF times to make the row full. Here, see that 1st contains YYYY and then NNNN.
- Do this for the next row dividing the repeating factor of previous row by a number of values. Here, RF for second row is 4/2 = 2. So, the repetition is YYNNYYNN covering all 8 columns.
- If all rows are exhausted stop. If not, then go to 3.

There is still one row left which is the 3rd row. RF for 3rd row is 2/2 = 1. Hence, it is entered as YNYNYNYN. All rows have been exhausted. Stop here. The decision table structure is made. In the following Table, in the third row, column

NOTES

3 and 4 have been shown with '-'. This shows that this combination does not apply to the implementation. This is called an indifferent combination.

		Combinations							
Causes	**Values**	1	2	3	4	5	6	7	8
Cause 1	*Y, N*	Y	Y	Y	Y	N	N	N	N
Cause 2	*Y, N*	Y	Y	N	N	Y	Y	N	N
Cause 3	*Y, N*	Y	N	-	-	Y	N	Y	N
Effects									
Effect 1		X			X				X
Effect 2			X				X		X

In the Table two columns are indifferent. These two columns are identical and hence, joined into one. In the following Table, the next step is shown. Third and fourth columns are identical so the third column is kept and the fourth column is dropped and now there are only 7 columns. Numbering of column has changed from fourth column onwards.

		Combinations						
Causes	**Values**	1	2	3	4	5	6	7
Cause 1	*Y, N*	Y	Y	Y	N	N	N	N
Cause 2	*Y, N*	Y	Y	N	Y	Y	N	N
Cause 3	*Y, N*	Y	N	-	Y	N	Y	N
Effects								
Effect 1		X						X
Effect 2			X			X		X

Make one row of checksum. For calculating checksum, the following rules are used:

- Find number of combination for every column.
- When a '-' is found in the column, each '-' shows number of combinations that the cause has.
- Multiply for every '-' below the column.
- Add up the total and then make the comparison with the second step.

Example 6.2: Find out the checksum for the following Table.

		Combinations				
Causes	**Values**	1	2	3	4	
Cause 1	*Y, N*	Y	Y	Y	N	
Cause 2	*Y, N*	Y	N	N	-	
Cause 3	*Y, N*	-	Y	N	-	
Effects						
Effect 1		X		X		
Effect 2						
Checksum		2	1	1	4	8

Solution: Checksum for the first column is 2 since there is one '-' and the number of values in the row is 2. For column 2 there is only one combination, hence the value of the checksum is 1. Same is true for the third column. Fourth column has two '-' and the value of checksum is 4 for this column. All added together comes to 8.

Then, read every column for determining the effects. One effect may take place in many combinations of test.

		Combinations				
Causes	**Values**	1	2	3	4	
Cause 1	*Y, N*	Y	Y	Y	N	
Cause 2	*Y, N*	Y	N	N	-	
Cause 3	*Y, N*	-	Y	N	-	
Effects						
Effect 1		X		X		
Effect 2		X	X			
Checksum		2	1	1	4	8

Implementation of decision table will be illustrated by an example. Suppose, a business establishment has to send mails to customers. Content of a mail tells about the present level of discount as well as potential levels for the discount. This may be different as per the types of customers. This business establishment creates three categories of customers into A, B and C. Normal letter is given to customers in the categories A, B and C. But type O customer gets special letter. Business house decides a criteria on which a customer having 2 or more current lines or having credit rating as 'X' gets addition of special paragraph that gives an offer for subscribing to another level of discounting.

These are shown in the following Table.

		Combinations															
Causes	**Values**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Customer Type	*A,B,C,O*	A	A	A	A	B	B	B	B	C	C	C	C	O	O	O	O
2 or more lines	*Y, N*	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N
Credit rating = X	*Y, N*	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Effects																	
Normal Letter		X	X	X	X	X	X	X	X					?	?	?	?
Special Letter										X	X	X	X	?	?	?	?
Add. Paragraph		?	X	X		?	X	X		?	X	X		?	?	?	?
No Letter														?	?	?	?
Checksum		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	16

Types of Decision Tables

Decision tables can be classified into two categories:

On the basis of the layout of the decision tables:

- **Expanded Decision Tables:** In expanded decision tables, all possible combinations of the conditions are shown in different columns. It is the condition oriented table representation of all single decision columns, used to check whether the table is correct and completely filled.
- **Contracted Decision Tables:** In some cases, it is possible to contract some combinations of conditions so that columns with the same actions are shown in one column. It is the compact, condition oriented, table representation of all decision columns.

NOTES

NOTES

On the basis of the **expression, statements** used for the conditions and actions:

- **Limited Entry Decision Tables:** The limited entry decision table is the simplest to describe. The condition alternatives are simple Boolean values, and the action entries are check-marks, representing which of the actions in a given column are to be performed.
- **Extended Entry Decision Tables:** In extended entry decision tables, the statements in the stub quadrants are more of open-ended questions. The question does not suggest the answer with limited options, but expects detailed information from a variety of possible options. The expression of conditions is partly given in the quadrants and the rest is expressed in the entries quadrants in the form of answers.
- **Mixed Entry Decision Tables:** The combinations of limited entry and extended entry rows results in mixed entry table. Mixed entry tables represent the flexibility of decision tables. The tables are convertible from one entry type to another and even a combination of both.

6.3.3 Decision Trees

Decision tree is another method for defining complex relationships. A decision tree has as many branches as there are logical alternatives within software. It depicts the logical structure based on a stated policy. Decision trees are used when combinations of conditions need to be present. When policies change, it is easy to change trees.

A decision tree approach represents a method for depicting alternative actions and conditions within a horizontal tree framework. It is used when complex branching occurs in a structured decision process. Trees are useful while keeping decisions in a particular sequence. It is important to distinguish between conditions and actions while drawing decision trees. A square node indicates an action and a circle indicates a condition. Figure 6.7 shows a decision tree structure.

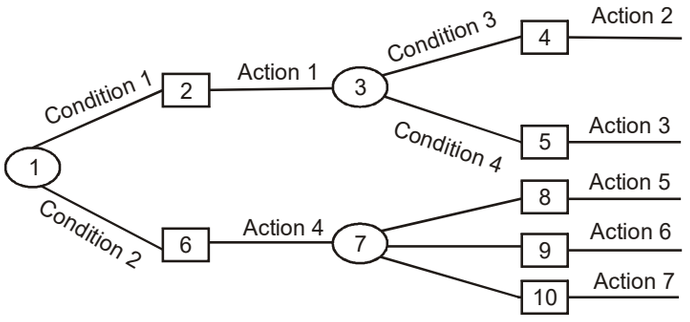


Fig. 6.7 *Structure of a Decision Tree*

Decision tree is a powerful tool which is popular as it is used for classification and prediction. It shows some rules for using a decision tree. It can also be directly used in query language, such as Structured Query Language (SQL) for access to a database and records that belong to a particular category may be retrieved.

NOTES

Different tree diagrams are used to determine the optimum course of action out of several possible alternatives with uncertain outcomes. Resulting chart or diagram looks like a cluster of tree branches and displays the structure of a particular decision. It also shows the interrelationships and interplay between various alternatives, decisions and possible outcomes.

In decision analysis, a decision tree is used to represent decisions and decision- making, visually and explicitly. When it is used for data mining, it describes only data and not decisions. Such a classification tree works as an input for decision-making.

For classifying an example, in a decision tree, start at the root of the tree and move through it until a leaf node is reached. By applying an inductive approach, decision tree is used to learn knowledge on classification. Decision tree is used for data mining to form a big database. Key requirements for mining by using decision trees are as follows:

- ***Attribute Value Description:*** The object is described as fixed collection of attributes. This requires conversion of continuous attributes in discrete form or already provided in the algorithm.
- ***Pre-Defined Classes:*** Categories must be established before assigning examples.
- ***Discrete Classes:*** Cases may or may not fall under a particular class. Number of cases have to be more than the number of classes.
- ***Data Sufficiency:*** Many cases are to be taken for this. Normally hundreds of cases should be taken but the number may even be thousands.

Thus, decision tree is a decision support tool making use of a tree like structure or graph. It works as a decision model that shows possible consequences, chance event outcomes, resource costs and utility. Decision trees find use in operations research, as a support in decision analysis which helps in identifying a strategy most suitable to realize a goal. Another use of decision trees is as a descriptive means for calculating conditional probabilities.

A decision tree has following types of nodes:

- ***Decision Nodes:*** Such nodes fall in-between terminal nodes or leaf nodes and root. These specify some kind of test that is to be carried out on a single attribute value having one branch and a subtree for every possible outcome of the test. In diagram, it is denoted by square.
- ***Chance Nodes:*** These are also intermediate nodes and are represented by circles.
- ***End Nodes:*** These nodes are also known as leaf nodes and indicate the value of the target attribute (class). These are represented by triangles in a diagram.

NOTES

Advantages of Decision Trees

The advantages of decision trees are as follows:

- ***Simple in Understanding and Interpretation:*** Strategies can be explained better with the help of decision tree models. A brief explanation may be sufficient.
- ***Values can be used:*** An expert can describe a situation by putting some numerical values. 'What if' analysis can be made by viewing various alternatives, probability of occurrence and costs involved and preferred outcome can be selected.
- ***Use a White Box Model:*** If the model provides a given result, its explanation on the result can easily be replicated by using simple math.
- ***Combination with Other Techniques of Decision-Making:*** Decision tree can be combined with other decision-making techniques. These techniques calculate Net Present Value (NPV), a linear distribution of expected outcomes and Present Value (PV).

The decision tree is a computational model in which an algorithm is used to show a sequence of branching operations which is based on comparisons of some quantities. There are many variants of decision tree models depending on the complexity of the operations of a single comparison and the way branching is done.

Decision trees models establish lower bounds for computational complexity that belong to certain classes of algorithms. Lower bound for worst case computational complexity is taken as proportional to the largest depth in the decision tree or all possible inputs. The model in which every decision requires comparison of two numbers within constant time is a simple decision tree model used to establish computational complexity of sorting and searching.

Simplest illustration for lower bound technique is the problem of finding the smallest number among n numbers by comparisons only and the decision tree model is a binary tree.

Decision tree, in communication complexity, is a decision-making model. It is a Boolean function having a series of queries as input and final decision as output. Each query depends on queries previous to it. For this reason, it is described as a binary tree. Hence, complexity of a decision tree is the complexity of query.

The three basic models are discussed as follows:

- ***Deterministic***

Let there be a decision tree having output as $f(x)$ for all, then, the decision tree computes f. In a decision trees depth is given by the number of queries made to reach a leaf node giving the result. $D(f)$ which shows complexity in deterministic model of f is given by the smallest depth in such a tree computing f.

- ***Randomized***

In this type of model, additional nodes are added and each node has a controlling probability of p_i. There is another definition which is equivalent to this. It lies in selecting the entire decision tree from a set of decision trees at the beginning which is based on some probability distribution. Complexity of the randomized tree is given by the highest depth among trees that have probabilities more than 0. $R_2(f)$ denotes complexity of randomized decision tree with least depth having result as $f(x)$ having minimum probability of 2/3 for all.

This $R_2(f)$ gives the complexity of randomized decision tree of the Monte Carlo model since here, the result may be incorrect bounded with error on both the sides. $R_1(f)$ is one-side-bounded error version.

- ***Quantum Decision Tree***

$Q(f)$ stands for quantum decision tree complexity. It is defined by lowest depth of the quantum decision tree giving $f(x)$ as result having minimum probability of 2/3 for all. $Q_E(f)$, is another quantity, defined as lowest depth of the quantum decision tree giving result $f(x)$ as a sure event (probability = 1) in all cases. These quantity, $Q(f)$ and $Q_E(f)$, are quantum query complexities. This is commonly name given as direct definition more complicated than that in the classical case.

Strength and Weakness of Decision Tree Methods

The strengths of the decision tree methods are as follows:

- Ability to generate understandable rules.
- Ability to classify without much computation.
- Ability to handle both continuous as well as categorical variables.
- Ability to give clear indication on most important fields for the purpose of classification or prediction.

The weaknesses of the decision tree methods are as follows:

- Less appropriate for tasks of estimation in predicting the value of an attribute of continuous nature.
- Classification is prone to errors when there are small number of samples and many classes.
- Computationally expensive in growing a decision tree.
- Decision tree algorithms, mostly, check only one field at a time. This may not be compatible to the actual distribution of records available in the decision space.

6.3.4 Structured English

Structured English is used to analyse decision process. It has a restricted vocabulary. It uses action oriented verbs like add, multiply, and so on. It does not use a subject and makes use of the terms defined in data dictionary. It is a tool that enables one to state the rules precisely. It uses logical constructs and imperative sentences that

NOTES

are instructive. It is a special language that uses procedural logic derived from structured programming. Keywords of many programming languages are derived from such strong verbs in assembly language or other languages sometimes using mnemonics. Words, for example do, add, sum, etc., are used. Shortened form, such as EQ for equal is used in FORmula TRANslator (FORTRAN). Query languages make use of words of structured English, such as SELECT, JOIN, WHERE along with other combination of special characters, such as =, >, <, etc.

Words, such as IF, THEN, ELSE, UNTIL, REPEAT and SO are used for decision-making. Structures show logical hierarchy. It is most useful if it is required to take into account loops and sequences in a program. It is also good for use in procedure manuals.

When the decision structure is not complex, structured English can be used. By using this tool, structured instructions are grouped and nested in an organized manner. To write structured English, the following points should be kept in mind:

- Use sequential structures, iterations or decision structures while expressing logic.
- Make use of keywords that exist already. Such keywords are IF, THEN, ELSE, DO, DO WHILE, etc.
- Statement blocks should be properly indented to reflect hierachical structure or nesting.
- If phrases or words that you use are already defined in the data dictionary, underline them. This shows that they have special meaning.
- Before starting the coding stage, clarify logical statements.

Two basic building blocks are logic or instructions that are organized as grouped procedure and are nested and strong verbs that are used for actions. These verbs are move, add, multiply, etc. Structured English is used for labelling decision trees and creating decision tables. Four major considerations for using structured English in decision trees are as follows:

- Identifying conditions
- Identifying outcomes for every decision
- Identifying actions
- Identifying rules

In decision tables, structured English words are used. They are needed when there are complex combinations, actions, rules and conditions on which actions are performed. Here, structured English provides a method that can help avoiding contradictions, ambiguity due to duplication and impossible situations. Decision trees are useful when every condition is not relevant for every action.

Structured English is used in three basic constructs that are used in every programming language. These are as follows:

- Sequence

- Selection (if-else conditional)
- Iteration (loops)

Pseudocode

NOTES

Pseudocode is used to express logic in the English language and has no conformation to any particular programming language. It is used while implementing physical design and even after it. It is not used for any coding. Pseudocode may be a good replacement of flowcharts used for program design. It is used to explain the program logic in combination with structured programming. This technique makes utilization of verbs, such as DO WHILE, UNTIL, PERFORM, ENDIF, and so on easier. It is structured but distorted English.

6.3.5 Data Flow Diagrams

Data Flow Diagrams (DFDs) are used to specify how the data flows between the functions of a system. The basic purpose of DFDs is to show how the system is currently implemented. Larry Constantine first developed DFDs as a way of expressing system requirements in a graphical form. A DFD is the starting point of the design phase that functionally decomposes the requirement specifications down to the lowest level of detail.

Basic Elements of a DFD

A data flow diagram illustrates the flow of data through a system and the work performed by that system. In data flow diagrams, the symbol set comprises of diagram entity, process, data store and data flow.

An entity is used to define the boundaries of the system. It is an external component of the system, for example a department, a person or a business that interacts with the existing system. It also provides data to the system and receives data from the system. An entity is represented by a rectangle. Table 6.3 shows the basic symbols of data flow diagrams.

A process is defined as a work or action performed by people, machines, etc., within a system. It is used to transform the input of a system into an output and is represented by bubbles.

A data store is used to store data. Data stores are represented by open boxes.

A data flow is any item that carries data to, within or from the system. That is, it is used to represent inputs and outputs of the system. Data flows are represented by arrows.

The two types of data flows are physical DFD and logical DFD.

- ***Physical DFD:*** A physical DFD is an implementation dependent view of the current system showing what functions are performed. A physical diagram provides details about hardware, software, files and people involved in the

NOTES

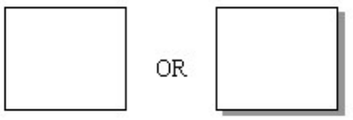
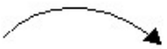
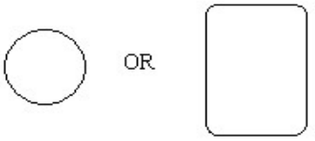
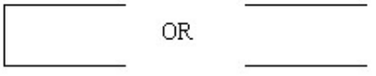
implementation of the system. Physical characteristics include names of the people, names of the departments, names of files, names of hardware, locations, names of procedures, form names, document names, etc.

- ***Logical DFD:*** Logical DFD is an implementation independent view of a system that focuses only on the flow of data between different processes or activities. Logical diagrams show how the business operates. They do not show how the system can be implemented. They explain the events of the system and the data required by each event of the system.

  Physical DFD differs from the logical DFD in the following ways:

  - Physical DFD is implementation dependent whereas logical DFD is implementation independent.
  - Physical diagrams in the physical DFDs provide low level details, such as hardware and software requirements of a system whereas logical diagrams in the logical DFDs explain only the events involved in the system and the data required to implement each event of the system.

Table 6.3 *Basic Symbols of a Data Flow Diagram*

Symbol Name	Symbol	Symbol Meaning	Comments
Square		Source or destination of data	May be one customer or a number of customers with customer orders
Arrow		Data flow	May be physically contained in a purchase order, invoice, etc.
Circle		Process that transforms that data flow	May be an accountant calculating discounts and preparing invoices
Open Rectangle		Data store	Can be a file, magnetic tape, a database on disk, etc.

Context Diagram

Context diagram helps in understanding the general characteristics of the system under study. It contains a single process and defines the whole system that will be studied. Anything that is not inside the process of the context diagram will not be part of the system study. The inputs and outputs specified at this level remain constant for the other lower levels as well. Figure 6.8 shows the context level DFD for a customer order processing system.

NOTES

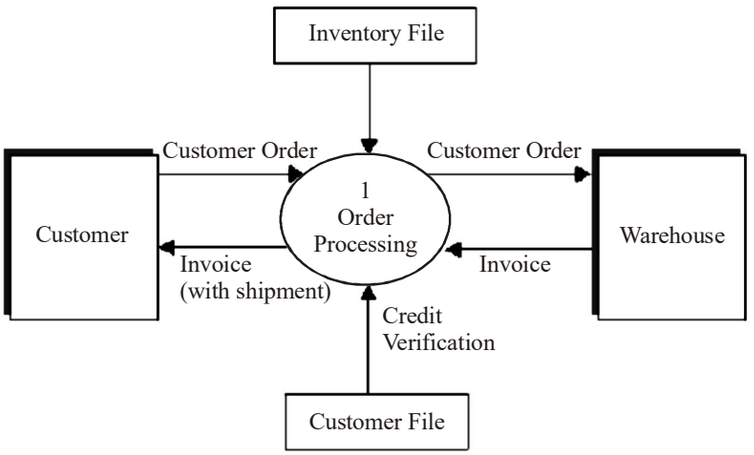


Fig. 6.8 *Context Level DFD for Customer Order Processing System*

Rules to Construct a DFD

The rules that must be followed to construct the DFD of a particular system are as follows:

- Each process involved in a system should be named and numbered for easy reference.
- The name of each process should be symbolic.
- The direction of flow of information in a DFD should be from the top to the bottom and from the left to the right.
- When a process is divided into several lower level processes, each low level process should be numbered.
- The names of the data stores, sources and destination should be in capital letters.

Level 0 DFD

Level 0 DFD provides more detail than the context diagrams. By dividing context diagram processes into sub-processes, the systems analyst begins to fill in the details about the flow of data within the system. Each sub-process in the Level 0 DFD is numbered with an integer. Generally, this number starts from the upper left

NOTES

corner and proceeds towards the lower right corner of the Level 0 DFD. Figure 6.9 shows the Level 0 DFD for customer order and credit verification system.

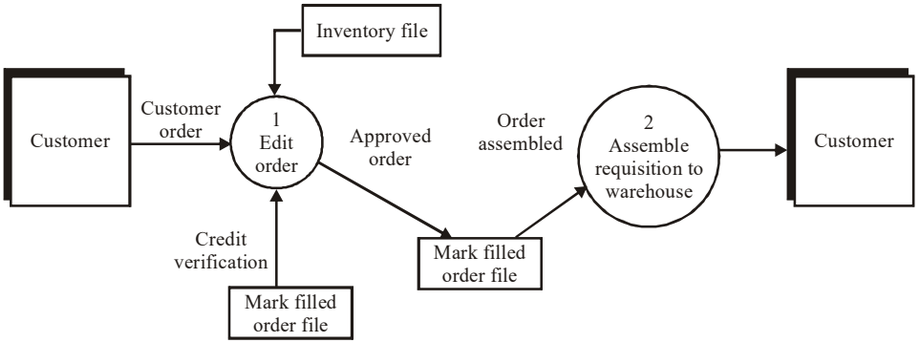


Fig. 6.9 *Level 0 DFD for Customer Order and Credit Verification System Child Diagrams*

Each process in the Level 0 DFD may, in turn, be divided to create a more detailed child diagram. The process of level 0 DFD that is divided is called the 'parent process' and the resultant diagram is called the 'child diagram.'

The primary rule while creating child diagrams is that the child diagram cannot receive input or produce output that is not produced or received by its parent process. That means any data flowing in or out of the parent process should be shown flowing into or out of the child diagram.

The processes in the child diagram are numbered by using the parent process number, a decimal point and a unique number assigned for each child process of the child diagram. Figure 6.10 shows a first Level DFD elaborating customer order processing and shipment system.

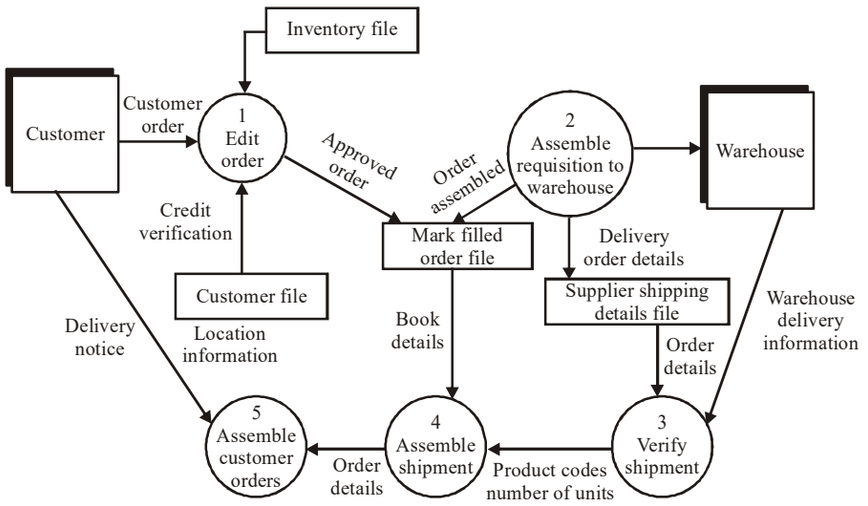


Fig. 6.10 *First Level DFD, Elaborating Customer Order Processing and Shipment System*

NOTES

Data Structures

A data structure is a set of data items that are related to one another and are collectively used to describe a component in a system. For example, a data item DATE can be further classified as DAY, MONTH, YEAR or a collection of these elementary data items form the data structure DATE.

The first stage of defining data structures is a logical design. Logical design shows what data the business needs for its day to day operations. For example, name, address, etc., are the elements that the user would see. Logical design is considered as a basis by the analyst for designing physical data structures. Physical data structures include additional elements necessary for implementing the system, such as key fields that are used to locate records in a file. Similarly, an item number which is not required for a business is necessary to identify and locate computer records.

Describing Data Structures

Data structures are arranged in four relationships:

- Sequence
- Selection
- Iterative
- Optional

Sequence relationship defines data items that are always included in a particular data structure or a concatenation of two or more data items. For example, CUSTOMER DATA includes:

- NAME
- FIRST NAME
- MIDDLE NAME
- LAST NAME
- ADDRESS
- STATE
- PIN

In certain situations, the data structure consists of alternatives for data items. The selection relationship represents such alternatives and indicates either/or situation. A choice of one item must be made from a set of two or more items.

Iteration relationship implies repetition. In this type of relationship, the data elements that compose the structure are repeated.

In certain cases, some of the data items, such as MIDDLE NAME, TEL_NUMBER may be optional. Such relationships are known as optional relationships.

NOTES

Data Flows

Data flows represent the data movement within the system. For example, the flow of an order form from customer to data collection centre. The information captured for each data flow may be summarized by using a form containing the following information:

- ID, an optional identification number.
- Unique name for the data flow.
- A general description of the data flow.
- The source of the data flow which could be an external entity, a process or a data flow coming from a data store.
- The destination of the data flow (a process, data store).
- An indication of whether the data flow is a record entering or leaving a file or a record containing a report, form or screen.
- The name of the data structure describing the elements found in this data flow.
- The volume per unit of time. This could be record per hour or per day, etc.
- An area of further comments about the data flow.

Data Stores

A data store is a place to store data. For example, CUSTOMER FILE, INVOICE FILE, STUDENT FILE, etc., that store all the information about a particular entity. Information that should be recorded regarding data stores are as follows:

- Data store ID
- Data store name
- Alias
- A short description of the data store
- The file type (manual or computerized)
- The maximum and average number of records on the file
- The data set name (file name)
- The data structure which must also appear in a data dictionary

Data dictionaries are supported by Computer Aided Software Engineering (CASE) products which greatly ease the task of creating and maintaining data dictionaries. Data dictionaries are the basis of automatic program generation.

Decision Analysis

Decision analysis evaluates various alternatives of complex nature in an uncertain environment in terms of some monetary values which are expressed as number.

These numbers have monetary values concerning a business and those who manage the business. An analysis of this kind needs an insight into the way these alternatives are defined while creating a mathematical model differing from one another. Such differences generate various suggestions providing improved alternatives. Numbers quantify subjective values by taking into consideration the uncertainties that enable you to take stock of a situation and lead you towards decision-making. These quantified results are then translated back into words generating qualitative insight.

Humans are capable of understanding, comparing and manipulating numbers. For creating a decision analysis model, you have to create a model structure assigning probabilities and values in the model. These assignments are essential for making computations. Assigning values for probabilities, functions, evaluating alternatives, risk evaluation and value weights for trade-off objectives is done to complete the model.

After finalizing the model structure and putting numbers in its place, analysis can be started. Computation of the expected utility of alternatives is not the only requirement of decision analysis. It requires much more than that. You should examine the sensitivity of the outcomes, proper weight for key probabilities and risk parameters. It is possible to calculate the value of perfect information in an atmosphere of uncertainty carefully modelled as part of the sensitivity analysis.

Here, two additional comparisons by using quantitative approaches are recognized. The first makes direct comparison of weighted utility keeping all the objectives in mind for alternatives that are under consideration. The second compares alternatives on any two objectives selected for the purpose showing Pareto optimality for these objectives.

A rational decision-making model is needed due to the growing complexity in the modern world where information plays an important part in the atmosphere of risk and uncertainty. Decision analysis provide guidance, based on information and develop an insight by putting the same in a structure that eases the decision making process to make it better and more 'rational' in nature.

Elements of Decision Analysis Models

The models and techniques have concerns based on prescriptive theories of choice for action. This shows a way to the decision-maker on how he should behave or react when he has to face a situation where outcomes are governed by chance or the actions or moves of his competitor.

Decision analysis enables decision-makers to choose one option from a set of possible alternatives. Such decision analysis is done when there are uncertainties regarding future outcomes.

The elements of decision analysis problems are as follows:

- An individual is designated as the decision-maker, for example the Chief Executive Officer (CEO) of a corporate body.

NOTES

- Number of possible events which is finite that is, set of possible scenarios. Circumstances or constraints under which a decision is made.
- Number of decision alternatives available to the decision-maker. There should be a selection of action out of these. Seeking among a better set of alternatives in comparison to those initially presented. Being brief on the logical and reasoning part of a decision.
- Pay-off means return of decision-making. Different combinations of decisions and uncertaintes generates different pay-offs.

Source of Errors in Decision-Making

The sources of errors in decision-making are incorrect assumptions, inaccurate estimation of probabilities, undue expectations, incorrect measurement of utility function and errors in forecast. Figure 6.11 shows the components of a probabilistic model.

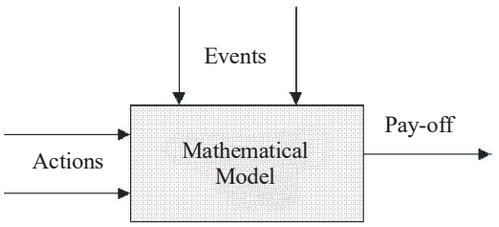


Fig. 6.11 *Components of a Probabilistic Model*

Decision analysis is a method of analysing all aspects of the factors affecting the decision-making process and help decision-makers to take correct decisions in the shortest possible time. This requires good communication between decision analysts in academia, business, industry and government.

Decision analysis carries on a systematic study in a framework that enables decision-makers to choose courses of action in a situation having uncertainty, complexity and conflicts. This consists of a brainstorming session to analyse all choices and possible actions, predicting the outcomes that are expected and derived by carrying out a logical analysis of factors affecting the decision-making situation. The decision theory finds its origin from economics that uses the utility function of pay-offs.

The flowchart in Figure 6.12 depicts the process of decision analysis for making a correct and logical decision.

NOTES

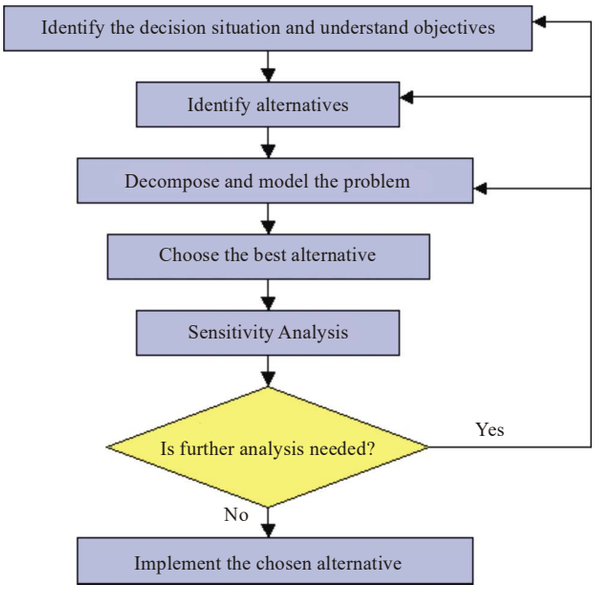


Fig. 6.12 *Flow Chart Depicting Decision Analysis Process*

For decision analysis, setting objectives is of utmost importance. Without a goal, it would be a blind and directionless movement. This is important in identifying problems as well as in evaluating the alternative solutions. Evaluation of an alternative course of action requires the expression of decision-maker objectives as criteria and reflecting the attributes of alternatives relevant to the choice.

Data is converted into information and information is turned into a meaningful fact. Fact, in turn, becomes knowledge, making decision analysis and using it for the successful completion of a decision-making process. Figure 6.13 illustrates the statistical thinking process based on the factual data for constructing statistical models.

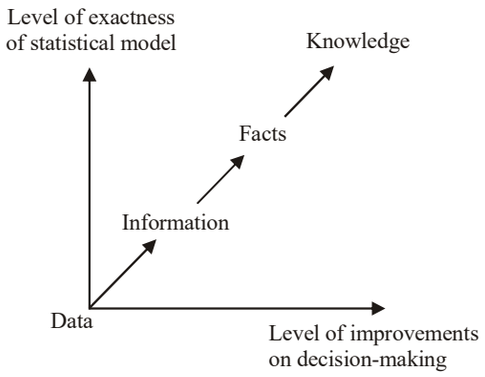


Fig. 6.13 *Statistical Thinking Process*

Figure 6.13 reveals the increase in the exactness of a statistical model as the level of improvements in decision-making increases. Hence, a probabilistic modelling is required.

NOTES

Decision-Making Process

In a decision-making process under uncertainty, variables are many and more difficult to measure and exercise control. Steps are the same like the deterministic decision-making process. These steps are as follows:

- Simplifying.
- Creating a decision model.
- Model testing.
- Finding solution with model has following features:
  - o It represents a simplified presentation of the actual situation.
  - o It might not be exact.
  - o It concentrates on most essential relationships.
  - o It is easily understood as compared to empirical situation.
  - o It can be used frequently for similar problems or can be modified.

With the growing advancement in mathematical analysis and complexities in real life situations these models and methods for analysis and decision-making fall under various types. The advent of computers made it possible to analyse such practical applications. A few examples are given as follows:

- By using random sampling techniques an auditor can do audit for clients.
- Factory managers use statistical quality control techniques to assure the quality of production.
- A financial analyst uses regression and correlation to understand the financial aspects to other variables in business.
- A market researcher may use this technique to decide on acceptance or rejection of a hypothesis on a group of buyers.
- A sales manager uses statistical techniques to forecast sales figures.

Data Dictionary

Data dictionary is one of the most important reference works that contain information about data which is also known as metadata. It is compiled by systems analysts and guides developers during analysis and design. It is created to assign specific meaning to every term used in the design leaving no chance of ambiguity on the interpretation of the meaning of the term used. Contents of the data dictionary have descriptions and specifications on system data but it does not contain the data itself.

NOTES

Data dictionary offers the following important functionalities:

- It provides basic documentation.
- It eliminates redundancy.
- It acts as starting point in the development of reports and screens.
- It validates DFD for accuracy and completeness.
- It aids the development of the logic for processes contained in DFD.

Using data dictionary, duplication of efforts is avoided and better communication is established between various departments of an organization that share a common database. This also makes maintenance easier and utilizes it as a standard for consistent data elements.

A data dictionary consists of:

- Name of the data item that the program handles.
- Alias, if any, for data item(s).
- A description for every data element in English.
- List of other data elements that are related to some specific data items.
- Acceptable values for every data item.

Naming is an important part in the creation of data dictionary. The name should be meaningful and distinct. There should be no duplication of the name. To derive maximum benefit from the data dictionary, it has to be tied to other programs within the system. This gives an advantage. When an item gets deleted/updated from data dictionary it gets deleted/updated automatically from the database. It is strongly recommended to let the computer maintain dictionary by using cross referencing whenever there are changes made in the system.

Data dictionaries can:

- Generate reports.
- Create screens.
- Create forms.
- Generate source code for computer program.
- Analyse system design for its completion and detection of flaws in design.

A data dictionary is also known as database dictionary. It defines basic database organization. This dictionary contains a list of every file with the number of records per file and information on each data field for names and types. Data dictionary being one of the most vital data structure is kept hidden from users for preventing accidental destruction of its contents.

An element in data dictionary is used in other important charts and documents. Data elements defined in a data dictionary are used in an Entity-Relationship (E-R) diagram as shown in Figure 6.14.

NOTES

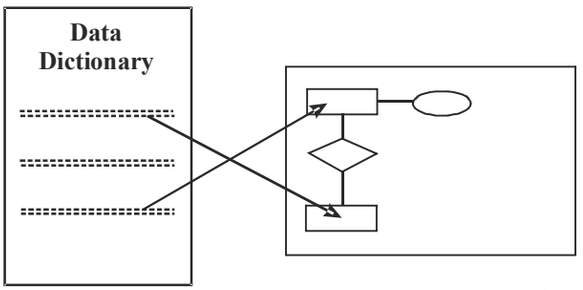


Fig. 6.14 *E-R Diagram*

Users of database and developers of application can derive benefits by using a data dictionary that is an authoritative data document that catalogues vital contents, organization's specific information and conventions for some databases. Such a dictionary may contain names with descriptions of many tables/fields in every database. It may also contain additional details, such as the type and length of every data element. Creation of data dictionary or database dictionary is a skilled task and there are no universal standards on the level of detail to be given in this type of document. A data dictionary may provide important information that describes the way data elements have been encoded. One great advantage offered by data dictionary is that it is meticulously designed and properly documented. It lies in its capability to establish consistency in a big database having a complex nature.

Data dictionary is an organized collection of all the relevant system data elements. A data dictionary helps both the user and system analyst have a common understanding of all the inputs, outputs, data stores and intermediate calculations. It is a catalogue of the data elements in the system. The data dictionaries are an integral part of the structured analysis because data flow diagrams themselves do not fully describe the data elements of the system. The data dictionaries provides additional information about the system. It stores the details and descriptions of data flows, data stores and processes. It also specifies the related values and units for the information in the data flows and data stores. A data dictionary is compiled by the system analyst during the analysis and design phase of the software. Table 6.4 shows the data dictionary user attributes.

Table 6.4 *Attributes of a Data Dictionary User*

Data Name	Data Description	Number of Characters
FNAME	First name	15
LNAMAE	Last name	16
ADD	Address	50
PHN	Phone number	12
DOB	Date of birth	8
SSN	Social security number	9
CARD	Card number	10

NOTES

Data Element

A data element is a field or an elementary item which cannot be classified further. For example, an INVOICE NUMBER, AMOUNT DUE, etc., are data elements.

Each item is identified by a data name, description, alias, length and specific values that are permissible for the system being studied.

Data Names

To differentiate data items from each other, meaningful and understandable data names must be used throughout the system development process. For example, a customer name is more meaningful if it is named as CUSTOMER NAME or CUST_NAME rather than ABCD. Different organizations suggest their own standards for developing data names.

Data Descriptions

Data descriptions briefly describe what the data item represents in the system. CUSTOMER NAME or CUST_NAME indicates the name of the customers that interacts with the system. Descriptions must be written with an assumption that the people who are reading the description do not know anything about the system. Jargons should be avoided in writing.

Aliases

Aliases are the additional names of the data items and can be referred to as synonyms. For example, a CUSTOMER can be referred to as ACTIVE CUSTOMER or INACTIVE CUSTOMER in other areas of the system. On the other hand, when the data is augmented through processing and reflected in the data names the data items are no longer aliases.

Length

Length identifies the number of spaces required for each data item.

Data Values

Data values are the specifications about the permissible values or the range of values that can be accepted. For example, MONTH cannot exceed a value of 12 or SEX can only be male or female. Table 6.5 shows the data dictionary book attributes.

NOTES

Table 6.5 Data Dictionary Book Attributes

Data Name	Data Description	Number of Characters
ISBN	ISBN number	10
TITL	Book title	60
SUB	Book title	80
AFNAME	Author first name	15
ALNAME	Author last name	16
PUBLD	Publication date	8
PUBLR	Publisher	20
ACNUM	Acquisition number	6

Notations used in Data Dictionary

A data dictionary uses many symbolic notations carrying some meaning that are defined as follows:

'=' is composed of
'+' and
'**()**' element is optional
'**{}**' ' interation
'**[ ]**' select one of a list of elements
'|' separates choices of elements
'******' comments
'**@**' identifier for a store (unique id)

Check Your Progress

8. Give the definition of structured analysis.
9. Why flow chart is used?
10. What are the various benefits of flowchart symbols?
11. Define the significance of decision tables.
12. Why decision trees is used?
13. What does data flow diagram illustrate?
14. Give the definition of data structures.
15. What are the sources of errors in decision-making?
16. Define the term data dictionary.

6.4 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

NOTES

1. Costs and benefits may be categorized as follows:
   - Tangible or intangible
   - Fixed or variable
   - Direct or indirect
2. Tangible costs and benefits are those that can be measured easily. The amount of cash dispensed for any specific item or service is known as its tangible cost. Such costs are clearly known and are estimated accurately. The same applies to tangible benefits.
3. Costs, directly associated with a system, are known as direct costs. For example, purchase of a pen drive for ₹ 500 is a direct cost as the association of the pen drive with the money spent is direct. Similarly, you can measure direct benefits.
4. Indirect costs, on the other hand, are not directly associated with a particular activity in the system. They are classified as overhead expenses. For example, to install a computer system, you need some space. Thus, the cost of space, the cost of maintenance of a computer centre, the cost of lighting and air conditioning are all tangible costs. But it is not easy to quantify the proportion attributable to any specific activity.
5. Fixed costs are sunk costs and are constant. They are one-time costs. For example, investment on installation of equipment for a computer system is a fixed cost, which is constant whether in use or not. Similarly, insurance and purchase of software are fixed costs. On the other hand, variable costs depend on the quantity of product and are incurred regularly; thus, these are proportional to the volume of work and continue along with the system in operation. For example, the cost of computerized forms varies in proportion to the amount of processing or the size of the reports.
6. Net Present Value (NPV) is the difference between the present value of cash inflows and the present value of cash outflows. NPV is used in capital budgeting to analyse the profitability of an investment or project. In other words, NPV is calculated by summing the rupee-valued benefits and then subtracting all of the rupee-valued costs, with discounting applied to both the benefits and the costs as appropriate.
7. Payback analysis is simply a calculation of how long it will take to recover your investment. Simple payback is a common economic analysis method and is understood by most business owners. Simple payback is the amount of time it will take to recover installation costs based on annual energy cost savings.

NOTES

8. Structured analysis is a development method that analyses an existing system and directs it towards the development of specifications for the proposed system. This method permits the analyst to learn more about the system as well as the individual processes or activities in a logical manner.
9. It is used to depict the sequence of activities that a system must perform. The assumption in a flowchart is that activities occur in a linear sequence in which one event occurs first always followed by a specific second event, then a third, and so on.
10. Following are the benefits of flowchart symbols:
    - *Communication*
    - *Effective Analysis*
    - *Proper Documentation*
    - *Efficient Coding*
    - *Proper Debugging*
    - *Efficient Program Maintenance*
11. Decision table is an excellent tool for expressing complex logical relationships in a highly understandable and precise manner. A decision table is a matrix containing rows and columns which are used to define relationships. A decision table consists of two parts in which one is STUB and the other is ENTRY. The STUB is further divided into the upper and lower quadrant. The upper quadrant is known as condition STUB and the lower quadrant is known as action STUB.
12. Decision tree is another method for defining complex relationships. A decision tree has as many branches as there are logical alternatives within software. It depicts the logical structure based on a stated policy. Decision trees are used when combinations of conditions need to be present. When policies change, it is easy to change trees.
13. Data Flow Diagrams (DFDs) are used to specify how the data flows between the functions of a system. The basic purpose of DFDs is to show how the system is currently implemented. Larry Constantine first developed DFDs as a way of expressing system requirements in a graphical form.
14. A data structure is a set of data items that are related to one another and are collectively used to describe a component in a system. For example, a data item DATE can be further classified as DAY, MONTH, YEAR or a collection of these elementary data items form the data structure DATE.
15. The sources of errors in decision-making are incorrect assumptions, inaccurate estimation of probabilities, undue expectations, incorrect measurement of utility function and errors in forecast.

16. Data dictionary is one of the most important reference works that contain information about data which is also known as metadata. It is compiled by systems analysts and guides developers during analysis and design. It is created to assign specific meaning to every term used in the design leaving no chance of ambiguity on the interpretation of the meaning of the term used.

NOTES

6.5 SUMMARY

- A Cost-Benefit Analysis (CBA) is done to determine how well or how poorly a planned action will turn out.
- Although a CBA can be used for almost anything, it is most commonly done on financial questions. Since the CBA relies on the addition of positive factors and the subtraction of negative ones to determine a net result, it is also known as running the numbers.
- The benefits of a CBA are it finds, quantifies and adds all the positive factors. Then, it identifies, quantifies and subtracts all the negatives—the costs. The difference between the two indicates whether the planned action is advisable.
- In practical terms, problems arise in making a cost-benefit analysis due to the fact that quantification of various components of costs is not easy in all cases.
- All aspects of a project, whether positive and negative, should be expressed in a common unit to decide the viability of any project, and money is the most convenient common unit.
- Apart from expressing the benefits and costs of a project as equivalent money value, the time at which this money value is provided is also important.
- An amount five years from now is not the same as an amount available at the present moment. One rupee, if invested now, to earn an interest for five years, will be worth more than a rupee in five years.
- Cost-benefit analysis valuation should represent the valuation of consumers or producers, as per their actual behaviour. Such valuations are required to be reflected in the preferences revealed by the choices made.
- The value is not just what transportation planners think about the worth of the time saved.
- While purchasing things at market price, consumers feel that the money they are spending is worth what they are spending on. Consumers have a tendency to increase consumption of a commodity till the marginal benefit it provides equals the marginal cost.
- Expenditure made on installing a system refers to the costs associated with the system. Extra expenditure incurred due to the inefficient working of a system is also a cost.

NOTES

- Tangible costs and benefits are those that can be measured easily. The amount of cash dispensed for any specific item or service is known as its tangible cost. Such costs are clearly known and are estimated accurately. The same applies to tangible benefits.
- There are intangible costs that exist but are not visible immediately. If management is sure that certain activities, such as timely procurement of raw material, are essential and that avoiding these activities may result in losses, then these are called intangible costs.
- Tangible benefits like doing jobs in fewer hours or producing error-free output data are quantifiable.
- Intangible benefits that result in the form of more satisfied customers or improvement in corporate image (as user of modern and efficient systems) cannot be easily quantified.
- Costs, directly associated with a system, are known as direct costs. For example, purchase of a pen drive for ₹ 500 is a direct cost as the association of the pen drive with the money spent is direct. Similarly, you can measure direct benefits.
- Direct benefits can also be attributed to a given project.
- Indirect costs, on the other hand, are not directly associated with a particular activity in the system.
- Indirect costs, on the other hand, are not directly associated with a particular activity in the system. They are classified as overhead expenses. For example, to install a computer system, you need some space. Thus, the cost of space, the cost of maintenance of a computer centre, the cost of lighting and air conditioning are all tangible costs. But it is not easy to quantify the proportion attributable to any specific activity.
- Fixed costs are sunk costs and are constant. They are one-time costs. For example, investment on installation of equipment for a computer system is a fixed cost, which is constant whether in use or not. Similarly, insurance and purchase of software are fixed costs. On the other hand, variable costs depend on the quantity of product and are incurred regularly; thus, these are proportional to the volume of work and continue along with the system in operation. For example, the cost of computerized forms varies in proportion to the amount of processing or the size of the reports.
- Measure and value the benefit in different components and aggregate them (add them suitably because there could be benefits of different types/ components flowing over a period of time).
- The same thing applies to all cost components: identify them, measure and value them and then aggregate them suitably.

NOTES

- Once the values of the total benefits and total costs are computed, it is simple to arrive at the net benefit as the difference between the two.
- The break-even analysis is an analysis device used in business organizations.
- Usually in long-term projects, it is difficult to calculate and compare the cost incurred today with the complete value of tomorrow's benefits.
- Present value is the value on a given date of a future payment or series of future payments, discounted to reflect the time value of money and other factors such as investment risk.
- Net Present Value (NPV) is the difference between the present value of cash inflows and the present value of cash outflows. NPV is used in capital budgeting to analyse the profitability of an investment or project. In other words, NPV is calculated by summing the rupee-valued benefits and then subtracting all of the rupee-valued costs, with discounting applied to both the benefits and the costs as appropriate.
- Cash flow is the inflow and outflow of money from business to determine the business' solvency.
- Cash flow analysis is the study of the cycle of your business cash inflows and outflows, with the purpose of maintaining an adequate cash flow for your business and providing the basis for cash flow management.
- Cash flow analysis involves examining the components of your business that affect cash flow, such as accounts receivable, inventory, accounts payable and credit terms.
- Payback analysis is simply a calculation of how long it will take to recover your investment. Simple payback is a common economic analysis method and is understood by most business owners. Simple payback is the amount of time it will take to recover installation costs based on annual energy cost savings.
- Structured analysis is a development method that analyses an existing system and directs it towards the development of specifications for the proposed system. This method permits the analyst to learn more about the system as well as the individual processes or activities in a logical manner.
- Structured analysis contains various components, such as graphic symbols, data dictionary, procedure and process description and rules.
- Diagrams and charts are very useful tools for communicating how processes work for the proposed system.
- It is used to depict the sequence of activities that a system must perform. The assumption in a flowchart is that activities occur in a linear sequence in which one event occurs first always followed by a specific second event, then a third, and so on.

NOTES

- Flowchart helps all concerned persons, such as systems analysts and programmers in developing an intermediate interface for understanding the logic of a system using symbols.
- A flowchart is a diagrammatic representation that illustrates the sequence of operations to be performed to get the solution of a problem.
- Flowcharts are generally drawn in the early stages of formulating computer solutions.
- The flowchart is a way of visually presenting the flow of data through an information processing systems and the operations performed within the system and the sequence in which they are performed.
- Symbols used in flowcharting describe what operations and in what sequence are required to solve a given problem.
- The domain chart is used to create partition for the system in terms of domains and sub-systems.
- The Gantt chart is also known as business charts or time lines chart. This chart is a horizontal bar chart developed as a production control tool in 1917 by Henry L. Gantt who is an American engineer and social scientist.
- The Input/Output (I/O) chart is an important tool that is used to visualize the order flow during system implementation and designing. It is generally associated with funnel model which basically filters the output.
- Decision table is an excellent tool for expressing complex logical relationships in a highly understandable and precise manner. A decision table is a matrix containing rows and columns which are used to define relationships. A decision table consists of two parts in which one is STUB and the other is ENTRY. The STUB is further divided into the upper and lower quadrant. The upper quadrant is known as condition STUB and the lower quadrant is known as action STUB.
- Decision tree is another method for defining complex relationships. A decision tree has as many branches as there are logical alternatives within software. It depicts the logical structure based on a stated policy. Decision trees are used when combinations of conditions need to be present. When policies change, it is easy to change trees.
- Decision tree is a powerful tool which is popular as it is used for classification and prediction. It shows some rules for using a decision tree.
- Decision trees models establish lower bounds for computational complexity that belong to certain classes of algorithms.
- Structured English is used to analyse decision process. It has a restricted vocabulary. It uses action oriented verbs like add, multiply, and so on.
- Pseudocode is used to express logic in the English language and has no conformation to any particular programming language.

- A data flow diagram illustrates the flow of data through a system and the work performed by that system.
- Context diagram helps in understanding the general characteristics of the system under study.
- A data structure is a set of data items that are related to one another and are collectively used to describe a component in a system. For example, a data item DATE can be further classified as DAY, MONTH, YEAR or a collection of these elementary data items form the data structure DATE.
- The sources of errors in decision-making are incorrect assumptions, inaccurate estimation of probabilities, undue expectations, incorrect measurement of utility function and errors in forecast.
- Data dictionary is one of the most important reference works that contain information about data which is also known as metadata. It is compiled by systems analysts and guides developers during analysis and design. It is created to assign specific meaning to every term used in the design leaving no chance of ambiguity on the interpretation of the meaning of the term used.
- Data values are the specifications about the permissible values or the range of values that can be accepted.

6.6 KEY WORDS

- **CBA:** CBA known as Cost-benefit analysis, it is done to determine how well or how poorly a planned action will turn out.
- **Monetary benefit:** It is the benefit that can be calculated in monetary units, such as rupees and dollars.
- **Tangible benefit:** It is the benefit that can be measured in numbers or can be quantified and calculated.
- **Intangible benefit:** It is the benefit that is neither tangible nor monetary but is qualitative in nature.
- **Gantt chart:** This chart is a horizontal bar chart developed as a production control tool in 1917 by Henry L. Gantt who is an American engineer and social scientist.
- **System flowchart**: It is a symbolic representation of solution of a given system design that is processed the flow control of entire system.
- **Visual table of contents**: It consists of tree or graph structured directory, summary of contents in each overview diagram, and a legend of symbol definitions.
- **Domain chart**: It is used to create partition the system into domains and subsystems.

- **Input/output chart**: It is an important tool that is used to visualize the order flow during system implementation and designing.

NOTES

6.7 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. What is cost-benefit analysis?
2. Define the concept of tangible or intangible costs and benefits.
3. Differentiate between direct and indirect costs.
4. List the various cost and benefit evaluation methods.
5. Write the formula of net present value analysis.
6. What is the formula for calculating payback period?
7. Name the three types of decision tree nodes.
8. Give the two advantages of decision tree.
9. Elaborate on the term structured English.
10. What does level 0 DFD provide?
11. Give the definition of decision-making process.

Long-Answer Questions

1. Explain the characteristic reatures of the following:
   (i) Cost-benefits analysis
   (ii) Tangible or Intangible Costs and Benefits
   (iii) Cost and Benefit Evaluation Methods
   (iv) Net present value
   (v) Payback Analysis
2. Briefly explain the basic tools of system analysis with the help of examfples and diagram.
3. Discuss briefly about the flowchart giving its significance and the various symbols used.
4. Explain the characteristics of the following charts:
   (i) Control Chart
   (ii) Domain Chart
   (iii) Gantt Chart
   (iv) Input/Output Chart

5. Describe the decision tables and its types giving examples.
6. Discuss about the decision tree and also explain the advantages.
7. Briefly explain the concept of structured English and write the four major consideration for using structured English in decision trees.
8. Elaborate on the basic elements of DFD with the help of examples and diagrams.
9. Discuss briefly about the data dictionary with the help of examples and E-R diagrams.

NOTES

6.8 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

NOTES

BLOCK - III
SYSTEM DESIGN

UNIT 7 SYSTEM DESIGN: AN INTRODUCTION

Structure

7.0 Introduction
7.1 Objectives
7.2 The Process and Stages of System Design
7.2.1 Logical and Physical Design
7.3 Design Methodologies
7.4 Structured Design
7.5 Form Driven Methodology
7.6 Major Development Activities
7.7 Answers to Check Your Progress Questions
7.8 Summary
7.9 Key Words
7.10 Self Assessment Questions and Exercises
7.11 Further Readings

7.0 INTRODUCTION

The entire process of designing the software system is an exercise which specifies the 'What' and the 'How' of the system. In order to design information systems, it is necessary for the systems analyst to comprehend the flow of information in an organization to assist in making decisions and to associate with the goals and objectives of the organization. Therefore, it is said that the systems analysis and systems design phases are deeply related. Input design, output design and file design which are considered as prime factors, are decided at the time of system design phase.

For the purpose of entering data, form and screen design are needed. The basic objectives of the input form design must be achieved by a good designed input form and visual display screen which can only take place by using the essential design principles and information about which type of data is required to be used in the system and recognizing the way by which the users react to various components of forms and screens.

The development activities include coding guidelines, implementing coding methodology, programming practice, code verification techniques, coding tools

and coding documentation. Programming refers to the method of creating a sequence of instructions to enable the computer to perform a task. While writing a software code, several coding tools are used along with the programming language to simplify the tasks of writing. On the successful conclusion of software design phase, the implementation phase begins where design specifications are translated into source code. Source code clarity is enhanced by following a good coding style, coding guidelines and verification techniques.

In this unit, you will study about the process and stages of system design, logical and physical design, design methodologies, structured design, form driven methodology and major development activities.

7.1 OBJECTIVES

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

- Understand the process and stages of system design as well as input design, output design and file design
- Know about the logical and physical design
- Explain design methodologies
- Discuss the basics of form design as well as styles and types of form
- Describe development activities

7.2 THE PROCESS AND STAGES OF SYSTEM DESIGN

The entire process of designing the software system is an exercise that specifies the 'What' and the 'How' of the system. This is not done in one step but is done in an iterative process that is based on 'What' the system will do as shown in the feasibility report. The systems design also tells how the system will do a particular job. This is given in the form of various programming constructs that designers specify. In any system there are three basic things: input, process and output. A customer who installs a system would want it to accept efficiently and reliably process the inputs made by him so as to produce corresponding outputs. Programs or routines that are stored in various files (linked together) do the processing.

In general, there are five components of a system design process. These are as follows:

(i) Output Design: Systems requirement tells what outputs are required. Before asking for a system, a customer wants certain outputs. This is the starting point of the design process. The output design requires proper knowledge of the systems requirements.

NOTES

(ii) **Input Design:** This is the second most important part of a system design. After deciding on the output requirements and finalizing the same, data that is required as an input to produce the desired output has is to be identified. The basic documents containing the details of input data which is a part of system requirement specification and used to serve as a guideline. It may become necessary to revise these documents and create new documents for this purpose. As far as possible, requirements should be finalized in Software Requirement Specification (SRS). Introducing fresh requirement at the design stage creates difficulties for systems designers.

(iii) **File Design:** When a system accepts input data, it has to be stored for processing purposes, either for a short or a long period. Also, this data is stored in a structured form generally kept in the form of files in a logical manner. Once data is stored, it is retrieved by the computer program either to process it or to display it. Hence, systems designers devise techniques of storage and retrieval of data from these files.

(iv) **Procedure Design:** This is the part that processes the data fed to the system and produces output. Procedures are written as programs which specify the manner in which processing will be carried out. Designers have to focus on various programming constructs to write efficient procedures or routines to process data. The two kinds of procedures adopted for this purpose are computer procedure and non-computer procedure. Computer procedure specifies functions that are to be carried out on computer. These procedures may have a different sequence in different programs that are run on the computer. These procedures are mostly application oriented. Non-computer procedures specify basic activities performed by the system to support computer procedure in feeding input data, receiving outputs, etc.

(v) **Control Design:** The control system is designed to properly sequence procedures or micro codes. It ties up procedures with data so that the design indicates the necessary procedures to ensure correct processing along with accuracy and timeliness of data. Then, seeing the results you feel assured that the system is functioning as per plan.

Although these components are classified under five categories, they are not separate from each other and are interdependent. Some of them are often used together with backtracking and traversing to arrive at a design that meets the requirements. This can be best described as two steps forward and then one step backward. In an iterative design, such look back type strategy is always better and results in a modified and upgraded product.

Figure 7.1 shows the four stages of the project, namely analysis, design, construction and implementation in an ideal situation in which one follows the other.

NOTES

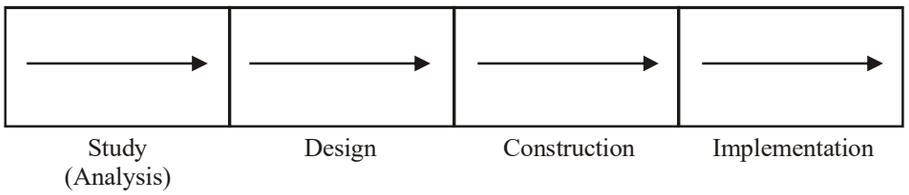


Fig. 7.1 *'Ideal' Project Progress*

An aspect that should be kept in mind by a system developer is the presence of constraints that may hamper work. These may be resource constraints, time constraints, and so on. For example, time constraints may lead to a hasty decision and an unsatisfactory systems design. Input design, output design and file design are the three operations preformed while processing with system design. These concepts are discussed in subsequent section.

Input Design

Input design is an important phase in system design phase. Businesses use the new technology to speed up the input process, reduce costs and capture data in new forms such as digital signature. Digital signature follows authentication mechanism. A code is attached with messages in the process of digital signature. Primarily, the signature is generated by hashing the message and then later this message is encrypted with the sender's private key. Digital signature is based on public key encryption. A signature confirms that integrity and source of message is correct. National Institute of Standards and Technology (NIST) standard recognized the Decision Support System (DSS) standard that basically uses the Secure Hash Algorithm (SHA). Message authentication protects digital signature because in that mechanism messages are exchanged from the third party. Digital signature is analogous to manual signature. The main objective of input design is to ensure the quality, accuracy and timeliness of data. The main objectives of input design are to select a suitable input and data entry method, reduce input volume, design attractive data entry screens, use validations checks to reduce input errors, design required source documents as well as forms and develop effective input controls. Systems analysts apply business process engineering techniques when studying transactions and business operations to determine how and when data should enter the system. Data entry is the process of manually entering data into the information system usually in the form of keystrokes and mouse clicks. Data input is the operation of encoding the data and writing them to the database. From a user's point of view, the interface is the most critical part of the system design because it is considered as a checkpoint through which users's can interact with the system. For this, it is essential to construct models and prototypes for user approval. You can prepare initial input screen design to users in the form of storyboard which represents a rough layout showing the sample screen for proposed system. Information systems development must address following broader organizational issues as well:

- What is the justification for the type of application which has to be developed?

NOTES

- What actual or desired processes should the application perform?
- How will we verify that the application performs as per design?

Software Development Life Cycle (SDLC) methodology allows a project team to successfully build an application which is suited for the organization's needs. It also allows the organizations to incorporate the new requirements, technology and human resources to Information Technology (IT) development. Input data is considered as prime tool for SDLC steps. Following are the activities performed in design methodology:

- System planning involves the project requirements summary, project team description and preliminary work schedule and service area's demographic analysis.
- System analysis is processed with E-R diagram, data flow diagrams and high level functional description.
- System design involves working with Relational DataBase Management System (RDBMS), data dictionary, identification or description of database objects and Web site map.
- Systems implementation involves working with Microsoft Access database, Web site content and elementary test plan.

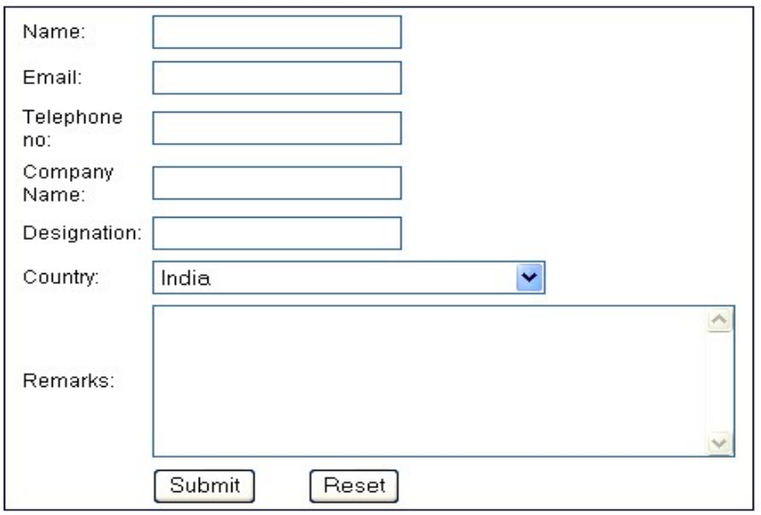
Input design in SDLC has been drastically changed. In the network era, many input devices and techniques are available. Business and organizations use new technology to speed up the input process, reduce costs and capture data in new forms, such as digital signature. Digital signature supports authentication mechanism. A code is attached with messages in digital signature. Primarily, the signature is generated by hashing the message and then later this message is encrypted with the sender's private key. This concept is based on public key encryption. A signature confirms that integrity and source of the message is correct. National Institute of Standards and Technology (NIST) standard recognized the digital signature standard to use basically the SHA. Message authentication protects digital signature because in that messages are exchanged with the help of third party. Digital signature is analogous to manual signature. Hence this signature is considered as a prime tool for input design in online transaction. It attaches date and time along with author of the signature. It authenticates the contents when signature has been completing. It solves the disputes using third party generally in online payment by PayPal. It also ensures that message has not been altered while sending and receiving from one user to another. The message can be electronic documents, such as e-mail, text file, spreadsheet, etc. The data entered into an information system checks the quality of the output as well as quality of input. This concept is sometimes known as Garbage In, Garbage Out (GIGO). The main objective of input design is to ensure the quality, accuracy and timeliness of input data. Let us take an example of Geographical Information System (GIS) in which input data refers to data sources for creating new data that includes remotely

NOTES

sensed data, for example satellite images, aerial photographs, Global Positioning System (GPS) data and paper maps. Remotely sensed data and GPS data are the primary data sources and paper maps are secondary data sources which are discussed below:

- **Remotely Sensed Data:** Remotely sensed data, such as digital orthophotos and satellites images are the types of data which are acquired by a sensor from distance whereas remotely sensed data are raster data but they are useful for vector data input. Digital orthophotos are digitized aerial photographs that have been differentially rectified or corrected to remove image displacements.
- **GPS Data:** GPS data includes the horizontal location based on the geographic grid or a coordinate system. It has become a useful tool for spatial data input.
- **Paper Maps:** These include all types of hard copy maps.

Following screen shows the screen of various types of input data in which **Name:, E-mail:, Company Name:, Designation:, Remarks:** fields have been taken as string data type whereas **Telephone no:** has been taken as integer data type for inputting the data. Two buttons, known as Submit and Reset have been interfaced on the screen to submit and reset operations respectively and **Remarks:** field has been set for user's feedback for the designed screen.



Input data is determined while the time of input designs. It ensures the quality, accuracy and timeliness of input data. It is determined how data will be captured and entered into the system during input design. Data capture uses an automated or manually operated device to identify source data and convert it into computer readable form, for example credit card scanners, bar code readers, etc. Table 7.1 summarizes the list of various input devices and their functions which are used to input data for designed system.

NOTES

Table 7.1 *Input Devices and their Functions*

Input Devices	Functions
Feedback devices	These types of devices create a digital image of biological data, such as fingerprints, retina patterns and facial characteristics.
Brain computer interface	This interface is used to translate the neural brain signals into digital commands.
Digital camera	This device records the photographs in digital forms rather than traditional film. The resulting data file can be stored, displayed or manipulated by the computer.
Electronic whiteboard	This device captures and stores text or graphics which are displayed on the board and functions as a touch screen.
Handheld computer stylus	This device allows users to form the characteristics on the screen of a handheld computer. Programs are handwriting recognition software that can translate the characters into digital input.
Internet workstations	This device enables the users to provide input to Web based Internet or intranet recipients who can integrate with information system output or personal computer applications.
Keyboard	This device is considered as prime device for inputting the data.
Microphone	This device is used to convert sound waves into digital information that can be stored, transmitted or attached to e-mail. Users can input data and issue commands using spoken words.
Motion sensors	This device is used to estimate an object's motion in a three dimensional position. This device is also used in virtual reality programs and gaming applications.
Pointing stick	This device is also known as pressure sensitive pointing device which is located between the keys on a notebook computer. This device resembles a button or pencil eraser.
Scanner and optical recognition	This device reads printed bar codes, characters and images. An Optical Character Recognition (OCR) device is used to convert scanned material to get the digital information.
Terminal	This device is especially used in intelligent screen, i.e., keyboard for independent processing.
Touch screen or pad	The sensors allow users to select options by touching the specific locations on the screen. The touch pads are used as pointing devices that use touch screen input.
Video input	It produces result in digital form. The produced data can be stored and replayed for further requirements.

Input data controls the necessary measures, such as input values are correct, complete and secure. This control is focussed on every phase of input design which starts with source documents and promotes data accuracy and quality. If batch input method is used then the computer can produce an input log file which identifies and documents the data entered. Examples of input controls include the following factors while the time of inputting data.

- **Check Digits:** Check digits are numbers produced by mathematical calculations performed on input data, such as account numbers. The

NOTES

calculation confirms the accuracy of input by verifying the calculated number against other data in the input data, typically the final digit.

- **Completeness Checks:** Completeness checks confirm that blank fields are not input and that cumulative input matches control totals.
- **Duplication Checks:** Duplication checks confirm that duplicate information is not input.
- **Limit Checks:** Limit checks factor confirms that a value does not exceed predefined limits.
- **Range Checks:** Range checks confirm that a value is within a predefined range of parameters.
- **Reasonableness Checks:** Reasonableness checks confirm that a value meets predefined criteria.
- **Sequence Checks:** Sequence checks confirms that a value is sequentially input or processed.
- **Validity Checks:** Validity checks confirm that a value conforms to valid input criteria.

An audit trail must be provided so that the source of each data item is entered into the system. The data security policies and procedures protect data from damaging. Input controls help to ensure the employees accurately for input information. Systems properly record input and systems either reject or accept and record and also input errors for later review and correction while the time of system development. An input device is attached to the computer and is helpful in data processing. This device is also known as online input device and data is said to be entered online. For this, computer terminal is required to be attached to the computer. It consists of keyboard for data entry and other means of displaying entered data. Data can be entered using typing devices (keyboard), pointing devices (mouse, joystick, light pens) and voice recognition devices (microphone). Input processes must be logically arranged while the time of setting transactions. Business operations determine how and when data should enter is the system. It has two prime methods known as batch input and online input methods.

- **Batch Input:** Data entry is usually performed in a specified time schedule, such as daily, weekly, monthly and for longer periods. For example, batch input occurs if a payroll department collects time cards at the end of the week and enters data as a batch. Another example of batch input can be to enter all the grades for academic terms with the help of batch processing system.
- **Online Input:** The batch input is used in specific situations in which most business activity requires online data entry.

NOTES

The detailed design phase of SDLC carries out construct and test activities to check, redefine and then revalidate the design. The development team checks the following factors in the detailed design phase:

Elaborate Design

This kind of design is built on the high level design to incorporate the designs and mechanisms with reference to its state, attributes and association and interdependencies.

Finalizing Architecture of the Project

This part of the detailed design considers all the factors before finalizing the architecture of the project and conveys decision to the developers and the systems analysts.

Integrating with Target Environment

This type of design is considered in the development phase based on the choice and applications it has to work with. For example, one may choose SQL (Structured Query Language) Server, WebSphere, EXtensible Markup Language (XML) or any other application with which the system will be integrated. During this process some infrastructure constraints should also be kept in mind.

Plan Deployments

This factor dictates the way and the time of the deployment with the system with which it is to be linked. The system under development may be implemented with Enterprise ARchive (EAR) files, Java ARchive (JAR) files, etc.

Specify Test Details

This factor identifies all those cases that qualify as test cases. This also includes descriptions of the tests to be carried out, inputs needed for these tests, the condition of execution and results expected.

Detailed Design Phenomena

The detailed design in the systems designing phase is taken as an input–process–output design. During the stage of input design, developers have to consider devices to be used for input. Such input devices may be a desktop PC, a pager, a Magnetic Ink Character Reader (MICR), an Optical Character Reader (OCR), an Optical Mark Reader (OMR), etc.

Input data shown in a Data Flow Diagram (DFD) turns into input modules and data in the structural chart. Each input enters a process on a DFD. Automation of input is worked with electronic notepads or barcode scanning, and this reduces the likelihood of human error while making data entries. Once the devices, controls and input fields are identified, the developers take into consideration the input format either through forms or input screens. There must also be a dialog or interaction between the system and the user. Java, C++, Visual Basic and C# are

development languages that contain libraries which speed up the process that generates reports and screen displays.

Objectives of Input Design

NOTES

Input is the raw data that is processed to give output in an information system. Therefore, the quality of system input determines the quality of system output. Well designed input forms and screens should meet the following objectives:

(i) **Effectiveness:** It must serve specific purposes. Forms are created to serve one or more purposes, such as recording, processing, storing and retrieving information for business.

(ii) **Accuracy:** It assures proper completion. Error rates associated with collecting data drop sharply if the forms are designed to ensure accurate completion.

(iii) **Ease of Use:** It should be straightforward. To reduce errors, check speed completion and facilitate the entry of data, it is essential that forms be easy to fill in. Forms should be designed with proper flow either from left to right or top to bottom. Logical grouping of information makes it easy for people to fill out forms correctly. The groupings are as follows:

- ***Consistency*:** Groups the data similarity from one application to the next.
- ***Simplicity:*** Keeps data uncluttered.
- ***Attractiveness:*** Focuses on user's attention.

All of these objectives are attainable through the use of basic design principles, through the knowledge of what is needed as input for the system and through an understanding of how end users respond to different elements of forms and screens. Forms have certain primary sections including instructions, heading, signature and verification, body, totals and comments.

Selection of Input Media and Devices

To achieve the main objective of developing a software system, the input data entered by the end user needs to be converted into a format that the computer understands. Also, to receive user input, one needs to use devices for conversion of data on documents at source into a variety of storage media like floppy diskettes, magnetic tapes, Compact Disks (CD) or flash drives. The two ways of data conversion include data conversion in which human beings are responsible for performing the conversion and those where hardware devices are responsible for performing the conversion activity.

As hardware devices are more reliable and less error prone when the conversion activity takes place, the second method of data conversion is preferred. However, since hardware devices are inflexible, they are more suitable for such situations that involve a large amount of specific type of data conversion as

NOTES

processing of cheques by banks and payments by credit or debit cards at grocery stores. In real life situations of business data processing, conversion is performed by the combined activities of humans and hardware. For example, human beings make data entry through keyboard and this is followed by conversion by a hardware device that converts the data into a code that is machine readable. In a situation like this, editing is very important as it minimizes the number of errors. Therefore, the main objective of input designing is to minimize the human element for balancing the slow input speed and error proneness of the end user input.

Key Tasks in Input Design

The following are the essential tasks involved in the input design:

- Designing data entry and input procedures.
- Designing source documents for data capture or devising other data capture methods.
- Designing input data records.
- Designing data entry screens.
- Designing screens for user interface.
- Designing security measures for the system and audit trails.

Data Capture

Data capturing is the identification and recording of source data. The following factors are involved in this process:

- Decision on data to be captured or accepted as input. Once input to the system is decided computer does the rest.
- Ensuring enough capacity with the system to handle the entered data. Fewer steps in data input will have fewer errors. For example, online forms are started with a well designed form so that a source document is used for data input.
- It is at times appropriate to reveal information via a code. For example, a salesperson uses codes to locate goods and goods codes allow meaningful data entry. The functions code can be used to request appropriate action from a computer or decision maker.

Guidelines for Establishing Code

The following points, with respect to codes, should be kept in mind:

- They should be kept short.
- They should be kept stable.
- They should be unique.

- They should not be confusing.
- They should be uniform.
- They should be modifiable.
- They should be made meaningful.

NOTES

Input Integrity Controls

Input integrity controls cover a number of techniques that reduce errors while the end users make inputs. An input control is done by checking on the value of individual fields. Here, just filling the format is not sufficient. Completeness of all inputs such that all the relevant fields are filled is important. Transaction logs are used to create audit trails for data entry and other system activities. This keeps a track of modifications that are made in the table of database providing security and means of recovery in case of a hardware or power failure, etc.

Screen Design Guidelines

The following are the guidelines to develop a screen design:

- The display screen should be simple and consistent and should show only that which is necessary for the end user.
- Screens can be kept consistent if information is located in the same area each time a new screen is accessed. Similar information should be consistently grouped together.
- Facilitate the navigation of end users among various screens. Web based forms facilitate movement with the use of hyperlinks.

Output Design

A smart systems analyst has to choose among existing alternatives before implementing an output format in which the output may be desired. A suitable output format should be selected depending upon the requirement of the output. Several formats are available from which selection can be made to communicate information. Narrative format uses standard text, numbers and pictures. It is a format that has been exploited by word processing technology for reports, personalized form letters as well as business letters.

Tabular output is the format that uses columns of text and numbers to present information. This method is most common for presenting computer outputs. Zoned output places the text and numbers in the selected areas of a form or a screen. Graphic output uses graphs or charts to represent information. This format enables end users to better understand the relationships between data and trends. Pie chart, bar chart, column chart, line diagram and scatter diagrams are the most commonly used graphs.

Pie charts depict the relationship between parts to the whole at some specific period. Certain pie chart styles are used to emphasize a particular item. A pie chart should be used to compare seven or fewer portions.

NOTES

Bar chart is an output format that represents information using bars. Categories or classes for comparison are organized in a vertical manner whereas the values are arranged in a horizontal manner. A stacked bar chart portrays the relationship of individual items to its 'whole.'

Column chart is a simple variation of the bar chart. Column charts enable the user to depict comparisons among items and show the variation in an item over a period of time. This chart organizes categories horizontally and values vertically. Such appearance emphasizes variations over a period of time.

Line charts show trends of an item over a period of time. It organizes items on the horizontal axis and measurements on the vertical axis.

Those formats that plot data values of two different items that reflect uneven intervals of data are called scatter charts. Standard statistical methods are applied to establish the degree of existing correlation.

Output Integrity Controls

Such controls consist of routing codes identifying the receiving system as well as verification messages which confirm the successful receipt of messages. The network protocol can handle routing and verification. Therefore, these services may not be required by the application. The common forms of output are messages, screens and reports. Screen format reports and printed reports should indicate the date or time of the data. It should also indicate the date or time at which the report was printed. It should contain other information, such as the time of the report data along with description and title of the report as well as the pagination of those reports that have multiple pages. Usually a version number along with an effective date is mentioned on pre-printed forms.

File Design

The file design includes the database design with reference to SDLC because in big organization the databases are required to maintain the various types of files. A file is a collection of data that can be read from or written to. A file can be a program. You can create a file to write the data and store in suitable device. Commands, printers, terminals and application programs are all stored in files which allow users to access diverse elements of the system in a uniform way and give the operating system great flexibility. No format is implied when a file is created. Files are used for all Input/Output (I/O) of information in the operating system. Input occurs when the content of a file is modified for read from or written to. Output occurs when the content of one file is read or transferred to another file. For example, to take a hardcopy printout of a text file, the system reads the information from the text file and writes the data to the file. The collections of files are often related to each other and storing them in a structure of directories keeps them organized. There are many ways to create, use and manipulate the files. In SDLC, the file design depends on the mode which is chosen for the scope of

software project. It depends on the framework of the scope of the project, project budget, organizational environment and available resources. The file design basically optimizes performance of database designing. The significant improvement of machine is recognized after the file designing phase. The physical file design holds the addressing techniques, compaction techniques, storage devices and physical layout of data. The physical layout is concerned with partitioning of data, clustering of data, organization of files and data elements. The file blocking is used to maximize the performance which reduces the requirements of memory space. The addressing techniques are used for database tables and flat files. The file design supports sequential layout, indices, hashed, inverted lists, etc. The network and hierarchical file design supports contiguous list and pointer methods. The compaction techniques use the packed fields to support compression of images, text and executables. The compressed file design reduces the requirements of increasing amount of data that can be transmitted via compressing and decompressing data. The trade-offs are evaluated via storage devices. These devices can be main memory, optical disks, solid state disk, magnetic disk, etc. The storage device matches the response time requirements. The devices are used to cache or buffer and hence are helpful in designing and creating the file. The system analyst and software development team collectively generate prototypes and mockups of the screens, reports and processes. If prototype is generated for the required file, the minimum amount of code justifies the type of fields including database. A prototype is required for necessary elements that produces production module. The prototype is not intended for business point of view. The file is to be considered for a computer as one dimensional sequence of bytes. The content of file can be of different types, such as integer values, texts, image pixels, audio/video content, multimedia effect, etc. The file represents the following interfaces:

- **Tree of Directories and Files:** This interface contains file and directory names which are named with path names.
- **Object:** It deals files and directories along with other kinds of objects, for example devices.
- **Path Name:** It contains a component name for each directory in the path.

File is a passive container of bytes on disk. The open file option is an active source or sink of bytes in a running program. These files are connected to a device. The file Open and file Read are the two frequently used options in computer applications. The file name extension represents the set of characters which help to make the operating system know what program is used to open it and what kind of information is kept into the file. The extension name appears at the end of the file name following a period and having three characters. Sometimes, you might occasionally encounter an error when copying a file with a very long file name to a location that has a longer path than its current location.

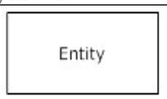
NOTES

You must know the basic concept of file and other related terminologies**.** A number of main memory blocks are associated with message tags sent from client side. The tags are required for addressable memory. The file design technique considers basically the concept of caching, system calls and hierarchical file naming systems, objects and operations, and required connection in protocols mechanisms. The role of system analyst in file designing is to include those aspects of database design that are usually not visible to the users and applications. The objective of physical file design is to optimize the performance. When a computer system is powered up a process known as cache block used for replacement can take place during the process of loading the operating system where complex parallel programs run. The required arranged programs for file design run logically in the operating system. The hierarchical file system arranges all the directories in which the basic feature of file design supports a notion of records which contain named fields. Directory is a database containing descriptive information of entries. For example, a directory can contain files related to log in users along with name, telephone number, e-mail address, network resources, such as printers and faxes. In the computer world, the basic directory structure for all types of digital library collections is same. A Web directory is a directory on the World Wide Web (WWW) which specializes in linking to other Web sites and categorizing those links. The searchers look at sites organizing in a series of categories and menus. Web directories are smaller than search engine databases. File designer and systems analyst consider various tools that are used to design a file. The tools used to design a file by file and system designer are as follows:

Entity Relation Diagram: Entity Relation Diagram (ERD) is a high level data model that includes all major entities and relationships. It illustrates the logical structure of databases. Data models are tools used in analysis to describe the data requirements and assumptions in the system from a top-down perspective. They are also set the phase for the design of databases in SDLC. The entire system and its components are required to develop an ERD. ERD brings out issues, such as ambiguities, entities and their relationships, the types of data needed to be stored and the degree of a relationship. In 1976, Peter Chen developed ERD but Charles Bachman and James Martin have introduced more basic ERD principles. Table 7.2 displays the ERD notations and their corresponding symbols.

NOTES

Table 7.2 ERD Notation and their Symbols

ERD Notations	Details	Symbols
Entity	Entities are objects or concept about which information is stored.	
Attribute	Attributes are the properties of an entity.	
Relationships	Relationships control the data flow in which two entities share information in the database structure.	
Cardinality	It specifies the maximum number and the absolute minimum number of relationships.	

An entity relationship diagram is a data modelling technique that creates a graphical representation of the entities and the relationships within an information system. An entity may be optional, for example a sales report could have no customers or could have one or many customers or mandatory, for example there must be at least one product listed in an order. There are several types of cardinality notation, such as crow's foot notation which is frequently used in entity relationship diagram. In entity relationship diagram, a single bar indicates one, a double bar indicates one and only one, for example a single instance of a product can only be stored in one warehouse, a circle indicates zero and a crow's foot indicates many. The three main cardinal relationships are popularly known as one-to-one (1:1), one-to-many (1: M) and many-to-many (M:N). Cardinality and modality are indicated at both ends of the relationship line. Once it is done the relationships are declared as being one-to-one (1:1), 1 to many (1: M) or many to many (M: M).

System Flowcharts: System flowchart is a symbolic representation of solution of a given system design that is processed with the flow control of entire system. This flowchart is basically used by system analysts through which they go through to design the successful system implementation. Let us take an example of system flowchart of online air ticket booking system. For this, two master files are taken for successful system implementation. One of the files is maintained for admin side and other one is maintained for travellers. For the above system design, analysts will go through a basic questionnaire as follows:

- How many flight numbers will be taken for landing and taking off?
- Travelling would be taken as round trip or one way and from where to where?
- The classes and choices of meals of travellers will be recorded at which stage in system flowchart?
- How will the whole system transactions run successfully?

NOTES

- What type of front end support (applications) and back end support (database server) will be implemented to design the system?
- What total cost will come in the whole transaction during system implementation?

Basically, system flowchart describes the data flow for a data processing system. It provides a logical diagram of how the system operates. It represents the flow of documents and the operations performed in data processing system. It also reflects the relationship between inputs, processing and outputs. A system flowchart, also known as data flowchart, is used to describe the flow of data through a complete data processing system. Different graphic symbols represent the clerical operations for input, storage and output equipment. Although the flowchart may indicate the specific programs but no details are given that how the programs process the data. The features of system flowcharts are as follows:

- It includes the sources from which data is generated and device.
- It involves logical step for various processing.
- It supports to prepare the intermediate and final output for the devices used for storage purpose.

The document symbol is used to display the procedure which is defined in the flowchart along with text. Arrow sign of system flowchart shows how flow of control will go. Shared file or shared disk keeps the record of information of stored transaction. After inputting the transactions, control will go to validate the transactions. If any error occurs at this stage, error report will be displayed otherwise the validation part will be checked completely. Transactions will be sorted either in ascending or descending order to make the master file. File updating process is maintained by the system analysts at regular interval. At last, financial report is generated to produce the total cost during system implementation.

Functional Decomposition Diagram: The Functional Decomposition Diagram (FDD) is basically business planning tool shown in the hierarchy of business functions, processes and sub-processes within an organization. The functions are decomposed or broken down into processes to make complex systems easy to understand and analyse. The objectives of FDD are as follows:

- It helps to understand the rules and style guidelines for FDDs.
- It helps to understand the process which is helpful in creating FDDs.
- It shows, on a single page, the capabilities of an organization that are relevant to the consideration of architecture.
- It breaks down a system step-by-step, beginning with the main function of a system and continuing with the interim levels down to the level of elementary functions.

NOTES

In system development life cycle procedure, decomposition is the process of starting at a high level and moving into smaller subsystems. It is a type of planning tool to identify the business functions and processes that comprise them collectively. Data flow diagram is taken at the starting point which provides a detailed process diagrams whereas the functional decomposition diagram is not able to depict the process. This diagram shows the hierarchical organization of functions and the processes that they include. This diagram contains a group of relevant functions which are cross organizational collections of activities that have to be performed further during system design. The functions are related to various departments of organizations, such as research and development, sales, purchasing, marketing, etc., where system design has to be implemented. Each function is defined along with name and descriptions. Descriptions in functions describe the collection of activities which are used by system analysts. System analysts make up their mind what systems are required within the organization to support the business. Functions are declared in the FDD that are connected with a straight line. The function declaration depends on organizations whether they are complex or simple. Analysts break down the functions into simpler components to make easy plan for system designing. A function is a parent to a sub-function and a sub-function is a child of a function. If a function only has one child, the child usually is not included on the FDD. Also, a child can have only one parent. FDD's are also called structure charts. They are used by system analysts to model business functions and show how they are organized into lower level processes. Those processes are translated into program modules during application development. FDD is created as an organizational chart, i.e., starting at the top and work the way down step-wise-step. In the fundamental analysis technique, complex problem is broken into successive layers of more manageable and comprehensive tasks resulting in a hierarchically structured function chart which describes the problem as well as solution in levels of increasing detail. In this technique, the lower level functions or processes completely describe the parent. A function or process at a lower level cannot exist unless it is included within its parent function or process. It drives the analysis process. The logical functions and processes become the subject and therefore they focus on the organizing interviews and verification activities. Similarly, the logical processes will eventually be organized into application systems. FDD approach is arranged in a systematic way to ensure that each function and process is conceptually and operationally independent. The aim of FDD is to illustrate functions which are highly cohesive and loosely coupled. This approach makes them conceptually and operationally independent and promotes more stable business systems. It takes two frequently used approaches, such as coupling and cohesion. Coupling is a measure of the degree to which two functions are interdependent. Loose coupling is considered as good process because changes to one function are made with less impact on other functions, i.e., you do not have to know about other functions to make changes to the function. Cohesion is a measure of the strength of the processes within a function. High cohesion is

NOTES

considered as good because highly cohesive functions perform well defined objective which is easy to understand and maintain. The FDD helps the system analysis in the following ways:

- The simplicity of the structure and representation aids help in understanding the breakdown of functions and processes.
- It specifies the precise requirements and features in which each function becomes easier because the functions and processes are broken down into smaller units.
- The partitioning and independence of the functions localizes errors and minimizes system faults.
- It allows the customer to view and discuss the organization in a form that can be used with a collection of functions rather than as a continuous process.

There are many loopholes in FDD that hinder the system designing in subsequent stages:

- It hinders to understand an organization's business when functions and processes overlap.
- It creates unnecessarily numerous and complex interfaces.
- It requires extensive rework of other Process Modelling (PM) methods, such as data flow diagrams in later stages as detail problems emerge.

Structured Flowcharts: The structured programming was introduced between 1960 and 1970 that is brought with the concept of structured flowcharts. The structured flowcharts specify the conventions to link the various symbols together into a complete flowchart. Flowcharts are structured because the structured programming paradigm evolved with it from the mathematically proven concept in which all problems can be solved by three types of control structures, such as sequence, decision or selection and iterative or looping. The definition of structured flowcharts used further defines the types of sequential structures, for example process, input or output, and subroutine call and the types of decision structures, such as single branch, double branch and case. In structured flowcharts, sequences make statements that are executed one after another, decisions make one or two blocks of program code which is based on a test for some condition and loops called iterations make one or more statements that are executed repeatedly as long as a specified condition is true. The four types of iterative structures are also dealt in structured flowcharts, such as test at the top, test at the bottom, counting and exiting. For example, the program is designed to aid the programmer in designing and presenting structured flowcharts. These types of flowcharts have many advantages as follows:

- They take less time to comprehend. They produce fewer errors in understanding. They give students more confidence in their understanding of an algorithm.

- They reduce the time spent answering questions about an algorithm and also reduce the number of times the students need to look at an algorithm.
- The advantage of structured flowchart is to create correct programs that are easy to write, understand and change. The relationship between the various procedures shows the modular design of the flowchart using flow control.
- A correctly designed structured flowchart is easily understood by another programmer.
- Modular design increases the programmer's productivity that looks at the file design first. Several programmers can work on a single, large program, each working on a different module. Studies show structured programs take less time to write than standard programs.
- Each procedure is specialized to perform just one task in structured flowchart in which a procedure can be checked individually. The logic of such type of flowchart is cluttered with details and therefore difficult to follow.

7.2.1 Logical and Physical Design

Software implementation requires proper planning. A design is of no avail if it can not be implemented. Implementation is a process that converts logical design into physical design that can be made operational such that users can take over to operate and evaluate it. Once the design phase is over, developers have to put these together and tested for operation involving users. In the design stage individual components are tested but the real task of putting these together so that it works as a system to the satisfaction of the user who wanted this system to behave the way he/she wanted.

In the implementation stage, no changes are made. In all previous stages, the developers had freedom of making changes, if any, in the software design plan. In implementation stage there is back tracking.

Just putting these together is not important. It is important to make the system operational and user is able to perform tasks that it desired to perform. This is the stage when developers get the first taste of their actual work and evaluate themselves and their work. In this stage, developers see the system from use points of view to see whether it is up to the mark and whether user will be able to get the expected output and performance.

This stage is most important phase in the whole project as since bugs and errors are found in the system for the first time. Different nature of errors is determined in this stage and debugging and error corrections are made. Final documentation of the developed software is also done in this stage. Such documentation has great bearing as these are referred later, whenever problem arises in subsequent stages of evaluation and maintenance. Documentation provides enough scope of re-evaluation of the processes created by developers.

NOTES

Uses of Implementation in Different Models

Interpretation of implementation stage is different for different Software Development Life Cycle (SDLC) models. Implementation suggests next development stage and is dependent on development model adopted. In general there are two types of model and all other models are combination of features of these two models. Viewed from implementations, there are two general types of the SDLC point of view model:

- Waterfall model
- Iterative model

Implementation in Waterfall Model

Implementation stage in this model can be understood better by having some basic knowledge of the model first.

This model is traditional in development of software. It follows a simple sequence of planning, design, implementation, testing and integration with the client's working environment. This workflow makes one understand how this model looks at implementation stage. Real hard work is put by developers since only after this stage, there would be testing and evaluation before final acceptance. If everything has gone well, it will lead to successful implementation with ease.

In this model, documentation is quite easy. Developers may refer to documentation in case something goes wrong or a bug is found in the developed software. If the waterfall model has been followed by they must ensure that documentations is complete; failing which they may have to work real hard in going back to the previous starting point, i.e., in the system design phase.

Implementation in Iterative model

This model is different and the implementation will also be different. Implementation stage in an iterative model is less critical when compared to waterfall model. This model creates a prototype from the beginning. Changes are easy to make in an iterative model. That means there will always be implementation in the iterative model. In this model, checking can be done at every stage and it is easy to see how they have developed so far. The user participates from the beginning and thus pressure of demonstration on the operational aspect is not that critical. This model ensures better implantation with minimum problems.

Software implementation should be done with proper planning, hasty decisions lead to problems and delays. The process of implementation of software should be consistent with least amount of disturbance.

Prerequisite for Smooth Implementation of Software

Following are the some of the basic considerations that should be kept in mind for smooth implementation of software:

NOTES

Proper Equipment

The hardware and software requirements should be first re-examined with the software supplier. The equipments for general-purpose applications should be delivered several weeks before the installation of an application. This helps to have a basic idea about the hardware before major implementation of the applications. Sufficient time for networking should be given, if the system uses the network resources. Good quality wiring and the fastest hubs should be used to gain best performance from the system. Proper connectivity devices, such as wiring, hubs and routers should be used to reduce bottleneck arising in system performance.

Conversion

Conversion methodology ensures a professional result in line with client expectations and within budgeted time period and cost. The conversion methodology clearly improves communication between the project team and management by providing a readily understandable and structured approach.

Training

Formal training about the functioning of software should be provided to the employees for the successful use of the application software. Hardly ever, one reads a manual and implements the application in a smooth manner. Application training should be designed to teach users how to use the software.

Implementing Applications

Follow a definite sequence to install all the inter-related applications. For example, an application consists of general accounting, payroll and utility billing. In the application, accounting will come first because the other systems require its availability.

Backups

Regular system backups should be taken so that the users can revert to an older copy of the data files when they commit any mistake. The backup procedure can be performed during the free hours or at the end of the day. You can take the backup of only the critical points in an application. Backups are stored in the removable disk or in high capacity tapes. One can also store backups in another folder of the hard disk drive of the system.

NOTES

Table 7.3 summarizes the difference between logical and physical design.

Table 7.3 *Difference between Logical Design and Physical Design*

Logical Design	Physical Design
In logical design, decisions are made about how the interaction and integration will take place between these elements.	In physical design, decisions are made to implement the logical design successfully.
At the time of designing, all the logical errors can be checked while integrating the various system components.	The technology used to implement the design is decided in physical design phase. This approach supports the constraints found during the implementation.
During logical design, we identify the important objects that need to be represented in the database and the relationships between these objects.	During physical design, we decide how the logical design is to be physically implemented (as tables) in the target DBMS. Therefore, it is necessary to have selected the target DBMS before proceeding to physical design.
Logical design is the process of constructing a model of the data used in a company based on a specific data model but independent of a particular database and other physical considerations, such as files and documents.	Physical design is the process of producing a description of the implementation of the database on secondary storage.
It describes the base tables, file organizations, and indexes used to achieve efficient access to the data, and any associated integrity constraints and security restrictions.	It does not describe indexes used to achieve efficient access to the data.
The logical design phase of the methodology is divided into following main steps: • In Step 1, we create a data model and check that the data model has minimal redundancy and is capable of supporting user transactions. The output of this step is the creation of a logical data model which is a complete and accurate representation of the company or part of the company that is to be supported by the database. • In Step 2, the E-R model is mapped to a set of tables. The structure of each table is checked using normalization. Normalization is an effective means of ensuring that the tables are structurally consistent and logical with minimal redundancy. The tables are also checked to ensure that they are capable of supporting the required transactions. The required integrity constraints on the database are also defined.	Physical design is divided into six main steps: • Step 1 involves the design of the base tables and integrity constraints using the available functionality of the target data. • Step 2 involves choosing the file organizations and indexes. Typically, a number of alternative file organizations for data are provided which tend to have a fixed storage structure. • Step 3 involves the design of the user views originally identified in the requirements analysis and collection stage of the system development life cycle. • Step 4 involves designing the security measures to protect the data from unauthorized access. • Step 5 considers relaxing the normalization constraints imposed on the tables to improve the overall performance of the system. This is a step that you should undertake only if necessary because of the inherent problems involved in introducing redundancy while still maintaining consistency. • Step 6 is an ongoing process of monitoring and tuning the operational system to identify and resolve any performance problems resulting from the design and to implement new or changing requirements.
Logical design is concerned with the 'what'. Logical design is independent of implementation details, such as application programs, programming languages or any other physical considerations. The output of this process is a logical data model that includes a set of relational tables together with supporting documentation, such as a data dictionary. These represent the sources of information for the physical design process which provides you a mechanism for making trade-offs.	Physical design is concerned with the 'how'. However, physical design is not an isolated activity because there is often feedback between physical, logical and application design. For example, decisions taken during physical design to improve performance, such as merging tables together might affect the logical data model.
In the logical database design phase, the logical representation of the database system is built which includes identification of the important entities and relationships and then translate this representation to a set of tables.	In the physical database design phase we decide how the logical design is to be physically implemented in the target relational DBMS. This phase allows the designer to make decisions on how the database is to be implemented.

7.3 DESIGN METHODOLOGIES

NOTES

In order to design information systems, it is necessary for the systems analyst to comprehend the flow of information in an organization, how it assists in making decisions, and how it is associated with the goals and objectives of the organization. Therefore, one can conclude that the systems analysis and systems design phases are deeply related. While collecting data about the problem and determining the end user requirements, the systems analyst obtains so that the organization's ideas to improve the flow and usage of information goals are achieved.

Various design methodologies, such as structured design and Hierarchical Input Process Output (HIPO) charts are available that the systems analyst can use to design a good software system. While developing the design of a software system, the systems analyst can determine various design specifications such as input/output subsystems, processing and database design, and so on. Therefore, choosing appropriate tools and techniques helps the systems analyst to hasten the design process. The finished design should meet the end user requirements and should also help in the proper maintenance of the software system.

To develop a good design, it is very important to select the best design methodology. Each design methodology has both good and bad points. Selection of design methodology depends on the size of the development time, the background and experience of analysts or designers, resources allocated to development, the type of application under development and the time allowance for the project.

The following issues should be kept in mind while selecting a design methodology:

- Is it easy to use and learn?
- Will it help the team to arrive at an understanding of the system under development?
- How much will it cost?
- Can this technique be managed?
- Which data structures are used?
- What data flow and control features does it have?
- Will it serve user analyst communication?

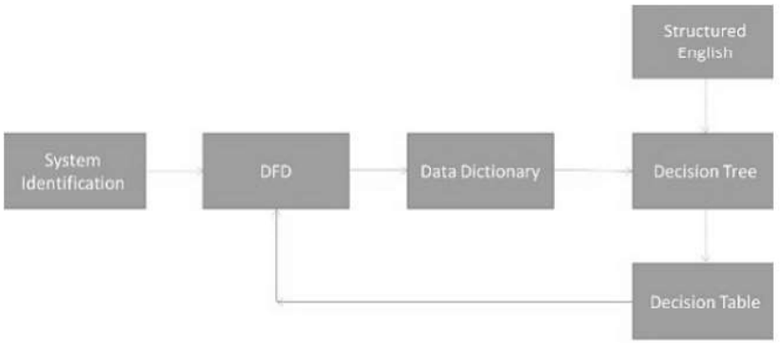
Some important designs that are common include structured design, Input-Process-Output (IPO), HIPO structured walkthrough, pseudocode, System Analysis and Design Technique (SADT), data structure diagrams, entity relationship model, Computer Aided Software Engineering (CASE) method and semantic data model.

NOTES

7.4 STRUCTURED DESIGN

Structured design is a data-flow based methodology that helps in identifying the input and output of the developing system. The main objective of structured design is to minimize the complexity and increase the modularity of a program. Structured design also helps in describing the functional aspects of the system.

In structured designing, the system specifications act as a basis for graphically representing the flow of data and sequence of processes involved in a software development with the help of Data Flow Diagram DFDs. After developing the DFDs for the software system, the next step is to develop the structure chart.



Modularization

Structured design partitions the program into small and independent modules. These are organized in top down manner with the details shown in bottom.

Thus, structured design uses an approach called Modularization or decomposition to minimize the complexity and to manage the problem by subdividing it into smaller segments.

Advantages

- Critical interfaces are tested first.
- It provide abstraction.
- It allows multiple programmers to work simultaneously.
- It allows code reuse.
- It provides control and improves morale.
- It makes identifying structure easier.

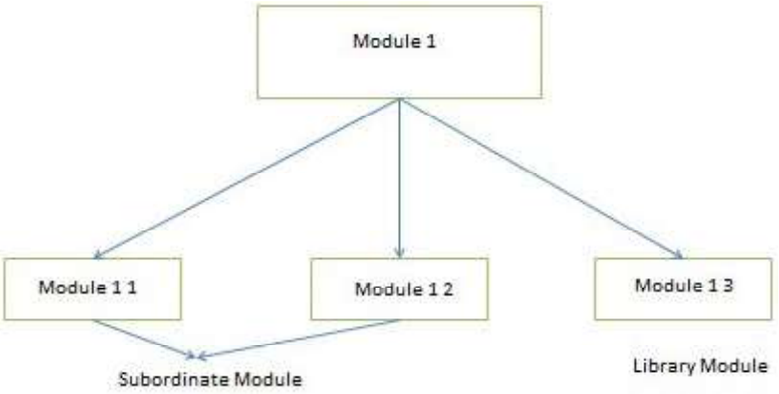
Structured Charts

Structured charts are a recommended tool for designing a modular, top down systems which define the various modules of system development and the relationship between each module. It shows the system module and their relationship between them.

NOTES

It consists of diagram consisting of rectangular boxes that represent the modules, connecting arrows, or lines.

- **Control Module** " It is a higher-level module that directs lower-level modules, called **subordinate modules**.
- **Library Module** " It is a reusable module and can be invoked from more than one point in the chart.



We have two different approaches to design a structured chart "

- **Transform-Centered Structured Charts** " They are used when all the transactions follow same path.
- **Transaction–Centered Structured Charts** " They are used when all the transactions do not follow the same path.

Objectives of Using Structure Flowcharts

- To encourage a top-down design.
- To support the concept of modules and identify the appropriate modules.
- To show the size and complexity of the system.
- To identify the number of readily identifiable functions and modules within each function.
- To depict whether each identifiable function is a manageable entity or should be broken down into smaller components.

7.5 FORM DRIVEN METHODOLOGY

Every rule related to the logical and physical data structure that has been obligated by the DataBase Management System (DBMS) must be followed by the entire database description managing the database. Therefore, steps are related to viewing the physical database design, transforming logical description with a physical model, describing the organization of the database and accessing the physical storage devices.

NOTES

The user interface usually comprises of various documents, reports and different screens of a system. It is the sole component of the system that is viewed by the user. Hence, for the user, the user interface is a vital component of the system. It should be designed in such a way that besides delivering information to the user it should also be visible.

For the purpose of designing an interface, the steps required are stated as follows:

1. Generation of the interface summary table for system use.
    - Analysing the human – computer limit in the DFD.
    - Filing the interface and analysing the design for the purpose of defining the data elements that have to be added in every form/ document, input screen and output report.
2. Designing the external input forms.
3. Designing the human – computer dialogues.
    - Choosing the kind(s) of dialogue.
    - Designing every dialogue.
4. Designing input (data entry) screens.
5. Report designing.
    - Choosing the output technique for every report.
    - Internal report designing.
    - External report designing.

Input Form Design

The excellence of the system output is concluded by the quality of the system input. After understanding such significant relations, it becomes necessary that the input forms and screens as well as output reports should be designed.

The basic objectives of the input form design must be achieved by a good designed input form and visual display screen which can only take place by using the essential design principles. The type of data is required for recognizing the way by which the users react to various different components of forms and screens.

The term 'effectiveness' basically refers that the input forms and screens play a particular role in the Management Information System (MIS) whereas the term 'accuracy' highlights the design to guarantee a suitable conclusion.

The objectives of the forms designing are to extract and capture the needed information through the members of the organization will be frequently added to the computer with the help of form designing. Forms act as the source documents regarding the data entry personnel with the help of this process.

The guidelines of form design are as follows:

- They ease in filling out the forms.
- They provide certainty of the forms meeting the reason of their being.

- They make the forms eye-catching.
- They design the formats for confirming precise completion.

Input Screen Design

NOTES

For the purpose of entering the data in the input forms or reports of the computer and updating the data, the data entry is made use of. The four guidelines for screen design that are important but not complete, include:

- Maintaining the simplicity of the screen.
- Keeping the screen presentation constant.
- Improving the user movement amongst screens.
- Generating an eye-catching screen.

Following this will result in two problems that are essential in context with the designing data entry screens:

1. The momentum of data entry
2. The preciseness of data entry

Maintaining the Simplicity of the Screen

The initial guideline regarding a perfect screen design is simplifying the screen display. Only the required material for the specific action carried out should be shown by the display screen. Practicing such steps will turn amateur users into smart ones as this will reduce the chances of them causing any serious error.

Keeping the Screen Consistent

The next guideline for creating a good screen design is to maintain the consistency of the display which can only be achieved through locating information in the similar area every time when the user is accessing a new screen. Moreover, all information that has logical similarities should be grouped together.

Facilitating Movement Between the Screens

Next guideline is making the browsing among various screens easier. This can be done by making the users develop a feeling of physically moving to a novel screen which can be done through the use of the scroller. The designing of the screens should be done in such a way which provides awareness in the user about the status of every action. The following are the ways by which this can be done:

- Usage of error messages for providing feedback on every error.
- For the purpose of providing feedback on update actions, usage of conformation message.
- In cases where there is any occurrence of backend process, the usage of status messages.

NOTES

- When the actions are far-reaching, permitting the user for reversing an action whenever probable.

Generation of an Attractive Screen

The last guideline is creating an eye-catching screen for the user by trying not to make it completely highlighted. The productivity of the user will increase if he finds the screens appealing and would also require reduced supervision which would result in decreasing the amount of errors. Screens should be such that attract the attention of the user and make them concentrate at a single position. This may be accomplished through using more of open area adjoining the data entry fields, thereby making the screen have a tidy look.

Basic Parts of a Form

Each form in a Windows application has by default Minimize, Maximize and Close buttons in the top left right corner of the title bar and they are known as basic parts of form. In addition, clicking on the icon on the right hand side of the title bar drops down a control box containing the same options and plus options to resize the Window. These options make it easy for the user to perform tasks, such as minimizing or maximizing a form. If, however, you do not want these controls to be available for a particular form they can be disabled from the Properties panel for the form. A form can be resized by the user. By default, every form in an application is shown in the Windows desktop taskbar when it is created. A standard form is interfaced with Minimize, Restore Down and Close buttons which are considered as basic parts of the form.



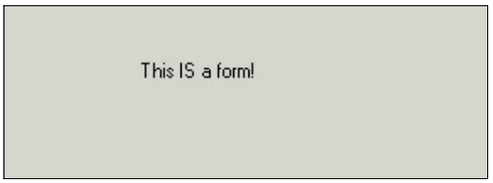
Styles and Types of Forms

A form has four main styles which are considered as template and can be used while the time of form design which set how the form looks. The styles 0 is used for None, 1 is used for Fixed Single, 2 is used for Sizable, 3 is used for Fixed

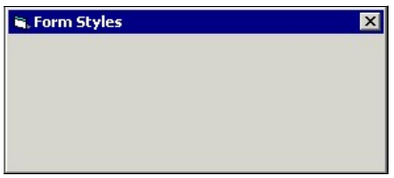
NOTES

Dialog and 4 is used for Fixed using programming language. Styles of form are discussed below:

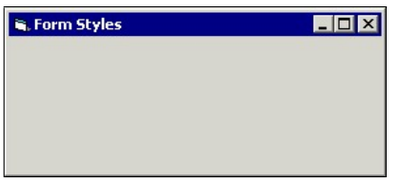
0 – None Style: This form style has no border, no title bar at the top, and no maximize, minimize and close buttons. This style is usually used for splash screens. The term 'splash screen' is used for one of those screens that are shown as an application in loading. Default is no icon in the taskbar.



1 - Fixed Single: This form style has a border, a title bar and a close button with optional minimize and maximize.



2 – Sizable: This form style provides a form with a border, a blue box and the entire standard buttons. The dialog can be resized.



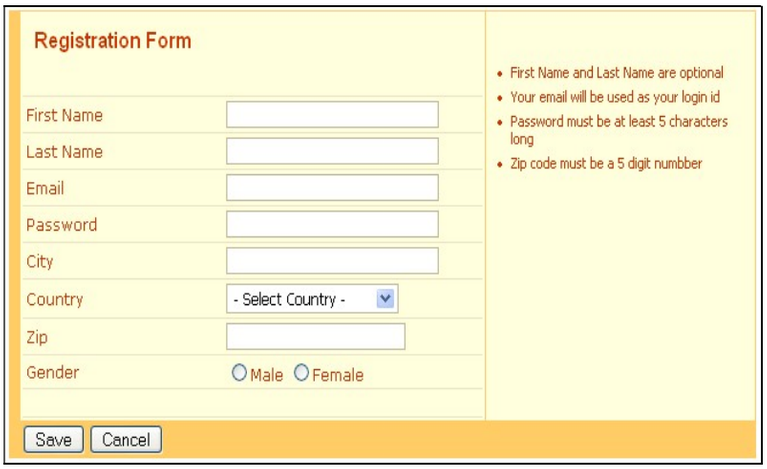
3 - Fixed Dialog: This form style is a form with a border, a blue box and a close button. No minimize and maximize buttons are allowed. Default is no icon in the taskbar.

4 - Fixed: This form style cannot be resized.

You can select any one of the forms style and specified form type and create a suitable form as per requirement. Every Web application that collects information from the user should have an interface which is understood by the users easily for interaction. If the interface is easy for the user to understand, the information supplied by the user will be more accurate. The proposed form (Registration Form) is a frequently used form in which the form labels are highlighted with different background color. Instructions or rules can be added at the top or bottom of every field as you see in the Zip field in the form screen below. The form shown below is designed with the help of two tables. The outer table creates border, padding and inner table is a container for the form. To design the form first it is split

NOTES

into two columns (panels), first column contains form fields and the other column provides space for instruction or rules on filling the form. The design is very easy to implement with minimum Cascading Style Sheets (CSS) styles and Hypertext Markup Language (HTML) tags. CSS is a style sheet language used to describe the look and formatting of a document written in a markup language. The CSS rules are applied style on input and selection controls in the form, changing the font, border and padding.



Flat Forms

A flat form is a single copy form prepared manually or by a machine and printed on a paper. For additional copies of the original carbon papers are inserted between copies. It is the easiest forms to design, print and reproduce. It uses less volume and is inexpensive.

Unit Set/Snapout Forms

Carbon interleaved unit sets are papers with one time carbons interleaved into unit sets for either handwritten or machine use. Carbons may be either blue or black, standard grade medium intensity. Generally, blue carbons are best for handwritten forms while black carbons are best for machine use.

Continuous Strip/Fanfold Forms

These multiple unit forms joined in a continuous strip with perforations between each pair of forms. It is a less expensive method for large volume use.

No Carbon Required or NCR Paper

Carbonless unit sets uses carbonless papers (often referred to as NCR paper). Carbonless papers use two chemical coatings (capsules), one on the face and the other on the back of a sheet of paper. When pressure is applied, i.e., handwriting, typing or machine impact the two capsules interact and create an image.

NOTES

Principles of Forms Design

Forms and reports are produced by input and output design. A business document comprising certain predefined data and providing fields to the end user for putting in additional data, can be referred to as a form.

An illustration of a form can be obtained from the database records. Order forms and credit and employment applications are examples of forms. A report is a set of predefined data collected from many, related or unrelated records or transactions. Weekly sales summaries, credit statements and invoices are examples of reports. Since a report is a business document containing only pre-defined data, it is used exclusively for reading or viewing. A report comprises data from various unrelated records or transactions.

Forms and reports are generated by prototyping. Some of this prototyping activity takes place while analysis is being done to confirm the requirements and demands of the user with regard to the new system.

It is important for those who design forms and reports to have thorough knowledge of the purpose of the form, its users, time of usage, the place of delivery and the information required by the user to finish the job that the report or form represents.

Automated design tools dramatically enhance the developer's ability to prototype forms and reports and present them to end users for evaluation during the design stage.

Good Form Design

The data entered by the user is received in a standardized manner using a printed paper called a form. The guidelines for a systems analyst for designing a good form are as follows:

Keep the Screen Simple

Following are the techniques used to keep the screen simple:

- Form should be designed in a proper sequence, i.e., from left to right and top to bottom.
- Information such as heading, identification and access, instructions, body, signature and verification, totals and comments should be grouped logically.
- Give clear captions of different types of information. Caption may be on the left, above or below the line, box - caption may be inside, above or below the box or space caption may be for a table.

Meet the Intended Purpose

Systems analysts may employ various types of forms serving different purposes in order to design a form that will fulfil the intended purpose. These purposes may

include processing, storing, retrieving and recording of information for different entities in the system.

Assure Accurate Completion

NOTES

Error rates related to data collection can be reduced by designing forms that are completely accurate. In other words, forms should help people do what is right with the form.

Check Your Progress

1. State about the input design.
2. Write the screen design guidelines.
3. Define the term file design.
4. Give the definition of structured flowchart.
5. How the waterfall model is documented?
6. Name the factors on which selection of design methodology depends.
7. Write the main objective of structured design.
8. What are the features of '0 – None Style' form?

7.6 MAJOR DEVELOPMENT ACTIVITIES

On the successful culmination of software design phase, the implementation phase begins, where design specifications are translated into source codes. In this phase, the programs are constructed in such a manner that they are easy to read and comprehend. This helps to ease the work of testing, debugging and maintenance. Note that simplicity, limpidity and elegance are the characteristics of good programs while obscurity and complexity in source codes indicate the presence of poor software design and misconceptions/misunderstandings in the minds of programmers. Also, there are many programming languages, each with distinct features and uses. Therefore it is essential to select a language that is able to meet the users' requirements in the best possible manner.

Writing code in an easy manner and in less number of lines shows the creativity of a software developer. For this, a thorough knowledge of programming languages and their tools is a prerequisite. Source code clarity is enhanced by following a good coding style, coding guidelines, structured coding techniques and various code verification techniques. For this, it is important to provide the programmers with a set of well defined software requirements, architectural design specifications and detailed design descriptions.

NOTES

Features of a Software Code

The code written for a software should be according to the requirements of the users. A program is said to be good if the software code is flawless or contains minimum errors. For the effective performance of a software, some particular features that are required in almost all languages that are used to write the software code (refer Figure 7.2). These features are listed below:

- ***Simplicity:*** A software code should be written in a simple and concise manner. Simplicity should be maintained in the organization, implementation, and design of the software code.
- ***Modularity:*** Breaking the software into several modules not only makes it easy to understand but also easy to debug. With the modularity feature, the same code segment can be reused in one or more software programs.
- ***Design:*** A software code is properly designed if it is presented in a proper manner. The design of the software should be decided before beginning to write a software code. Writing the software code in a specific, consistent style helps other software developers in reviewing it.
- ***Efficiency:*** A program is said to be efficient if the software program makes optimal use of the available resources.
- ***Clarity***: Software codes should be clear so that developers are able to understand the program without any complexity. Clarity can be achieved by using features such as simplicity, readability and modularity. Note that clarity comprises clarity of code, clarity of design, and clarity of purpose so that one knows what occurs at each level in the software program.
- ***Accessibility:*** Software codes should be written in a way that the software components (for example, files and functions) are easily available and accessible. For the files and functions to be accessible, they should have meaningful names as well as supporting captions and text for each image. Similarly, there should be hyperlinks and navigation aids to assist the users in searching information from different sections of the software code.
- ***Stability:*** Software codes are said to be stable if they are able to work correctly on different platforms without affecting their layout and consistency. To check for stability, software codes should be tested for errors and inconsistency.

NOTES

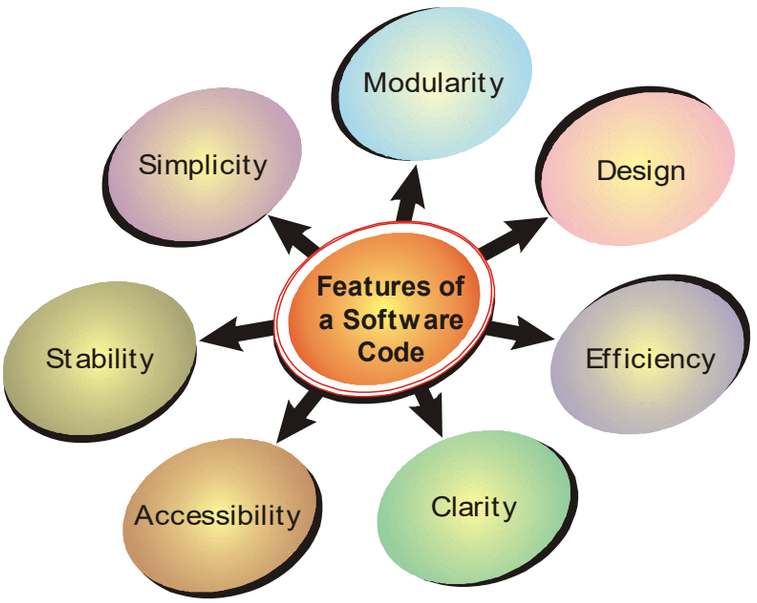


Fig. 7.2 *Features of a Software Code*

Coding Guidelines

Writing an efficient software code requires a thorough knowledge of programming. This knowledge can be implemented by following a coding style which comprises several guidelines that help in writing the software code efficiently and with minimum errors. These guidelines, known as coding guidelines, are used to implement individual programming language constructs, comments, formatting, and so on. The objective of using these guidelines is to prevent programming errors, control complexities, and to promote the understandability of the source code.

A set of comprehensive coding guidelines encompasses all aspects of code development. To ensure that all developers work in a harmonized manner (the source code should reflect a harmonized style, as though a single developer had written the entire code in one session), the developers should be aware of the coding guidelines before the inception of a software project. Moreover, coding guidelines should state how to deal with the existing code when the software incorporates it or when maintenance is performed.

Since there are numerous programming languages for writing software codes, each having different features and capabilities, coding style guidelines differ from one language to another. However, there are some basic guidelines which are followed in all programming languages. These include *naming conventions*, *commenting conventions*, and *formatting conventions*.

NOTES

Table 7.4 Naming Conventions for Variables and Constants

Variable Naming Conventions	Constant Naming Conventions
The variable names should be in mixed case letters starting with a lower case letter. For example, use 'total Amount' instead of 'Total Amount'.	All the names of constants should be in upper case. In case the name of a constant is too long, it should be separated by an underscore. For example, sales tax rate should be written as 'SALES_TAX_RATE'.
The temporary storage variables that are restricted to a segment of code should be short. For example, the variable 'temp' can be used for a temporary variable. It is important to note that a single temporary variable should not be re-used in the same program. For example, variables 'i', 'j', or 'k' are declared while using loops.	The use of literal should be avoided. Literal numbers, such as '15' used in the software code confuse the reader. These numbers are counted as integers and result in wrong output of the program. However, the numbers '0' and '1' can be used as constants.
The use of numbers with variables should be avoided. For example, 'first Number' should be used instead of 'number1'.	

Naming Conventions

There are certain rules for naming variables, functions and methods in the software code. These naming conventions help software developers in understanding the use of a particular variable or function. The guidelines used to assign a name to any variable, function or method are as follows:

- All the variables, functions or methods should be assigned names that make the code more understandable to the reader. By using meaningful names, the code can be self explanatory, thus minimizing the effort of writing comments for variables. For example, if two variables are required to refer to 'sales tax' and 'income tax', they should be assigned names, such as 'salesTax' and 'incomeTax'.
- For names, a full description in a commonly spoken language (for example, English) should be used. In addition, the use of abbreviations should be avoided. For example, variable names like 'contactNumber' and 'address' should be used instead of 'cno' and 'add'.
- Short and clear names should be assigned in place of long names. For example, 'multiplyTheTwoNumbers' can be shortened to 'multiplyNumber' as it is clear and short enough to be expressed in reasonable length.

In every programming language, there is a different naming convention for variables and constants in the software code. The commonly used conventions for naming variables and constants are listed in Table 7.4.

NOTES

Function Naming Conventions

As with variables and constants, there are some guidelines that should be followed while naming functions in the software code. These conventions are as follows:

- The names of functions should be meaningful and describe the purpose of the function with clarity and brevity. Like variables, the names should be self explanatory so that there is no additional need to explain the task of that function.
- The function name should begin with a verb. For example, the verb 'display' can be used to display the output of a code. In case the verb itself is not descriptive, an additional noun or adjective can be used with the verb. For example, the function name 'addMarks' should be used to clarify the function and its purpose.
- In case the function returns a Boolean value, the helping verbs 'is' and 'has' should be used as prefixes for the function name. For example, the function name 'isDeposited' or 'hasDeposited' should be used for functions that return true or false values.

Commenting Conventions

Comments are helpful in the proper understanding of the code segment used in a program. Commenting conventions should be used efficiently to make the code easy to grasp. Generally, two types of commenting conventions are used: file header comments and trailing comments. **File header comments** are useful in providing information related to a file as a whole and comprise identification information, such as date of creation, name of the creator and a brief description of the code included in the software code.

Trailing comments are used to provide explanation of a single line of code. These comments are used to clarify the complex code. These also specify the function of the abbreviated variable names that are not clear. In some languages, trailing comments are used with the help of a double slash (//). The commenting conventions that are commonly followed in the software code are as follows:

- Comments should not be used to include information that is clearly understandable from the software.
- Comments should be used with important segments of code and with code segments that are difficult to understand.
- Comments should be separated from the code to enhance readability of the software code.

Formatting Conventions

Formatting (way of arranging a program in order to enhance readability) consists of indentation, alignment and use of white spaces in the program. Consistency plays an important role while formatting a program in an organized way. A program

NOTES

with consistent formatting makes the code easier to read and understand. The commonly used formatting conventions are as follows:

- ***Indentation:*** This refers to one or more spaces left at the beginning and the end of statements in the program. Indentation is useful in making the code easily readable. However, the spaces used for indentation should be followed in the entire program. The guidelines that are commonly followed while indenting a program are listed below:
  - o Indentation should be used to highlight a nested block. Some nested blocks can be made with the help of 'if-else' and 'do-while' loops.
  - o Indentation is required if the statement is small enough to fit in a single line.
  - o Indentation should be consistent at the beginning and at the end of the braces in the program.
- ***White Spaces:*** These improve readability by minimizing the compactness of the code. Some of the guidelines for proper usage of spaces within the code are:
  - o There should be a space after placing a comma between two function arguments.
  - o There should be no space between a function name and parenthesis.
  - o There should be spaces to align the operators vertically to emphasize program structure and semantics.

Implementing Coding Guidelines

If coding guidelines are used in a proper manner, errors can be detected at the time of writing the software code. Such detection in early stages helps in increasing the performance of the software as well as reducing the additional and unplanned costs of correcting and removing errors. Moreover, if a well defined coding guideline is applied, the program yields a software system that is easy to comprehend and maintain. Some of the coding guidelines that are followed in a programming language are as follows:

- All the codes should be properly commented before being submitted to the review team.
- All curly braces should start from a new line.
- All class names should start with the abbreviation of each group. For example, AA and CM can be used instead of academic administration and course management, respectively.
- Errors should be mentioned in the following format: [error code]: [explanation]. For example, 0102: null pointer exception, where 0102 indicates the error code and null pointer exception is the name of the error.

NOTES

- Every 'if' statement should be followed by a curly brace even if there exists only a single statement.
- Every file should contain information about the author of the file, modification date and version information.

Similarly, some of the commonly used coding guidelines in a database (organized collection of information that is systematically organized for easy access and analysis) are as follows:

- Table names should start with TBL. For example, TBL_TUS.
- If table names contain one word, field names should start with the first three characters of the name of the table. For example, STU_FIRSTNAME.
- Every table should have a primary key.
- Long data type (or database equivalent) should be used for the primary key.

Advantages of Coding Guidelines

A coding guideline supplements the language standard by defining acceptable and unacceptable usage of the programming language used. Acceptable usage avoids troublesome situations, while unacceptable usage is conducive to errors or leads to misunderstanding of the written code. A properly implemented coding guideline helps the developer to limit program complexity, establish the basis for code review, and guard against compiler and common programming errors. The other advantages associated with coding guidelines listed forthwith and depicted in Figure 7.3.

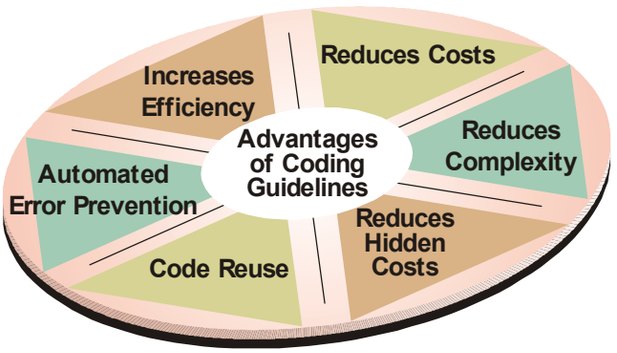


Fig. 7.3 *Advantages of Coding Guidelines*

- ***Increases Efficiency:*** Coding guidelines can be used effectively to save time spent on gathering unnecessary details. These guidelines increase the efficiency of the software team while the software development phase is carried out. An efficient software code is fast and economical. Software coding guidelines are used to increase efficiency by making the team productive, thus ensuring that the software is delivered to the user on time.

- ***Reduces Costs:*** Coding guidelines are beneficial in reducing the cost incurred on the software project. This is possible since coding guidelines help in detecting errors in the early stages of the software development. Note that if errors are discovered after the software is delivered to the user, the process of rectifying them becomes expensive as additional costs are incurred on late detection, rework and retesting of the entire software code.
- ***Reduces Complexity:*** The written software code can be either simple or complex. Generally, it is observed that a complex segment of software code is more susceptible to errors than a segment containing a simple software code. This is because a complex software code reduces readability as well as understandability. In addition, the complex software code may be inefficient in functioning and use of resources. However, if the code is written using the laid coding guidelines, the problem of complexity can be greatly avoided as the probability of error occurrence reduces substantially.
- ***Reduces Hidden Costs:*** Coding guidelines, if adhered to in a proper manner, help to achieve a high-quality software code. The software quality determines the efficiency of a software. Software quality is the degree to which user requirements are accomplished in the software along with conformity to standards. Note that if quality is not considered while developing the software, the cost for activities, such as fixing errors, redesigning the software and providing technical support increases considerably.
- ***Code Reuse:*** Using coding guidelines, software developers are able to write a code that is more robust and create individual modules of the software code. The reason for making separate code segments is to enable reusability of the modules used in the software. A reusable module can be used a number of times in different modules in one or more software.
- ***Automated Error Prevention:*** The coding guidelines enable Automated Error Prevention (AEP). This assures that each time errors occur in a software, the software development activity is improved to prevent similar errors in future. AEP begins with detecting errors in the software, isolating its cause, and then searching the cause of error generation. Coding guidelines are useful in preventing errors as they allow implementation of requirements that prevent the most common and damaging errors in the software code.

In addition to the afore-mentioned advantages, coding guidelines define appropriate metric thresholds. These thresholds help in reducing complexity, thus minimizing the occurrence of errors. Software developers face increasing demands to demonstrate that development practices meet the accepted coding guidelines. This is essential for companies developing safety critical software as well as those seeking Capability Maturity Model (CMM) and International Organization for Standardization (ISO) certification.

NOTES

Coding Methodology

This methodology refers to a set of well-documented procedures and guidelines used in the analysis, design and implementation of programs. Coding methodology includes a diagrammatic notation for documenting the results of the procedure. It also includes an objective set (ideally quantified) of criteria for determining whether the results of the procedure are of the desired quality or not. To use coding methodology, a number of steps are followed, such as:

1. The software development team begins its work by reviewing and understanding the design and requirements specification documents. These documents are essential for understanding user requirements and creating a framework for the software code.
2. In case the software development team is unable to understand user requirements correctly and further clarification is required, the queries are sent back to the user. The software development team also returns the requirements that are understood by them.
3. After the requirements are clearly understood by the software development team, the design and specifications are implemented in source code, supporting files and the header files. Note that while writing the software code, the coding style guidelines should be followed. In some cases, there may be a proposal of change in hardware or software specifications. However, the requests for change are implemented only after the approval of the user.
4. When the software code is completely written, it is compiled along with other required files.
5. Code inspection and reviews are conducted after the compilation. These methods are used to correct and verify errors in the software code.
6. Software testing is carried out to detect errors and correct in each module of the software code, its integration with hardware, the type of errors, and so on.
7. After the software code is tested, the software is delivered to the user along with the relevant code files, header files and documentation files.

Figure 7.4 shows the flowchart of the software coding process. In case further change and clarifications are required in the design or Software Requirements Specification (SRS) documents, the software development team raises a query, which is sent to the user with the document containing what the software development team understood from the documents sent by the user. Changes are made only when the user has a positive response to the queries raised by the software development team.

NOTES

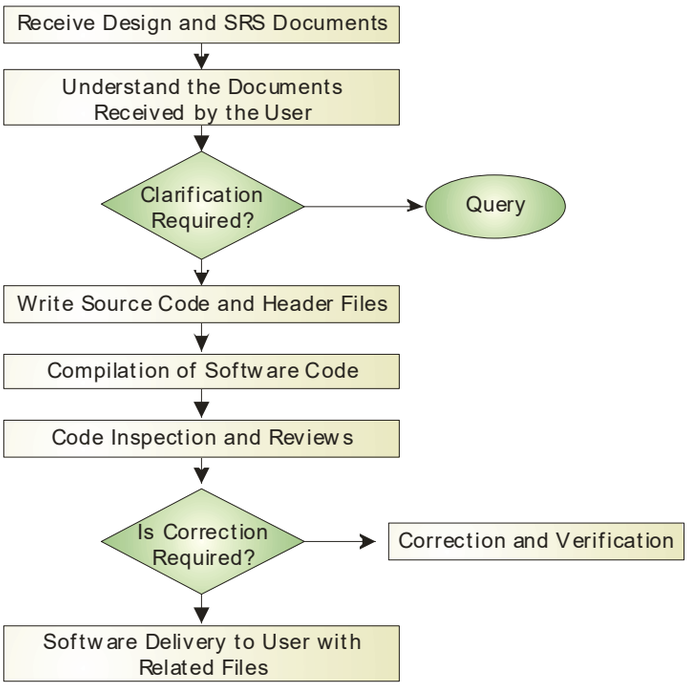


Fig. 7.4 *Flowchart of Software Coding*

Programming Practice

Programming refers to the method of creating a sequence of instructions to enable the computer to perform a task. It is done by developing logic and then writing instructions in a programming language. A program can be written using various programming practices available. A programming practice refers to the different ways of writing a software program and is used along with coding style guidelines. The commonly used programming practices include top-down programming, bottom-up programming, structured programming, and information hiding (refer Figure 7.5).

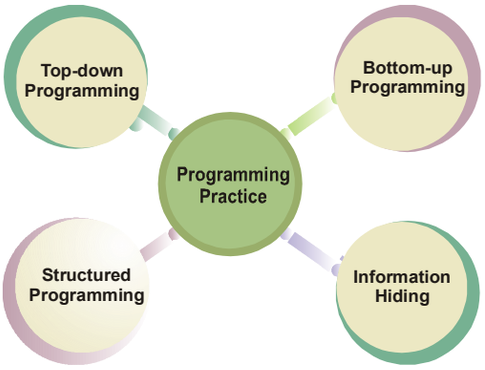


Fig. 7.5 *Commonly-used Programming Practices*

NOTES

Top-Down Programming

Top-down programming focuses on the use of modules. It is therefore also known as modular programming. The program is broken up into small modules so that it is easy to trace a particular segment of code in the software program. The modules at the top level are those that perform general tasks and proceed to other modules to perform a particular task. Each module is based on the functionality of its functions and procedures. In this approach, programming begins from the top level of hierarchy and progresses towards the lower levels. The implementation of modules starts with the main module. After the implementation of the main module, the subordinate modules are implemented and the process follows in this way. In top-down programming, there is a risk of implementing data structures as the modules are dependent on each other and they have to share one or more functions and procedures. In this way, the functions and procedures are globally visible. In addition to modules, the top-down programming uses sequences and the nested levels of commands.

Bottom-Up Programming

Bottom-up programming refers to the style of programming where an application is constructed with the description of modules. The description begins at the bottom of the hierarchy of modules and progresses through higher levels until it reaches the top. Bottom-up programming is just the opposite of top-down programming. Here, the program modules are more general and reusable than top-down programming.

It is easier to construct functions in bottom-up manner. This is because bottom-up programming requires a way of passing complicated arguments between functions. It takes the form of constructing abstract data types in languages, such as C++ or Java which can be used to implement an entire class of applications and not only the one that is to be written. It therefore becomes easier to add new features in a bottom-up approach than in a top-down programming approach.

Structured Programming

Structured programming is concerned with the structures used in a computer program. Generally, structures of computer program comprise decisions, sequences and loops. The decision structures are used for conditional execution of statements (for example, 'if' statement). The sequence structures are used for the sequentially executed statements. The loop structures are used for performing some repetitive tasks in the program.

Structured programming forces a logical structure in the program to be written in an efficient and understandable manner. The purpose of structured programming is to make the software code easy to modify when required. Some languages, such as Ada, Pascal, and dBase are designed with features that implement

NOTES

the logical program structure in the software code. Primarily, the structured programming focuses on reducing and removing the following statements from the program.

- 'Go To' statements.
- 'Break' or 'Continue' outside the loops.
- Multiple exit points to a function, procedure or subroutine. For example, multiple 'Return' statements should not be used.
- Multiple entry points to a function, procedure or a subroutine.

Structured programming frequently employs the top-down design because program structure is divided into separate subsections. A defined function or set of similar functions is kept separately. Due to this separation of functions, they are easily loaded in the memory. In addition, these functions can be reused in one or more programs. Each module is tested individually. After testing, they are integrated with other modules to achieve an overall program structure. Note that a key characteristic of a structured statement is the presence of single entry and single exit. This characteristic implies that during execution, a structured statement starts from one defined point and terminates at another defined point.

Information Hiding

Information hiding refers to hiding details of a function or structure in software code. It focuses on making the functions or structures inaccessible to other components of software. A software developer applies information hiding in software design and coding to hide unnecessary details from the rest of the program. The objective of information hiding is to minimize complexities among different modules of the software. Note that complexities arise when one program or module in software is dependent on several other programs and modules.

Information hiding is implemented with the help of interfaces. An interface is a medium of interaction for software components using the properties of software modules containing data. They are used in a programming language. However, their implementation depends on the syntax and process. Examples of interface include constants, data types, types of procedures, and so on. Interfaces protect other parts of programs when a software design is changed.

Generally, the interfaces act as a foundation to modular programming (top-down programming) and object oriented programming. In Object Oriented Programming (OOP), interface of an object comprises a set of methods, which are used to interact with the objects of the software programs. Using information hiding, a single program is divided into several modules. These modules are independent of each other and can be used interchangeably in other software programs.

To understand the concept of information hiding, let us consider an example of a program written for 'car'. The program can be organized in several ways.

NOTES

One is to arrange modules without using information hiding. In this case, the modules can be created as 'front part', 'middle part', and 'rear part'. On the other hand, creating modules using information hiding includes specifying names of modules, such as 'engine' and 'steering'.

On comparison, it is found that modules created without using information hiding affect other modules. This is because when a module is modified, it affects the data, which does not require modification. However, if modules are created using information hiding, then modules are concerned only with specific segments of the program and not the whole program or other parts of the program. In our example, this statement means that the module 'engine' does not have any affect on the module 'steering'.

Code Verification Techniques

Code verification is the process used for checking the software code for errors introduced in the coding phase. The objective of code verification process is to check the software code in all aspects. This process includes checking the consistency of user requirements with the design phase. Note that code verification process does not concentrate on proving the correctness of programs. Instead, it verifies whether the software code has been translated according to the requirements of the user or not.

The code verification techniques are classified into two categories, namely, *dynamic* and *static* techniques. The dynamic technique is performed by executing some test data. The outputs of the program are tested to find errors in the software code. This technique follows the conventional approach for testing the software code. In the static technique, the program is executed conceptually and without any data. In other words, the static technique does not use any traditional approach as used in the dynamic technique. Some of the static techniques commonly used are code reading, static analysis, symbolic execution, code inspection and reviews (refer Figure 7.6).

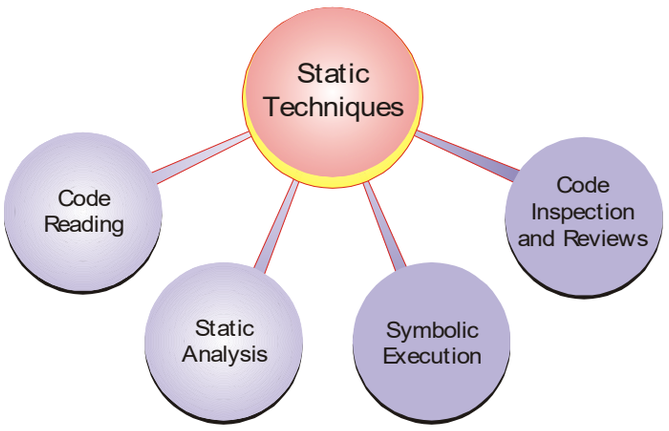


Fig. 7.5 *Static Techniques*

NOTES

Code Reading

Code reading is a technique that concentrates on how to read and understand a computer program. It is essential for a software developer to know code reading. The process of reading a software program in order to understand it is known as code reading or program reading. In this process, attempts are made to understand the documents, software specifications or software designs. The purpose of reading programs is to determine the correctness and consistency of the code. However, this skill is rarely developed in a systematic way while training software professionals. In addition, code reading is performed to enhance the software code without entirely changing the program or with minimal disruption in the current functionality of the program. Code reading also aims to inspect the code and remove (fix) errors from it.

Code reading is a passive process and needs concentration. An effective code reading activity primarily focuses on reviewing 'what is important'. The general conventions that can be followed while reading the software code are as follows:

- ***Figure Out What Is Important:*** While reading the code, emphasis should be on finding graphical techniques (bold, italics) or positions (beginning or end of the section). Important comments may be highlighted in the introduction or at the end of the software code. The level of details should be according to the requirement of software code.
- ***Read What Is Important:*** Code reading should be done with the intent to check syntax and structure, such as brackets, nested loops and functions rather than the non-essentials, such as name of the software developer who has written the software code.

Static Analysis

Static analysis comprises a set of methods used for analyzing the software source code or object code to understand how the software functions and to set up criteria to check its correctness. Static analysis studies the source code without executing it and reveals a variety of information like the structure of model used, data and control flows, syntactical accuracy and much more. Due to this, there are several kinds of static analysis methods.

- ***Control Analysis:*** This focuses on examining the controls used in software calling structure, control flow analysis and state transition analysis. The calling structure deals with the model by identifying the calling and the called structure. The calling structure can be a procedure, subroutine, function or method. The control flow analysis checks the sequence of control transfers. In addition, it identifies incorrect and inefficient constructs in the model. A graph of the model is constructed in which the conditional branches and model junctions are represented by nodes.
- ***Data Analysis:*** This ensures that proper operations are applied to data objects (for example, data structures and linked lists). In addition, this method

NOTES

also ensures that the defined data is properly used. Data analysis comprises two methods, namely, data dependency and Data flow analysis. **Data dependency** (which determines the dependency of one variable on another) is essential for assessing the accuracy of synchronization across multiple processors. **Data flow analysis** checks the definition and references of variables.

- ***Fault/Failure Analysis:*** This analyses the fault (incorrect model component) and failure (incorrect behaviour of a model component) in the model. This method uses input-output transformation descriptions to identify the conditions that are the cause for the failure. To determine the failures in certain conditions, the model design specification is checked.
- ***Interface Analysis:*** This verifies and validates the interactive and distributive simulations to check the software code. There are two basic techniques for the interface analysis, namely, model interface analysis and user interface analysis. **Model interface analysis** examines the submodel interfaces and determines the accuracy of the interface structure. **User interface analysis** examines the user interface model and checks for precautionary steps taken to prevent errors during the user's interaction with the model. This method also concentrates on how accurately the interface is integrated into the overall model and simulation.

Symbolic Execution

Symbolic execution concentrates on assessing the accuracy of the model by using symbolic values instead of actual data values for input. Symbolic execution, also known as symbolic evaluation, is performed by providing symbolic inputs, which produce expressions for the output.

Symbolic execution uses a standard mathematical technique for representing the arbitrary program inputs (variables) in the form of symbols. To perform the calculation, a machine is employed to perform algebraic manipulation on the symbolic expressions. These expressions include symbolic data meant for execution. The symbolic execution is represented as a symbolic state symbol, consisting of variable symbolic values, path, and the path conditions. The symbolic state for each step in the arbitrary input is updated. The steps that are commonly followed for updating the symbolic state considering all possible paths are:

1. The read or the input symbol is created.
2. The assignment creates a symbolic value expression.
3. The conditions in symbolic state add constraints to the path condition.

The output of symbolic execution is represented in the form of a symbolic execution tree. The branches of the tree represent the paths of the model. There is a decision point to represent the nodes of the tree. This node is labelled with the symbolic values of the data at that junction. The leaves of the tree are complete paths through the model and they represent the output of symbolic execution.

Symbolic execution helps in showing the correctness of the paths for all computations. Note that in this method the symbolic execution tree increases in size and creates complexity with growth in the model.

Code Inspection and Reviews

NOTES

This technique is a formal and systematic examination of the source code to detect errors. During this process, the software is presented to the project managers and the users for a comment of approval. Before providing any comment, the inspection team checks the source code for errors. Generally, this team consists of a moderator, reader, recorder and author.

- ***Moderator:*** conducts inspection meetings, checks errors and ensures that the inspection process is followed.
- ***Reader:*** paraphrases the operation of the software code.
- ***Recorder:*** keeps record of each error in the software code. This frees the task of other team members to think deeply about the software code.
- ***Author:*** observes the code inspection process silently and helps only when explicitly required. The role of the author is to understand the errors found in the software code.

As mentioned above, the reader paraphrases the meaning of small sections of code during the code inspection process. In other words, the reader translates the sections of code from a computer language to a commonly spoken language (such as English). The inspection process is carried out to check whether the implementation of the software code is done according to the user requirement or not. Generally, to conduct code inspections, a number of steps are followed such as:

1. ***Planning:*** After the code is compiled and there are no more errors and warning messages in the software code, the author submits the findings to the moderator who is responsible for forming the inspection team. After the inspection team is formed, the moderator distributes the listings as well as other related documents like design documentation to each team member. The moderator plans the inspection meetings and coordinates with the team members.
2. ***Overview:*** This is an optional step and is required only when the inspection team members are not aware of the functioning of the project. To familiarize the team members, the author provides details to make them understand the code.
3. ***Preparation:*** Each inspection team member individually examines the code and its related materials. They use a checklist to ensure that each problem area is checked. Each inspection team member keeps a copy of this checklist, in which all the problematic areas are mentioned.

NOTES

4. ***Inspection Meeting:*** This is carried out with all team members to review the software code. The moderator discusses the code under review with the inspection team members.

There are two checklists for recording the result of the code inspection, namely, code inspection checklist and inspection error list. **The code inspection checklist** contains a summary of all errors of different types found in the software code. This checklist is used to understand the effectiveness of inspection process. **The inspection error list** provides the details of each error that requires rework. Note that this list contains details only of those errors that require the whole coding process to be repeated.

All errors in the checklist are classified as major or minor. An error is said to be major if it results in problems and later comes to the knowledge of the user. On the other hand, minor errors are spelling errors and non-compliance with standards. The classification of errors is useful when the software is to be delivered to the user and there is little time to review all the errors present in the software code.

At the conclusion of the inspection meeting it is decided whether the code should be accepted in the current form or sent back for rework. In case the software code needs reworking, the author makes all the suggested corrections and then compiles the code. When the code becomes error free, it is sent back to the moderator. The moderator checks the code that has been reworked. If the moderator is completely satisfied with the software code, inspection becomes formally complete and the process of testing the software code begins.

Coding Tools

While writing a software code, several coding tools are used along with the programming language to simplify the tasks of writing a software code. Note that coding tools vary from one programming language to another as they are developed according to a particular programming language. However, sometimes a single coding tool can be used in more than one programming language. Generally, coding tools comprise text editors, supporting tools for a specific programming language, and the framework required to run the software code. Some of the commonly used coding tools are listed in Table 7.5.

NOTES

Table 7.5 Description of Coding Tools

Coding Tools	Languge	Description
JBuilder	Java, XML	Used to speed up Web applications and database applications. It provides interfaces to application servers, such as WebLogic, WebSphere, and EAServer as an editor and visual flow designer. In addition, it provides the Java 2 Standard Edition (J2SE) and Java 2 Enterprise Edition (J2EE) support in Java, enhanced performance tools, and code audits.
Dreamweaver	HTML, ASP.NET	Used for the server side scripting of languages, such as ASP.NET and HTML. It is also used for performing functions, such as creation of Web sites, database connections, querying a database, formatting the output of a software code, and displaying multiple records
Eclipse	Java	Used for Integrated Development Environment (IDE) for Java. It is a cross platform tool, integrated with other coding tools of Java and is relatively easy to set up.
Ant	Java	Used for writing code in Java. In addition, it is a cross platform tool and is flexible to use as any action being performed repeatedly can be standardized using Ant.
Junit	Java	Used for testing a framework by creating tests for the software code that are to be repeated as often as required.

In addition to the programming language and coding tools, there are some software programs that are essential to run the software code. For example, a debugger is used to detect the source of program errors by performing a step-by-step execution of the software code. A debugger breaks program executions at various levels in the application program. It supports features, such as breakpoints, displaying or changing memory and so on. Similarly, **compilers** are used to translate programs written in a high level language into their machine language equivalents.

Code Documentation

Code documentation is a manual-cum-guide that helps in understanding and correctly utilizing the software code. The coding standards and naming conventions written in a commonly spoken language in code documentation provide enhanced clarity for the designer. Moreover, they act as a guide for the software maintenance team (this team focuses on maintaining software by improving and enhancing the software after it has been delivered to the end user) while the software maintenance process is carried out. In this way, code documentation facilitates code reusability. While writing a software code, the developer needs proper documentation for reference purposes. Programming is an ongoing process and requires modifications from time to time. When a number of software developers are writing the code for

NOTES

the same software, complexity increases. With the help of documentation, software developers can reduce the complexity by referencing the code documentation. Some of the documenting techniques are comments, visual appearances of codes, and programming tools. **Comments** are used to make the reader understand the logic of a particular code segment. The visual appearance of a code is the way in which the program should be formatted to increase readability. The **programming tools** in code documentation are algorithms, flowcharts and pseudocodes.

Code documentation contains a source code, which is useful for software developers in writing the software code. The code documents can be created with the help of various coding tools that are used to auto-generate the code documents. In other words, these documents extract comments from the source code and create a reference manual in the form of text or HyperText Markup Language (HTML) file. The auto-generated code helps software developers to extract the source code from the comments. This documentation also contains application programming interfaces, data structures and algorithms. There are two kinds of code documentation, namely, internal documentation and external documentation.

Internal Documentation

Documentation which focuses on the information that is used to determine the software code is known as **internal documentation**. It describes the data structures, algorithms, and control flow in the programs. There are various guidelines for making the documentation easily understandable to the reader. Some of the general conventions to be used at the time of internal documentation are header comment blocks, program comments and formatting. Header comment blocks are useful in identifying the purpose of the code along with details, such as how the code functions, and how each segment of code is used in the program.

Since software code is updated and revised several times, it is important to keep a record of the code information so that internal documentation reflects the changes made by the software code. Internal documentation should explain how each code section relates to user requirements in the software. Generally, internal documentation comprises the following information:

- Name, type, and purpose of each variable and data structure used in the code.
- Brief description of algorithms, logic and the error-handling techniques.
- Information about the required input and expected output of the program.
- Assistance on how to test the software.
- Information on the upgradations and enhancements in the program.

NOTES

External Documentation

Documentation which focuses on general description of the software code and is not concerned with its detail is known as **external documentation**. It includes information such as function of code, name of the software developer who has written the code, algorithms used in the software code, dependency of code on programs and libraries, and format of the output produced by the software code. Generally, external documentation includes structure charts for providing an outline of the program and describes the design of the programs.

External documentation is useful for software developers as it consists of information such as description of the problem along with the program written to solve it. In addition, it describes the approach used to solve the problem, operational requirements of the program, and user interface components. For the purpose of readability and proper understanding, the detailed description is accompanied by figures and illustrations that show how one component is related to another. External documentation explains why a particular solution is chosen and implemented in the software. It also includes formulas, conditions and references from where the algorithms or documentation are derived. External documentation makes the user aware of the errors that occur while running the software code. For example, if an array of five numbers is used, it should be mentioned in the external documentation that the limit of the array is five.

Code Documentation Tools

While writing software code documentation, it is important to consider code documentation tools required for writing the software code. The software documentation tools conform to standards by generating the required elements automatically with configurable format and style. These tools combine the selected comment sections with the software code to generate a usable documentation with the essential level of details in it. Some of the code documentation tools are listed in Table 7.6.

Code documentation tools should be simple to use because easy-to-use documentation tools provide rapid feedback. However, the basic features of software code documentation tools are discussed here:

- ***Target Media:*** views software code easily in a web browser. The target media is useful in displaying the structure and layout of the page with sufficient precision. It is required for code documentation, so that the software code can be easily used in the web browser. An example of target media is HTML.
- ***Documentation Structure:*** includes the index to pages and chapters. The chapters in the documentation should include information such as title, introduction, table of contents and sections.
- ***Comment Extraction Capabilities:*** extracts the software code comments regardless of the style used in the software code.

NOTES

- ***Languages Supported:*** makes the code documentation consistent while writing the software code. The code documentation tools support multiple programming languages and are preferred for concentrating on a particular language.
- ***Formatting and Style Elements:*** make the format of the software code proper. Special elements, such as tags or mark-ups are required to determine the layout, structure and style of the code documentation.
- ***Code Readability:*** makes the software code consistent and easily readable. This is because code documentation is itself not sufficient and requires the comments in software code to make the code documentation readable and understandable.

Table 7.6 *Code Documentation Tools*

Documentation Tools	Language Supported	Description
Cocoon	C++	Used to process C++ library files and generates Web pages that are useful to document the libraries, classes and global functions.
Ccdoc	C++	Used for implementing the document standards in Java and C++.
Cxref	C	Used to generate documents in HTML, Rich Text Format (RTF), and so on. It also includes cross references from source code of C programs.
DOC++	C, C++, Java	Used for providing output for the documentations produced in C, C++, and Java.
JavaDoc	Java	Used as a standard for documentation in Java.
Perceps	C++	Used to break C and C++ header files into separate header files. It generates documentation in various formats according to class definitions, declarations, and comments included in those files.
RoboDoc	Assembler, C, Perl, LISP, Fortran, Shell scripts, COBOL	Used to convert formatted documentation into cross-referenced set of HTML pages, which describe the software code.
DocJet	Java, C, C++, Visual Basic	Used to generate documentation from comments in the source code.
ObjectManual	C++	Used to generate documentation in the form of HTML, XML and RTF pages.
Together	Java, C++	Used to generate documentation from Unified Modeling Language (UML) and its source code.
Doc-o-matic	C++, C#, ASP.NET, VB.NET, Java, JavaScript, JSP	Used to create documentations such as source code documentation, online help, and user manuals. It is integrated with easy-to-use interface for managing the documentation projects.

In addition to the afore mentioned features, the amount of detail provided is also an important feature. Too much detail makes the code documentation inefficient and proves unnecessary. The level of details should be according to the software developer and not according to coding tools used in the software code.

Check Your Progress

9. Write the two features of software code.
10. Give the definition of coding guidelines.
11. Write the types of commenting conventions.
12. What does coding methodology include?
13. Explain the purpose of structured programming.
14. What is code verification?
15. What does static analysis comprise of?
16. Give the definition of internal documentation.
17. Define the term external documentation.

7.7 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. Input design is an important phase in system design phase. Businesses use the new technology to speed up the input process, reduce costs and capture data in new forms such as digital signature.
2. The following are the guidelines to develop a screen design:
   - The display screen should be simple and consistent and should show only that which is necessary for the end user.
   - Screens can be kept consistent if information is located in the same area each time a new screen is accessed. Similar information should be consistently grouped together.
   - Facilitate the navigation of end users among various screens. Web based forms facilitate movement with the use of hyperlinks.
3. A file is a collection of data that can be read from or written to. A file can be a program. You can create a file to write the data and store in suitable device. Commands, printers, terminals and application programs are all stored in files which allow users to access diverse elements of the system in a uniform way and give the operating system great flexibility.

NOTES

4. The definition of structured flowcharts defines the types of sequential structures, for example process, input or output, and subroutine call and the types of decision structures, such as single branch, double branch and case.
5. In waterfall model, documentation is quite easy. Developers may refer to documentation in case something goes wrong or a bug is found in the developed software. If the waterfall model has been followed by they must ensure that documentations is complete; failing which they may have to work real hard in going back to the previous starting point, i.e., in the system design phase.
6. Selection of design methodology depends on the size of the development time, the background and experience of analysts or designers, resources allocated to development, the type of application under development and the time allowance for the project.
7. The main objective of structured design is to minimize the complexity and increase the modularity of a program. Structured design also helps in describing the functional aspects of the system.
8. This form style has no border, no title bar at the top, and no maximize, minimize and close buttons. This style is usually used for splash screens. The term 'splash screen' is used for one of those screens that are shown as an application in loading. Default is no icon in the taskbar.
9. These features are listed below:
   - *Simplicity:* A software code should be written in a simple and concise manner. Simplicity should be maintained in the organization, implementation, and design of the software code.
   - *Modularity:* Breaking the software into several modules not only makes it easy to understand but also easy to debug. With the modularity feature, the same code segment can be reused in one or more software programs.
10. Writing an efficient software code requires a thorough knowledge of programming. This knowledge can be implemented by following a coding style which comprises several guidelines that help in writing the software code efficiently and with minimum errors. These guidelines, known as coding guidelines.
11. Generally, two types of commenting conventions are used: file header comments and trailing comments. File header comments are useful in providing information related to a file as a whole and comprise identification information, such as date of creation, name of the creator and a brief description of the code included in the software code.

Trailing comments are used to provide explanation of a single line of code. These comments are used to clarify the complex code. These also specify the function of the abbreviated variable names that are not clear. In some languages, trailing comments are used with the help of a double slash (//).

12. Coding methodology includes a diagrammatic notation for documenting the results of the procedure. It also includes an objective set (ideally quantified) of criteria for determining whether the results of the procedure are of the desired quality or not.
13. The purpose of structured programming is to make the software code easy to modify when required. Some languages, such as Ada, Pascal, and dBase are designed with features that implement the logical program structure in the software code. Primarily, the structured programming focuses on reducing and removing the following statements from the program.
14. Code verification is the process used for checking the software code for errors introduced in the coding phase. The objective of code verification process is to check the software code in all aspects. This process includes checking the consistency of user requirements with the design phase.
15. Static analysis comprises a set of methods used for analyzing the software source code or object code to understand how the software functions and to set up criteria to check its correctness. Static analysis studies the source code without executing it and reveals a variety of information like the structure of model used, data and control flows, syntactical accuracy and much more.
16. Documentation which focuses on the information that is used to determine the software code is known as internal documentation. It describes the data structures, algorithms, and control flow in the programs.
17. Documentation which focuses on general description of the software code and is not concerned with its detail is known as external documentation. It includes information such as function of code, name of the software developer who has written the code, algorithms used in the software code, dependency of code on programs and libraries, and format of the output produced by the software code.

7.8 SUMMARY

- The entire process of designing the software system is an exercise that specifies the 'What' and the 'How' of the system.
- Systems requirement tells what outputs are required. Before asking for a system, a customer wants certain outputs.

NOTES

- This is the starting point of the design process. The output design requires proper knowledge of the systems requirements.
- The basic documents containing the details of input data which is a part of system requirement specification and used to serve as a guideline.
- File design system accepts input data, it has to be stored for processing purposes, either for a short or a long period. Also, this data is stored in a structured form generally kept in the form of files in a logical manner. Once data is stored, it is retrieved by the computer program either to process it or to display it. Hence, systems designers devise techniques of storage and retrieval of daa from these files.
- Procedures are written as programs which specify the manner in which processing will be carried out.
- The two kinds of procedures adopted for this purpose are computer procedure and non-computer procedure. Computer procedure specifies functions that are to be carried out on computer.
- The control system is designed to properly sequence procedures or micro codes. It ties up procedures with data so that the design indicates the necessary procedures to ensure correct processing along with accuracy and timeliness of data.
- An aspect that should be kept in mind by a system developer is the presence of constraints that may hamper work.
- Input design is an important phase in system design phase. Businesses use the new technology to speed up the input process, reduce costs and capture data in new forms such as digital signature.
- Digital signature follows authentication mechanism. A code is attached with messages in the process of digital signature.
- A signature confirms that integrity and source of message is correct. National Institute of Standards and Technology (NIST) standard recognized the Decision Support System (DSS) standard that basically uses the Secure Hash Algorithm (SHA). Message authentication protects digital signature because in that mechanism messages are exchanged from the third party.
- Software Development Life Cycle (SDLC) methodology allows a project team to successfully build an application which is suited for the organization's needs. It also allows the organizations to incorporate the new requirements, technology and human resources to Information Technology (IT) development.
- System planning involves the project requirements summary, project team description and preliminary work schedule and service area's demographic analysis.
- System analysis is processed with E-R diagram, data flow diagrams and high level functional description.

NOTES

- Input design in SDLC has been drastically changed. In the network era, many input devices and techniques are available.
- A signature confirms that integrity and source of the message is correct. National Institute of Standards and Technology (NIST) standard recognized the digital signature standard to use basically the SHA.
- Remotely sensed data, such as digital orthophotos and satellites images are the types of data which are acquired by a sensor from distance whereas remotely sensed data are raster data but they are useful for vector data input.
- GPS data includes the horizontal location based on the geographic grid or a coordinate system. It has become a useful tool for spatial data input.
- Input data is determined while the time of input designs. It ensures the quality, accuracy and timeliness of input data. It is determined how data will be captured and entered into the system during input design.
- Completeness checks confirm that blank fields are not input and that cumulative input matches control totals.
- Limit checks factor confirms that a value does not exceed predefined limits.
- Range checks confirm that a value is within a predefined range of parameters.
- Sequence checks confirms that a value is sequentially input or processed.
- Validity checks confirm that a value conforms to valid input criteria.
- Data entry is usually performed in a specified time schedule, such as daily, weekly, monthly and for longer periods.
- The batch input is used in specific situations in which most business activity requires online data entry.
- This type of design is considered in the development phase based on the choice and applications it has to work with.
- The detailed design in the systems designing phase is taken as an input–process–output design. During the stage of input design, developers have to consider devices to be used for input.
- Input is the raw data that is processed to give output in an information system.
- To achieve the main objective of developing a software system, the input data entered by the end user needs to be converted into a format that the computer understands.
- As hardware devices are more reliable and less error prone when the conversion activity takes place, the second method of data conversion is preferred.

NOTES

- The main objective of input designing is to minimize the human element for balancing the slow input speed and error proneness of the end user input.
- Decision on data to be captured or accepted as input. Once input to the system is decided computer does the rest.
- Ensuring enough capacity with the system to handle the entered data. Fewer steps in data input will have fewer errors. For example, online forms are started with a well designed form so that a source document is used for data input.
- Input integrity controls cover a number of techniques that reduce errors while the end users make inputs. An input control is done by checking on the value of individual fields. Here, just filling the format is not sufficient.
- A smart systems analyst has to choose among existing alternatives before implementing an output format in which the output may be desired.
- Tabular output is the format that uses columns of text and numbers to present information.
- Pie charts depict the relationship between parts to the whole at some specific period. Certain pie chart styles are used to emphasize a particular item. A pie chart should be used to compare seven or fewer portions.
- A file is a collection of data that can be read from or written to. A file can be a program. You can create a file to write the data and store in suitable device. Commands, printers, terminals and application programs are all stored in files which allow users to access diverse elements of the system in a uniform way and give the operating system great flexibility.
- If prototype is generated for the required file, the minimum amount of code justifies the type of fields including database. A prototype is required for necessary elements that produces production module.
- File is a passive container of bytes on disk. The open file option is an active source or sink of bytes in a running program.
- The extension name appears at the end of the file name following a period and having three characters.
- Entity Relation Diagram (ERD) is a high level data model that includes all major entities and relationships. It illustrates the logical structure of databases.
- The entire system and its components are required to develop an ERD. ERD brings out issues, such as ambiguities, entities and their relationships, the types of data needed to be stored and the degree of a relationship. In 1976, Peter Chen developed ERD but Charles Bachman and James Martin have introduced more basic ERD principles.

NOTES

- System flowchart is a symbolic representation of solution of a given system design that is processed with the flow control of entire system.
- One of the files is maintained for admin side and other one is maintained for travellers.
- The Functional Decomposition Diagram (FDD) is basically business planning tool shown in the hierarchy of business functions, processes and sub-processes within an organization.
- In system development life cycle procedure, decomposition is the process of starting at a high level and moving into smaller subsystems. It is a type of planning tool to identify the business functions and processes that comprise them collectively.
- This diagram shows the hierarchical organization of functions and the processes that they include. This diagram contains a group of relevant functions which are cross organizational collections of activities that have to be performed further during system design.
- The logical functions and processes become the subject and therefore they focus on the organizing interviews and verification activities. Similarly, the logical processes will eventually be organized into application systems.
- The definition of structured flowcharts used further defines the types of sequential structures, for example process, input or output, and subroutine call and the types of decision structures, such as single branch, double branch and case.
- A correctly designed structured flowchart is easily understood by another programmer.
- Software implementation requires proper planning. A design is of no avail if it can not be implemented.
- In the implementation stage, no changes are made. In all previous stages, the developers had freedom of making changes, if any, in the software design plan. In implementation stage there is back tracking.
- Just putting these together is not important. It is important to make the system operational and user is able to perform tasks that it desired to perform. This is the stage when developers get the first taste of their actual work and evaluate themselves and their work.
- Interpretation of implementation stage is different for different Software Development Life Cycle (SDLC) models. Implementation suggests next development stage and is dependent on development model adopted.
- In this model, documentation is quite easy. Developers may refer to documentation in case something goes wrong or a bug is found in the developed software. If the waterfall model has been followed by they must

NOTES

ensure that documentations is complete; failing which they may have to work real hard in going back to the previous starting point, i.e., in the system design phase.

- The hardware and software requirements should be first re-examined with the software supplier. The equipments for general-purpose applications should be delivered several weeks before the installation of an application.
- Conversion methodology ensures a professional result in line with client expectations and within budgeted time period and cost.
- Formal training about the functioning of software should be provided to the employees for the successful use of the application software. Hardly ever, one reads a manual and implements the application in a smooth manner.
- Regular system backups should be taken so that the users can revert to an older copy of the data files when they commit any mistake. The backup procedure can be performed during the free hours or at the end of the day.
- Selection of design methodology depends on the size of the development time, the background and experience of analysts or designers, resources allocated to development, the type of application under development and the time allowance for the project.
- The main objective of structured design is to minimize the complexity and increase the modularity of a program. Structured design also helps in describing the functional aspects of the system.
- DataBase Management System (DBMS) must be followed by the entire database description managing the database. Therefore, steps are related to viewing the physical database design, transforming logical description with a physical model, describing the organization of the database and accessing the physical storage devices.
- The excellence of the system output is concluded by the quality of the system input. After understanding such significant relations, it becomes necessary that the input forms and screens as well as output reports should be designed.
- A standard form is interfaced with Minimize, Restore Down and Close buttons which are considered as basic parts of the form.
- This form style has no border, no title bar at the top, and no maximize, minimize and close buttons. This style is usually used for splash screens. The term 'splash screen' is used for one of those screens that are shown as an application in loading. Default is no icon in the taskbar.
- A flat form is a single copy form prepared manually or by a machine and printed on a paper.
- Simplicity is a software code should be written in a simple and concise manner. Simplicity should be maintained in the organization, implementation, and design of the software code.

NOTES

- Writing an efficient software code requires a thorough knowledge of programming. This knowledge can be implemented by following a coding style which comprises several guidelines that help in writing the software code efficiently and with minimum errors. These guidelines, known as coding guidelines.
- Generally, two types of commenting conventions are used: file header comments and trailing comments. File header comments are useful in providing information related to a file as a whole and comprise identification information, such as date of creation, name of the creator and a brief description of the code included in the software code.

  Trailing comments are used to provide explanation of a single line of code. These comments are used to clarify the complex code. These also specify the function of the abbreviated variable names that are not clear. In some languages, trailing comments are used with the help of a double slash (//).

  Coding guidelines can be used effectively to save time spent on gathering unnecessary details. These guidelines increase the efficiency of the software team while the software development phase is carried out.
- The written software code can be either simple or complex. Generally, it is observed that a complex segment of software code is more susceptible to errors than a segment containing a simple software code.
- Coding methodology includes a diagrammatic notation for documenting the results of the procedure. It also includes an objective set (ideally quantified) of criteria for determining whether the results of the procedure are of the desired quality or not.
- Programming refers to the method of creating a sequence of instructions to enable the computer to perform a task.
- Top-down programming focuses on the use of modules. It is therefore also known as modular programming.
- Bottom-up programming refers to the style of programming where an application is constructed with the description of modules.
- The purpose of structured programming is to make the software code easy to modify when required. Some languages, such as Ada, Pascal, and dBase are designed with features that implement the logical program structure in the software code. Primarily, the structured programming focuses on reducing and removing the following statements from the program.
- Information hiding refers to hiding details of a function or structure in software code. It focuses on making the functions or structures inaccessible to other components of software.
- Code verification is the process used for checking the software code for errors introduced in the coding phase. The objective of code verification

NOTES

process is to check the software code in all aspects. This process includes checking the consistency of user requirements with the design phase.

- Code reading is a technique that concentrates on how to read and understand a computer program.
- Static analysis comprises a set of methods used for analyzing the software source code or object code to understand how the software functions and to set up criteria to check its correctness. Static analysis studies the source code without executing it and reveals a variety of information like the structure of model used, data and control flows, syntactical accuracy and much more.
- Symbolic execution concentrates on assessing the accuracy of the model by using symbolic values instead of actual data values for input.
- Code documentation is a manual-cum-guide that helps in understanding and correctly utilizing the software code.
- Documentation which focuses on the information that is used to determine the software code is known as internal documentation. It describes the data structures, algorithms, and control flow in the programs.
- Documentation which focuses on general description of the software code and is not concerned with its detail is known as external documentation. It includes information such as function of code, name of the software developer who has written the code, algorithms used in the software code, dependency of code on programs and libraries, and format of the output produced by the software code.

7.9 KEY WORDS

- **Procedure design:** This is a type of design to process the data which is to be fed to the system and produced output.
- **SDLC methodology**: This approach allows a project team to successfully build an application which is suited for the organization's needs.
- **Stacked bar chart:** This type of chart portrays the relationship of individual items to its 'Whole'.
- **Tree of directories and files:** This approach refers to an interface containing file and directory names which are named with path names.
- **Bottom-up programming:** It refers to the style of programming where an application is constructed with the description of modules.
- **Structured design:** Structured design is a data-flow based methodology that helps in identifying the input and output of the developing system.

7.10 SELF ASSESSMENT QUESTIONS AND EXERCISES

NOTES

Short-Answer Questions

1. Define the term control design.
2. Give the difference between the batch and online input.
3. Define the term output design.
4. How the file represents the interfaces?
5. Write the objectives of FDD.
6. How the iterative model is implemented?
7. Name some important designs which are included in design methodologies.
8. What is structured design?
9. Explain the term input form design.
10. What is software design?
11. Explain the concept of naming conventions.
12. Elaborate on the term bottom-up programming.
13. What do you understand by structured programming?
14. What is code reading?
15. Define the term coding tools.

Long-Answer Questions

1. Discuss about the process designing giving appropriate components of system design.
2. Briefly explain the input design and its objectives.
3. Describe the file design and also explain the entity relation diagram with the help of diagrams.
4. Differentiate between logical and physical design giving appropriate examples.
5. Explain the characteristic the features and examples following:
   (i) System flowchart
   (ii) Functional decomposition diagram
   (iii) Structured flowcharts
   (iv) Proper equipment
   (v) Structured design
   (vi) Structured charts

NOTES

6. Discuss briefly about design methodologies and their features.
7. Briefly explain the form design methodology giving examples.
8. Discuss about the features of a software code with the help of diagram and an example.
9. Elaborate on the advantages of coding guidelines giving examples.
10. Discuss design methodologies and their features.
11. Discuss the difference between top-down and bottom-up programming giving appropriate examples.
12. Describe the code documentation and explain its types.

7.11 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 8 INPUT OUTPUT AND FORM DESIGN

NOTES

Structure

8.0 Introduction
8.1 Objectives
8.2 Input Design
8.3 Output Design
8.4 Answers to Check Your Progress Questions
8.5 Summary
8.6 Key Words
8.7 Self Assessment Questions and Exercises
8.8 Further Readings

8.0 INTRODUCTION

This is the second most important part of a system design. After deciding on the output requirements and finalizing the same, data that is required as an input to produce the desired output has to be identified. The basic documents containing the details of input data which is a part of system requirement specification is used to serve as a guideline. It may become necessary to revise these documents and create new documents for this purpose. As far as possible, requirements should be finalized in Software Requirements Specification (SRS). Introducing fresh requirement at the design stage creates difficulties for systems designers.

Systems requirement tells what outputs are required. Before asking for a system, a customer wants certain outputs. This is the starting point of the design process. The output design requires proper knowledge of the systems requirements.

In this unit, you will study about the input design, capturing data for input, input validation, input design of on-line systems, output design, printed, display and audio.

8.1 OBJECTIVES

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

- Understand the input design
- Explain how the data is captured for input
- Define the input validation
- Discuss about the input design of on-line systems
- Know about the output design, printed, display and audio

NOTES

8.2 INPUT DESIGN

Input design is an important phase in system design phase. Businesses use the new technology to speed up the input process, reduce costs and capture data in new forms such as digital signature. Digital signature follows authentication mechanism. A code is attached with messages in the process of digital signature. Primarily, the signature is generated by hashing the message and then later this message is encrypted with the sender's private key. Digital signature is based on public key encryption. A signature confirms that integrity and source of message is correct. National Institute of Standards and Technology (NIST) standard recognized the Decision Support System (DSS) standard that basically uses the Secure Hash Algorithm (SHA). Message authentication protects digital signature because in that mechanism messages are exchanged from the third party. Digital signature is analogous to manual signature. The main objective of input design is to ensure the quality, accuracy and timeliness of data. The main objectives of input design are to select a suitable input and data entry method, reduce input volume, design attractive data entry screens, use validations checks to reduce input errors, design required source documents as well as forms and develop effective input controls. Systems analysts apply business process engineering techniques when studying transactions and business operations to determine how and when data should enter the system. Data entry is the process of manually entering data into the information system usually in the form of keystrokes and mouse clicks. Data input is the operation of encoding the data and writing them to the database. From a user's point of view, the interface is the most critical part of the system design because it is considered as a checkpoint through which users's can interact with the system. For this, it is essential to construct models and prototypes for user approval. You can prepare initial input screen design to users in the form of storyboard which represents a rough layout showing the sample screen for proposed system. Information systems development must address following broader organizational issues as well:

- What is the justification for the type of application which has to be developed?
- What actual or desired processes should the application perform?
- How will we verify that the application performs as per design?

SDLC (Software Development Life Cycle) methodology allows a project team to successfully build an application which is suited for the organization's needs. It also allows the organizations to incorporate the new requirements, technology and human resources to Information Technology (IT) development. Input data is considered as prime tool for SDLC steps. Following are the activities performed in design methodology:

- System planning involves the project requirements summary, project team description and preliminary work schedule and service area's demographic analysis.

NOTES

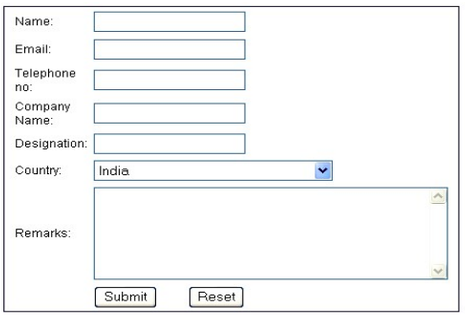
- System analysis is processed with E-R diagram, data flow diagrams and high level functional description.
- System design involves working with Relational DataBase Management System (RDBMS), data dictionary, identification or description of database objects and Web site map.
- Systems implementation involves working with Microsoft Access database, Web site content and elementary test plan.

Hence this signature is considered as a prime tool for input design in online transaction. It attaches date and time along with author of the signature. It authenticates the contents when signature has been completing. It solves the disputes using third party generally in online payment by PayPal. It also ensures that message has not been altered while sending and receiving from one user to another. The message can be electronic documents, such as e-mail, text file, spreadsheet, etc. The data entered into an information system checks the quality of the output as well as quality of input. This concept is sometimes known as Garbage In, Garbage Out (GIGO). The main objective of input design is to ensure the quality, accuracy and timeliness of input data. Let us take an example of Geographical Information System (GIS) in which input data refers to data sources for creating new data that includes remotely sensed data, for example satellite images, aerial photographs, Global Positioning System (GPS) data and paper maps. Remotely sensed data and GPS data are the primary data sources and paper maps are secondary data sources which are discussed below:

- **Remotely Sensed Data:** Remotely sensed data, such as digital orthophotos and satellite images are the types of data which are acquired by a sensor from distance whereas remotely sensed data are raster data but they are useful for vector data input. Digital orthophotos are digitized aerial photographs that have been differentially rectified or corrected to remove image displacements.
- **GPS Data:** GPS data includes the horizontal location based on the geographic grid or a coordinate system. It has become a useful tool for spatial data input.
- **Paper Maps:** These include all types of hard copy maps.

Following screen shows the screen of various types of input data in which **Name:, E-mail:, Company Name:, Designation:, Remarks:** fields have been taken as string data type whereas **Telephone no:** has been taken as integer data type for inputting the data. Two buttons, known as Submit and Reset have been interfaced on the screen to submit and reset operations respectively, and **Remarks:** field has been set for user's feedback for the designed screen.

NOTES



Input data is determined while the time of input designs. It ensures the quality, accuracy and timeliness of input data. It is determined how data will be captured and entered into the system during input design. Data capture uses an automated or manually operated device to identify source data and convert it into computer readable form, for example credit card scanners, bar code readers, etc. Table 8.1 summarizes the list of various input devices and their functions which are used to input data for designed system.

Table 8.1 *Input Devices and their Functions*

Input Devices	**Functions**
Feedback devices	These types of devices create a digital image of biological data, such as fingerprints, retina patterns and facial characteristics.
Brain computer interface	This interface is used to translate the neural brain signals into digital commands.
Digital camera	This device records the photographs in digital forms rather than traditional film. The resulting data file can be stored, displayed or manipulated by the computer.
Electronic whiteboard	This device captures and stores text or graphics which are displayed on the board and functions as a touch screen.
Handheld computer stylus	This device allows users to form the characteristics on the screen of a handheld computer. Programs are handwriting recognition software that can translate the characters into digital input.
Internet workstations	This device enables the users to provide input to Web based Internet or intranet recipients who can integrate with information system output or personal computer applications.
Keyboard	This device is considered as prime device for inputting the data.
Microphone	This device is used to convert sound waves into digital information that can be stored, transmitted or attached to e-mail. Users can input data and issue commands using spoken words.
Motion sensors	This device is used to estimate an object's motion in a three dimensional position. This device is also used in virtual reality programs and gaming applications.
Pointing stick	This device is also known as pressure sensitive pointing device which is located between the keys on a notebook computer. This device resembles a button or pencil eraser.
Scanner and optical recognition	This device reads printed bar codes, characters and images. An Optical Character Recognition (OCR) device is used to convert scanned material to get the digital information.
Terminal	This device is especially used in intelligent screen, i.e., keyboard for independent processing.
Touch screen or pad	The sensors allow users to select options by touching the specific locations on the screen. The touch pads are used as pointing devices that use touch screen input.
Video input	It produces result in digital form. The produced data can be stored and replayed for further requirements.

NOTES

Input data controls the necessary measures, such as input values are correct, complete and secure. This control is focussed on every phase of input design which starts with source documents and promotes data accuracy and quality. If batch input method is used then the computer can produce an input log file which identifies and documents the data entered. Examples of input controls include the following factors while the time of inputting data:

- **Check Digits:** Check digits are numbers produced by mathematical calculations performed on input data, such as account numbers. The calculation confirms the accuracy of input by verifying the calculated number against other data in the input data, typically the final digit.
- **Completeness Checks:** Completeness checks confirm that blank fields are not input and that cumulative input matches control totals.
- **Duplication Checks:** Duplication checks confirm that duplicate information is not input.
- **Limit Checks:** Limit checks factor confirms that a value does not exceed predefined limits.
- **Range Checks:** Range checks confirm that a value is within a predefined range of parameters.
- **Reasonableness Checks:** Reasonableness checks confirm that a value meets predefined criteria.
- **Sequence Checks:** Sequence checks confirms that a value is sequentially input or processed.
- **Validity Checks:** Validity checks confirm that a value conforms to valid input criteria.

An audit trail must be provided so that the source of each data item is entered into the system. The data security policies and procedures protect data from damaging. Input controls help to ensure the employees accurately for input information. Systems properly record input and systems either reject or accept the record and also input errors for later review and corrections at the time of system development. An input device is attached to the computer and is helpful in data processing. This device is also known as online input device and data is said to be entered online. For this, computer terminal is required to be attached to the computer. It consists of keyboard for data entry and other means of displaying entered data. Data can be entered using typing devices (keyboard), pointing devices (mouse, joystick, light pens) and voice recognition devices (microphone). Input processes must be logically arranged at the time of setting transactions. Business operations determine how and when data should enter in the system. It has two prime methods known as batch input and online input methods.

- **Batch Input:** Data entry is usually performed in a specified time schedule, such as daily, weekly, monthly and for longer periods. For example, batch input occurs if a payroll department collects time cards at the end of the week and enters data as a batch. Another example of batch input can be,

NOTES

to enter all the grades for academic terms with the help of batch processing system.

- **Online Input:** The batch input is used in specific situations in which most business activity requires online data entry.

The detailed design phase of SDLC carries out construct and test activities to check, redefine and then revalidate the design. The development team checks the following factors in the detailed design phase:

Elaborate Design

This kind of design is built on the high level to incorporate the designs and mechanisms with reference to its state, attributes and association, and interdependencies.

Finalizing Architecture of the Project

This part of the detailed design considers all the factors before finalizing the architecture of the project and conveys decision to the developers and the systems analysts.

Integrating with Target Environment

This type of design is considered in the development phase based on the choice and applications it has to work with. For example, one may choose SQL Server, WebSphere, eXtensible Markup Language (XML) or any other application with which the system will be integrated. During this process some infrastructure constraints should also be kept in mind.

Plan Deployments

This factor dictates the way and the time of the deployment with the system with which it is to be linked. The system under development may be implemented with Enterprise ARchive (EAR) files, Java ARchive (JAR) files, etc.

Specify Test Details

This factor identifies all those cases that qualify as test cases. This also includes descriptions of the tests to be carried out, inputs needed for these tests, the condition of execution and results expected.

Detailed Design Phenomena

The detailed design in the systems designing phase is taken as an input–process–output design. During the stage of input design, developers have to consider devices to be used for input. Such input devices may be a desktop PC, a pager, a Magnetic Ink Character Reader (MICR), an Optical Character Reader (OCR), an Optical Mark Reader (OMR), etc.

Input data shown in a Data Flow Diagram (DFD) turns into input modules and data in the structural chart. Each input enters a process on a DFD. Automation of input is worked with electronic notepads or barcode scanning and this reduces the likelihood of human error while making data entries. Once the devices, controls

NOTES

and input fields are identified, the developers take into consideration the input format either through forms or input screens. There must also be a dialog or interaction between the system and the user. Java, C++, Visual Basic and C# are development languages that contain libraries which speed up the process that generates reports and screen displays.

Objectives of Input Design

Input is the raw data that is processed to give output in an information system. Therefore, the quality of system input determines the quality of system output. Well designed input forms and screens should meet the following objectives:

(i) **Effectiveness:** It must serve specific purposes. Forms are created to serve one or more purposes, such as recording, processing, storing and retrieving information for business.

(ii) **Accuracy:** It assures proper completion. Error rates associated with collecting data drop sharply if the forms are designed to ensure accurate completion.

(iii) **Ease of Use:** It should be straightforward. To reduce errors, check speed completion and facilitate the entry of data, it is essential that forms be easy to fill in. Forms should be designed with proper flow either from left to right or top to bottom. Logical grouping of information makes it easy for people to fill out forms correctly. The groupings are as follows:

- *Consistency*: Groups the data similarity from one application to the next.
- *Simplicity:* Keeps data uncluttered.
- *Attractiveness:* Focuses on user's attention.

All of these objectives are attainable through the use of basic design principles, through the knowledge of what is needed as input for the system and through an understanding of how end users respond to different elements of forms and screens. Forms have certain primary sections including instructions, heading, signature and verification, body, totals, and comments.

Selection of Input Media and Devices

To achieve the main objective of developing a software system, the input data entered by the end user needs to be converted into a format that the computer understands. Also, to receive user input, one needs to use devices for conversion of data on documents at source into a variety of storage media like floppy diskettes, magnetic tapes, Compact Disks (CD) or flash drives. The two ways of data conversion include data conversion in which human beings are responsible for performing the conversion and those where hardware devices are responsible for performing the conversion activity.

As hardware devices are more reliable and less error prone when the conversion activity takes place, the second method of data conversion is preferred. However, since hardware devices are inflexible, they are more suitable for such situations that involve a large amount of specific type of data conversion as processing of cheques by banks and payments by credit or debit cards at grocery stores. In real life situations of business data processing, conversion is performed

NOTES

by the combined activities of humans and hardware. For example, human beings make data entry through keyboard and this is followed by conversion by a hardware device that converts the data into a code that is machine readable. In a situation like this, editing is very important as it minimizes the number of errors. Therefore, the main objective of input designing is to minimize the human element for balancing the slow input speed and error proneness of the end user input.

Key Tasks in Input Design

The following are the essential tasks involved in the input design:

- Designing data entry and input procedures.
- Designing source documents for data capture or devising other data capture methods.
- Designing input data records.
- Designing data entry screens.
- Designing screens for user interface.
- Designing security measures for the system and audit trails.

Data Capture

Data capturing is the identification and recording of source data. The following factors are involved in this process:

- Decision on data to be captured or accepted as input. Once input to the system is decided computer does the rest.
- Ensuring enough capacity with the system to handle the entered data. Fewer steps in data input will have fewer errors. For example, online forms are started with a well designed form so that a source document is used for data input.
- It is at times appropriate to reveal information via a code. For example, a salesperson uses codes to locate goods and goods codes allow meaningful data entry. The functions code can be used to request appropriate action from a computer or decision maker.

Guidelines for Establishing Code

The following points, with respect to codes, should be kept in mind:

- They should be kept short.
- They should be kept stable.
- They should be unique.
- They should not be confusing.
- They should be uniform.
- They should be modifiable.
- They should be made meaningful.

NOTES

Input Integrity Controls

Input integrity controls cover a number of techniques that reduce errors while the end users make inputs. An input control is done by checking on the value of individual fields. Here, just filling the format is not sufficient. Completeness of all inputs such that all the relevant fields are filled is important. Transaction logs are used to create audit trails for data entry and other system activities. This keeps a track of modifications that are made in the table of database providing security and means of recovery in case of a hardware or power failure, etc.

Screen Design Guidelines

The following are the guidelines to develop a screen design:

- The display screen should be simple and consistent and should show only that which is necessary for the end user.
- Screens can be kept consistent if information is located in the same area each time a new screen is accessed. Similar information should be consistently grouped together.
- Facilitate the navigation of end users among various screens. Web based forms facilitate movement with the use of hyperlinks.

Input Validation

Input validation, also known as data validation, is the proper testing of any input supplied by a user or application. Input validation prevents improperly formed data from entering an information system. Because it is difficult to detect a malicious user who is trying to attack software, applications should check and validate all input entered into a system. Input validation should occur when data is received from an external party, especially if the data is from untrusted sources. Incorrect input validation can lead to injection attacks, memory leakage, and compromised systems. While input validation can be either whitelisted or blacklisted, it is preferable to whitelist data. Whitelisting only passes expected data. In contrast, blacklisting relies on programmers predicting all unexpected data. As a result, programs make mistakes more easily with blacklisting.

Check Your Progress

1. What is a digital signature?
2. Explain the main objective of input design.
3. Define data entry.
4. Elaborate on the activities performed in design methodology.
5. Why we use Geographical Information System (GIS).
6. Write down the devices used in data entry.
7. Give the definition of data validation.

NOTES

8.3 OUTPUT DESIGN

A smart systems analyst has to choose among existing alternatives before implementing an output format in which the output may be desired. A suitable output format should be selected depending upon the requirement of the output. Several formats are available from which selection can be made to communicate information. Narrative format uses standard text, numbers and pictures. It is a format that has been exploited by word processing technology for reports, personalized form letters as well as business letters.

Tabular output is the format that uses columns of text and numbers to present information. This method is most common for presenting computer outputs. Zoned output places the text and numbers in the selected areas of a form or a screen. Graphic output uses graphs or charts to represent information. This format enables end users to better understand the relationships between data and trends. Pie chart, bar chart, column chart, line diagram and scatter diagrams are the most commonly used graphs.

Pie charts depict the relationship between parts to the whole at some specific period. Certain pie chart styles are used to emphasize a particular item. A pie chart should be used to compare seven or fewer portions.

Bar chart is an output format that represents information using bars. Categories or classes for comparison are organized in a vertical manner whereas the values are arranged in a horizontal manner. A stacked bar chart portrays the relationship of individual items to its 'whole.'

Column chart is a simple variation of the bar chart. Column charts enable the user to depict comparisons among items and show the variation in an item over a period of time. This chart organizes categories horizontally and values vertically. Such appearance emphasizes variations over a period of time.

Line charts show trends of an item over a period of time. It organizes items on the horizontal axis and measurements on the vertical axis.

Those formats that plot data values of two different items that reflect uneven intervals of data are called scatter charts. Standard statistical methods are applied to establish the degree of existing correlation.

Output Integrity Controls

Such controls consist of routing codes identifying the receiving system as well as verification messages which confirm the successful receipt of messages. The network protocol can handle routing and verification. Therefore, these services may not be required by the application. The common forms of output are messages, screens and reports. Screen format reports and printed reports should indicate the date or time of the data. It should also indicate the date or time at which the report was printed. It should contain other information, such as the time of the report data along with description and title of the report as well as the pagination of those reports that have multiple pages. Usually a version number along with an effective date is mentioned on pre-printed forms.

NOTES

Output Design Methods

Producing different types of output requires different technologies. Some common output methods are:

1. Printers

Printers are widely used output devices. It is used to generate had copy.

Advantages

(i) It is permanent.
(ii) It is portable.
(iii) Its information cannot be changed by the user.
(iv) It provides detailed information.
(v) Affordable for most organization.

Disadvantages

(i) May be noisy.
(ii) Compatibility problems with computer software.
(iii) May be slow.

2. Screens as Output

Screens are increasingly popular output technology. It becomes feasible because as their size and price decreases & their compatibility with other components increases.

Advantages

(i) Interactive
(ii) Quiet
(iii) Good for frequently accessed.
(iv) Works in online & real time transmission.

Disadvantages

(i) Require cabling & setup space.
(ii) Is expensive to develop.

3. Video, Audio & Animation

A video is a complex form of output since it combines the strength and potential emotional impact of audio with a stimulating visual channel.

NOTES

Advantages

(i) Good for the individual user.

(ii) Good for transient messages.

(iii) God where the worker needs hand free.

(iv) Good if the output is highly repetitive.

Disadvantages

(i) Is expensive to develop.

(ii) Needs a dedicated room where the output will not interfere with other tasks.

(iv) Has limited application.

4. CD-ROMs and DVDs

The display of material on **CD_ROMs** have increased because of the growing demand of multimedia output.

Advantages

(i) Speedy retrieval.

(ii) Allows multimedia output.

(iii) Has large capacity.

(iv) Less vulnerable to damage.

Disadvantages

(i) More difficult to develop.

(ii) Expensive to develop.

Design of Output Reports

In designing a printed report, the system analyst incorporates both functional & stylistic considerations so that report supplies the user necessary information in a readable format.

Check Your Progress

8. Why pie charts are used?
9. Define the concept of bar chart.

8.4 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

NOTES

1. Input design is an important phase in system design phase. Businesses use the new technology to speed up the input process, reduce costs and capture data in new forms such as digital signature. Digital signature follows authentication mechanism. A code is attached with messages in the process of digital signature. Primarily, the signature is generated by hashing the message and then later this message is encrypted with the sender's private key.
2. The main objective of input design is to ensure the quality, accuracy and timeliness of data. The main objectives of input design are to select a suitable input and data entry method, reduce input volume, design attractive data entry screens, use validations checks to reduce input errors, design required source documents as well as forms and develop effective input controls.
3. Data entry is the process of manually entering data into the information system usually in the form of keystrokes and mouse clicks. Data input is the operation of encoding the data and writing them to the database.
4. Following are the activities performed in design methodology:
   - System planning involves the project requirements summary, project team description and preliminary work schedule and service area's demographic analysis.
   - System analysis is processed with E-R diagram, data flow diagrams and high level functional description.
   - System design involves working with Relational Database Management System (RDBMS), data dictionary, identification or description of database objects and Web site map.
   - Systems implementation involves working with Microsoft Access database, Web site content and elementary test plan.
5. Geographical Information System (GIS) in which input data refers to data sources for creating new data that includes remotely sensed data, for example satellite images, aerial photographs, Global Positioning System (GPS) data and paper maps.
6. Data can be entered using typing devices (keyboard), pointing devices (mouse, joystick, light pens) and voice recognition devices (microphone).
7. Input validation, also known as data validation, is the proper testing of any input supplied by a user or application. Input validation prevents improperly formed data from entering an information system.
8. Pie charts depict the relationship between parts to the whole at some specific period. Certain pie chart styles are used to emphasize a particular item. A pie chart should be used to compare seven or fewer portions.

9. Bar chart is an output format that represents information using bars. Categories or classes for comparison are organized in a vertical manner whereas the values are arranged in a horizontal manner.

NOTES

8.5 SUMMARY

- Input design is an important phase in system design phase. Businesses use the new technology to speed up the input process, reduce costs and capture data in new forms such as digital signature. Digital signature follows authentication mechanism. A code is attached with messages in the process of digital signature. Primarily, the signature is generated by hashing the message and then later this message is encrypted with the sender's private key.
- Digital signature is based on public key encryption. A signature confirms that integrity and source of message is correct.
- The main objective of input design is to ensure the quality, accuracy and timeliness of data. The main objectives of input design are to select a suitable input and data entry method, reduce input volume, design attractive data entry screens, use validations checks to reduce input errors, design required source documents as well as forms and develop effective input controls.
- Data entry is the process of manually entering data into the information system usually in the form of keystrokes and mouse clicks. Data input is the operation of encoding the data and writing them to the database.
- SDLC (Software Development Life Cycle) methodology allows a project team to successfully build an application which is suited for the organization's needs.
- The message can be electronic documents, such as e-mail, text file, spreadsheet, etc.
- Geographical Information System (GIS) in which input data refers to data sources for creating new data that includes remotely sensed data, for example satellite images, aerial photographs, Global Positioning System (GPS) data and paper maps.
- Check digits are numbers produced by mathematical calculations performed on input data, such as account numbers.
- Limit checks factor confirms that a value does not exceed predefined limits.
- Range checks confirm that a value is within a predefined range of parameters.
- Sequence checks confirms that a value is sequentially input or processed.
- Validity checks confirm that a value conforms to valid input criteria.
- An audit trail must be provided so that the source of each data item is entered into the system.

NOTES

- Data can be entered using typing devices (keyboard), pointing devices (mouse, joystick, light pens) and voice recognition devices (microphone).
- Input validation, also known as data validation, is the proper testing of any input supplied by a user or application. Input validation prevents improperly formed data from entering an information system.
- Data entry is usually performed in a specified time schedule, such as daily, weekly, monthly and for longer periods.
- The batch input is used in specific situations in which most business activity requires online data entry.
- The detailed design in the systems designing phase is taken as an input–process–output design. During the stage of input design, developers have to consider devices to be used for input.
- Once the devices, controls and input fields are identified, the developers take into consideration the input format either through forms or input screens.
- Java, C++, Visual Basic and C# are development languages that contain libraries which speed up the process that generates reports and screen displays.
- Input is the raw data that is processed to give output in an information system.
- To achieve the main objective of developing a software system, the input data entered by the end user needs to be converted into a format that the computer understands.
- The two ways of data conversion include data conversion in which human beings are responsible for performing the conversion and those where hardware devices are responsible for performing the conversion activity.
- As hardware devices are more reliable and less error prone when the conversion activity takes place, the second method of data conversion is preferred.
- The main objective of input designing is to minimize the human element for balancing the slow input speed and error proneness of the end user input.
- Decision on data to be captured or accepted as input. Once input to the system is decided computer does the rest.
- Ensuring enough capacity with the system to handle the entered data. Fewer steps in data input will have fewer errors. For example, online forms are started with a well designed form so that a source document is used for data input.
- Input integrity controls cover a number of techniques that reduce errors while the end users make inputs. An input control is done by checking on the value of individual fields. Here, just filling the format is not sufficient.

NOTES

- Narrative format uses standard text, numbers and pictures. It is a format that has been exploited by word processing technology for reports, personalized form letters as well as business letters.
- Tabular output is the format that uses columns of text and numbers to present information. This method is most common for presenting computer outputs. Zoned output places the text and numbers in the selected areas of a form or a screen.
- Pie charts depict the relationship between parts to the whole at some specific period. Certain pie chart styles are used to emphasize a particular item. A pie chart should be used to compare seven or fewer portions.
- Bar chart is an output format that represents information using bars. Categories or classes for comparison are organized in a vertical manner whereas the values are arranged in a horizontal manner.
- Column chart is a simple variation of the bar chart. Column charts enable the user to depict comparisons among items and show the variation in an item over a period of time.
- Line charts show trends of an item over a period of time. It organizes items on the horizontal axis and measurements on the vertical axis.
- Such controls consist of routing codes identifying the receiving system as well as verification messages which confirm the successful receipt of messages. The network protocol can handle routing and verification.
- Usually a version number along with an effective date is mentioned on pre-printed forms.
- Producing different types of output requires different technologies.
- Screens are increasingly popular output technology. It becomes feasible because as their size and price decreases & their compatibility with other components increases.
- A video is a complex form of output since it combines the strength & potential emotional impact of audio with a stimulating visual channel.
- The display of material on CD_ROMs have increased because of the growing demand of multimedia output.
- In designing a printed report, the system analyst incorporates both functional and stylistic considerations so that report supplies the user necessary information in a readable format.

8.6 KEY WORDS

- **Geographical Information System (GPS) data:** GPS data includes the horizontal location based on the geographic grid or a coordinate system. It has become a useful tool for spatial data input.

NOTES

- **Paper maps:** Hard copy maps.
- **Validity checks:** Validity checks confirm that a value conforms to valid input criteria.
- **Tabular output:** Format that uses columns of text and numbers to present information.
- **System flowcharts:** A symbolic representation of solution of a given system design that is processed with the flow control of entire system.
- **Data capture:** Data capturing is the identification and recording of source data.

8.7 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. What is an input?
2. What does GPS data include?
3. Explain the term data capturing.
4. Define the term data validation.
5. Why the column charts are used?
6. Define the term line chart.
7. Explain about the output design methods.

Long- Answer Question

1. Describe the input design and explain the activities of design methodology giving examples.
2. Briefly explain the input devices giving the functions of each device.
3. Describe in detail the different types of factors involved it the time of inputting data.
4. Briefly explain the objectives of input design.
5. Discuss about the output design and its methods giving details.
6. Elaborate on the different types of charts used in output design.

8.8 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

NOTES

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 9 FORMS DESIGN

NOTES

Structure

9.0 Introduction
9.1 Objectives
9.2 Forms Design
- 9.2.1 Classification of Forms
- 9.2.2 Requirements of Form Design
- 9.2.3 Styles and Types of Form
- 9.2.4 Forms Control

9.3 Answers to Check Your Progress Questions
9.4 Summary
9.5 Key Words
9.6 Self Assessment Questions and Exercises
9.7 Further Readings

9.0 INTRODUCTION

The systems analyst should be capable of designing a complete and useful form. Unnecessary forms that waste an organization's resources should be eliminated. Forms are important instruments for steering the course of work. They are pre-printed papers that require people to fill in responses in a standardized way. Forms elicit and capture information required by organizational members that will often be input to the computer. Through this process, forms often serve as source documents for users or for input to ecommerce applications that humans must enter.

To reduce error, speed completion, and facilitate the entry of data, it is essential that forms be easy to fill in. The cost of the forms is minimal compared with the cost of the time employees spend filling them in and then entering data into the information system. It is often possible to eliminate the process of transcribing data that are entered on a form into the system by using electronic submission.

Designing a form with proper flow can minimize the time and effort expended by employees in form completion. Forms should flow from left to right and top to bottom. Illogical flow takes extra time and is frustrating. A form that requires people to go directly to the bottom of the form and then skip back up to the top for completion exhibits poor flow.

In this unit, you will study about the definition, classification of forms, requirements of forms design, types of forms and forms control.

9.1 OBJECTIVES

NOTES

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

- Explain about the forms
- Discuss about the classification of forms
- Understand the requirements of forms design
- Describe the types of forms
- Define the forms control

9.2 FORMS DESIGN

Form is a tool with a message. It is the physical carrier of data-information. I tis an either an authority for action or a request for action.

Every rule related to the logical and physical data structure that has been obligated by the DataBase Management System (DBMS) must be followed by the entire database description managing the database. Therefore, steps are related to viewing the physical database design, transformation of the logical description with a physical model describing the organization of the database and the method of accessing the physical storage devices.

The user interface usually comprises of various documents, reports and different screens of a system and is the sole component of the system that is viewed by the user. Hence, for the user the user interface is a vital component of the system. It should be designed in such a way that besides delivering information to the user it should also be visible.

For the purpose of designing an interface, the steps required are stated as follows:

- Generation of the interface summary table for system use by following methods:
  - o Analysing the human computer limit in the Data Flow Diagram (DFD).
  - o Filing the interface and analysing the design for the purpose of defining the data elements that have to be added in every form/ document, input screen and output report.
- Designing the external input forms.
- Designing the human computer dialogs.
  - o Choosing the kind(s) of dialog.
  - o Designing every dialog.

- Designing input (data entry) screens.
- Report designing
    - o Choosing the output technique for every report.
    - o Internal report designing.
    - o External report designing.

NOTES

Input Form Design

The excellence of the system output is concluded by the quality of the system input. After understanding such significant relations, it becomes necessary that the input forms and screens as well as output reports should be designed.

The basic objectives of the input form design must be achieved by a good designed input form and visual display screen which can only take place by using the essential design principles, information about which type of data is required to be used in the system and recognizing the way by which the users react to various different components of forms and screens.

The term 'effectiveness' basically refers that the input forms and screens play a particular role in the Management Information System (MIS) whereas the term 'accuracy' highlights the design that guarantees a suitable conclusion.

The objectives of the forms designing are that extraction and capturing the needed information through the members of the organization will be frequently added to the computer with the help of form designing. Forms act as the source documents regarding the data entry personnel with the help of this process.

The guidelines of form design are as follows:

- They ease in filling out the forms.
- They provide certainty of the forms meeting the reason of their being.
- They make the forms eye catching.
- They design the formats for confirming precise completion.

Following guidelines are considered while the time of input screen design:

Input Screen Design

For the purpose of entering the data in the input forms or reports of the computer and updating the data, the data entry is made use of. The four guidelines for screen design are as follows:

- It maintains the simplicity of the screen.
- It keeps the screen presentation constant.
- It improves the user movement amongst screens.
- It generates an eye catching screen.

NOTES

Following are the two problems that occur in context with the designing data entry screens:

- The momentum of data entry.
- The preciseness of data entry.

Maintaining the Simplicity of the Screen: The initial guideline regarding a perfect screen design is simplifying the screen display. Only the required material for the specific action carried out should be shown by the display screen. Practicing such steps will turn amateur users into smart ones as this will reduce the chances of them causing any serious error.

Keeping the Screen Consistent: The next guideline for creating a good screen design is to maintain the consistency of the display which can only be achieved through locating information in the similar area every time when the user is accessing a new screen. Moreover, all information that has logical similarities should be grouped together.

Facilitating Movement between the Screens: Next guideline is making the browsing among various screens easier. This can be done by making the users develop a feeling of physically moving to a novel screen which can be done through the use of the scroller. The designing of the screens should be done in such a way which provides awareness in the user about the status of every action. Following are the ways by which this can be done:

- Usage of error messages for providing feedback on every error.
- For the purpose of providing feedback on update actions and usage of conformation message.
- In cases where there is any occurrence of backend process, the usage of status messages.
- When the actions are far reaching, permitting the user for reversing an action whenever probable.

Generation of an Attractive Screen: The last guideline is creating an eye catching screen for the user by trying not to make it completely highlighted. The productivity of the user will increase if he finds the screens appealing and would also require reduced supervision which would result in decreasing the amount of errors. Screens should be such that attract the attention of the user and make them concentrate at a single position. This may be accomplished through using more of open area adjoining the data entry fields, thereby making the screen have a tidy look.

Basic Parts of a Form

To make it easy to navigate from one screen to another, systems analysts may use one of the following methods:

- Scrolling the screen back and forth.
- Calling up another screen for more detail.
- Using on-screen dialogue through the prompts.

Various features of electronic form design software are as follows:

- An electronic from design software is able to design paper, electronic or Web based forms.
- This software design a form using templates.
- This form design by cutting and pasting familiar shapes and objects.
- It facilitates completion through the use of software.
- It permits customized menus, toolbars, keyboards and macros.
- It supports popular databases.
- It enables broadcasting of electronic forms.
- It permits sequential routing of forms.
- It assists form tracking.
- It encourages automatic delivery and processing.
- It establishes security for electronic forms.

NOTES

Controlling Forms

Controlling business forms is an important task and usually organizations often have a form specialist that does this job. Basic duties for controlling forms are as follows:

- Making sure that each form fulfils its specific purpose and that specific purposes is integral to organizational functioning.
- Preventing duplication of the information that is collected and of the forms that collect it.
- Designing effective forms.
- Deciding on how to get forms reproduced in the most economical way.
- Establishing stock control and inventory procedures that make forms available at the lowest possible cost.
- Including unique form number and revision date on each form for controlling purposes.

9.2.1 Classification of Forms

A printed form is generally classified by what it does in the system. There are three primary classifications

1. **Action:** This type of form requests the user to do something.

   Example: *purchase orders.*

2. **Memory:** This form is a record of historical data that remains in a file, is used for reference, and serves as control on key details.

   Example: *Inventory records, purchase records.*

3. **Report:** This form guides supervisors and other administrators in the activities. It provides data on a project or a job.

   Example: *profit and loss statements, sales analysis report.*

NOTES

9.2.2 Requirements of Form Design

Form design follows analysing forms. Since the purpose of a form is to communicate effectively through forms design, there are several major requirements.

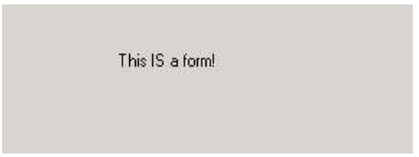
1. **Identification and Wording**: The form title must clearly identify its purpose. Columns and rows should be labelled to avoid confusion. The form should also be identified by firm name or code number to make it easy to reorder.
2. **Maximum Readability and Use:** The form must be easy to use and fill out. It should be legible, intelligible and uncomplicated. Ample writing space must be provided for inserting data.
3. **Physical Factors:** The forms composition, color, layout and paper stock should lend themselves to easy reading. Pages should be numbered when multipage reports are being generated for the user.
4. **Order of Data Items:** The data requested should reflect a logical sequence. Related data should be in adjacent positions. Data copied from source documents should be in the same sequence on both forms.
5. **Ease of Data Entry:** If used for data entry, the form should have field positions indicated under each column of data and should have some indication of where decimal points are.
6. **Size and Arrangement**: The form must be easily stored and filed. It should provide for signatures. Important items must be in a prominent action on the form.
7. **Use of Instructions**: The instructions that accompany a form should clearly show how it is used and handled.
8. **Efficiency Considerations:** The form must be cost effective. This means eliminating unnecessary data and facilitating reading lines across the form.
9. **Type of Report**: Forms design should also consider whether the content is executive summary, intermediate managerial information, or supporting data. The user requirements for each type determine the final form design.

9.2.3 Styles and Types of Form

A form has four main styles which are considered as template. These styles are used for interfacing the look and appearance of forms. Various values are used in programming languages for form styles, such as value 0 is used for None, 1 is used for Fixed Single, 2 is used for Sizable, 3 is used for Fixed Dialog and 4 is used for Fixed style. The styles of form are discussed below:

- **0 – None Style:** This form styles has no border, no title bar at the top and no maximize or minimize and close buttons. This style is usually used for

splash screens. The term 'splash screen' is one of those screens which are shown as an application is loading. Default is no icon in the taskbar.



NOTES

- **1 - Fixed Single:** This form style has a border, a title bar and a close button with optional minimize and maximize. If a form with its border style is set then it cannot be resized except using minimize and maximize buttons.



- **2 – Sizable:** This form style provides a form with a border, a blue box and the entire standard buttons. The dialog can be resized.



- **3 - Fixed Dialog:** This form style is a form with a border, a blue box and a close button. No minimize and maximize buttons are allowed. Default is no icon in the taskbar.
- **4 - Fixed:** This form style cannot be resized.

You can select any one of the abovementioned forms style and types to create a suitable form as per screen design. Every Web application that collects information from the user should have an interface which is understood by the users easily and interact with it. If an interface is user friendly then it is easy for the user to understand and handle. The information supplied by the user will be more accurate. The proposed form (Registration Form) is a frequently used form in which the form labels are highlighted with different background color. Instructions or rules are added at the top or bottom of every field as you see in the form screen below. Above form is designed with the help of two tables. The outer table creates border and padding whereas inner table refers to a container of the form. To design the form first it is split into two columns (panels) in which the first column contains form fields and the other column provides space for instruction or rules on filling the form. The design is very easy to implement with minimum Cascading Style Sheets (CSS) and Hyper Text Markup Language (HTML) tags. CSS is a style sheet language used to describe the presentation semantics, i.e., the look and formatting of a document written in a markup language. CSS is designed primarily to enable the separation of document content written in HTML or a similar markup

NOTES

language from document presentation including elements, such as the layout, colors and fonts. The CSS rules are applied style on input and selection controls in the form, changing the font, border and padding. Style sheets are simply text files saved as .css composed of lines of code that tell browsers how to display an HTML page. These sheets give the designer more control over the appearance of a Web page by allowing the styles for elements, such as fonts on the page.

Screen below displays the 'Registration Form' layout which contains various fields, such as 'First Name', 'Last Name', etc., filled by users. The information of regarding fields is also displayed on the form screen, for example Zip code must be a 5 digit number and 'Your email will be used as your login id'.



Following are the types of forms:

Flat Forms

A flat form is a single copy form prepared manually or by a machine and printed on a paper. For additional copies of the original carbon papers are inserted between copies. It is the easiest forms to design, print and reproduce. It uses less volume and is inexpensive.

Unit Set/Snapout Forms

Carbon Interleaved Unit Sets (CIUS) are papers with one time carbons interleaved which are set with unit sets for either handwritten or machine use. Carbons may be either blue or black standard grade medium intensity. Generally, blue carbons are best for handwritten forms while black carbons are best for machine use.

Continuous Strip/Fanfold Forms

These multiple unit forms joined in a continuous strip with perforations between each pair of forms. It is a less expensive method for large volume use.

No Carbon Required Paper

Carbonless Unit Sets (CUS) uses carbonless papers often referred to as No Carbon Required (NCR) paper. Carbonless papers use two chemical coatings (capsules), one on the face and the other on the back of a sheet of paper. When pressure is applied, i.e., handwriting, typing or machine impact the two capsules interact and create an image.

Principles of Form Design

A form is a printed paper used to receive the data entered by the user in a standardized way. Following are the guidelines that the systems analysts need to follow to design a good form:

NOTES

Keeping the Screen Simple

Following are the techniques used to keep the screen simple:

- Form should be designed in a proper sequence, i.e., from left to right and top to bottom.
- Information, such as heading, identification and access, instructions, body, signature and verification, totals and comments should be grouped logically.
- Give clear captions of different types of information. Caption may be on the left, above or below the line, box caption may be inside, above or below the box or space caption may be for a table.

Meeting the Intended Purpose

To design a form which meets the intended purpose, systems analysts may use different types of forms serving different purposes including recording, processing, storing and retrieving of information for different entities in the system.

Assuring Accurate Completion

To reduce error rates associated with data collection the forms should be designed to ensure complete accuracy. In other words, forms should be designed to make people do the right thing with the form.

Keeping Forms Attractive

To encourage people to complete forms, systems analysts should keep forms attractive. To make the screen more attractive, systems analysts may use different thickness of separation lines between subcategories, inverse video and blinking cursors, icons that are pictorial, on-screen representations symbolizing computer actions, different combinations of colours and different type fonts.

9.2.4 Forms Control

User Interface (UI) provides a way for the end users to communicate with a software system by making use of forms, reports, screens and error messages. While designing the interface design, the software developers identify certain procedures for performing each system activity and the relevant inputs for those activities. During the design process, end user involvement is must so that the appropriate user interface is designed which meets the user requirements. After completing user interface design, software developers record the reports, procedures, screens and dialogues that form a part of the system in a user manual. The manual may be printed or used as an online help file.

NOTES

User Interface Designing

The user interface design is one of the most important tasks of a systems analyst. The goal of user interface design is to help the end users and businesses to get the required information in and out of the system by addressing the following objectives:

- **Effectiveness:** Ensures that the end users can access their system in the way they want.
- **Efficiency:** Ensures that a good interface is used that increases the speed of data entry and reduces errors.
- **User Considerations:** Ensures the appropriate feedback to end users.
- **Productivity:** Ensures that a good interface is used that leads to an increase in productivity.

User Interface Types

The User Interface (UI) can be classified as natural language interface, Question and answer interface, menus, form-fill interface (input-output forms), command language interface and Graphical User Interface (GUI). The user interface has two main components called as presentation language and action language. Presentation language is the computer-to-human part of the transaction. Action language characterises the human-to-computer part of the transaction. Following are the various types of user interfaces:

- **Natural Language Interface:** Is ideal for inexperienced end users. It allows end users to interact in their natural language and does not require special skills.
- **Question and Answer Interface:** Is a user interface in which the computer displays a question to the user on screen and the user enters his/her answer. The computer then acts on that input information in a pre-programmed manner, typically by moving to the next question.
- **Menus Interface:** Is a menu interface that provides a list of available options to the user. In this interface, the number of options available to the user is limited and menus can be set up to use keyboard entry or mouse.
- **Form-Fill Interface:** Is the interface that consists of on-screen forms or Web based forms displaying fields containing data items. The main advantage of this form is that the printed version of this form provides excellent documentation. It puts the responsibility for accuracy onto the user and makes the form available for completion and submission on a 24 hour worldwide basis.
- **Command Language Interface:** Is the interface that allows the user to control the application with a series of commands. It manipulates the computer as a tool by allowing the user to control the communication between the computer and the user. This type of interface offers the user more flexibility and control.

- **Graphical User Interface:** Is the interface that allows direct manipulation of the graphical representation on the screen with the help of keyboard input or mouse. It allows the use of hyperlinks and icons in an effective manner.

NOTES

Dialogue Designing

Dialogue is the communication between the computer and a person. Following are the key points for designing good dialogue:

- Meaningful communication between the end user and the computer.
- Presenting clear information to the user.
- Interpreting correctly what user is entering.
- Minimizing user interference.
- Entering in code instead of the whole word.
- Prompting the user to enter only the unavailable data.
- Usage of default values should be maximised.
- Providing shortcut keys.
- Maintaining consistency.
- Locating titles, information and messages in the same places on all screens.
- Exiting each program by the same key or menu option.
- Cancelling a transaction in a consistent way.
- Obtaining help in a standardised way.
- Standardising the use of icons for similar operations while using a GUI.
- Standardising the colour used for all screens.

Feedback

All systems require feedback in order to monitor and improve its performance. Feedback can tell the user more details, for example if problems occur, what are the possible causes and solutions to the problems. Feedback is needed to tell the user that:

- The input has been accepted by the system.
- The input is in the correct form.
- There will be a delay in the processing.
- The request has been completed.
- The system is unable to complete the request.
- More detailed feedback is available.

Integrity Controls

While considering the user interface, developers must also consider the procedures that help to protect both the existence and the quality of the data in the system.

NOTES

Integrity controls are the techniques used to insure the quality of the data entered into the system. These controls help establish the types of transactions the end user is allowed to perform.

They also ensure that the data entered is accurate and of good quality by bringing data entry errors to the notice of the end users. Integrity controls ensure the correct processing of transactions. These controls help in maintaining the integrity of input and output.

Access controls help in determining who has access to an information system. Access controls determine who has access to an information system and which modules those are accessible to various end users. Access restrictions may be applicable to entire applications or specific data views within the system. Systems administrators control the access list to applications and therefore establish the authorized or registered end users and the unauthorized end users.

Techniques of access control implementation include login scripts, user ID/ passwords as well as physical measures, for example user specific ID cards. User ID is important because numerous systems facilitate access to all connected applications from a single user login. Systems that possess Internet access are particularly prone to attacks from external or unauthorized end users. Therefore, additional control elements, such as firewalls and proxy server should be employed to restrict entry—into internal networks from external sources—to only specific applications or authorized end users. Apart from imposing checks on accessing particular applications, many organizations restrict physical access to computer system facilities as well.

Controlling Forms

Controlling business forms is an important task and usually organisations often have a form specialist that does this job. Basic duties for controlling forms are as follows:

- Making sure that each form fulfils its specific purpose and that specific purposes is integral to organizational functioning
- Preventing duplication of the information that is collected and of the forms that collect it
- Designing effective forms
- Deciding on how to get forms reproduced in the most economical way
- Establishing stock control and inventory procedures that make forms available at the lowest possible cost
- Including unique form number and revision date on each form for controlling purposes.

Check Your Progress

1. Write the methods for generation of the interface summary table for a system.
2. Explain the objective of input form design.
3. State about the guidelines of form design.
4. Write down the methods for navigation from one screen to another.
5. What are the features of '0 –None Style' form?
6. Explain the term splash screen.
7. Write the type of primary classifications.
8. What is dialogue?
9. Define the term integrity control.

NOTES

9.3 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. Generation of the interface summary table for system uses the following methods:
   - Analysing the human computer limit in the Data Flow Diagram (DFD).
   - Filing the interface and analysing the design for the purpose of defining the data elements that have to be added in every form/document, input screen and output report.
2. The basic objectives of the input form design must be achieved by a good designed input form and visual display screen which can only take place by using the essential design principles, information about which type of data is required to be used in the system and recognizing the way by which the users react to various different components of forms and screens.
3. The guidelines of form design are as follows:
   - They ease in filling out the forms.
   - They provide certainty of the forms meeting the reason of their being.
   - They make the forms eye catching.
   - They design the formats for confirming precise completion.
4. To make it easy to navigate from one screen to another, systems analysts may use one of the following methods:
   - Scrolling the screen back and forth.
   - Calling up another screen for more detail.
   - Using on-screen dialogue through the prompts.

NOTES

5. 0 –None Style form styles has no border, no title bar at the top and no maximize or minimize and close buttons.
6. The term 'splash screen' is one of those screens which are shown as an application is loading. Default is no icon in the taskbar.
7. There are three primary classifications
   1. Action: This type of form requests the user to do something.

      Example: purchase orders.
   2. Memory: This form is a record of historical data that remains in a file, is used for reference, and serves as control on key details.

      Example: Inventory records, purchase records.
   3. Report: This form guides supervisors and other administrators in the activities. It provides data on a project or a job.

      Example: profit and loss statements, sales analysis report.
8. Dialogue is the communication between the computer and a person.
9. Integrity controls are the techniques used to insure the quality of the data entered into the system. These controls help establish the types of transactions the end user is allowed to perform.

9.4 SUMMARY

- Every rule related to the logical and physical data structure that has been obligated by the DataBase Management System (DBMS) must be followed by the entire database description managing the database.
- The user interface usually comprises of various documents, reports and different screens of a system and is the sole component of the system that is viewed by the user.
- The excellence of the system output is concluded by the quality of the system input. After understanding such significant relations, it becomes necessary that the input forms and screens as well as output reports should be designed.
- The basic objectives of the input form design must be achieved by a good designed input form and visual display screen which can only take place by using the essential design principles, information about which type of data is required to be used in the system and recognizing the way by which the users react to various different components of forms and screens.
- The term 'effectiveness' basically refers that the input forms and screens play a particular role in the Management Information System (MIS) whereas the term 'accuracy' highlights the design that guarantees a suitable conclusion.
- The objectives of the forms designing are that extraction and capturing the needed information through the members of the organization will be frequently added to the computer with the help of form designing.

NOTES

- For the purpose of entering the data in the input forms or reports of the computer and updating the data, the data entry is made use of.
- The initial guideline regarding a perfect screen design is simplifying the screen display. Only the required material for the specific action carried out should be shown by the display screen.
- Next guideline is making the browsing among various screens easier. This can be done by making the users develop a feeling of physically moving to a novel screen which can be done through the use of the scroller.
- For the purpose of providing feedback on update actions and usage of conformation message.
- Controlling business forms is an important task and usually organizations often have a form specialist that does this job.
- Making sure that each form fulfils its specific purpose and that specific purposes is integral to organizational functioning.
- Establishing stock control and inventory procedures that make forms available at the lowest possible cost.
- A form has four main styles which are considered as template.
- The term 'splash screen' is one of those screens which are shown as an application is loading. Default is no icon in the taskbar.
- The information supplied by the user will be more accurate. The proposed form (Registration Form) is a frequently used form in which the form labels are highlighted with different background color.
- The outer table creates border and padding whereas inner table refers to a container of the form. To design the form first it is split into two columns (panels) in which the first column contains form fields and the other column provides space for instruction or rules on filling the form.
- Style sheets are simply text files saved as .css composed of lines of code that tell browsers how to display an HTML page. These sheets give the designer more control over the appearance of a Web page by allowing the styles for elements, such as fonts on the page.
- A flat form is a single copy form prepared manually or by a machine and printed on a paper. For additional copies of the original carbon papers are inserted between copies.
- Carbon Interleaved Unit Sets (CIUS) are papers with one time carbons interleaved which are set with unit sets for either handwritten or machine use.
- Carbons may be either blue or black standard grade medium intensity. Generally, blue carbons are best for handwritten forms while black carbons are best for machine use.

NOTES

- Carbonless Unit Sets (CUS) uses carbonless papers often referred to as No Carbon Required (NCR) paper.
- A form is a printed paper used to receive the data entered by the user in a standardized way.
- To design a form which meets the intended purpose, systems analysts may use different types of forms serving different purposes including recording, processing, storing and retrieving of information for different entities in the system.
- To reduce error rates associated with data collection the forms should be designed to ensure complete accuracy. In other words, forms should be designed to make people do the right thing with the form.
- To encourage people to complete forms, systems analysts should keep forms attractive. To make the screen more attractive, systems analysts may use different thickness of separation lines between subcategories, inverse video and blinking cursors, icons that are pictorial, on-screen representations symbolizing computer actions, different combinations of colours and different type fonts.
- User Interface (UI) provides a way for the end users to communicate with a software system by making use of forms, reports, screens and error messages.
- During the design process, end user involvement is must so that the appropriate user interface is designed which meets the user requirements.
- The user interface design is one of the most important tasks of a systems analyst.
- Ensures that a good interface is used that leads to an increase in productivity.
- The user interface has two main components called as presentation language and action language.
- The computer then acts on that input information in a pre-programmed manner, typically by moving to the next question.
- Is a menu interface that provides a list of available options to the user. In this interface, the number of options available to the user is limited and menus can be set up to use keyboard entry or mouse.
- The main advantage of this form is that the printed version of this form provides excellent documentation. It puts the responsibility for accuracy onto the user and makes the form available for completion and submission on a 24 hour worldwide basis.
- Is the interface that allows direct manipulation of the graphical representation on the screen with the help of keyboard input or mouse. It allows the use of hyperlinks and icons in an effective manner.

- Dialogue is the communication between the computer and a person.
- All systems require feedback in order to monitor and improve its performance. Feedback can tell the user more details.
- Integrity controls are the techniques used to insure the quality of the data entered into the system. These controls help establish the types of transactions the end user is allowed to perform.
- Access controls help in determining who has access to an information system. Access controls determine who has access to an information system and which modules those are accessible to various end users.
- Techniques of access control implementation include login scripts, user ID/ passwords as well as physical measures, for example user specific ID cards.
- Controlling business forms is an important task and usually organisations often have a form specialist that does this job.
- Preventing duplication of the information that is collected and of the forms that collect it

9.5 KEY WORDS

- **User interface:** This interface comprises various documents, reports and different screens of a system and is the sole component of the system that is viewed by the user.
- **0 – None Style**: 0 – None Style form has no border, no title bar at the top, and no maximize, minimize and close buttons.
- **Flat form:** It is a single copy form prepared manually or by a machine and printed on a paper.
- **Carbon Interleaved Unit Sets (CIUS) :**Carbon Interleaved Unit Sets (CIUS) are papers with one time carbons interleaved which are set with unit sets for either handwritten or machine use.
- **Carbonless Unit Sets (CUS):** Carbonless Unit Sets (CUS) uses carbonless papers often referred to as No Carbon Required (NCR) paper.

9.6 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. Define the term form.
2. Write down the guidelines for screen design.
3. Give the definition of flat form.

NOTES

4. Define the term fanfold form.
5. State about the main components of user interface.
6. List the different types of user interface.
7. Explain the term dialogue design.

Long-Answer Questions

1. Describe the basics of form design giving appropriate examples.
2. Briefly explain the steps required for designing an interface.
3. Discuss about the different styles of forms.
4. Describe the types of forms in detail with the help of examples.
5. Briefly explain the principles of form design.
6. Explain briefly the layout considerations and forms control giving examples.
7. Elaborate the key points of designing good dialogue design.

9.7 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

BLOCK - IV
FILE AND DATABASE DESIGN

NOTES

UNIT 10 FILE CONCEPTS

Structure

10.0 Introduction
10.1 Objectives
10.2 Introduction to File Organization
  10.2.1 File Organization
10.3 Types of Files
10.4 Database Design
10.5 Answers to Check Your Progress Questions
10.6 Summary
10.7 Key Words
10.8 Self Assessment Questions and Exercises
10.9 Further Readings

10.0 INTRODUCTION

A file is organized to ensure that records are available for processing. It should be designed in the line with the activity and volatility of the information and the nature of the storage media and devices. There are four methods of organizing files: sequential organization simply means storing and sorting in physical, contiguous blocks within files on tape or disk according to key. Indexed sequential organization stores records sequentially but uses an index to locate record. Records are related through chaining using pointers. Inverted list organization uses an index for each key type. Records are not necessarily in a particular sequences. Direct access organization has records placed randomly throughout the file. Records are updated directly and independently of the other records.

A DataBase is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make information access easy, quick, inexpensive and flexible for the user. Data base design minimizes the artificially embedded in using separate files. The primary objectives are fast response time to inquiries, more information at low cost, control of redundancy, clarity and ease of use, data and program independence, accuracy and integrity of the system, fast recovery, privacy and security of information, and availability of powerful end user languages.

In this unit, you will study about the types of files, methods of file organization, sequential, direct and indexed, database design, and database concept.

NOTES

10.1 OBJECTIVES

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

- Understand the methods of file organization
- Explain the types of files
- Define the sequential and direct file organization
- Elaborate on the concept of DataBase

10.2 INTRODUCTION TO FILE ORGANIZATION

The modular design of the software consists of a number of files. The file creation follows a predefined file structure. Figure 10.1 shows the various elements in hierarchy of a file structure.



Fig. 10.1 *Hierarchical File Structure*

Figure 10.1 shows the following elements of file structure:

- **Byte:** Byte is a set of eight bits that represent a character. It is the smallest addressable unit in a computer system.
- **Data Item:** Combination of more than one byte is called a data item or element which defines attributes of object. The data item is also called field.
- **Record:** The data items related to the object are combined into a record.
- **File:** It consists of a collection of related records. The size of a file depends upon the size of memory or the storage space of computer system. File has two types of characteristics which help determine the file structure. Following are characteristics of file structure:
  - **File Activity:** It specifies the percentage values of the records which are processed in single run.
  - **File Volatility:** This specifies that the properties of record change while processing.

- **Database:** It is the upper layer in hierarchy of file structure. It is a set of interrelated files for real time processing. It contains the data, which is required for problem solving and can be used by several users.

NOTES

10.2.1 File Organization

File organization is the methodology which is applied to structured computer files. Files contain computer records which can be documents or information. It refers primarily to the logical arrangement of data which can itself be organized in a system of records with correlation between the fields or columns in a file system. There are certain basic types of computer file which can include files stored as blocks of data and streams of data where the information streams out of the file while it is being read until the end of the file is encountered. Therefore, file organization refers to the way records are physically arranged on a storage device. The two main types of file organization are record type and record access. These types are described below:

- Record type refers to whether records in a file are all the same length are of varying length or use other conventions to define where one record ends and another begins.
- Record access refers to the method used to read records from or write records to a file regardless of its organization. The way a file is organized does not necessarily imply the way in which the records within that file will be accessed.

Following are the two prime types of file organizations:

Sequential File Organization

A sequentially organized file consists of records arranged in the sequence in which they are written to the file, i.e., the first record written is the first record in the file, the second record written is the second record in the file and so on. As a result, records can be added only at the end of the file. Attempting to add records at some place other than the end of the file will result in truncating the file at the end of the record. Sequential files are usually read sequentially, starting with the first record in the file. Sequential files with a fixed length record type that are stored on disk can also be accessed by relative record number (direct access). A sequential file contains records organized by the order in which they were entered. The order of the records is fixed. Records in sequential files can be read or written only sequentially.

Relative File Organization

Within a relative file are numbered positions, called cells. These cells are of fixed equal length and are consecutively numbered from 1 to n, where 1 is the first cell and n is the last available cell in the file. Each cell either contains a single record or is empty. Records in a relative file are accessed according to cell number. A cell number is a record's relative record number where its location relative to the beginning of the file. By specifying relative record numbers, you can directly retrieve, add or delete records regardless of their locations. Within the cells, you can store records of varying length as long as their size does not exceed the cell size.

NOTES

It is necessary to organize the file properly in order to ensure that the records are available for processing. The two main characteristics that determine how the file is organized are activity and volatility. File activity determines the total number of records that can be processed in a single run. If the total number of records that can be accessed at a specific time is small, then the file is organized on a disk or on a tape. File volatility focus on the properties of the records that changes. There are four methods to access a file:

- **Sequential Access:** In this file organization method, records are stored in contiguous blocks of memory. Tape and disk are the examples of physical devices of sequential organization method. In a sequential organization, adding a record in a file takes place at the end of a file. It is not possible to add a record at the middle of the file. All records in a sequential file organization are scanned to access a particular record.
- **Indexed-Sequential Access:** Indexed-sequential file organization method also stores records in the contiguous blocks of memory but it uses indexes to access a particular record. The disk storage is divided into three areas: primary area, overflow area and index area. Primary area is the main area of the disk where the records are stored by key or ID numbers. The overflow area holds those records that cannot be placed in the logical sequence in the primary area. The index area is the area on a disk that stores the key of records and the memory addresses of their locations. Indexed-sequential organization takes less time to locate records as compared to the sequential file organization. The disadvantage of this organization method is the extra space that is required to store the indexes.
- **Chaining Access:** In chaining access file organization method, pointers are used with link records to each other. The pointers give the address of the next part of the same class.
- **Inverted List Access:** Inverted list organization method is similar to the indexed sequential organization method but in indexed sequential organization, multiple indexes are stored of a given key whereas in inverted list organization a single index is used for a single key. The records stored in inverted list organization are not necessarily stored in a particular sequence. They are stored anywhere in the data storage area.

Advantages:

1. Simple to design.
2. Easy to program.
3. Variable length and blocked records available.
4. Best use of storage space.

Disadvantages:

1. Records cannot be added at the middle of the file.

Indexed- Sequential Organization

NOTES

Like sequential organization, keyed sequential organization stores data in physically contiguous blocks. The difference is in the use of indexes to locate records. To understand this method, we need to distinguish among three areas in disk storage: prime area, overflow area and index area. The prime area contains file records stored by key or ID numbers. All records are initially stored in the prime area. The overflow area contains records added to the file that cannot be placed in logical sequence in the prime area. The index area is more like a data dictionary. It contains keys of records and their locations on the disk. A pointer associated with each key is an address that tells the system where to find a record.

In an airline reservation file, the index area might contain pointers to the Chicago and Delhi flights. The Chicago flight points to the Chicago flight information stored in the prime area. The Delhi flight points to the Delhi flight information in the prime area.

Lack of space to store the Brisbane flight in sequential order make it necessary to load it in the overflow area. The overflow pointer places it logically in sequential order in the prime area. The same arrangement applies to the other flights.

Indexed-sequential organization reduces the magnitude of the sequential search and provides quick access for sequential and direct processing. The primary drawback is the extra storage space required for the index. It also takes longer to search the index for data access or retrieval.

Chaining

File organization requires that relationships be established among data items. It must show how characters form fields, fields form files, and files relate to one another.

Establishing relationships is done through chaining or the use of pointers. The example on airline reservation file showed how pointers, link one record to another. Part number retrieves a record. A better way is to chain the records by linking a pointer to each. The pointer gives the address of the next part type of the same class. The search method applies similarly to other parts in the file.

Direct - Access Organization

In direct – access file organization, records are placed randomly throughout the file. Records need not be in sequence because they are updated directly and rewritten back in the same location. New records are added at the end of the file or inserted in specific locations based on software commands.

NOTES

Records are accessed by addresses that specify their disk locations. An address is required for location a record, for linking records, or for establishing relationships.

Addresses are of two types: absolute and relative. An absolute address represents the physical location of the record. It is usually stated in the format of sector/track/record number. For example, 3/14/6 means go to sector 3, track 14 of that sector, and the sixth record of the track. One problem with absolute addresses is that they become invalid when the file that contains the records is relocated on the disk. One way around this is to use pointers for the updated records.

A relative address gives a record location relative to the beginning of the file. There must be fixed-length records for reference. Another way of locating a record is by the number of bytes it is from the beginning of the file (refer figure 10.2). Unlike relative addressing, if the file is move, pointers need not be updated, because the relative location of the record remains the same regardless of the file location.

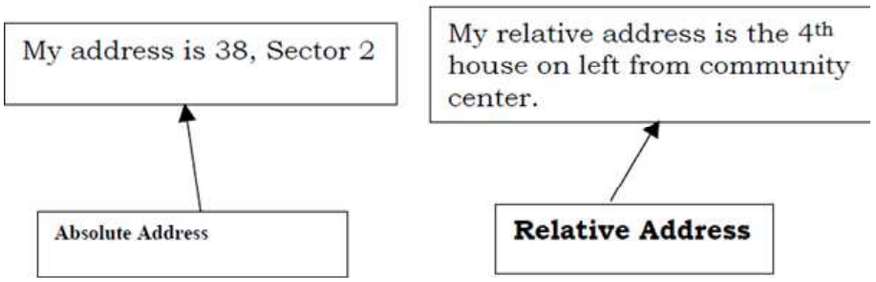


Fig. 10.2 *Absolute and Relative Addressing an Example*

10.3 TYPES OF FILES

Following are the types of files used in an organization system:

- **Master File :** It contains the current information for a system. For example, customer file, student file, telephone directory.
- **Table File :** It is a type of master file that changes infrequently and stored in a tabular format. For example, storing Zip code.
- **Transaction File :** It contains the day-to-day information generated from business activities. It is used to update or process the master file. For example, addresses of the employees.
- **Temporary File :** It is created and used whenever needed by a system.
- **Mirror File :** They are the exact duplicates of other files. Help minimize the risk of downtime in cases when the original becomes unusable. They must be modified each time the original file is changed.

- **Log Files :** Log filer contain copies of master and transaction records in order to chronicle any changes that are made to the master file. It facilitates auditing and provides mechanism for recovery in case of system failure.
- **Archive files :** Backup files that contain historical versions of other files.

Check Your Progress

1. What is byte?
2. Give the definition of file organization.
3. Explain the concept of cell in relative file organization.
4. Why are pointers used in chaining process?
5. State about the log file.

10.4 DATABASE DESIGN

Database is a collection of data stored for a specific purpose. Database Management System (DBMS) is a software system that allows you to define, construct and manipulate databases that are used for various purposes, such as for storing the customer and financial information of an organization. DBMS provides significant features that make a database system efficient and reliable. It also provides multiple User Interfaces (UI) that help a user in interacting with the system. The interfaces provided by DBMS include the query language for users and the programming language interface for application programmers. DBMS prevents the data redundancy problems that occur in other management systems. DBMS provides security to the system by restricting the unauthorised users to access the data of an organization. The backup and recovery feature of DBMS allows you to recover data that is lost due to hardware or software failure. In addition, DBMS users can also perform a variety of operations, such as insert, delete, retrieve and update on the database.

Client-server architecture is used to retrieve information from the databases. Clients are the machines that request for the services and server is the machine that processes the requests send by the clients. Server processes the information based on the information that is stored either in flat file database or relational database. Utilize the information obtained from the SQL Server database, you need to install SQL Server clients. A SQL Server can be defined as database a collection of related data items that can be stored in database objects, such as tables. Databases provide an easy way to access, manage and update the data in a database. There are two types of databases:

- Flat file database
- Relational database

NOTES

Flat File Databases

Flat file databases are simple text files that store data in a single table. Consider an example of an enterprise that needs to store the sales information, such as customer details (customer name, customer address and customer mobile number), product details (product id, product features and product price) and billing details (purchase date, discount, sale price). All the information will be stored in a single table if flat file system is used in the enterprise.

Flat file database is less flexible in terms of maintaining records. Consider that an enterprise needs to maintain a database for every unit of product sold. The database is maintained using flat file databases. The information stored in database includes customer details (customer id, customer name and customer address), product details (product id, product name and product features). Each time a product is sold, the information regarding customer details (provided the customer is same) and product details (product name and product features) needs to be repeated in the database. This additional information will bring redundancy in the database making it less flexible.

Relational Databases

Relational databases are complex databases that store information in multiple tables. In relational database, every table that stores data is linked to another table using a common column. Various software programs that store data in relational database format are Oracle and Structure Query Language (SQL). Relational databases are flexible as compared to flat file databases in terms of accessing, managing and upgrading the data.

Consider the above example; in this case, you need to maintain the records for each unit of product sold. In relational databases, the information will be stored in separate tables depending on the similarity. All the customer information will be stored in the Customer_details table, all the product information will be stored in the Product_details table and all the billing information will be stored in the Billing_details table.

Each time a product unit is sold, the records need to be updated in the billing information and not in Customer_detail and Product_details. In this way, data stored in relational databases avoids redundancy.

Data Requirements

For a manager, it is necessary to gather information about partially-completed activities and to forecast the completion date of project. Without proper data it would be difficult to forecast accurately. The project manager can design several forms to collect data to be filled by the team leader and team members. These forms include a weekly timesheet, a progress review form and an activity assessment sheet. After receiving the assessment forms for all activities, the project manager uses these to evaluate the status of project.

Objectives of Database Design

NOTES

Database design refers to a system that stores and retrieves data systematically from the database. However, first you must know the description of a few basic terms of DBMS so that you can design a suitable database as per the requirement of an organization. These are as follows:

- **Data:** Data means known facts that can be recorded and used to produce useful information, e.g., names, telephone numbers and addresses of the employees of an organization.
- **Database:** A database is a collection of interrelated data. For example, you may create a database by recording the names, telephone numbers and addresses of employees in an indexed addressed book or on a diskette, using a personal computer and software, such as Microsoft Access or Microsoft Excel. This recorded information is a collection of related data with an implicit meaning and hence is called a database.
- **Defining a Database:** This involves specifying the data types, structures and constants for data to be stored in the database.
- **Constructing a Database:** This is a process of storing real data on some storage medium with the help of DBMS.
- **Manipulating a Database:** This involves performing functions, such as querying the database to retrieve specific data, updating the database to reflect changes made by the user and generating reports from data.
- **Database System:** The database and DBMS software together form a database system.

The objectives of database design vary from implementation to implementation. Some of the important factors like efficiency, integrity, privacy, security, implementability and flexibility have to be considered in the design of database.

- **Efficiency:** Efficiency is generally considered to be the most important factor. Given a piece of hardware on which the database will run and piece of software Database Management System (DBMS) to run it and the design should make full and efficient use of the facilities provided. If the database is made online, then the users should interact with the database without any time delay.
- **Integrity:** The term integrity means that the database should be as accurate as possible. The problem or preserving the integrity of data in a database can be viewed at a number of levels. At a low level, it concerns ensuring that the data are not corrupted by hardware or software errors. At a higher level, the problem of preserving database integrity concerns maintaining an accurate representation of the real world.
- **Privacy:** The database should not allow unauthorized access to files. This very important in the case of financial data, for example the bank balance of one customer should not be disclosed to other customers.

NOTES

- **Security:** The database, once loaded, should be safe from physical corruption whether from hardware or software failure or from unauthorized access. This feature is a general requirement of most databases.
- **Implementation:** The conceptual model should be simple and effective so that mapping from conceptual model to logical model is easy. Moreover while designing the database; care has to be taken such that application programs should interact effectively with the database.
- **Flexibility:** The database should be implemented in a rigid way that assumes the business will remain constant forever. Changes will occur and the database must be capable of responding readily to such change.

Other than the factors which were mentioned above, the design of the database should ensure that data redundancy is not there.

To understand the basics of DBMS design, you must know the terms and definitions that are used in DBMS technology. DBMS is a software program that may run on a user machine or a server computer. DBMS accepts queries from users and responds to these queries. The DBMS design has the following features:

- **Structured Data:** DBMS enables you to structure the data as tables, records or objects.
- **Query Language:** DBMS provides a query language, such as SQL to process the user requests.
- **Multi-User Access:** DBMS allows users to access data stored in a database. At the same time, it provides security features that restrict some users from viewing or manipulating data.
- **Data Dictionary:** DBMS provides a data dictionary that contains the structure of a database.
- **Data Abstraction:** It allows a DBA to logically separate data from the programs that use the data. There are three levels of abstraction in a database: external, conceptual and internal. The external level represents the user view of the database; the conceptual level allows you to map the internal and external levels; and the external level represents the operating system and DBMS level.

A well designed database takes time and effort to build and refine. While the time of designing a database you have to make decisions regarding how best to take some system in the real world and model it into a selected database. This consists of deciding which tables to create, what columns they will contain as well as the relationships between the tables. The benefits of a database designed according to the relational model are as follows:

- Efficient data entry, updates and deletion functions can be used.
- Well organized and prompt data retrieval, summarization and reporting function can be generated.

Suitable database design is based on schema. One major advantage to the database schema is easy to make. Tables in the relational model are used to represent

NOTES

'things in the real world'. Each table should represent only one thing. These things refer to entities and can be real world objects or events, for example a real world object might be a customer, an inventory item or an invoice. Examples of events include patient visits, orders and telephone calls, etc. Tables are made up of rows and columns. The relational model is used in designing the database which dictates that each row in a table be unique. If you allow duplicate rows in a table then there is no way to uniquely address a given row via programming. Uniqueness property provides guarantee for a table by designating a primary key, i.e., a column that contains unique values for a table. Each table can have only one primary key, even though several columns or combination of columns may contain unique values. All columns in a table with unique values are referred to as candidate keys from which the primary key must be drawn. All other candidate key columns are referred to as alternate keys. Keys can be simple or composite. A simple key is a key made up of one column. A composite key is made up of two or more columns. Following steps are required to design a good database:

- Start to rough out on paper the data entry forms.
- If this system was not computerized, take the existing paper based system and draw a rough table design based on these forms. It is likely that these forms will be non-normalized.
- If the database will be converted from an existing computerized system, use its tables as a starting point. Use the existing schema, table by table and the existing data entry forms for the designing process.
- Start the normalization process. First, identify candidate keys for every table and using the candidates and then choose the primary key. Select a primary key that is minimal, stable, simple and familiar. Every table must have a primary key.
- Draw relationships between the tables, noting if they are one-to-one or one-to-many. If many-to-many relationships occur then linking tables are created.
- Determine whether the tables are in First Normal Form (1NF). Are all fields atomic? Use decomposition, if 1NF occurs.
- Determine whether the tables are in Second Normal Form (2NE). Does each table describe a single entity? Are all non-key columns fully dependent on the primary key? Does the primary key imply all of the other columns in each table? Decompose to meet 2NF.
- Determine if the tables are in Third Normal Form (3NF). Are there any computed columns? Are there any mutually dependent non-key columns? Remove computed columns.
- Using the normalized tables refine the relationships between the tables and create the tables using database software and add sample data to the tables.

NOTES

- Create prototype queries, forms and reports. While creating these objects design deficiencies refine the design as needed.
- Create the final forms, reports and queries. Develop the application and refine the design as necessary.
- Deliver the final database design system.

A single database can contain hundreds of tables and each playing its own unique role in the database schema. Database architecture and schema begins with a thorough understanding of tables. The tables are comprised of table rows and columns. Table columns are responsible for storing many different types of data, such as numbers, texts, dates and even files. Table 10.1 summarizes the details of Emp_Official_Info table**.**

Table 10.1 *Emp_Official_Info*

Emp_Code	E_Name	CL (in Days)	JD	Department	Designation	Basic_Salary
12	Richa Gulati	12	2007-3-5	HR	Business Exec.	40000
5	Gul Awasthi	8	2005-1-12	HR	Office Asstt.	23000
2	Neha Verma	19	2000-5-4	HR	Senior Manager	60000
3	Natasha Vij	24	1997-11-11	IT	Software Engineer	30000
7	Nisha Sawant	5	2006-10-10	Marketing	Marketing Exec.	17000

To define the functional dependency keys are clarified. Let us take an example to define the functional dependency in database designing. Following properties are taken if Column A is the primary key for table T if:

- **Property 1:** All columns in T are functionally dependent on A.
- **Property 2:** No sub collections of the columns in table T also have Property 1.

If all the fields in a defined database are dependent on one and only one field then that field is the key.

Check Your Progress

6. What does the backup and recovery feature of DataBase Management System (DBMS) allow?
7. Define the significance of relational databases.
8. Elaborate on the term integrity in database design.
9. Name the three levels of data abstraction.
10. Explain the term database table.

10.5 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. Byte is a set of eight bits that represent a character. It is the smallest addressable unit in a computer system.

NOTES

2. File organization is the methodology which is applied to structured computer files. Files contain computer records which can be documents or information. It refers primarily to the logical arrangement of data which can itself be organized in a system of records with correlation between the fields or columns in a file system.
3. Within a relative file are numbered positions, called cells. These cells are of fixed equal length and are consecutively numbered from 1 to n, where 1 is the first cell and n is the last available cell in the file. Each cell either contains a single record or is empty. Records in a relative file are accessed according to cell number. A cell number is a record's relative record number where its location relative to the beginning of the file.
4. In chaining access file organization method, pointers are used with link records to each other
5. Log filer contain copies of master and transaction records in order to chronicle any changes that are made to the master file. It facilitates auditing and provides mechanism for recovery in case of system failure.
6. The backup and recovery feature of DBMS allows you to recover data that is lost due to hardware or software failure. In addition, DBMS users can also perform a variety of operations, such as insert, delete, retrieve and update on the database.
7. Relational databases are complex databases that store information in multiple tables. In relational database, every table that stores data is linked to another table using a common column. Various software programs that store data in relational database format are Oracle and Structure Query Language (SQL). Relational databases are flexible as compared to flat file databases in terms of accessing, managing and upgrading the data.
8. The term integrity means that the database should be as accurate as possible. The problem or preserving the integrity of data in a database can be viewed at a number of levels. At a low level, it concerns ensuring that the data are not corrupted by hardware or software errors. At a higher level, the problem of preserving database integrity concerns maintaining an accurate representation of the real world.
9. There are three levels of abstraction in a database: external, conceptual and internal. The external level represents the user view of the database; the conceptual level allows you to map the internal and external levels; and the external level represents the operating system and DBMS level.
10. The tables are comprised of table rows and columns. Table columns are responsible for storing many different types of data, such as numbers, texts, dates and even files.

10.6 SUMMARY

NOTES

- The modular design of the software consists of a number of files. The file creation follows a predefined file structure.
- Combination of more than one byte is called a data item or element which defines attributes of object. The data item is also called field.
- File organization is the methodology which is applied to structured computer files. Files contain computer records which can be documents or information. It refers primarily to the logical arrangement of data which can itself be organized in a system of records with correlation between the fields or columns in a file system.
- File organization refers to the way records are physically arranged on a storage device. The two main types of file organization are record type and record access.
- Record type refers to whether records in a file are all the same length are of varying length or use other conventions to define where one record ends and another begins
- Record access refers to the method used to read records from or write records to a file regardless of its organization.
- A sequentially organized file consists of records arranged in the sequence in which they are written to the file, i.e., the first record written is the first record in the file, the second record written is the second record in the file and so on.
- A sequential file contains records organized by the order in which they were entered. The order of the records is fixed. Records in sequential files can be read or written only sequentially.
- Within a relative file are numbered positions, called cells. These cells are of fixed equal length and are consecutively numbered from 1 to n, where 1 is the first cell and n is the last available cell in the file. Each cell either contains a single record or is empty. Records in a relative file are accessed according to cell number. A cell number is a record's relative record number where its location relative to the beginning of the file.
- In chaining access file organization method, pointers are used with link records to each other.
- All records are initially stored in the prime area.
- The overflow area contains records added to the file that cannot be placed in logical sequence in the prime area. The index area is more like a data dictionary.
- File organization requires that relationships be established among data items. It must show how characters form fields, fields form files, and files relate to one another.

NOTES

- Establishing relationships is done through chaining or the use of pointers.
- A better way is to chain the records by linking a pointer to each. The pointer gives the address of the next part type of the same class.
- The search method applies similarly to other parts in the file.
- In direct – access file organization, records are placed randomly throughout the file. Records need not be in sequence because they are updated directly and rewritten back in the same location.
- Addresses are of two types: absolute and relative.
- An absolute address represents the physical location of the record. It is usually stated in the format of sector/track/record number.
- A relative address gives a record location relative to the beginning of the file. There must be fixed-length records for reference.
- Database is a collection of data stored for a specific purpose.
- DataBase Management System (DBMS) is a software system that allows you to define, construct and manipulate databases that are used for various purposes, such as for storing the customer and financial information of an organization.
- DBMS provides significant features that make a database system efficient and reliable.
- The backup and recovery feature of DBMS allows you to recover data that is lost due to hardware or software failure. In addition, DBMS users can also perform a variety of operations, such as insert, delete, retrieve and update on the database.
- Client-server architecture is used to retrieve information from the databases. Clients are the machines that request for the services and server is the machine that processes the requests send by the clients.
- Server processes the information based on the information that is stored either in flat file database or relational database.
- Flat file databases are simple text files that store data in a single table.
- Flat file database is less flexible in terms of maintaining records. Consider that an enterprise needs to maintain a database for every unit of product sold.
- Relational databases are complex databases that store information in multiple tables. In relational database, every table that stores data is linked to another table using a common column. Various software programs that store data in relational database format are Oracle and Structure Query Language (SQL). Relational databases are flexible as compared to flat file databases in terms of accessing, managing and upgrading the data.
- Database design refers to a system that stores and retrieves data systematically from the database.

NOTES

- Data means known facts that can be recorded and used to produce useful information, e.g., names, telephone numbers and addresses of the employees of an organization.
- A database is a collection of interrelated data. For example, you may create a database by recording the names, telephone numbers and addresses of employees in an indexed addressed book or on a diskette, using a personal computer and software, such as Microsoft Access or Microsoft Excel.
- The database and DBMS software together form a database system.
- The term integrity means that the database should be as accurate as possible. The problem or preserving the integrity of data in a database can be viewed at a number of levels. At a low level, it concerns ensuring that the data are not corrupted by hardware or software errors. At a higher level, the problem of preserving database integrity concerns maintaining an accurate representation of the real world.
- The database should not allow unauthorized access to files. This very important in the case of financial data, for example the bank balance of one customer should not be disclosed to other customers.
- The database, once loaded, should be safe from physical corruption whether from hardware or software failure or from unauthorized access. This feature is a general requirement of most databases.
- The database should be implemented in a rigid way that assumes the business will remain constant forever. Changes will occur and the database must be capable of responding readily to such change.
- To understand the basics of DBMS design, you must know the terms and definitions that are used in DBMS technology.
- DBMS enables you to structure the data as tables, records or objects.
- DBMS provides a query language, such as SQL to process the user requests.
- DBMS allows users to access data stored in a database. At the same time, it provides security features that restrict some users from viewing or manipulating data.
- There are three levels of abstraction in a database: external, conceptual and internal. The external level represents the user view of the database; the conceptual level allows you to map the internal and external levels; and the external level represents the operating system and DBMS level.
- Suitable database design is based on schema. One major advantage to the database schema is easy to make.
- Tables in the relational model are used to represent 'things in the real world'.
- Each table can have only one primary key, even though several columns or combination of columns may contain unique values. All columns in a table with unique values are referred to as candidate keys from which the primary key must be drawn.

- A simple key is a key made up of one column. A composite key is made up of two or more columns.
- If the database will be converted from an existing computerized system, use its tables as a starting point. Use the existing schema, table by table and the existing data entry forms for the designing process.
- The tables are comprised of table rows and columns. Table columns are responsible for storing many different types of data, such as numbers, texts, dates and even files.

10.7 KEY WORDS

- **Byte:** It is a set of eight bits that represent a character.
- **Data item:** Combination of more than one byte is called a data item.
- **Record:** The data items related to the object.
- **File:** A collection of related records.
- **Database:** The upper layer in hierarchy of file structure.
- **DataBase Management System (DBMS):** DBMS is a software system that allow you to define, construct and manipulate database.

10.8 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. Write the types of file organization.
2. Explain the types of files.
3. Define the significance of inverted list organization method.
4. Explain the concept of direct access organization.
5. Elaborate on the term transaction file.
6. State about the flat file database system.
7. How the database is manipulated?
8. Explain the concept of data abstraction.

Long-Answer Questions

1. Briefly explain the file organization with the help of appropriate examples.
2. Describe the concept of sequential file organization and relative file organization with the help of examples and illustrations.
3. Discuss about the types of files with the help of examples.

NOTES

4. Briefly explain the database design and objectives giving appropriate examples.
5. Discuss the difference between flat file databases and relational databases with the help of examples.
6. How a good DataBase is designed? Explain giving appropriate examples to support your answer.

10.9 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 11 TYPES OF DATABASES

NOTES

Structure

11.0 Introduction
11.1 Objectives
11.2 Types of Databases
11.3 Hierarchical Model
11.4 Network Model
11.5 Relational Model
11.6 Answers to Check Your Progress Questions
11.7 Summary
11.8 Key Words
11.9 Self Assessment Questions and Exercises
11.10 Further Readings

11.0 INTRODUCTION

The heart of the data base is DataBase Management System (DBMS). It manages and controls the data base file and handle request from the application program. A data structure defines relationships among ententes. There are three types of relationships one-to-one, one-to-many and many-to-many. Although all DBMS have a common approach to data management, they are differ in the way they structure data. The three types of data structure are hierarchical, network and relational.

A data model determines the structure of data. It is considered as foundation of a database which determines the sequence of storing, organizing and manipulating the data in a database system. It defines the infrastructure offered by a particular database system. Communication and precision are the two key benefits that make a data model important to applications that use and exchange data. Data models are often complemented by function models. Hierarchical model, network model, relational model, object oriented model and object relational model are the types of data model. A hierarchical database model is a data model in which the data is organized into a tree-like structure. The structure allows representing information using parent and child relationships where each parent can have many children but each child has only one parent. All attributes of a specific record are listed under an entity type. The relational data model represents a database as a collection of relation values or relations where a relation resembles a two-dimensional table of values presented as rows and columns. A relation has a heading, which is a tuple of attribute names, and a body, which is a set of tuples having the same heading. The purpose of the relational model is to provide a declarative method for specifying data and queries. In network model, a record can have any number of parent records which can have multiple child records. In the network data model, data

NOTES

manipulation language is used for searching and retrieving records from the DataBase.

In this unit, you will study about the hierarchical, network and relational DataBase models.

11.1 OBJECTIVES

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

- Explain the significance of data model
- Describe the features of hierarchical model
- Explain the network model
- Understand the function of relational model

11.2 TYPES OF DATABASES

A data model is an abstract model that documents and organizes the business data for communication between database team members and is used as a plan for developing applications specifically how data is stored and accessed. According to Hoberman, *'A data model is a way of finding tool for both business and Information Technology or IT professionals which uses a set of symbols and text to precisely explain a subset of real information to improve communication within the organization and thereby lead to a more flexible and stable application environment'*. A data model is used to determine the structure of data or structured data. Applications of data models include database models, design of information systems and enabling exchange of data. The data models are specified in a data modelling language. The primary function of information systems is to manage large quantities of structured and unstructured data. Data models describe structured data for storage in data management systems, such as relational databases. They do not describe unstructured data, such as Word processing documents, e-mail messages, pictures, digital audio and video. A data model describes the following components:

- **Structure of Database:** This approach represents how the data is arranged and organized in a database.
- **Integrity of Database:** This approach provides a set of rules which indicate whether or not defined structure can be used to organize data in a selected database.
- **Manipulation of Database:** This approach supports a language in which you can update data in the given database.
- **Querying Data:** This approach supports SQL (Structured Query Language) which provides a language in which user can ask query from selected database and get the result.

Role of Data Models

The main objective of data models is to support the development of information systems by providing the definition and format of data. According to West and Fowler, *'If this is done consistently across systems then compatibility of data can be achieved. If the same data structures are used to store and access data then different applications can share data. The results of this are indicated above. However, systems and interfaces often cost more than they should to build, operate and maintain. They may also constrain the business rather than support it'*. Communication and precision are the two key benefits that make a data model important to applications that use and exchange data. A data model is the medium which project team members from different backgrounds and with different levels of experience can communicate with one another. Precision means that the terms and rules on a data model can be interpreted only one way and are not ambiguous. A data model is also referred to as a data structure especially in the context of programming languages. Data models are often complemented by function models especially in the context of enterprise models.

NOTES

Perspectives of Data Models

The American National Standards Institute/Standards Planning and Requirements Committee or ANSI/SPARC three level architecture shows that a data model can be an external model (or view), a conceptual model or a physical model. Following are the components of data models:

- **Conceptual Schema:** This component describes the scope of the model and consists of entity classes representing kinds of things of significance in the domain and relationships assertions about associations between pairs of entity classes. A conceptual schema specifies the kinds of facts or propositions that can be expressed using the model.
- **Logical Schema:** Logical Schema component is represented by a particular data manipulation technology and consists of descriptions of tables and columns, object oriented classes, and XML tags.
- **Physical Schema:** This component describes the physical means by which data are stored. This is concerned with partitions, Central Processing Units or CPUs, tablespaces, etc.

The importance of this approach, according to American National Standards Institute or ANSI is that it allows the three perspectives to be relatively independent of each other. The table/column structure can change without affecting the conceptual model. Communication and precision are the two key benefits that make a data model important to applications that use and exchange data. A data model is considered as a medium which project team members from different backgrounds and with different levels of experience can communicate with one another. Precision refers to that the terms and rules on a data model can be interpreted

NOTES

only one way and are not ambiguous. Data models have been used for many years to design information systems. This type of software typically uses a Relational DataBase Management System or (RDBMS) based on Structured Query Language (SQL). Logical data models are used to organize and understand information structure. Physical data models incorporate data types and other design details needed to generate SQL code. In addition to creating logical and physical data models, MacA&D and WinA&D can customize diagram presentations, model non-visual database elements and generate SQL for Oracle, DB2, SQL Server, Sybase, Informix and InterBase. A data model comprises of three parts:

- A structural part consisting of a set of rules according to which databases can be constructed.
- A manipulative part defining the types of operation that are allowed on the data. This approach includes the operations that are used or updating or retrieving data from the database and for changing the structure of the database.
- A set of integrity rules which ensures that the data is accurate.

The types of data models are discussed in subsequent sections.

11.3 HIERARCHICAL MODEL

Data models can be defined as a collection of various concepts used to describe the structure of a database. Implementing a data model includes specifying data types, relationships among data types and constraints on the data. In the hierarchical model, also called hierarchical schema, data is organized in the form of a tree structure. The hierarchical model supports the concept of data independence. Data independence is the ability to change the representation of data at one level of a database system without the compulsion of changing the data representation at the next higher level.

The hierarchical model uses two types of data structures, records and parent–child relationship to define the data and relationship among data. Records can be defined as a set of field values which are used to provide information about an entity. An entity is a collection of objects in a database which can be described by using a set of attributes. Records that have the same type can be easily grouped together to form a record type and assigned a name. The structure of a record type can be defined by using a collection of named fields or data items. Each data item or field has a certain data type such as character, float or integer. The Parent Child Relationship (PCR) can be defined as a 1:N relationship between two different record types. The record type on the 1-side is called the parent record type and the record type on the N-side is called the child record type. Occurrence of the PCR type, also called instance, consists of one record of the parent record type and a number of records of the child record type. Figure 11.1 shows an example

of 1:N relationship between a Finance department and its employees.

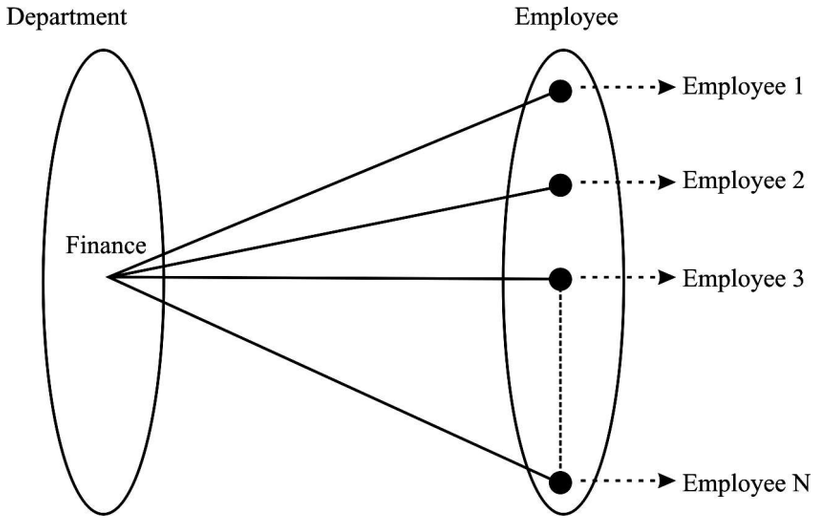


Fig. 11.1 *1:N Relationship between a Finance Department and its Employees*

Hierarchical schema consists of a number of record types and PCR types. In the hierarchical schema record types are represented by rectangular boxes and PCR types are represented by the lines, which are used to connect a parent record type to a child record type. Figure 11.2 represents a hierarchical schema which has three record types and two PCR types. Department, Employee and Project are the record types in Figure 11.3.

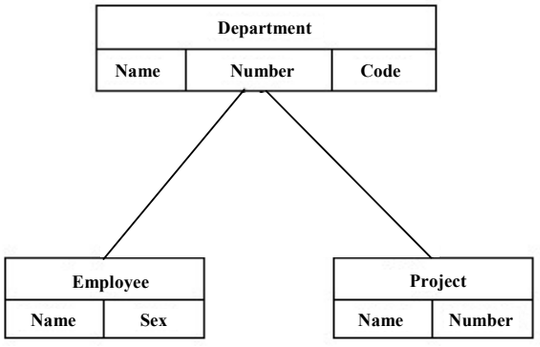


Fig. 11.2 *Hierarchical Schema*

Each record type can have a set of data items or fields. For example, the record type department can have department name, department number and department code as the fields or data items. PCR type can be represented by listing pair in parentheses. For example, in Figure 11.3, there are two PCR types, which can be represented as (Department, Employee) and (Department, Project). In Figure 11.2, each occurrence of the (Department, Employee) PCR type relates one department record to the records of many employees, who work in that department. The occurrence of (Department, Project) PCR type relates a department record

NOTES

NOTES

to the records of projects controlled by that department. Figure 11.3 represents the tree-like structure of the hierarchical schema shown.

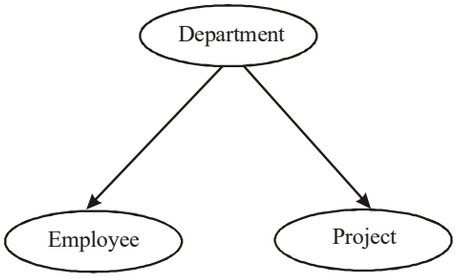


Fig. 11.3 *Tree Representation of Hierarchical Schema*

In a tree-like structure, a record type is represented by the node of the tree and PCR type is represented by the arc of the tree. The following are the properties of the hierarchical schema which contains the numbers of record types and PCR types:

- One record type, called the root of the hierarchical schema, does not participate as a child record type in any PCR type.
- In the hierarchical model, each record can have only one parent record but can have many child records.
- Every record type except the root participates as a child record type in only one PCR type.
- A record type can participate as a parent record type in a number of PCR types.
- A record type which does not participate as a parent record type in any PCR type is called leaf node in hierarchical schema.
- If a record type participates as a parent node in more than one PCR type, then its child record types must be in a left to right ordered sequence.

The advantages of the hierarchical data model are as follows:

- It is simple to construct and operate on data in the hierarchical model.
- It involves hierarchically organized domains, such as product info in manufacturing and employee information in organization.
- It uses constructs, such as `GET`, `GET UNIQUE` and `GET NEXT`.

The disadvantages of the hierarchical data model are as follows:

- It requires the navigational and procedural processing of data.
- It provides less scope of query optimization.

11.4 NETWORK MODEL

The network data model can be defined as a database model used to represent objects and the relationships among these objects. In this model, a record can have any number of parent records which can also have multiple child records. Like the hierarchical

model, the network model also supports the concept of data independence, which can be defined as the ability to change the representation of data at one level of a database system without the compulsion of changing the data representation at the next higher level. In the network data model, Data Manipulation Language (DML) is used for searching and retrieving records from the database. DML can also be used for connecting records from the set of instances, deleting and modifying records.

The network data model uses two types of data structures, records and set type, to define the data and relationship among data. Figure 11.4 represents a record type Employee that has three data items: Name, Sex and Birth Date.

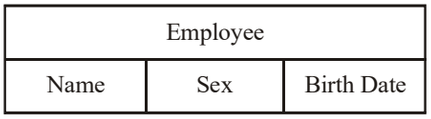


Fig. 11.4 *Employee Record Type*

Set type is a description of a 1:N relationship between two record types. Each set type definition has the following elements:

- Name for set type
- Owner record type
- Number record type

Figure 11.5 represents a set type R_Dept as an arrow. This representation is known as Bachman diagram. In the figure, Department is the owner record type and Employee is the child record type. This represents a 1:N relationship between the department of the company and the employees that are working in that department.

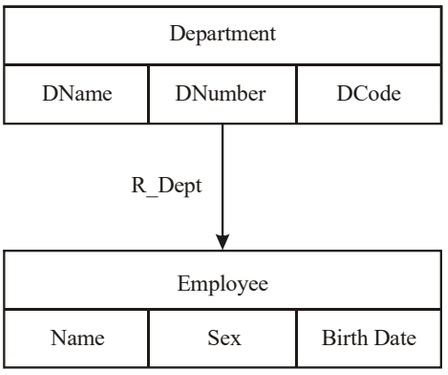


Fig. 11.5 *Set Type R_Dept*

In a database, there are set occurrences, also called set instances, corresponding to a set type. Each instance is used to relate one record from owner record type, i.e., Department to the set of records of member record types, i.e., Employee. Owner serves as parent node and member serves as a child node. Each set occurrence consists of the following elements:

- One owner record from owner record type.
- A member of related member records from the member record type.

NOTES

A record from the member record type cannot belong to more than one set occurrence of a particular set type. This represents a 1:N relationship. A set occurrence can be easily identified by the owner record or by any number of records. The following are the differences between the set instance of a database and the set in mathematics:

- The set instance in a database has one distinguished element called owner record. whereas in mathematics, there is no such type of distinction among set elements.
- In a database, all member records of a set instance are ordered. On the other hand, in mathematics, the elements of a set are not ordered.

The most commonly used implementation of a set type in a network model is the system-owned set. A system-owned set can be defined as a set that does not have any owner record type. In this set, the system can be regarded as an owner record type. It provides the following services to the network model:

- System-owned sets provide entry points into the database through the records of a specified member record type. Processing can be performed through the fields or data items of the member record type.
- System-owned sets can be used to order the records of a given record type by using set ordering specifications. By specifying the number of system-owned sets on the same record type, you can access your records in a different order.

Figure 11.6 shows a network model.

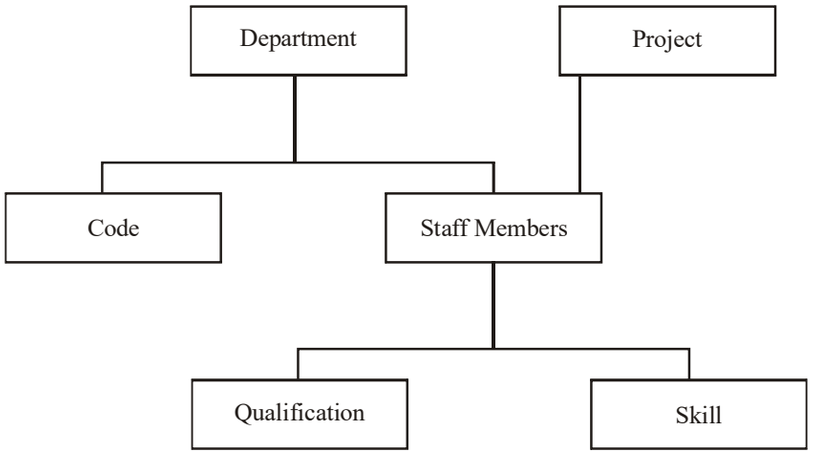


Fig. 11.6 *Network Model*

In Figure 11.6, Department, Project and Staff Members are the owner record types and Code, Qualification and Skill are the member record types.

The advantages of the network data model are as follows:

- It enables the representation of complex relationships and effect of operations, such as add and delete, on the relationships.

- It uses constructs, such as FIND, FIND OWNER and FIND NEXT within a set that allows users to navigate through the database.
- It can inherit the advantages of the hierarchical model.
- Many-to-Many (M:N) relationships are easier to implement in a network model as compared to a hierarchical model.
- It ensures data integrity.

The disadvantages of the network data model are as follows:

- It provides a complex array of pointers that thread through a set of records that are not dealt with easily.
- It provides less scope for query optimization.

Comparison of Network and Hierarchical Models

Table 11.1 summarizes the similarities and differences between network and hierarchical models.

Table 11.1 *Comparison of Network and Hierarchical Models*

Factor	Network Model	Hierarchical Model
Data Independence	Yes	Yes
Structural Independence	No. Changes in the database structure require to be made in all related application programs.	No. Changes in the database structure require to be made in all related application programs.
Programming	Extensive programming required, as network model is implemented using linked list.	Difficult to design, as you need to implement it using a tree.
Data Definition	Network Data Definition Language (NDDL) is used to define data in network models.	Hierarchical Data Definition Language (HDDL) is used to define data in hierarchical models.
Data Manipulation	Network Data Manipulation Language (NDML) is used to modify data in network models.	Hierarchical Data Manipulation Language (HDML) is used to define data in network models.
Constraint	A link depends on its start node and end node. If a start node or end node is deleted, the link is also deleted.	A record can only be occur if it is related to a parent record and not to a root record.

11.5 RELATIONAL MODEL

In 1970, E. F. Codd formally introduced the relational model. Predicate logic and set theory form the basis of the relational model for database management. This model

provides a simple, yet rigorously defined concept of the manner in which data is perceived by users.

Strengths

NOTES

Some of the strengths of relational models are given in Table 11.2.

Table 11.2 *Strengths of Relational Model*

Simplicity	End-users' requests are formulated in terms of the information content. These requests do not reflect any complexities due to system oriented aspects. A relational data model is what one sees, and not necessarily, what will be implemented physically.
Non-procedural Request	Requests focus on 'What is to be done' rather than 'How it is done'.
Data Independence	Removes the details of storage structure and access strategy from the user interface. Structural flexibility is provided by relational databases. It is easier to maintain applications written for those databases. It also allows retrieval of combinations of data that may not have been anticipated as required or needed when the database was designed. To be able to make use of this characteristic, however, the design of the relations must be complete and accurate.
Mathematical Backbone	It is based on a formal theoretical model and is not only studied extensively but proven in practice. Almost every known aspect of it is actually proven in the form of mathematical theorems.

Components

The main principle of the relational model is the information principle—all information is represented by data values in relations. The three components—structural, manipulative and integrity—make up the relational model. These components are described as follows:

- The structural component is concerned with how data is represented. A set of relations represents the conceptual view of the database.
- The manipulative component is concerned with how data is operated upon. It comprises a set of high level operations which produces whole tables and acts upon them.
- The integrity component is concerned with determining which states are valid for a database. It is a cluster of rules for the maintenance of the integrity of the database.

The relational data model represents a database as a collection of relation values or relations where a relation resembles a two-dimensional table of values presented as rows and columns. A relation has a heading which is a tuple of attribute names, and a body which is a set of tuples having the same heading. The heading of a relation is also referred to as **relation schema** or **intension** and the body of the relation is referred to as **extension**. Thus, the intension of the EMP relation would be:

```
EMP(EmpCode, EmpName, Salary, Date_of_join, Deptno)
```

Intensions provide a convenient way of describing a database depicting the schema of the database. An extension refers to the rows of data values.

Relational Terminologies

NOTES

The relational terminologies are discussed below:

Domain: A domain is the set of defined atomic values for an attribute. It is a pool of values from which specific relations draw their actual values. A domain is specified in terms of data type (possibly system-defined or user defined) and optionally, in terms of size, range, etc.

Attribute: Attribute is the name of a role played by a domain in the relation. Each attribute A_i is defined over a domain D_i (the set of values that A_i can take on) and is the name of a feature of the real world entity or relationship that the relation is representing. Formally, it is an ordered pair (N, D), where N is the name of the attribute and D is the domain that the named attribute represents, for example, EMPNAME, M GHOSH.

Relational Schema: A relational schema is made up of a relation name and a list of attributes. A relation schema R is denoted by $R(A_1, A_2, \ldots, A_n)$, where R is the name of the relation and $A_1, A_2, \ldots, A_n$ is a list of attributes. A relation schema is used to describe a relation.

Relational Database Schema: A relation schema R, denoted by $R (A_1, A_2, \ldots, A_n)$, is made up of a relation name R and a list of attributes $A_1, A_2, \ldots, A_n$. Each attribute A_i is the name of a role played by some domain D in the relation schema R. D is called the domain of A_i and is denoted by dom (A_i). A relation schema is used to describe a relation; R is called the name of this relation. The degree or parity of a relation is the number of attributes n of its relation schema. A relation schema is sometimes called a relation scheme.

A relation or relation state r of the relation schema $R (A_1, A_2, \ldots, A_n)$, also denoted by r (R) is a set of *n*-tuple t is an ordered list of n values $t = <v_1, v_2, \ldots, v_n>$, where each value v_i, $1 \leq i \leq n$, is an element of dom (A) or is a special Null value. The ith value in tuple t which corresponds to the attribute A_1 is referred to as $t [A_i]$. The terms 'Relation' for the schema R and relation extension for a relation state r (R) are commonly used.

Relation: A relation (or relation state) r of the relation schema $R(A_1, \ldots, A_n)$ is a set of n-tuples,

i.e., $r = \{t_1, t_2, \ldots, t_n\}$.

Tuple: Each row in a relation is a set of related data values and is called a tuple. Formally, an **n-tuple**, is an ordered list of values $t = <v_1, \ldots, v_n>$ where each v_i is an element of D_i where D_i is the domain of A_i.

Degree of a Relation Schema: The degree of the relation is the number of attributes (n).

NOTES

Cardinality of a Relation State: The cardinality m is the number of tuples in a particular relation state.

Formally, a **relation** is defined as the subset of the Cartesian product of domains. In order to do so, first we define the Cartesian product of two sets and then the expanded Cartesian product. The Cartesian product of two sets A and B, denoted by A × B is

$$A \times B = \{(a, b): a \in A \text{ and } b \in B\}$$

The expanded Cartesian product of n sets $A_1, A_2,..., A_n$ is defined by

$(A_1,A_2,...,A_n) = \{(a_1,a_2,...,a_n): a_j \in A_j, 1<=j<=n)\}$. The element $(a_1, a_2,... ,a_n)$ is called an n-tuple.

A relation r(R) is a subset of the Cartesian product of the domains $D(A_i)$ that define R. Therefore,

$$r(R) \subseteq D(A_1) \times D(A_2) \times \ldots \times D(A_n).$$

A *relation state* r of the relation schema $R(A_1, \ldots, A_n)$ is a set of n-tuples, i.e., $r = \{t_1, t_2, \ldots, t_m\}$.

Let us consider the following relation schema EMPLOYEE describing the employee information of a company. The relation EMPLOYEE is depicted in Figure 11.7.



EMPLOYEE	ECODE	ENAME	ADDRESS	DT_JN	BASIC	DEPT
Tuple	E01	M GHOSH	107 PRATAP GARH	10-JAN-85	6000	PROJECT

Fig. 11.7 *Relation Schema Employee*

Characteristics of Relations

Following are the characteristics of relations:

- A relation has a name that is distinct from all other relation names in the relation schema.
- Each attribute value of a tuple is atomic. Hence, composite and multi-valued attributes are not allowed in a relation.

> According to this property, repeating groups or arrays should not form columns in a relational table. Such tables are said to be in the 'First Normal Form' (1NF). The foundation of the relational model is the atomic value property of relational tables and there it is important. The primary advantage of the one value property is that it makes the data manipulation logic simple.

NOTES

- A distinct name is given to each attribute.
- In a relation, all the values of an attribute come from the same domain.
- There is no semantic significance in the order of attributes as long as correspondence between the attributes and their values in the relation is maintained.

> This property is derived from the fact that the heading of the relation is a mathematical set (of attribute). According to this property, the ordering of the columns in the relational table is meaningless. Columns can be retrieved in various sequences and in any order. The advantage of this property is that it allows multiple users to share the same table without any concern for the manner in which it is organized. It also allows the physical structure of the database to alter without any impact on the relational tables.

- Each tuple is distinct; there are no duplicate tuples.

> This property is based on the fact that the body of the relation is a mathematical set (of tuples). In mathematics, sets do not include duplicate elements. Therefore, theoretically, this property makes sure that two rows are never identical in a relational table; the values of at least one column, or set of columns, uniquely identify each row in the table. Such columns are referred to as primary keys.

- The order of tuples has no semantic significance.

> This property is based on the fact that in mathematics, a set is not ordered. Since the body of the relation is represented following the set theory, this property is analogous to the one mentioned earlier. However, it applies to rows rather than columns. The primary advantage is that in a relational table, the rows are retrievable in varying sequences and orders. Addition of information to a relational table becomes simple and does not impact the existing queries.

- Derived attributes are not captured in a relation schema.

> In an SQL schema, only two types of relation schema may be defined, i.e., VIEWS and BASE RELATION. These are called NAMED RELATIONS. Other tables, called UNNAMED RELATIONS, may be derived from these using relational operations, such as join and projection.

- **View:** It is a virtual or derived relation or a named relation that does not necessarily exist in its own rights but may be dynamically derived from one or more base relations. Its purposes may be cited as follows:
  - It provides a powerful and flexible security by hiding parts of the database from certain users.
  - It permits users to access data in a way that is customized to their needs so that the same data can be seen in different ways at the same time.
  - It can simplify complex operations on the base relations.
- **Base Relation:** This implies a named relation which corresponds to an entity in the conceptual schema whose tuples are physically stored in the database. A relational system must provide a means for creating the base relations (specifically tables) in the first place. In SQL, this function is performed by the CREATE TABLE command. Base tables have independent existence.

NOTES

Disadvantages of Relational Model

As it has already been mentioned, of all data models, the relational model is the most dominant. However, it suffers from certain limitations. Like the hierarchical and network models, the relational model has been developed to meet the requirements of business information processing. While applying the relational model to the application areas, such as Computer Aided Design (CAD), simulation and image processing, many shortcomings have been noticed in this model. It is being suggested that a more sophisticated data model should be developed. The various short comings of this model may be discussed as follows:

- **Difficulty in Modelling Complex Objects:** In certain circumstances, the strength of the relational model—its simple tabular data model—becomes its weakness. The reason for this is that compressing some of the complex relationships, that exist in the real-world, into tables is a cumbersome exercise. Thus, the modelling of such complex, nested entities in a relational data model is not easy.
- **Lack of Semantic Knowledge:** 'Semantic knowledge' refers to knowledge about the meaning of data, i.e., how to interpret data and the legitimate processes for which the data may be used. In the relational database model, this knowledge is scarce. Only the domain, entity and referential integrity rules possess semantic information. Moreover, many RDBMS (Relational Database Management System) do not fully support the domain concept. In such circumstances, application programmers are left with no other option but to compensate for the inability of the basic relational model to carry semantic knowledge, by building such knowledge into application programs.
- **Limited Data Types:** This limitation is also related to the two limitations just mentioned. An RDBMS can recognize only simple atomic data types, such as integers, characters, etc. It is one of the most critical disadvantages of RDBMS

Check Your Progress

1. Give the definition of database model according to Hoberman.
2. What are the two key benefits that make a data model important for any application?
3. How is logical schema represented?
4. Name the two types of data structures used by hierarchical model.
5. What does hierarchical schema contain?
6. Which database language is used for searching and retrieving records from the database in network model?
7. Write the advantages of network data model.
8. Who introduced relational model?

11.6 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

NOTES

1 According to Hoberman, 'A data model is a way of finding tool for both business and Information Technology or IT professionals which uses a set of symbols and text to precisely explain a subset of real information to improve communication within the organization and thereby lead to a more flexible and stable application environment'.

2. Communication and precision are the two key benefits that make a data model important to applications that use and exchange data. A data model is the medium which project team members from different backgrounds and with different levels of experience can communicate with one another. Precision means that the terms and rules on a data model can be interpreted only one way and are not ambiguous.
3. Logical Schema component is represented by a particular data manipulation technology and consists of descriptions of tables and columns, object oriented classes, and XML tags.
4. The hierarchical model uses two types of data structures, records and parent–child relationship to define the data and relationship among data. Records can be defined as a set of field values which are used to provide information about an entity.
5. Hierarchical schema consists of a number of record types and PCR types. In the hierarchical schema record types are represented by rectangular boxes and PCR types are represented by the lines, which are used to connect a parent record type to a child record type.
6. In the network data model, Data Manipulation Language (DML) is used for searching and retrieving records from the database. DML can also be used for connecting records from the set of instances, deleting and modifying records.
7. The advantages of the network data model are as follows:
   - It enables the representation of complex relationships and effect of operations, such as add and delete, on the relationships.
   - It uses constructs, such as FIND, FIND OWNER and FIND NEXT within a set that allows users to navigate through the database.
   - It can inherit the advantages of the hierarchical model.
   - Many-to-Many (M:N) relationships are easier to implement in a network model as compared to a hierarchical model.
   - It ensures data integrity.
8. In 1970, E. F. Codd formally introduced the relational model.

NOTES

11.7 SUMMARY

- According to Hoberman, 'A data model is a way of finding tool for both business and Information Technology or IT professionals which uses a set of symbols and text to precisely explain a subset of real information to improve communication within the organization and thereby lead to a more flexible and stable application environment'.
- Data models describe structured data for storage in data management systems, such as relational databases.
- Communication and precision are the two key benefits that make a data model important to applications that use and exchange data. A data model is the medium which project team members from different backgrounds and with different levels of experience can communicate with one another. Precision means that the terms and rules on a data model can be interpreted only one way and are not ambiguous.
- The American National Standards Institute/Standards Planning and Requirements Committee or ANSI/SPARC three level architecture shows that a data model can be an external model (or view), a conceptual model or a physical model.
- Logical Schema component is represented by a particular data manipulation technology and consists of descriptions of tables and columns, object oriented classes, and XML tags.
- The importance of this approach, according to American National Standards Institute or ANSI is that it allows the three perspectives to be relatively independent of each other.
- The table/column structure can change without affecting the conceptual model.
- Data models have been used for many years to design information systems. This type of software typically uses a Relational DataBase Management System or (RDBMS) based on Structured Query Language (SQL).
- A structural part consisting of a set of rules according to which databases can be constructed
- Data models can be defined as a collection of various concepts used to describe the structure of a database.
- The hierarchical model supports the concept of data independence.
- The hierarchical model uses two types of data structures, records and parent–child relationship to define the data and relationship among data. Records can be defined as a set of field values which are used to provide information about an entity.

NOTES

- Hierarchical schema consists of a number of record types and PCR types. In the hierarchical schema record types are represented by rectangular boxes and PCR types are represented by the lines, which are used to connect a parent record type to a child record type.
- Each record type can have a set of data items or fields.
- One record type, called the root of the hierarchical schema, does not participate as a child record type in any PCR type.
- In the hierarchical model, each record can have only one parent record but can have many child records.
- A record type can participate as a parent record type in a number of PCR types.
- A record type which does not participate as a parent record type in any PCR type is called leaf node in hierarchical schema.
- If a record type participates as a parent node in more than one PCR type, then its child record types must be in a left to right ordered sequence.
- The network data model can be defined as a database model used to represent objects and the relationships among these objects.
- In the network data model, Data Manipulation Language (DML) is used for searching and retrieving records from the database. DML can also be used for connecting records from the set of instances, deleting and modifying records.
- The network data model uses two types of data structures, records and set type, to define the data and relationship among data.
- In a database, there are set occurrences, also called set instances, corresponding to a set type
- A record from the member record type cannot belong to more than one set occurrence of a particular set type. This represents a 1:N relationship.
- The most commonly used implementation of a set type in a network model is the system-owned set.
- A system-owned set can be defined as a set that does not have any owner record type.
- The advantages of the network data model are as follows:
  - It enables the representation of complex relationships and effect of operations, such as add and delete, on the relationships.
  - It uses constructs, such as FIND, FIND OWNER and FIND NEXT within a set that allows users to navigate through the database.
  - It can inherit the advantages of the hierarchical model.
  - Many-to-Many (M:N) relationships are easier to implement in a network model as compared to a hierarchical model.
  - It ensures data integrity.

NOTES

- In 1970, E. F. Codd formally introduced the relational model.
- The main principle of the relational model is the information principle—all information is represented by data values in relations.
- The relational data model represents a database as a collection of relation values or relations where a relation resembles a two-dimensional table of values presented as rows and columns.
- A relation has a heading which is a tuple of attribute names, and a body which is a set of tuples having the same heading. The heading of a relation is also referred to as **relation schema** or **intension** and the body of the relation is referred to as **extension**.
- A domain is the set of defined atomic values for an attribute. It is a pool of values from which specific relations draw their actual values.
- Attribute is the name of a role played by a domain in the relation. Each attribute A_i is defined over a domain D_i (the set of values that A_i can take on) and is the name of a feature of the real world entity or relationship that the relation is representing.
- A relational schema is made up of a relation name and a list of attributes. A relation schema R is denoted by $R(A_1, A_2, \ldots, A_n)$, where R is the name of the relation and $A_1, A_2, \ldots, A_n$ is a list of attributes.
- A relation (or relation state) r of the relation schema $R(A_1, \ldots, A_n)$ is a set of n-tuples,

  i.e., $r = \{t_1, t_2, \ldots, t_n\}$.
- Each row in a relation is a set of related data values and is called a tuple.
- The cardinality m is the number of tuples in a particular relation state.
- A **relation** is defined as the subset of the Cartesian product of domains.
- A relation has a name that is distinct from all other relation names in the relation schema.
- Each attribute value of a tuple is atomic. Hence, composite and multi-valued attributes are not allowed in a relation.
- There is no semantic significance in the order of attributes as long as correspondence between the attributes and their values in the relation is maintained.
- Each tuple is distinct; there are no duplicate tuples.
- The order of tuples has no semantic significance.
- Derived attributes are not captured in a relation schema.
- As it has already been mentioned, of all data models, the relational model is the most dominant.

11.8 KEY WORDS

- **Relation schema:** The heading of a relation.
- **Extension:** The body of the relation.
- **Domain:** The set of defined atomic values for an attribute.
- **Relational database schema:** A set of relation schema having distinct name
- **Tuple:** An ordered list of elements.
- **Cardinality:** The number of tuples in a particular relation state.
- **Degree of a Relation Schema**: The degree of the relation is the number of attributes (n).
- **Data independence:** The ability to change the representation of data at one level of a database system without the compulsion of changing the data representation at the next higher level.
- **Records:** A set of field values.
- **Entity:** A collection of objects in a database.
- **Network data model:** A database model used to represent objects and the relationships among these objects.

NOTES

11.9 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. Define the role of data model according to West and Fowler.
2. Write the perspectives of data models.
3. What does conceptual schema specify?
4. What are the advantages of hierarchical data model?
5. Define the term network model.
6. Explain the term system-owned sets.
7. Write the advantages of relational model.

Long-Answer Questions

1. Discuss about the data models giving its definitions, appropriate components and examples.
2. Briefly explain the tree representation of hierarchical model with the help of an example.

NOTES

3. Describe the network model with the help of diagram. Also explain and disadvantages.
4. Discuss about the differences between network and hierarchical models giving suitable examples.
5. Briefly explain the relational model discussing the strengths and components.

11.10 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 12 SYSTEM DEVELOPMENT

NOTES

Structure

12.0 Introduction
12.1 Objectives
12.2 Software Design
12.3 System Flowcharts
12.4 Program Flowchart
12.5 HIPO Charts
12.6 Warnier-Orr Diagram
12.7 Design Methodologies
  12.7.1 Structured Design
  12.7.2 Structured Walkthrough
12.8 Overview of System Control and Quality
  12.8.1 Levels of Quality Assurance (QA)
12.9 System Testing
  12.9.1 Special Systems Tests
12.10 Answers to Check Your Progress Questions
12.11 Summary
12.12 Key Words
12.13 Self Assessment Questions and Exercises
12.14 Further Readings

12.0 INTRODUCTION

No program or system design is perfect. Communication between the user and the designer is not always complete or clear and time is usually short. The result is errors. The number and nature of errors in a new design depend on several factors. The two operational design objectives continually sought by developers are systems reliability and maintainability. There are three approaches to reliability namely, error avoidance, error detection and error tolerance. Under error avoidance, developers and programmers make every attempt to prevent errors from occurring at all. The emphasis on early and careful identification of user requirements in another way this objective is pursued. Correcting user errors, such as misspelling keywords or entering invalid commands, is one remedy.

Error detection in programs is handled in a similar manner. Error tolerance strategies keep the system running even in the presence of errors. When systems are installed, they generally are used for long periods. The average life of a system is 4 to 6 years, with the oldest application often in use for over 10 years. Several studies of maintenance have examined the type of tasks performed under maintenance. The broad classes maintenance found in information systems environments are corrective, adaptive and perfective. Once systems are installed, the need for debugging and correcting errors or failures on an emergency basis is comparatively low: less than 20 percent of the tasks are for correction.

NOTES

Software design should be guided by modularity and partitioning, coupling, cohesion, span of control, size and shared use. Well – designed, modular software is more likely to meet the maintenance, reliability, and testing requirements. Three specific tools are discussed: Structured flow-charts, Hierarchical Input Process Output (HIPO) diagrams, and Warnier - Orr diagrams. Quality assurance is the review of software products and related documentation for completeness, correctness, reliability, and maintainability. And, of course, it includes assurance that the system meets the specifications and the requirements for its intended use and performance. Four levels of quality assurance: testing, verification, validation, and certification. The philosophy behind testing is to find errors. There are two general plans for testing software: The strategies of code testing ad specification testing. Systems are not designed as entire systems nor are they tested as single systems. The analyst must perform both unit and integration testing. There are other tests that are in special category, since they do not focus on the normal running of the system. Six tests are essential. Peak load testing, storage testing, performance time testing, recovery testing, procedure testing, and human factor testing. A well-designed system should have controls to ensure proper operation and routine auditing. A candidate systems failure often results from lack of emphasis on data control. Therefore, standards of accuracy, consistency and maintainability must be specified to eliminate errors and control for fraud.

.In this unit, you will study about the software design , top down approach, flow chart, system flow chart, program flow chart, Hierarchical Input Process Output (HIPO) model, Input–Process–Output (IPO) model, Visual Text Organizational Chart (VTOC) , warnier-ORR diagram, structured walkthrough, quality assurance, levels of assurance, system testing and special tests for systems.

12.1 OBJECTIVES

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

- Explain design methodologies
- Understand the concept of system flowchart
- Describe the basic concept of Hierarchical Input Process Output (HIPO) model, Input–Process–Output (IPO) model, and Visual Text Organizational Chart (VTOC)
- Discuss the significance of Warnier-Orr diagrams and NS charts
- Construct the different types of charts
- Know about the structured design and structured walkthrough
- Understand the basic functioning of the system control and quality assurance
- Elaborate on the concept of testing

12.2 SOFTWARE DESIGN

NOTES

Two different strategies of developing the design of a software system are available. These strategies are:

1. Top-Down Strategy
2. Bottom-Up Strategy

Top-Down Strategy

The top-down strategy follows the modular approach to develop software design. It is known as the top-down strategy, because it starts from the top or the highest-level module and moves towards the lowest level modules. This technique first identifies of all, the highest-level module or the main module to develop the software. This main module is divided into several smaller and simpler submodules based on the task performed by each module. Then, each submodule is further subdivided into several submodules of the next lower level. This process of dividing each module into several submodules continues until the lowest level modules, which cannot be further subdivided, are identified.

Figure 12.1 shows the hierarchy of the modules of a software system in the top-down design strategy.

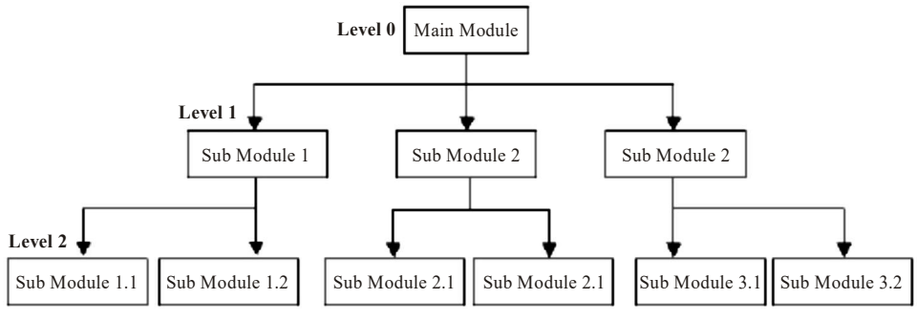


Fig. 12.1 *Hierarchy of the Modules in Top-Down Design Strategy*

Bottom-Up strategy

The bottom-up strategy also follows the modular approach to develop the software design. It is known as the bottom-up strategy, because it starts from the bottom or the most basic level modules and moves towards the highest level modules. This technique first identifies the modules at the most basic or the lowest level. They are then grouped together based on the function performed by each of them to form the next higher-level modules. Then, these modules are further combined to form the next higher-level modules. This process of grouping several smaller and simpler submodules to form higher-level module continues until the main or the highest-level module of the software development process is achieved.

NOTES

Figure 12.2 shows the hierarchy of the modules of a software system in the bottom-up design strategy.

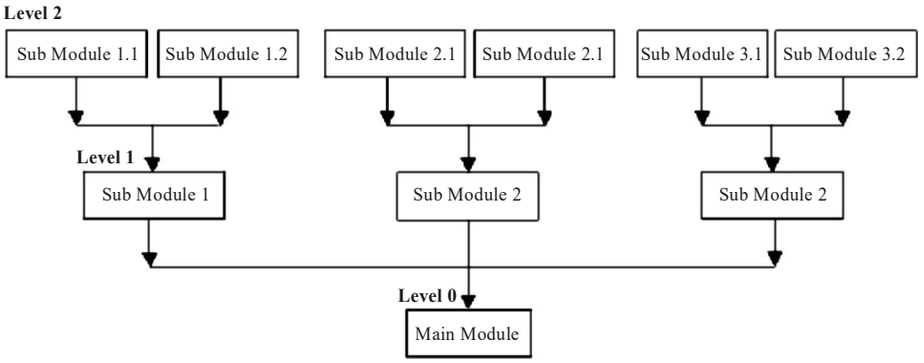


Fig. 12.2 *Hierarchy of the Modules in Bottom-up Design Strategy*

12.3 SYSTEM FLOWCHARTS

System flowchart is a symbolic representation of solution for a given system design that is processed the flow control of entire system. This flowchart is basically used by system analyst through which they go through to design the successful system implementation. Let us take an example of system flowchart of an online air ticket booking system. For this, there would be basically two master files which will be maintained for successful system implementation. One of the files is maintained for admin side and other one is maintained for travellers. For the above system design, analysts will go through a basic questionnaire as follows:

- How many flight numbers will be taken for landing and taking off?
- Travelling would be taken as round trip or one way and from where (source) to where (destination)?
- The classes and choices of meals of travellers will be recorded at which stage in system flowchart?
- How the whole system transactions will run successfully?
- What type of front end (applications) and back end support (database server) will be implemented to design the system?
- What total cost will come in the whole transaction during system implementation?

Basically, system flowchart describes the data flow for a data processing system and also provides a logical diagram of how the system operates. It represents the flow of documents and the operations performed in data processing system. It also reflects the relationship between inputs, processing and outputs. A system flowchart, also known as data flowchart, is used to describe the flow of data through a complete data processing system. Different graphic symbols represent

NOTES

the clerical operations involved and the different input, storage and output equipment required. Although the flowchart may indicate the specific programs used but no details are given of how the programs process the data.

The document symbol is used to display the procedure which is defined in the flowchart along with text, for example arrow sign of system flowchart represents the flow of control. Shared file or shared disk keeps the record of information of stored transaction.

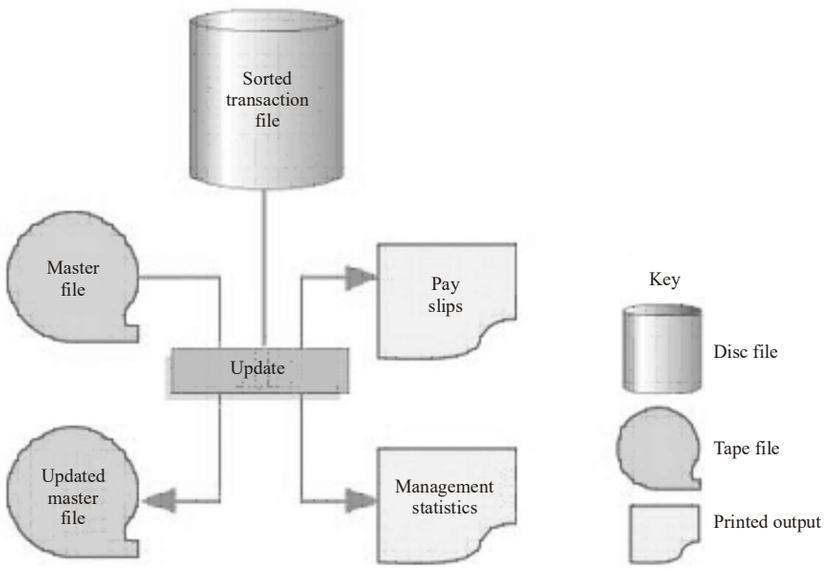


Fig. 12.3 *Sorted Transaction File in System Flowchart*

Figure 12.3 shows the system flowchart in which the transactions control will go to validate the transactions after inputting the values. If any error occurs at this stage, error report will be displayed otherwise the validate part will be checked completely. Transactions will be sorted either in ascending or descending order to make the master file. Updating of this file is maintained and updated by the system analysts at regular interval. At last, financial report is generated to produce the total cost during system implementation. Let us take an example of accounting system.

Systems flowcharts are graphic illustrations of the physical flow of information through the entire accounting system. It is commonly used in analysis and design. Flow lines represent the sequences of processes and other symbols represent the inputs and outputs to a process. Accountants use system flowcharts to describe the computerized processes, manual operations and inputs and outputs of an application system. System flowchart is a concrete and physical model that is documented in an easily visualized and graphical form including programs, procedures, files, reports, screens, etc. A system flowchart is a valuable presentation aid because it shows how the system's major components fit together and interact. In effect, it serves as a system roadmap. During the information gathering stage, a system flowchart is an excellent tool for summarizing the technical information about the existing system. A system flowchart can also be used to map hardware requirements.

NOTES

When faced with a complex system, a good approach is to draw a high level flowchart showing key functions as predefined processes and then explode those predefined processes to the appropriate level on subsequent pages. Predefined processes are similar to subroutines. The shipment is checked (in a predefined process) against the Shipping documents (the printer symbol) and Record shipment (the rectangle) in both for Inventory file and the Vendor file using data from the Item ordered file in a system flowchart for processing a shipment into inventory processing system (Refer Figure 12.4).

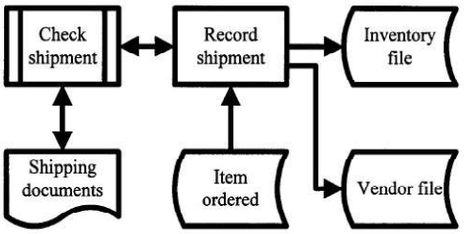


Fig. 12.4 *Predefined Processes for Complex System Flowchart*

Figure 12.4 is a flowchart of the predefined process named Check Shipment. When a Shipment arrives, it is inspected manually and either rejected or tentatively accepted. The role of Vendor File is to store the detail data of various vendors. The appropriate data are then input via a keyboard and display unit to the Record Shipment program (Refer Figure 12.4). Unless the program finds something wrong with the Shipment the newly arrived stock is released to the warehouse. If the Shipment is rejected for any reason then the Shipping documents are marked and sent to the reorder process.

Let us take an example, how a system flowchart is designed after taking two input files that will become one new master file as output. The new master file is created by the following formula:

Old Master File + Transaction File (Sorted) = New Master File

Transactions will be sorted either in ascending or descending order to make the master file. Updating file is maintained by the system analysts at regular interval. At last, financial report is generated to produce the total cost during system implementation. The variants basically depend on the software product in which professionals can choose an SDLC model to develop or modify their software. The various models are used to decide the factors of variants as follows:

- ***Waterfall Model:*** This SDLC model divides software product development phase into several developed phases. After completing the one phase the next phase starts.
- ***V-Shaped Model:*** During the developing time of software development phase, this model uses sequential path of execution. The testing procedures are developed as the base document.
- ***Incremental Model:*** The various software product developments use this model frequently. This data model is able to pass the iteration to gather the raw data for file designing, coding and testing phases.

- ***Spiral Model:*** This model is considered as incremental model that rotates between planning, risk analysis, engineering and evaluation. The phases of iterations include spirals. It is also known as baseline spirals in which data is gathered and analysed.

Information Oriented System Flowcharts

A methodology is a formalized approach to implement the SDLC as per list of steps and deliverables. There are many different systems development methodologies and each one is unique because of its emphasis on processes versus data and the order and focus it places on each SDLC phase. Some methodologies are formal standards used by government agencies while others have been developed by consulting firms to sell the clients. Many organizations have their own internal methodologies that have been refined over the years and they explain exactly how each phase of the SDLC is to be performed in that company. All system development methodologies lead to a representation of the system concept in terms of processes and data however they vary in terms of whether the methodology places primary emphasis on business processes or on the data that supports the business. The activities and information used in producing the payroll for an organization. The open ended rectangles in the diagram represent data storage containers and the rounded rectangles represent activities performed whereas arrows represent activity inputs and outputs.

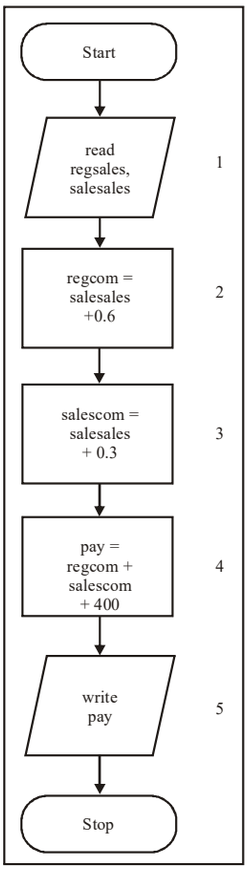


Fig. 12.5 *Sales Problem Program Flowchart*

NOTES

In Figure 12.5, *read* is followed by *regsales* and *salesales*. *Regsales*, salesales are variables representing the data items whose values can be changed. These values can vary while the time of processing the code. When a variable is used with reference to *read* statement the value of a variable is known to the computer and termed as defined value. The use of variables helps to ensure that the program is data independent which performs the required processing steps on any set of input data. Instead of *read* other terms, such as *enter*, *get* or *input* can also be used in the flowchart to accept the input values. Figure 12.4 shows the flowchart of sales problem program to compute a weekly sales report is to be written and executed. But, it does not show which mathematical functions are needed to perform which operations. The sequence of instructions is used to solve this problem:

- Input two weekly sales totals for any employee.
- Compute the regular commission by multiplying the regular sales amount by 6 per cent.
- Compute the sales commission by multiplying the reduced sales amount by 3 per cent.
- Compute the total pay due to base pay, regular commission and sales commission.
- Output the total pay on the payroll report.

Above instructions also can be expressed pictorially whereby the steps to be followed can be seen. A program flowchart of the required steps is done in the following way:

- Define the problem to be solved.
- Develop a solution algorithm in which steps are to be taken to solve the problem.

This flowchart is very useful in showing the inputs, major processing functions and output of an information processing system which presents no detail about how the system performs using specific processing steps.

Process Oriented System Flowcharts

Process oriented system flowcharts define the activities associated with the system, i.e., the processes. These methodologies utilize the process models as the core of the system concept. Analysts concentrate initially on representing the system concept as a set of processes with information flowing into and out of the processes, for example pay check details flow in to the Produce Pay Checks (PPC) process and pay checks are produced as output. Data centred methodologies focus first on defining the contents of the data storage containers and how the contents are organized. Data centred methodologies utilize data models as the core of the system concept. For example, analysts concentrate initially on identifying the data that must be available to produce the output. Flowcharting, a tool for clarifying situations and thus improving knowledge and understanding, is particularly useful when used

NOTES

by a group or team. For drawing the process oriented system flowchart, following functions are required:

- This flowchart develops a common understanding of the given situation or given problem.
- This flowchart contributes a large chain of knowledge than an individual can assuming team members are well chosen for their knowledge and experience.
- This flowchart concentrates with a common approach for solving problems, resolving ambiguities and making improvements.

After taking the process by flowcharting the final form of the flowchart should describe the process clearly. It can then be used to communicate to others. Flowcharts can either be distributed or unchanged to communicate processes or they can be used to help prepare logically structured and clearly written policies and procedures. Communication of processes in flowcharting is important for the following reasons:

- People who are new to particular processes need to learn them and be able to refer to information about them.
- The organization benefits considerably from standardized processes and these must be clearly communicated to be effective.

Process flowchart is used to measure the output of processes which is commonly referred to as product verification. Measuring is concerned with the determination of quantities of an entity, such as time, speed and capability indices whereas monitoring is concerned with continual observation part from periodic measurement. In order to measure the process following indicators are needed:

- Process objectives in which what the process is designed to achieve, for example a sales process can be designed to convert customer enquiries into sales orders into the product or service generation process.
- Indicators of performance which refer to the units of measure, for example a sales measure can be measured as the ratio of confirmed orders to be enquired.
- Defined performance standard and sensors to detect variance before during or after operations.

Flowchart is used for process monitoring that should be effective if potential and actual variations from the norm refer to ship's engine room where there are lots of dials, gauges and data logging on a continuous basis. The process oriented flowchart follows a sequence and they can be implemented in system flowcharting.

Following steps are required for aprocess oriented flowchart:

- Define the key performance indicators and method of measurements.
- Establish current performance against indicators. The indicators are constraints which are required to check the loopholes of flowcharting.

NOTES

- Produce a process flow chart to perform a task analysis to determine who does, what, when, where, when, how and why?
- Identify constraints on the process of system designing and test the data validity.
- Conduct relationship analysis to establish conflicts.
- Perform a cultural analysis to establish the behavioural factors that are caused success or failure.

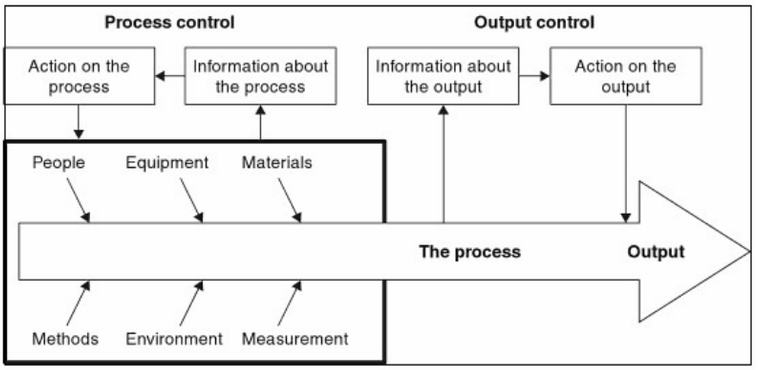


Fig. 12.6 *Process Control Flowchart*

Figure 12.6 shows the process control in which action on the process is worked with people, equipment, materials, methods, environment and measurement are processed for getting the information about the output. Arrow shows the factors used in controlling the output.

12.4 PROGRAM FLOWCHART

A program flowchart is the simplest way to figure out the bugs in a program before carrying out, saving a lot of time, labour, and money.

Definition

The programme flowchart is a diagram that represents the sequence of coded instructions fed into a machine using a set of standard graphic symbols, allowing it to perform specified logical and arithmetical operations. It is a perfect instrument for improving the quality of work. The programme flowchart, start, method, decision, and end have four basic symbols. Each symbol represents a piece of the code written for the program.

The Oval or Pill Shape - Represents the start/end.

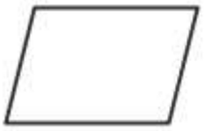


NOTES

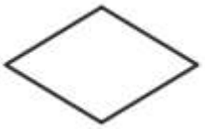
The Rectangle Shape - Represents a process.



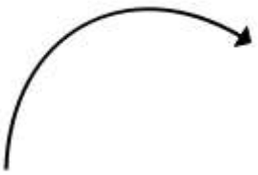
The Parallelogram Shape - Represents the input/output of the information.



The Diamond Shape - Represents a decision.



The Arrow Shape - Represents the flow of the sequence.



Steps to Create Program Flowcharts

With an easy-to-use program flowchart maker, you can create program flowcharts in minutes.

1. **Drag relevant symbols of the program flowchart and drop them on the page.**

   The purpose of the program chart is to make a complex program easy and readable, which means you should confirm your core topic and state it merely with several steps.

2. **Drag relevant vector symbols and drop them on the page.**

   Each symbol has its own function within the program, and the flowchart will not work if there is a wrong symbol.

3. **Text information into the right symbols.**

   Single keywords or short phrases will make a flowchart much clear and concise.

4. **Connect the steps with arrow according to their correct order.**

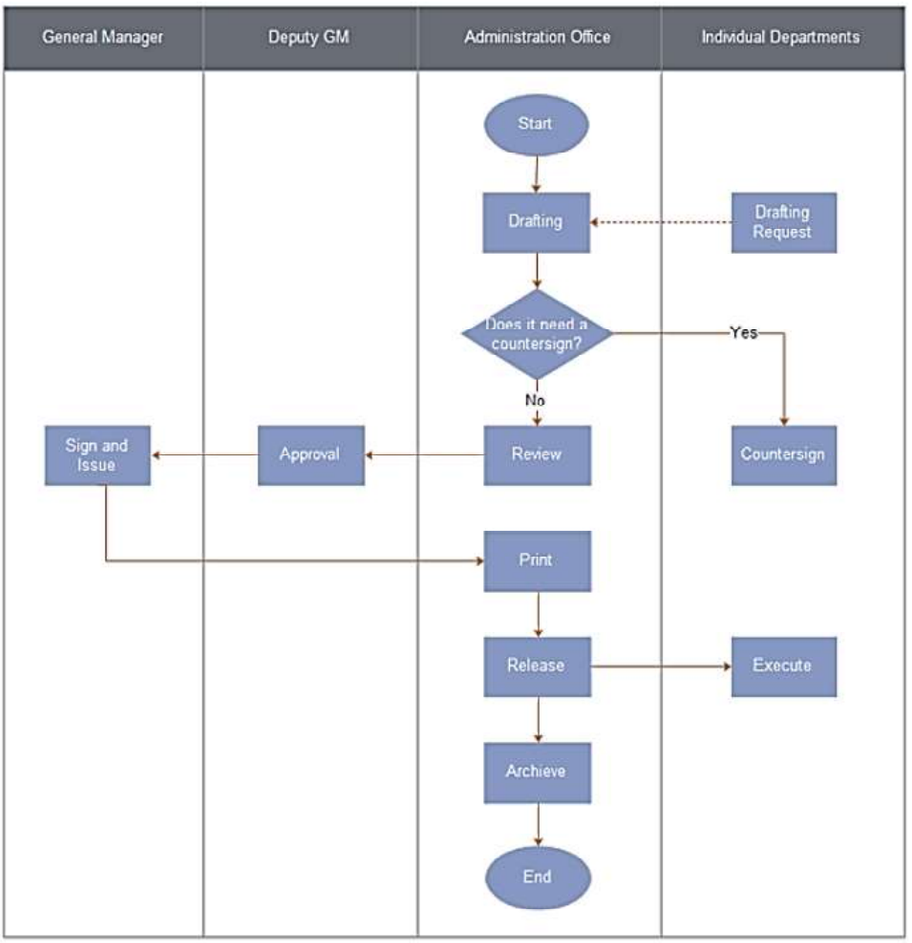
   You can connect the shapes one by one or click the floating button, which saves a lot of time.

NOTES

5. Complete and check the programming flowchart.

After completing the program flowchart, check it before carrying it out to find any bug.

Below is a program flowchart example for you.



Benefits of Program Flowchart

The advantages of program flowchart are as follows:

1. Program flowchart can help programmers to find the bug in the process before carrying out.
2. It works as a blueprint when analysing the systems and developing programs, which makes coding more efficient.

3. It improves programmers' efficiency in maintaining the operating program.
4. With the help of program flowchart, communicating the logic of a system to all concerned gets much easier.

NOTES

12.5 HIPO CHARTS

The HIPO (Hierarchy Process Input Output) chart is a tool used to analyse a problem and visualize a solution using top-down design approach.

Hierarchy Process Input Output (HIPO) diagrams were developed by International Business Machines (IBM) in the 1970s as a design notation for representing system design. These diagrams further became an excellent input for detailed program design. In addition, HIPO diagrams have been used by systems analysts to present a high level view of the functions performed by a system and decomposition of the functions into sub-functions and so on. Thus, it can be said that the HIPO diagram is as useful to analysts as the program design language is to programmers.

A HIPO diagram or chart generally consists of:

- A set of overview diagrams.
- A set of detailed diagrams.
- A Visual Table Of Contents (VTOC) consists of tree or graph structured directory, summary of contents in each overview diagram and a legend of symbol definitions.

Note that in some user environments HIPO diagrams can be used as a modelling tool. This is because they are similar to the organization chart that describes the hierarchy of managers, sub-managers, and so on.

Figure 12.7 shows a hierarchy of HIPO diagram in which UPDATE MASTERFILE is placed at the root directory which contains five branches. The branches are used to get master record, get valid transaction, update master record, write master record and print error message. This hierarchy is used by some systems analysts to present a high level view of the functions performed by a system as well as the decomposition of functions into sub-functions and so on. But, this diagram does not show the data used by or produced by the system while it is understandable that one might want to de-emphasize data relationships in a model. The details of the data are indeed shown at this level because they are not shown on the high level VTOC diagram. For this, interfaces are created between the various system components. GET VALID FIELD contains two branches known as EDIT FIELD and GET NEXT FIELD. The two sub-branches of GET NEXT FIELD are EXTRACT FIELD and GET NEXT FIELD.

NOTES

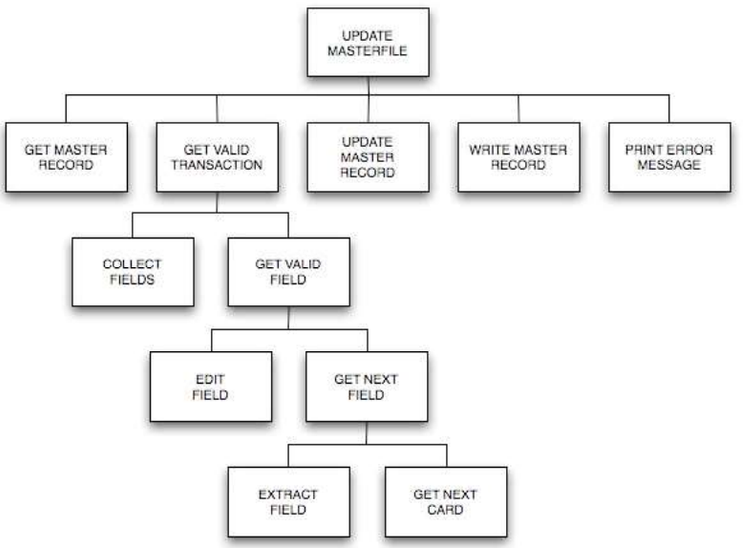


Fig. 12.7 *Structure of HIPO Diagram*

There is another component of the HIPO diagram that does show the data. The diagram shown in Figure 12.7 is known as a VTOC format. Each function represented by a rectangular box can be described in further detail in an Input-Process-Output (IPO) diagram (Refer Figure 12.8).

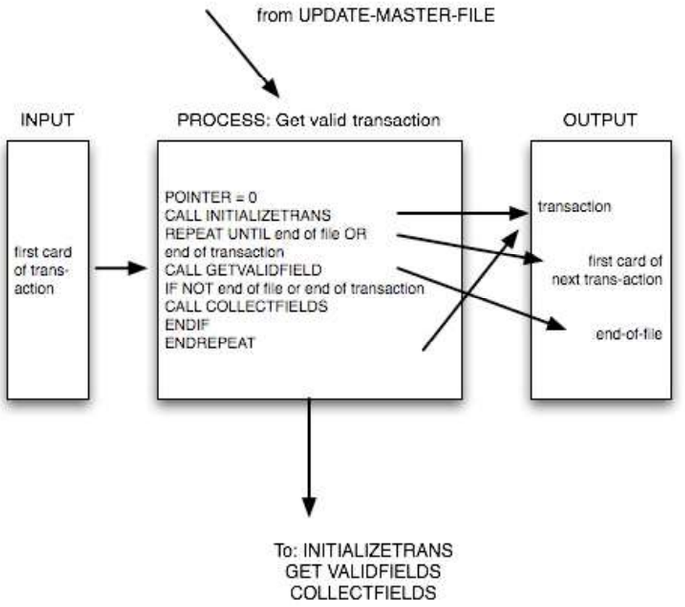


Fig. 12.8 *IPO Diagram*

First card of transaction is taken as INPUT phase which includes 'Get valid transaction, file for PROCESS phase. To get the transaction OPUTPUT following pseudocode is required:

```
POINTER=0
CALL INITIALIZETRANS
REPEAT UNTIL end of file OR
end of transaction
CALL GETVALIDFIELD
IF NOT end of file or end of transaction
CALL COLLECTFIELDS
ENDIF
ENDREPEAT
```

Check Your Progress

1. Elaborate on the top-down strategy.
2. Explain the concept of bottom-up strategy.
3. What is system flowchart?
4. Write the various models used to decide the factors of variants.
5. Defines the term process oriented system flowchart.
6. Give the definition of program flowchart.
7. What does Hierarchical Input Process Output (HIPO) diagram consist of?
8. Give the definition of Visual Table Of Organizational (VTOC).

12.6 WARNIER-ORR DIAGRAM

A Warnier-Orr diagram is a type of hierarchical flowchart that depicts the description of the organization of data and procedures. It is also known as a logical construction of a program or a system. They were initially developed in France by Jean-Dominique Warnier and in the United States by Kenneth Orr. This diagram depict the design of program structures by identifying the output and processing the results and then working backwards to determine the steps and combinations of input needed to produce the results/output.

Warnier-Orr diagrams demonstrate the processes and sequences in which they are performed. Each process is defined in a hierarchical manner. At each level, the process is shown in bracket that groups its components. Since a process can have many different subprocesses, Warnier-Orr diagram uses a set of brackets to show each level of the system. A Warnier-Orr diagram shows critical factors in system definition and development through iteration, repetition or alteration.

NOTES

Using Warnier-Orr Diagrams

To develop a Warnier-Orr diagram, the analyst works backwards, starting with systems output and using output oriented analysis. On paper, the development moves from right to left. First, the intended output or results of the processing are defined. At the next level, shown by inclusion with a bracket, the steps needed to produce the output are defined. Each step in turn is further defined. Additional brackets group the processes required to produce the result on the next level. Warnier-Orr diagram offer some distinct advantages to systems experts. They are simple in appearance and easy to understand. Yet they are powerful design tools. They have advantage of showing groupings of processes and the data that must be passed from level to level. In addition, the sequence of working backwards ensures that the system will be result oriented. This method is useful for both data and process definition. It can be used for each independently, or both can be combined on the same diagram.

Constructs in Warnier-Orr Diagrams

There are four basic constructs used in Warnier-Orr diagrams: Hierarchy, Sequence, Repetition, and Alternation. There are also two slightly more advanced concepts that are occasionally needed: Concurrency and Recursion.

- **Hierarchy:** Hierarchy is the most fundamental construct among other constructs of the Warnier-Orr. It is simply a nested group of sets and subsets shown as a set of nested brackets. Each bracket on the diagram represents one level of hierarchy. The hierarchy or structure that is represented on the diagram can show the organization of data or processing. However, both data and processing are never shown on the same diagram.

  Brackets serve two purposes. They enclose the members of a set of logically related items and separate each hierarchical level. A bracket must be given a unique user defined name. Each bracket in the diagram represents a breakdown of the item name at the head of the bracket. The Warnier-Orr Diagram is read from left to right and from top to bottom within a bracket.

- **Sequence:** Sequence is the simplest structure to show on a Warnier-Orr diagram. Within one level of hierarchy, the features listed are shown in the order in which they occur. In other words, the step listed first is the first that will be executed (if the diagram reflects a process), while the step listed last is the last that will be executed. Similarly with data, the data field listed first is the first that is encountered when looking at the data, the data field listed last is the final one encountered. The Warnier-Orr Diagram uses three parts to represent the structure of a module:
  - o BEGIN
  - o Process steps
  - o END

  To represent sequence, the process steps are written at the same hierarchical level in a column one after the other and are read from top to bottom.

- **Repetition:** Repetition is the representation of a classic loop in programming terms. It occurs whenever the same set of data occurs over and over again (for a data structure) or whenever the same group of actions is to occur over and over again (for a processing structure). Repetition is indicated by placing a set of numbers inside parentheses beneath the repeating set. Typically there are two numbers listed in the parentheses, representing the fewest and the most number of times the set will repeat. By convention the first letter of the repeating set is the letter chosen to represent the maximum.

  While the minimum bound and maximum bound can technically be anything, they are most often either (1,n) or (0,n). When used to depict processing, the (1,n) repetition is classically known as a DoUntil loop, while the (0,n) repetition is called a DoWhile loop. On the Warnier-Orr diagram, however, there is no distinction between the two different types of repetition, other than the minimum bound value. Sometimes the minimum and maximum bound are predefined and not likely to change, for example, the set 'Day' occurs within the set 'Month' from 28 to 31 times (since the smallest month has 28 days, the largest months, 31). This is not likely to change. And sometimes, the minimum and maximum are fixed at the same number. The number of times clause is always an operator attached to some set (i.e., the name of some bracket), and is never attached to an element (a diagram feature which does not decompose into smaller features). The reason for this will become more apparent when you to work with the diagrams but now consider this as a formation rule for a correct diagram.
- **Alternation:** Alternation, or selection, is the traditional 'Decision' process whereby a determination is made to execute one process or another. The Exclusive OR symbol (the plus sign inside the circle) indicates that the sets immediately above and below it are mutually exclusive (if one is present the other is not). For example, diagram for Employee-Management indicates that an Employee is either Management or Non-Management, one Employee cannot be both. It is also permissible to use a 'Negation bar' above an alternative. The bar is read by simply using the word 'Not'.
- **Concurrency:** Concurrency is one of the two advanced constructs used in the methodology. It is used whenever sequence is unimportant. For instance, years and weeks operate concurrently (or at the same time) within our calendar. The concurrency operator is rarely used in program design (since most languages do not support true concurrent processing anyway), but does come into play for resolving logical and physical data structure clashes.
- **Recursion:** Recursion is the least used of the constructs. It is used to indicate that a set contains an earlier or a less ordered version of itself. In the classic 'Bill of Materials' problem components contain parts and other sub-components. Sub-components also contain sub-sub-components, and so on. The doubled bracket indicates that the set is recursive.

NOTES

12.7 DESIGN METHODOLOGIES

In order to design information systems, it is necessary for the systems analyst to comprehend the flow of information in an organization, how it assists in making decisions, and how it is associated with the goals and objectives of the organization. Therefore, one can conclude that the systems analysis and systems design phases are deeply related. While collecting data about the problem and determining the end-user requirements, the systems analyst obtains so that the organization's ideas to improve the flow and usage of information goals are achieved.

Various design methodologies, such as structured design and charts are available that the systems analyst can use to design a good software system. While developing the design of a software system, the systems analyst can determine various design specifications, such as input/output subsystems, processing and database design, and so on. Therefore, choosing appropriate tools and techniques helps the systems analyst to hasten the design process. The finished design should meet the end-user requirements and should also help in the proper maintenance of the software system.

To develop a good design, it is very important to select the best design methodology. Each design methodology has both good and bad points. Selection of design methodology depends on:

- On the size of the development time
- The background and experience of analysts or designers
- Resources allocated to development
- The type of application under development
- The time allowance for the project

The following issues should be kept in mind while selecting a design methodology:

- Is it easy to use and learn?
- Will it help the team to arrive at an understanding of the system under development?
- How much will it cost?
- Can this technique be managed?
- Which data structures are used?
- What data flow and control features does it have?
- Will it serve user-analyst communication?

12.7.1 Structured Design

Structured design is a data-flow based methodology that helps identify the input and output of the developing system. Its main objective is to minimize the complexity

and maximize the modularity of a program. Structured design also helps describe the functional aspects of the system. In structured designing, system specifications with the help of DFDs (Data Flow Diagram) act as a basis for graphically representing the flow of data and sequence of processes involved in software development. In structured designing, the complex software program is divided into several smaller and independent programs that are arranged in a proper sequence. After developing DFDs for the software system, the next step is to develop structured flow charts. Structured flow charts define the various modules of software development and their interrelationship.

The objectives of using structured flow charts are to:

- Encourage a top-down design
- Support the idea of modules and identify the right modules
- Identify and limit data couples and control flags that go in between modules

Structure chart modules fall into one of the following three categories:

- Control
- Transformation
- Function

12.7.2 Structured Walkthrough

A study of the results of an investigation and the models that are built on the basis of these results are defined as structured walkthrough. It is called 'Structured' walkthrough review process as it is formalized into a collection of processes. The purpose of a structured walkthrough is to locate mistakes and issues. The basic idea of documentation is to provide a detailed review of any mistakes, lapses, oversights, irregularities or issues. As part of the analysis phase, the first thing to be studied is the documentation that was developed during a structured walkthrough. It can be a flow chart presenting a workflow, a model diagram documenting a complete process, or a narrative describing a procedure.

To create a structured walkthrough, the results can be studied by the group members of the project group. The two key groups engaged in walkthroughs are the individuals who require their work to be re-evaluated and the team that re-evaluates it. For confirmation and to make certain that internal accuracy and consistency exists it is better to involve other analysts who are experienced in the walkthrough. Their core job is to look for issues and inconsistencies.

For a structured walkthrough, it is necessary to follow a structure similar to the interview, preparation, execution and follow-up process. The analyst whose work is getting re-evaluated should keep all the material handy for re-assessment. The right participants are identified and copies of the content under review are provided. Finally, a time and place is scheduled for the walkthrough and all participants are notified of the same. At the time of walkthrough, the material is

NOTES

presented by the analyst point by point. If it is a diagram or flow chart, he walks through the flow explaining each component. The main task of the reviewers as explained earlier, is to identify and highlight issues and inconsistencies. All comments made by the reviewer are recorded by the scribe. Necessary rectifications are made at this point. In case of major mistakes and issues, an added walkthrough may be essential.

12.8 OVERVIEW OF SYSTEM CONTROL AND QUALITY

The basic goal of system control and quality assurance is to check the sample of a developed system and implement methodologies to find errors, if any, in the processed data. The type and quality of the developed system needs to be assessed, planned and implemented. At this phase, the quality management team fixes the criteria to assess the data quality and arrange training programs for an effective implementation of system control and quality assurance. These steps ensure that the developed system is authentic and full of security safeguards; and it is properly documented to comply with the cost effectiveness of the program requirements. The system control is basically oriented to prevent system deficiency, such as complexity, low quality of system execution and desired output screen.

Figure 12.9 shows how the hierarchy of the quality control team works together to confirm quality assurance. The complete system is first scrutinized. The team then analyses the performance of the system and the team prepares a user manual to guide the users. This manual is prepared technically, i.e., it is scrutinized by gaining the Incoming Quality Control (IQC), Process Quality Control (PQC) and Outgoing Quality Control (IQC). Table 12.1 shows the 'FPA' phase known as Functionality' → Processing' → Adaptabilty mechanism.

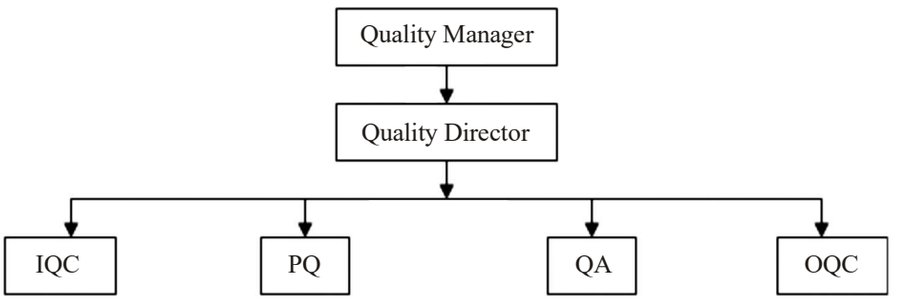


Fig. 12.9 *Hierarchy of Quality Control Team*

Table 12.1 Typical Software Quality Control Factors

Functionality (Exterior Quality)	Processing (Interior Quality)	Adaptability (Future Quality)
Correctness	Efficiency	Flexibility
Reliability	Testability	Reusability
Usability	Documentation	Maintainability
Integrity	Structure of the system	

NOTES

Process and product-level quality are parts of software quality engineering. Process level quality is referred to as quality assurance, whereas, product level quality is referred to as testing. Software Quality Assurance (SQA) comprises the methods, procedures and tools applied by the software professionals to ensure that the software fulfils the standards already specified and set. The purpose of SQA is to provide the administration with proper visibility of the software project and of the products being developed. It produces an efficient, optimized and acceptable software code as fault-free as possible by reviewing and auditing the software products and activities.

SQA monitors the methods and standards to confirm that the software specialists have properly implemented their expertise.

During the initial stages of the software development process, the SQA group along with the software team establishes plans, standards and procedures for the software that satisfy the objectives of the organization. The SQA group reviews and audits the software product and activities throughout the System Development Life Cycle (SDLC). The following are the goals of SQA:

- To plan the SQA activities.
- To monitor the software development process and the final software developed.
- To ensure that the software project is implementing the standards and procedures set by the management.
- To notify groups and individuals about the SQA activities and results of these activities.
- To ensure that the issues that are not solved within the software are addressed by the upper management.
- To recognize any insufficiency in the product, process or the standards and fix them.

SQA Activities

The following are the top-level activities performed by the SQA group:

- Prepare the SQA plan for the software according to a documented procedure.

NOTES

- Ensure that the tasks performed by SQA are according to the plan.
- Help in preparation and review of the standards and procedures of the software development process.
- Review the activities of the software development team to confirm compliance.
- Inform the results of its activities to the software development team.
- Handle the differences found in the result of the software activities and the software products according to a documented procedure.

12.8.1 Levels of Quality Assurance (QA)

In order to certify a software product, several levels of QA and testing have to be performed. All QA levels are not needed for all the different types of software described.

The levels of quality assurance are:

Level 1: Code walk-through

At this level, the software is checked for any violations of the official coding rules in the off-line software. In particular, emphasis is placed on the examination of the documentation and the level of in-code comments.

Level 2: Compilation and linking

At this level, it is checked if the software can compile and link all official platforms and operating systems.

Level 3: Routine running

Routine running assures that the software can run properly under a variety of conditions, such as certain number of events and small and large event sizes, etc.

Level 4: Performance test

At this final level, the software is evaluated for its performance. This test checks whether the performance of the software satisfies the previously specified performance level. The performance of the software can be evaluated by determining the time taken to reconstruct an event and the Central Processing Unit (CPU) time per event.

12.9 SYSTEM TESTING

System testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements.

In system Testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behaviour of a component or a system when it is tested.

System testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both. System testing tests the design and behaviour of the system and also the expectations of the customer. It is performed to test the system beyond the bounds mentioned in the Software Requirements Specification (SRS).

System Testing is basically performed by a testing team that is independent of the development team that helps to test the quality of the system impartial. It has both functional and non-functional testing.

Types of System Testing

- **Performance Testing:** Performance testing is a type of software testing that is carried out to test the speed, scalability, stability and reliability of the software product or application.
- **Load Testing:** Load testing is a type of software testing which is carried out to determine the behaviour of a system or software product under extreme load.
- **Stress Testing:** Stress testing is a type of software testing performed to check the robustness of the system under the varying loads.
- **Scalability Testing:** Scalability testing is a type of software testing which is carried out to check the performance of a software application or system in terms of its capability to scale up or scale down the number of user request load.

12.9.1 Special Systems Tests

There are other tests that are in special category, since they do not focus on the normal running of the system. Six tests are essential.

1. Peak Load Testing

There are critical times in many systems, particularly online systems. For example, in a banking system, analysts want to know what will happen if all teller sign on at their terminals at the same time before the start of the business day. Will the system handle them one at a time without incident, will it attempt to handle all of the at once and be so confused that it "locks up" and must be restarted, or will terminal addresses be lost?

The only sure way to find out is to test for it. The same situations can arise when tellers sign out during lunch periods and at the end of the day, so testing is looking at real situations.

2. Storage Testing

Analysts specify a capacity for the system when it is designed and constructed. Capacities are measured in terms of the number of records that a disk will handle

NOTES

or a file can contain. These capacities are linked to disk space and the size of indices, record keys, and so on. But they too must be tested. If the documentation for a new system to be run on a microcomputer claims that a disk file can store up to 10,000 records, each 393 bytes long, the claim must be verified before implementation.

Storage testing often requires entering data until the capacity is reached.

Comparing the actual and claimed capacities will verify the accuracy of the documentation on the one hand and allow a judgement about actual capacity at the same time. Many, systems are never tested in this way. Users find out too late that claims made during installation are not true: there is not enough storage capacity for transactions and master file records.

3. Performance Time Testing

When analysts are developing a design, their concerns are more on reports, inputs, files, and processing sequences than on performance time, although this changes with experience. During simple unit and integration testing, relatively small sets of data are used to find errors or cause failures. Therefore, users frequently find out how slow or fast the response time of the system is only after it has been installed and loaded up with data.

That may be too late. Systems are rarely too fast for users.

Performance time testing is conducted prior to implementation to determine how long it takes to receive a response to an inquiry, make a backup copy of a file, or send a transmission and receive a response. It also includes test runs to time indexing or resorting of large files of the size the system will have during a typical run or to prepare a report.

A system that runs well with only a handful of test transactions may be unacceptably slow when full loaded. And the time to know about this is prior to implementation, when adjustments can be more easily made. Once files are fully loaded and the user is relying on the system for daily activities, it is difficult to pull it back and being large-scale changes. The user needs the system and the analyst will not want to risk the loss of live data.

4. Recovery Testing

Analysts must always assume that the system will fail and data will be damaged or lost. Even though plans and procedures are written to cover these situations, they also must be tested. By creating a failure or data loss event where the users are forced to reload and recover form a backup copy, analysts can readily determine whether recovery procedures are adequate. The best designed plans usually are adjusted or augmented after this test.

NOTES

5. Procedure Testing

Documentation and run manuals tell the user how to perform certain functions are tested quite easily by asking the user to follow them exactly through a series of events. It is surprising how not including instructions about when to depress the enter key, about removing diskettes before powering down, or what to do when the paper – out light on the printer lights up can raise questions.

There is, of course, no substitute for a well – designed set of procedure manuals.

Analysts concentrate on the major and critical details of a systems design and include them in the documentation. They also pay attention to the little details, when designing the system. But often descriptions of the details do not get into the documentation. This type of testing not only shows where they are needed but also where they are wrong, that is, where actions suggested in the documentation do not match those that must actually be taken to make the system.

6. Human Factors Testing

What do users do if, after submitting a transaction through a terminal, the screen goes blank while the data are being processed? They may not take the actions the analyst wants or expects, instead responding in unusual ways: they may depress the send key several times, turn the power switch on the terminal off and back on, unplug it and replug it, or beat on the terminal. Obviously, they will do just about anything if the analyst has not given them some message on the screen to indicate that their request has been received, that it is being processed, and that there will be a short delay. This is what human factors testing is all about – finding answers to questions about how people will react to the system in ways not anticipated. And as a general rule, as strange as the above actions may sound, the people are right; they are taking actions that are normal under the circumstances.

It is the responsibility of the analyst to anticipate questions that will arise in the minds of the users as they interact with the system. If a screen will go blank during transaction processing, the analyst should make sure that it displays a message informing the user that processing is occurring. Even that is not enough if the delay will be more than a second or two. For processing that will take long periods, the analyst should have the screen give the user a message telling approximately how long it will take and providing an option to cancel the request. The user may decide to have that one – hour job run some other time when the system is not so busy.

If the system is going into a long sorting step, the analyst should keep the user informed about how much of the sort is completed. Users appreciate systems that display the numbers of records sorted or the percentages completed.

Also the analyst should be sure to watch how people enter data. Do they use different keystroke from those anticipated, such as the top row of numbers on

NOTES

the typewriter pad rather than those on the numeric keypad)? Are any keystrokes awkward? And therefore error prone (for example, having to hold down the shift key with the little finger while depressing the + key with the index finger)?

How will the user of a system feel after working with the system for a lengthy? Period of time? Glare on the screen or simply too much detail on one display is physically and mentally irritating. Slight modifications in the display contents or the location of the equipment are important human factor concerns that dramatically affect the user and, therefore, the system over time.

These simple testing questions are of monumental importance and extremely helpful in finding flaws that can cause the system to fail. Some analysts will find these flaws the hard way – through bad experiences. It is difficult to forget the system that was damaged because a user banged on the terminal when data were submitted and accepted by the system without displaying a response. But, following the guidelines above, the analyst can avoid those situations.

Check Your Progress

9. Elaborate on the Warnier-Orr diagram.
10. Explain the term concurrency.
11. Define the term structured design.
12. Explain the concept of structured walkthrough.
13. Write the goal of system control and quality assurance.
14. Name the levels of quality assurance.
15. Write the types of system testing.

12.10 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. The top-down strategy follows the modular approach to develop software design. It is known as the top-down strategy, because it starts from the top or the highest-level module and moves towards the lowest level modules. This technique first identifies of all, the highest-level module or the main module to develop the software.
2. The bottom-up strategy also follows the modular approach to develop the software design. It is known as the bottom-up strategy, because it starts from the bottom or the most basic level modules and moves towards the highest level modules. This technique first identifies the modules at the most basic or the lowest level.
3. System flowchart is a symbolic representation of solution for a given system design that is processed the flow control of entire system. This flowchart is

basically used by system analyst through which they go through to design the successful system implementation.

4. The various models are used to decide the factors of variants as follows:
   - *Waterfall Model:* This SDLC model divides software product development phase into several developed phases. After completing the one phase the next phase starts.
   - *V-Shaped Model:* During the developing time of software development phase, this model uses sequential path of execution. The testing procedures are developed as the base document.
   - *Incremental Model:* The various software product developments use this model frequently. This data model is able to pass the iteration to gather the raw data for file designing, coding and testing phases.
   - *Spiral Model:* This model is considered as incremental model that rotates between planning, risk analysis, engineering and evaluation. The phases of iterations include spirals. It is also known as baseline spirals in which data is gathered and analysed.
5. Process oriented system flowcharts define the activities associated with the system, i.e., the processes. These methodologies utilize the process models as the core of the system concept. Analysts concentrate initially on representing the system concept as a set of processes with information flowing into and out of the processes.
6. The programme flowchart is a diagram that represents the sequence of coded instructions fed into a machine using a set of standard graphic symbols, allowing it to perform specified logical and arithmetical operations. It is a perfect instrument for improving the quality of work.
7. A HIPO diagram generally consists of:
   - A set of overview diagrams.
   - A set of detailed diagrams.
8. A Visual Table Of Contents (VTOC) consists of tree or graph structured directory, summary of contents in each overview diagram and a legend of symbol definitions.
9. A Warnier-Orr diagram is a type of hierarchical flowchart that depicts the description of the organization of data and procedures. It is also known as a logical construction of a program or a system. They were initially developed in France by Jean-Dominique Warnier and in the United States by Kenneth Orr. This diagram depict the design of program structures by identifying the output and processing the results and then working backwards to determine the steps and combinations of input needed to produce the results/output.
10. Concurrency is one of the two advanced constructs used in the methodology. It is used whenever sequence is unimportant. For instance, years and weeks operate concurrently (or at the same time) within our calendar. The

NOTES

NOTES

concurrency operator is rarely used in program design (since most languages do not support true concurrent processing anyway), but does come into play for resolving logical and physical data structure clashes.

11. Structured design is a data-flow based methodology that helps identify the input and output of the developing system. Its main objective is to minimize the complexity and maximize the modularity of a program. Structured design also helps describe the functional aspects of the system. In structured designing, system specifications with the help of DFDs (Data Flow Diagram) act as a basis for graphically representing the flow of data and sequence of processes involved in software development.
12. A study of the results of an investigation and the models that are built on the basis of these results are defined as structured walkthrough. It is called 'Structured' walkthrough review process as it is formalized into a collection of processes.
13. The basic goal of system control and quality assurance is to check the sample of a developed system and implement methodologies to find errors, if any, in the processed data.
14. The levels of quality assurance are:

    Level 1: Code walk-through

    Level 2: Compilation and linking

    Level 3: Routine running

    Level 4: Performance test
15. Following are the types of system testing
    - Performance Testing: Performance testing is a type of software testing that is carried out to test the speed, scalability, stability and reliability of the software product or application.
    - Load Testing: Load testing is a type of software testing which is carried out to determine the behaviour of a system or software product under extreme load.
    - Stress Testing: Stress testing is a type of software testing performed to check the robustness of the system under the varying loads.
    - Scalability Testing: Scalability testing is a type of software testing which is carried out to check the performance of a software application or system in terms of its capability to scale up or scale down the number of user request load.

12.11 SUMMARY

- The top-down strategy follows the modular approach to develop software design. It is known as the top-down strategy, because it starts from the top or the highest-level module and moves towards the lowest level modules.

NOTES

This technique first identifies of all, the highest-level module or the main module to develop the software.

- The bottom-up strategy also follows the modular approach to develop the software design. It is known as the bottom-up strategy, because it starts from the bottom or the most basic level modules and moves towards the highest level modules. This technique first identifies the modules at the most basic or the lowest level.
- System flowchart is a symbolic representation of solution for a given system design that is processed the flow control of entire system. This flowchart is basically used by system analyst through which they go through to design the successful system implementation.
- System flowchart describes the data flow for a data processing system and also provides a logical diagram of how the system operates.
- Systems flowcharts are graphic illustrations of the physical flow of information through the entire accounting system.
- A system flowchart can also be used to map hardware requirements.
- During the developing time of software development phase, this model uses sequential path of execution. The testing procedures are developed as the base document.
- A methodology is a formalized approach to implement the SDLC as per list of steps and deliverables.
- The open ended rectangles in the diagram represent data storage containers and the rounded rectangles represent activities performed whereas arrows represent activity inputs and outputs.
- Process oriented system flowcharts define the activities associated with the system, i.e., the processes. These methodologies utilize the process models as the core of the system concept. Analysts concentrate initially on representing the system concept as a set of processes with information flowing into and out of the processes.
- Process objectives in which what the process is designed to achieve, for example a sales process can be designed to convert customer enquiries into sales orders into the product or service generation process.
- Defined performance standard and sensors to detect variance before during or after operations.
- The programme flowchart is a diagram that represents the sequence of coded instructions fed into a machine using a set of standard graphic symbols, allowing it to perform specified logical and arithmetical operations. It is a perfect instrument for improving the quality of work.
- The purpose of the program chart is to make a complex program easy and readable, which means you should confirm your core topic and state it merely with several steps.

NOTES

- A Visual Table Of Contents (VTOC) consists of tree or graph structured directory, summary of contents in each overview diagram and a legend of symbol definitions.
- The details of the data are indeed shown at this level because they are not shown on the high level VTOC diagram.
- There is another component of the HIPO diagram that does show the data.
- A Warnier-Orr diagram is a type of hierarchical flowchart that depicts the description of the organization of data and procedures. It is also known as a logical construction of a program or a system. They were initially developed in France by Jean-Dominique Warnier and in the United States by Kenneth Orr. This diagram depict the design of program structures by identifying the output and processing the results and then working backwards to determine the steps and combinations of input needed to produce the results/output.
- Hierarchy is the most fundamental construct among other constructs of the Warnier-Orr. It is simply a nested group of sets and subsets shown as a set of nested brackets.
- Sequence is the simplest structure to show on a Warnier-Orr diagram. Within one level of hierarchy, the features listed are shown in the order in which they occur.
- Repetition is the representation of a classic loop in programming terms.
- Alternation, or selection, is the traditional 'Decision' process whereby a determination is made to execute one process or another.
- Concurrency is one of the two advanced constructs used in the methodology. It is used whenever sequence is unimportant. For instance, years and weeks operate concurrently (or at the same time) within our calendar. The concurrency operator is rarely used in program design (since most languages do not support true concurrent processing anyway), but does come into play for resolving logical and physical data structure clashes.
- Various design methodologies, such as structured design and charts are available that the systems analyst can use to design a good software system.
- Structured design is a data-flow based methodology that helps identify the input and output of the developing system. Its main objective is to minimize the complexity and maximize the modularity of a program. Structured design also helps describe the functional aspects of the system. In structured designing, system specifications with the help of DFDs (Data Flow Diagram) act as a basis for graphically representing the flow of data and sequence of processes involved in software development.
- A study of the results of an investigation and the models that are built on the basis of these results are defined as structured walkthrough. It is called 'Structured' walkthrough review process as it is formalized into a collection of processes.

NOTES

- To create a structured walkthrough, the results can be studied by the group members of the project group.
- The basic goal of system control and quality assurance is to check the sample of a developed system and implement methodologies to find errors, if any, in the processed data.
- SQA monitors the methods and standards to confirm that the software specialists have properly implemented their expertise.
- To ensure that the issues that are not solved within the software are addressed by the upper management
- Handle the differences found in the result of the software activities and the software products according to a documented procedure.
- The performance of the software can be evaluated by determining the time taken to reconstruct an event and the Central Processing Unit (CPU) time per event.
- System testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements.
- System testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both.
- Load testing is a type of software testing which is carried out to determine the behaviour of a system or software product under extreme load.
- Analysts specify a capacity for the system when it is designed and constructed. Capacities are measured in terms of the number of records that a disk will handle or a file can contain.
- Storage testing often requires entering data until the capacity is reached.
- A system that runs well with only a handful of test transactions may be unacceptably slow when full loaded.
- Analysts must always assume that the system will fail and data will be damaged or lost.
- The best designed plans usually are adjusted or augmented after this test.
- Documentation and run manuals tell the user how to perform certain functions are tested quite easily by asking the user to follow them exactly through a series of events.
- The analyst should make sure that it displays a message informing the user that processing is occurring. Even that is not enough if the delay will be more than a second or two.
- If the system is going into a long sorting step, the analyst should keep the user informed about how much of the sort is completed.

NOTES

12.12 KEY WORDS

- **Software Design:** A process of problem-solving and planning for a software solution which involves the identification of the architecture, components, interfaces, etc. of the software/ component of a system.
- **Structured design:** It is a data flow based methodology that helps identify the input and output of the developing system.
- **System flowchart**: It is a symbolic representation of solution of a given system design that is processed the flow control of entire system.
- **Logical data flow diagram:** It is an implementation independent view of a system that focuses only on the flow of data between different processes or activities.
- **Structured walkthrough:** A study of the results of an investigation and the models that are built on the basis of these results are called structured walkthrough.
- **System testing:** System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements.

12.13 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. Differentiate between top-down and bottom-up strategy.
2. State about the system flowcharts.
3. How a new master file is created?
4. What is a spiral model?
5. Why information oriented system flowchart is used?
6. Write the advantages of program flowchart.
7. What are the basic constructs used in Warnier-Orr diagrams?
8. Define the term hierarchy.
9. State about the design methodologies.
10. Explain the concept of structured design.
11. Write a short note on the levels of quality assurance.
12. Elaborate on the term system testing.

Long-Answer Questions

1. Briefly explain the top-down and bottom-up strategy with the help of examples.
2. Describe the system flowcharts with the help of illustration.
3. Discuss about the program flowchart its benefits giving appropriate examples.
4. Explain the concept of information oriented flowcharts with the help of examples.
5. Briefly explain the Hierarchical Input Process Output (HIPO) chart with the help HIPO diagram.
6. Discuss about the Warnier-Orr diagram with the help of suitable examples.
7. Describe briefly the design methodologies.
8. Differentiate between structured design and structured walkthrough.
9. Diagrammatically explain the various architectural styles that are commonly used by system controls.
10. Briefly explain the special system tests with the help of examples.

NOTES

12.14 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

NOTES

BLOCK - V
SYSTEM EVALUATION, IMPLEMENTATION AND MAINTENANCE

UNIT 13 SYSTEM EVALUATION AND IMPLEMENTATION

Structure

13.0 Introduction
13.1 Objectives
13.2 Training
  13.2.1 Training Strategies
  13.2.2 Training Personnel
13.3 Conversion
  13.3.1 Conversion Plan
  13.3.2 Operating Plan
13.4 Post Implementation Review
13.5 Answers to Check Your Progress Questions
13.6 Summary
13.7 Key Words
13.8 Self Assessment Questions and Exercises
13.9 Further Readings

13.0 INTRODUCTION

For administration putting a new system into operation is usually complicated by the fact that there is an older system already in operation. The analyst has to deal with changing from something familiar to something new and different, while also attending to the mechanics of implementation.

Since the concern for simultaneous conversion and implementation is usual. Planning for the formal training of user personnel in the operation of the new system is important. A new method may drastically affect people's lives by changing their work methods, work style and relationships with other employees. One of the most effective ways of dealing with the potential impact of these changes is to provide a well-designed program of training.

Generate a feeling among employees that the new system is "Their" system. The quality of training received by the personnel involved with the system in various capacities helps or hinders, and may even prevent, the successful implementation of an information system. Conversion is the process of changing form the old system to the new one. The conversion plan includes a description of all the activities

that must occur to implement the new system and put it into operation. It identifies the persons responsible for each activity and includes a timetable indicating when each activity will occur. The operating plan is checking of all arrangements. It includes reviewing conversion plans, verifying the delivery of equipment, software, forms, preparing the site and preparing the data and files.

In this unit, you will study about training personnel ,training methods ,conversion, conversion methods ,parallel, direct, pilot and phase-in, conversion plan ,site preparation , data file preparation and post implementation review .

13.1 OBJECTIVES

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

- Know about the system training personnel
- Discuss about the purpose of giving training to the user of system
- Elaborate on the different types of training
- Explain how conversion takes place from existing system to new system
- Define the conversion plan
- Understand the post implementation review

13.2 TRAINING

An essential part of the computer-based information system development is end-user training. However, most managers, unfortunately, ignore the training of end-users quite frequently. Many training methods are being made use of in organizations presently, both instructor-led and self-paced. Most of them are aimed at providing the end-users with the required skills.

Employees can be empowered if managers first create a vision and then go about enforcing the mission statement of the company. The managers must then build trust within the organization by not only providing workplace security but also changing the attitude of the management. Employees should be provided with some financial information so that they feel more committed and responsible. This will also help them make better decisions that will add to the company's success.

Lastly, training helps the employees to solve their own problems.

Therefore, it is important for the managers to decide whether or not the benefits of one training technique will outweigh the benefits of others. It is up to the employers to see to it that the training provided matches with the job of the employees or the positions held by them. Each training module should have a set goal.

NOTES

In addition, the training courses must be designed to facilitate fast mobilization for the organization. This is considered the most essential policy in providing internal end-user training for the company.

13.2.1 Training Strategies

The task of designing the information system curricula and the training programmes can be given to the Electronic Data Processing (EDP) centre along with outside companies or academic institutions. This ensures that the training courses or programmes are suited to the trainees. The success of the training sessions can be ensured and increased in the following ways:

(i) The company must hire high-quality Information Technology (IT) personnel.

(ii) The hiring process should be fast, focussed, decisive and candid.

(iii) Once high-quality, sincere and hard working employees are in place, the IT department managers must realize that these employees need to be continuously educated. Therefore, it is important to honour the commitments about the education of employees.

(iv) The company should use training to update the employees about the latest training techniques.

(v) The training courses should be aimed at providing the employees with the skills to input information and generate reports. In addition, they should also provide knowledge about the business processes behind the system.

(vi) Training courses can be successful if the managers see to it that the users focus on the advantages of their hard work in training, as well as attempt to prevent the negative feelings of the users toward the new system.

(vii) Instead of relying on the traditional techniques, the end-users should be tutored, supported and trained by their peers The latter is much more convenient and understandable. Also, the peer tutoring technique does not concentrate totally on the technical aspects of the Computer Based Information System (CBIS). Instead, it focusses on the use of the CBIS in daily tasks. This method also alerts the end-user to self-training and can be employed to avoid problems resulting from lack of domain knowledge and physical distance.

(viii) Industrial organizations with fewer resources for in-house research and development (R&D) should focus on projects in conjunction with non-profit research institutions. Such organizations must construct suitable ground conditions for technology institutions to host joint R&D projects.

(ix) If the employees are trained too early, the likelihood of them forgetting their skills is high. As a result, the employer would have to spend on refresher courses or new training programmes.

(x) Delayed trainings may also result in the delivery of problems at the help desk or other employees, which may be costlier thant the actual training. Therefore, the right formula to be adopted is Just-In-Time (JIT) training. The monolithic and instructor-led courses must be divided into several chunks to be made available when required by the users.

(xi) Employees can be provided with Web and computer-based training in combination with instructor-led training.

(xii) The training strategies should be determined by the trainer and the trainee. The analyst ensures that a person whose work is affected by the new information system is properly trained. Possible training sources include:

- In-house trainers
- Vendors
- External/guest trainers
- Systems Analysts

The guidelines for training are as follows:

- Use of correct and appropriate training techniques.
- Establishment of measurable objectives.
- Selection of appropriate training sites.
- Employment of understandable training materials.

13.2.2 Training Personnel

Even well designed and technically elegant systems can succeed or fail because of the way they are operated and used. Therefore, the quality of training received by the personnel involved with the system in various capacities helps or hinders, and may even prevent, the successful implementation of an information system. Those whose will be associated with or affected by the system must know in detail what their roles will be, how they can use the system, and what the system will or will not do. Both systems operators and users need training.

Training systems operators

Many systems depend on the computer – centre personnel, who are responsible for keeping the equipment running as well as for providing the necessary support service. Their training must ensure that they are able to handle all possible operations, both routine and extraordinary. Operator training must also involve the data entry personnel.

If the system calls for the installation of new equipment, such as a new computer system, special terminals, or different data entry equipment, the operators training should include such fundamentals as how to turn the equipment on and use it, how to power it down, and a knowledge of what constitutes normal operation and use. The operators should also be instructed in what common malfunctions may occur, how to recognize them, and what steps to take when they arise.

NOTES

As part of their training, operators should be given both a troubleshooting lists that identifies possible problems and remedies for them, as well as the names and telephone numbers of individuals to contact when unexpected or unusual problems arise.

Training also involves familiarization with run procedures, which involves working through the sequence of activities needed to use a new system on an ongoing basis. These procedures allow the computer operators to become familiar with the actions they need to take (such as mounting magnetic disks or tapes, copying files, changing printer forms, or turning on communication systems), and when these actions must occur. In addition, they find out how long applications will run under normal conditions. This information is important both to enable users to plan work activities and to identify systems that run longer or shorter than expected – a sign that typically indicates problems with the run.

User Training

User training may involve equipment use, particularly in the case where, say, a microcomputer is in use and the individual involved is both operator and user. In these cases, user must be instructed first in how to operate the equipment. Questions that seem trivial to the analyst, such as how to turn on a terminal, how to insert a diskette into a microcomputer, or when it is safe to turn off equipment without danger of data loss, are significant problems to new users who are not familiar with computers. User training must also instruct individuals in troubleshooting the system, determining whether a problem that arise is caused by the equipment or software or by something they have done in using the system. Including a troubleshooting guide in systems documentation will provide a useful reference long after the training period is over. There is nothing more frustrating than working with a system, encountering a problem, and not being able to determine whether it is the user's fault or a problem with the system itself. The place to prevent this frustration is during training. Most user training deals with the operation of the system itself. Training in data coding emphasizes the methods to be followed in capturing data form transactions or preparing data needed for decision support activities. For example, in an accounting system, it may be important to translate customer names into customer account numbers that are input as part of the accounting transaction. Users must be trained so that they know how to determine the customer account number, that it is four digits in length, and that there are no alphabetic characters in it.

Data – handling activities receiving the most attention in user training are adding data (how to store new transactions), editing data (how to change previously stored data), formulating inquiries (finding specific records or getting responses to questions) and deleting records of data.

The bulk of systems use involves this set of activities, so it follows that most training time will be devoted to this area.

NOTES

From time to time, users will have to prepare disks, load paper into printers, or change ribbons on printers. No training program is complete without some time devoted to systems maintenance activities. If a microcomputer or data entry system will use disks, users should be instructed in formatting and testing disks. They should also actually perform ribbon changes, equipment cleaning and other routine maintenance. It is not enough to simply include this information in a manual, even though that is essential for later reference.

As the above discussion demonstrates, there are two aspects to user training: familiarization with the processing system itself (that is, the equipment used for data entry or processing) and training in using the application (that is, the software that accepts the data, processes it, and produces the results). Weaknesses in either aspect of training are likely to lead to awkward situation that produce user frustration, errors, or both. Good documentation, although essential, does not replace training. There is no substitute for hands – on – operation of the system while learning its use.

Systems analysts engage in educational processes with users, i.e., training. The following are the descriptions of the current training methods.

Instructor-Led Training

Instructor-led training method involves trainers as well as trainees who are required to meet at the same time, but not necessarily at the same place. Two broad categories of this type of training are: virtual and normal classrooms.

Virtual Classroom

The characteristics of a virtual classroom are:

- The trainees and trainers must meet at the same time, but not necessarily at the same place.
- The training session could be collaborative or one-on-one.
- The tools used for training could include videoconferencing, text-based Internet relay chat tools or virtual reality packages.

The following are the merits and demerits of virtual training:

Merits

- This technique helps save money, which is otherwise spent on travelling.
- The outcomes of this technique are similar to the face-to-face method but cheaper.

Demerits

- The interaction between trainer and trainee is still in a limited context.
- The technologies must be sufficient to support this category of training.
- A test of the objectives for the training is hard to conduct.

NOTES

Normal Classroom

The features of a normal classroom are as follows:

- The trainers and trainees must meet at the same time and at the same place.
- The tools used to provide training are the same as those used in the normal classroom (i.e., blackboard, overhead projectors, Liquid Crystal Display (LCD) projector, etc.).
- The training session could be either collaborative or one-on-one.

The following are the merits and demerits for this category of training:

Merits

- The user's questions should be answered by experts.
- The companies reflect an increase in productivity following the training.
- The context of the training is unlimited. Information can also be transferred by the trainer using computer technologies.
- Most of the courses guarantee a certificate for the trainee announcing their ability following course completion.
- The abilities of the trainees are easy to test.

Demerits

- It is an expensive technology as employers need money to meet travel expenses.
- In addition, the travel time leads to the employee's downtime being extended.
- The trainees involved in face-to-face training will have less interest and perceived learning than those who participate in a computer-based training.
- It is difficult to set up this method as it requires time and resources.

Self-Paced

This method of training involves both trainers and trainees, who do not require to meet at the same time or at the same place. This type of training can be divided into two main classes, that is, multimedia and Web-based. The details for each category are as follows:

Multimedia

This category has the following features:

- The skills are learnt by the trainees themselves, at their own pace and in their own time.
- This category of training is commonly developed and stored in a CD-ROM (Computer Disc-Real-Only Memory) where it will run on a PC (Personal Computer).
- Normally, the courses are presented in a multimedia format.

The following are the merits and demerits for this category of training:

Merits

- Trainees have the option of accessing the courses at their convenience.
- Certain courses also let the instructors change the outline of the course easily by allowing users to modify the graphics, the screen layout and training questions without help from external programmers. This could help to reduce the cost in developing an in-house training course.
- It can be used to provide JIT training to the end-users.
- The courses are normally represented in a multimedia form. Thus, they can stimulate the users' interests.

Demerits

- The training must be imparted in a conservative manner as this technology cannot be used completely in lieu of an instructor.
- PC-based training is only appropriate for conveying the basic information.
- It is not easy to test the abilities of the trainees.

Web-Based Training

This category has the following features:

- It lets trainees learn at their own pace and convenience.
- It is capable of supporting both the Internet and Intranet.
- The structures of the courses are often presented in hyper-media format.

The following are the merits and demerits for this category of training:

Merits

- Trainees can access the courses at their own convenience.
- Online training lets the employees spend more time in the office.
- The course can be customized to suit individual needs.
- It can be used to provide JIT training for the end-users.
- Web-based training technology permits organisations to tailor training requirements to fulfil individual needs.
- This technology results in cost reduction as the users are not required to pay for travel expenses and the employee's downtime is shorter than ever before.

Demerits

- The medium has its limitations and the courses are still in the early phase of development. The developer will, therefore, be forced to minimize the video context and use a text-based style for the instructions.

NOTES

- On its own, this technology cannot substitute instructor-led training. Therefore, the training should be given conservatively.
- There is no simple manner of conducting a test of objectives of the courses.

NOTES

Check Your Progress

1. Write about the success of training sessions.
2. Define the concept of training personal.
3. What is instructor-led training?
4. Write the features of normal classroom.
5. What are the demerits of web-based training?

13.3 CONVERSION

Conversion is a method of replacing the old software with a new one. A conversion plan includes a description of all those activities which must occur during the implementation of a new system. The following are the activities that are included in the conversion method:

- Name all files for conversion.
- Identify the data requirements for the development of new files during conversion.
- List new documents and procedures that are required.
- Identify the controls to be used in each activity.
- Identify the persons responsible for each activity.
- Verify all schedules.

The four methods of conversion are as follows:

- Parallel conversion
- Direct cutover
- Phase-in method
- Pilot approach

Parallel Conversion

In parallel conversion, both the old and the new system are used simultaneously. This method has two advantages. First, the continued use of the old system provides a fallback when the new system fails. Any new information system, no matter how exhaustively it has been tested, may fail or have problems shortly after implementation. The old system provides business continuation capabilities while the new system is being modified.

NOTES

Second, this method serves as an ultimate test for the new system. Parallel conversion is typically conducted for at least two accounting cycles. Both systems should produce exactly the same results or differences should be able to reconcile any. The disadvantage of this system is that keeping two systems running at the same time causes cost overruns. However, studies have shown that it is cheaper in the long run to use the parallel conversion method.

Most companies have found that the extra work and expense of the parallel conversion method are a small price to pay for the convenience of having an existing system to provide backup support when needed.

Direct Cutover Conversion

In direct conversion, the new system is implemented and the old system is totally replaced. This method is cheap and fast, but is quite hazardous. The direct cutover method converts from the old to the new system abruptly. As mentioned earlier, testing never uncovers all problems. With the direct cutover, 'You pay your money and you take your chances!'

It forces all system users to make the new system work. Direct cutover requires careful planning. Training sessions must be scheduled and maintained. The installation of all devices must be on time. Direct cutover works best when the conversion team anticipates the occurrence of problems and is ready to tackle them. A support team is essential in this case.

Phase-In Method

This technique supports a phased approach to system conversions. The approach has several advantages including the ability to come up with improved and more accurate estimates of the amount of work to be undertaken, thereby avoiding large contingencies for risk management. This results in a higher quality solution which is more in keeping with expectations without any reduction in quality due to the pressures of meeting time scales and tight budgets.

The phase-in method is used when it is not possible to install a new system throughout the organization at once. The long-term phase-in periods create difficulties for analysts whether the conversions go well or not. If the system is working well, early users will communicate this information to other personnel who are waiting for implementation. On the other hand, if there are problems in the early stages of implementation, this will also spread among the users. This may lead the users to react negatively to the smallest mistakes, even their own.

This technique forms the basis of formal project management and control procedures that need to be followed on all projects. The project execution plan which is drawn up at the commencement of the project provides various details about the tasks to be performed along with resource allocation. On a weekly/fortnightly basis, achievement against the schedule is reviewed and assessment of the time left for completion is done for each job/task.

NOTES

A project status report is generated every fortnight and discussed with the client project manager. An assessment of the results of the reviews and inspections confirms the progress.

Pilot Approach

When new systems also involve new techniques or drastic changes in organization performance, this approach is preferred. In the pilot approach, a working version of the information system is implemented in one part of the organization. When the system is approved for its performance, it is installed throughout the organization all at once or one stage at a time.

Types of Conversion

Conversion efforts depend on the type of conversion carried out. Software conversion falls into the following categories: DBMS type, language type and network type.

There are different versions of the same software, whether language or database system. For example, a Database Management System (DBMS) has different versions. Earlier dBaseI, dBaseII, dBaseIII, etc., were different versions of the same DBMS. Similarly, there are different versions of the same relational database. MS Access is a relational database having many versions. Similar is the case of distributed database that is widely used in client/server architecture. The same thing applies to the native operating systems or network operating systems. Similar is the case with conversions of other systems, whether high-level language or low-level language or compiler. These conversions are aimed at enhancing programs conforming to improved standards.

Conversion Strategy

To carry out a work where so many factors get involved, it is always better to follow a plan for conversion.

Hardware Conversion Strategy

This strategy is very important as it requires working on system hardware. All hardware may not support all kinds of software. More powerful microprocessors are developed now a days and a clear-cut strategy for the conversion of system hardware is to be decided. The conversion strategy should be such that the existing system does not stop functioning or need not be brought to a halt. For a particular period, it may be required to work in parallel and even make a comparison between these two systems. For example, when an organization that was using the earlier system with 5.25 inches floppies got 3.5 inches floppy diskettes, it required the transfer of all data from 5.25 inches floppies to 3.25 inches floppies. In recent days, so much of high memory consuming software is being used, particularly

graphic based software that one or two floppies are not found adequate and the general trend has shifted from floppy drives to (Compact Disk) CD-drives and (Digital Video Disk) DVD-drives. Instead of bootable floppies, bootable CDs are used. Strategies to carry out hardware conversion, thus, are very essential and should be mentioned in the planning document.

Software Conversion Strategy

A section is to be devoted to software conversion strategy describing strategies to be followed in case of software. Hardware is not much varied as compared to software. Each hardware or peripheral comes with its driver software. It is also the fact that all hardware do not support all software. For example, Intel processor 386 was not capable of supporting Windows XP, but 486 was capable of doing so but at a speed. Soon Pentium appeared and other manufacturers too started manufacturing microprocessors equivalent to Pentium processors. There are certain software which are put on ROM (Read Only Memory) chips and made hardwired. These are essentially software that hardwired permanently to that hardware and are known as firmware.

Most of the software are upward compatible and advanced versions support the earlier ones. But the converse is not true.

Data Conversion Strategy

In conversion strategy, strategy related to data conversion is described. All the factors related to assurance of data quality and controls in data conversion are to be clearly mentioned.

Data Conversion Approach

This part of the plan describes preparation of some specific data for use in the system conversion. For transferring data from the original system, a detailed description of procedures adopted for this purpose should be described. If machine-readable media is used for such transfer, their characteristics should be described.

Interfaces

If a user decides to change the hardware platform, then such conversion, for example, from main frame to client-server or from mini frame to client-server platform may require re-engineering. Thus, in the section describing such conversion affected interfaces are to be mentioned and revisions required in each should be clearly spelt out.

13.3.1 Conversion Plan

The conversion plan includes a description of all the activities that must occur to implement the new system and put it into operation. It identifies the persons

NOTES

responsible for each activity and includes a timetable indicating when each activity will occur.

During the pre-implementation stages, when the conversion is being planned, analysts should assemble a list of all tasks, including the following:

1. List all files for conversion.
2. Identify all data required to build new files during conversion.
3. List all new documents and procedures that go into use during conversion.
4. Identify all controls to be used during conversion. Establish procedures for crosschecking the old and new systems. Determine how team members will know if something has not been completed properly.
5. Assign responsibility for each activity.
6. Verify conversion schedules.

Possible issues and ways to deal with them should be expected in the conversion plan. Missing records, mixed data formats between existing and new files, data conversion errors, missing data or misplaced files, and circumstances that were ignored during system development are among the most commonly occurring issues. The conversion manager is expected to guard against the omission of conversion measures. A checklist will avoid missing moves. Personnel absences must also be planned and appropriate fallback plans stated.

Conversion scheduling is difficult, as there are so many facets of the conversion, ranging from equipment installation to type and supply ordering.

13.3.2 Operating Plan

The operating plan is checking of all arrangements. It includes reviewing conversion plans, verifying the delivery of equipment, software, forms, preparing the site and preparing the data and files.

1. **Site Preparation:** Analysts often work with vendor personnel to outline site – preparation guidelines. Due importance should be paid to electrical using, air conditioning needs, humidity controls, space requirements, etc.
2. **Data and File Preparation:** Before the regular running of the system, master files and system files need to be entered into the system for a new system to start. Generally, master files are manually generated. The number of records in the old master file of the system should correspond to the number of records in the new master file.

13.4 POST IMPLEMENTATION REVIEW

NOTES

In-process reviews are informal reviews intended to be held during the conduct of each SDLC phase. Informal review implies that there is little reporting to the customer on the results of the review. They are often called peer reviews because they are usually conducted by the producer's peers. The scheduling of the reviews, while intentional and a part of the overall project plan, is rather flexible. This allows for reviews to be conducted when necessary earlier if the product is ready and later if the product is late. One scheduling rule is used to review that the producer is willing to throw away. Another rule is used to have an in-process review every two weeks. Phase end reviews are formal reviews that usually occur at the end of the SDLC phase or work period and establish the baselines for work in succeeding phases. Unlike the in-process reviews which deal with a single product phase end reviews include examination of all the work products that are to be completed during that phase. A review of project management information, such as budget, schedule and risk are considered as important factors. For example, the Software Requirements Review (SRR) is a formal factor of the requirements document and sets the baseline for the activities in the design phase to follow. It should also include a review of the system test plan, a draft of the user documentation, perhaps an interface requirements document, quality plans, safety and risk plans and the actual status of the project with regard to budget and schedule plans. In general, anything expected to have been produced during that phase is to be reviewed. The participants include the producer and software quality practitioner as well as the user or customer. Phase-end reviews are a primary means of keeping the user or customer aware of the progress and direction of the project. A phase end review is not considered finished until the action items have been closed, software quality has approved the results and the user or customer has approved going ahead with the next phase. These reviews permit the user or customer to verify that the project is proceeding as intended or to give redirection as needed. They are also major reporting points for software quality to indicate the management how the project is used to adhere the standards, requirements and resource budgets. Phase end reviews should be considered go or no-go decision points. At each phase end review, there should be sufficient status information available to examine the project's risk including its viability. Two formal audits, the Functional Audit (FA) and the Physical Audit (PA) are held. These two events, held at the end of the SDLC (Software Development Life Cycle), are the final analysis of the software product to be delivered against its approved requirements and its current documentation respectively.

NOTES

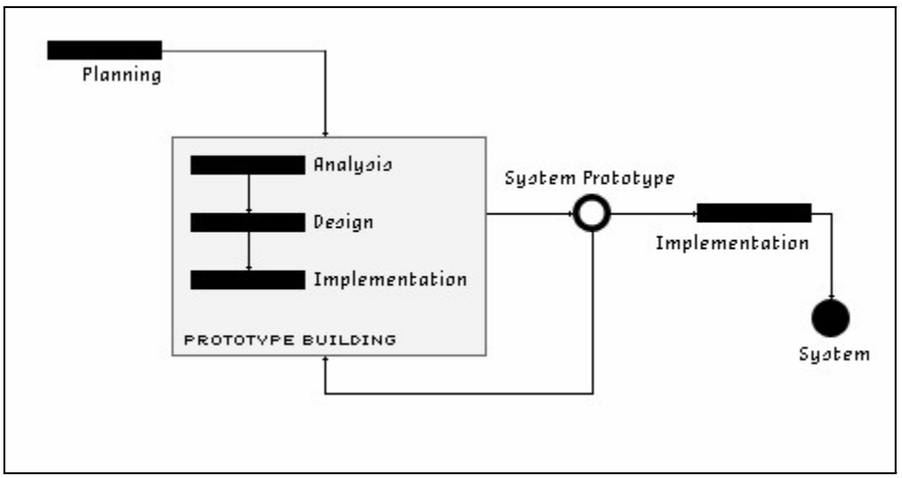


Fig. 13.1 *Implementation Phase in SDLC*

Figure 13.1 represents the implementation phase of SDLC in which system planning is sent for prototype building. The concept of prototyping is used to make system prototype which refers to analysis, design and implementation phase. The FA compares the software system being delivered against the currently approved requirements for the system. This is usually accomplished through an audit of the test records. The PA is intended to assure that the full set of deliverables is an internally consistent set, i.e., the user manual is the correct one for this particular version of the software. The PA relies on the records for the delivered products. The software quality practitioner frequently is charged with the responsibility of conducting these two audits. In any case, the software quality practitioner must be sure that the audits are conducted to find the audits to management. The Post Implementation Review (PIR) is held once the software system is in production. The PIR usually is conducted six to nine months after implementation. Its purpose is to determine whether the software has met the user's expectations for it in actual operation. Data from the PIR is intended for use by the software quality practitioner to help improve the software development process. Table 13.1 summarizes some of the approaches used in PIR.

Table 13.1 *Approaches used in Post Implementation Review*

Approaches	**3 To 6 Months after Software System Implementation**
Software system goals versus experience	This approach supports return on investment, schedule results, user response and defect history.
Review results usage	This approach supports input values for process analysis.

The role of the software quality practitioner is often that of conducting the PIR. This review is best used as an examination of the results of the development process.

Problems with the system as delivered can point to opportunities to improve the development process and verify the validity of previous process changes.

A Post-Implementation Evaluation Review (PIER) enables you to verify that the project has achieved the desired outcome within specified time period and planned cost. Once a project is implemented you need to evaluate the performance of the system in-order to determine the success of the project. The post implementation review ensures that the project has met its objectives by evaluating the development and management processes of the project. It also enables you to compare actual expenditures on the project with that to the estimated expenditure of a project.

A PIR highlights the strong and weak points of a project. Following are the objectives of a PIR:

- To demonstrate the success of a project against the projected costs, benefits and timelines in the business case.
- To identify the opportunities to add additional value to the project.
- To identify strengths and weaknesses of the project for future reference and appropriate action.
- To make recommendations on the future of the project.

Purpose of PIER

A PIER (Post-Implementation Evaluation Review) enables you to evaluate and measure the value gained by the implementation of the project. PIER is conducted after implementation of a project to observe whether the implementation supports the strategic objectives. The PIER provides reasons for the success or failure of a project. It is a tool that enables you to determine whether the project is yielding the expected benefits to the processes, products or services supported by the agency. It provides a standard approach to evaluate the outcome of the project.

PIER helps in analysing actual expenditures with that of budgeted expenditures. This enables the project team to improve future project estimates by refining the cost estimating techniques. This type of systematic and formal evaluation creates a common language that can be used to communicate performance of a project to all interested and involved parties.

Post Implementation Review Process Description

The post-implementation review process begins during the initial stages of the project development. The project team and the primary users of the project define the performance measures and benefits expected at the time of planning of the project. The approving authority then approves these performance measures and expected benefits. Finally, the project team ensures that appropriate measures are implemented so that project performance can be maintained throughout the life cycle of the project.

NOTES

Upon completion of a major project, they prepare a post implementation evaluation review. The project manager is responsible for implementation process. Following are the important points that are considered by the project manager at the time of implementation:

- Whether all the recommendations are implemented correctly.
- Whether all the recommendations are effective.
- Whether the recommendations address any new issues that may have arisen at the time of their implementation.

The project manager ensures that the performance measures are gathered prior to implementation and again three to six months after implementation. The post implementation allows a more valid comparison to the earlier measurements. These project measures report the actual benefits received from the Information System (IS). Final project measurements are reported to the Quality Assurance (QA) team. Project manager also ensures that appropriate staff and management participate in the post implementation review. Following are the staff members that should be included in the review process:

Project Team and Management

Project team focuses on the changes and improvements that can be incorporated in the future projects.

User Staff

User staff focuses on the comparison of performance measures to the initial measurements and determine if the project realized the planned performance.

Strategic Management

Strategic management focuses on the benefits derived by the system. It is responsible for approving the project and ensures that the project has actually achieved the benefits.

External Users

External users focus is on service. They participate to measure the benefits that apply to them.

Elements of a Post Implementation Review

The following are the elements of post implementation review:

Project History Description

Provides an overview of the project objectives and technical solution selected to satisfy those objectives. Project description describes changes in the original design

that took place during the development of a project. It contains an overview of the dates of the projects that highlights turning points in the development and implementation of the system.

NOTES

Background

Provides a brief summary of the project and the situations that led to the implementation of the project.

Cost History

Provide an estimate of the total costs incurred in the development of the project by using the various cost schedule formats. Cost history also includes the differences between the actual and planned costs and the reason for those differences. The explanation of the difference between the actual cost and the estimated cost contains the description about how the cost changed, the influence of those changes and cause of such changes etc.

Project Management and Systems Development Methodology Descriptions

It includes the description about the parts of the systems development methodology that were used and those that were not used and explain the impact of these decisions. These descriptions include the comparison of the initially planned dates and the actual delivery dates and the reason and impact for the difference in the schedules.

Performance Measures

Provide a list of the measures taken to justify the performance of a project. It provides comparison of the initial performance measures to that of actual performance of the project. It also includes the impact of the project on the organisation.

Experiences

Provides the important information gather during the implementation of the project. These experiences provide help in the future planning and development process of a project. Following are some of the information included in the experiences:

- The project management process.
- The systems development process.
- The contracting methodology used.
- The training received or provided.
- The conversion tasks.

NOTES

- Other parts of the project development process that appear to need changing for future projects.
- The technology/hardware/software that was used.
- Improvements in the project team.

Audit Issues

- Provides description about the existing control and security measures taken to evaluate their adequacy.

Impact of the System

Describes the impact of the project on the users such as managers and external users. Provide feedback to the management about the success and failures of the system. It also includes the feedback from the user about the performance of the system.

Check Your Progress

6. Define the term conversion.
7. State about the direct conversion and direct cutover conversion methods.
8. Elaborate on the term hardware conversion strategy.
9. Explain the concept of conversion plan.
10. Define the term operating plan.
11. How does PIER help?
12. What does project description describe?

13.5 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. The success of the training sessions are follow as:
   (i) The company must hire high-quality Information Technology (IT) personnel.
   (ii) The hiring process should be fast, focussed, decisive and candid.
2. Technically elegant systems can succeed or fail because of the way they are operated and used. Therefore, the quality of training received by the personnel involved with the system in various capacities helps or hinders, and may even prevent, the successful implementation of an information system. Those whose will be associated with or affected by the system must know in detail what their roles will be, how they can use the system, and what the system will or will not do. Both systems operators and users need training.

NOTES

3. Training method involves trainers as well as trainees who are required to meet at the same time, but not necessarily at the same place. Two broad categories of this type of training are: virtual and normal classrooms.
4. The features of a normal classroom are as follows:
   - The trainers and trainees must meet at the same time and at the same place.
   - The tools used to provide training are the same as those used in the normal classroom (i.e., blackboard, overhead projectors, Liquid Crystal Display (LCD) projector, etc.).
   - The training session could be either collaborative or one-on-one.
5. Follwing are the demerits of web-based trainig:
   - The medium has its limitations and the courses are still in the early phase of development. The developer will, therefore, be forced to minimize the video context and use a text-based style for the instructions.
   - On its own, this technology cannot substitute instructor-led training. Therefore, the training should be given conservatively.
   - There is no simple manner of conducting a test of objectives of the courses.
6. Conversion is a method of replacing the old software with a new one. A conversion plan includes a description of all those activities which must occur during the implementation of a new system.
7. In direct conversion, the new system is implemented and the old system is totally replaced. This method is cheap and fast, but is quite hazardous. The direct cutover method converts from the old to the new system abruptly. As mentioned earlier, testing never uncovers all problems. With the direct cutover.
8. Hardware conversion strategy is very important as it requires working on system hardware. All hardware may not support all kinds of software. More powerful microprocessors are developed now a days and a clear-cut strategy for the conversion of system hardware is to be decided. The conversion strategy should be such that the existing system does not stop functioning or need not be brought to a halt. For a particular period, it may be required to work in parallel and even make a comparison between these two systems.
9. The conversion plan includes a description of all the activities that must occur to implement the new system and put it into operation. It identifies the persons responsible for each activity and includes a timetable indicating when each activity will occur.
10. The operating plan is checking of all arrangements. It includes reviewing conversion plans, verifying the delivery of equipment, software, forms, preparing the site and preparing the data and files.

NOTES

11. PIER (Post-Implementation Evaluation Review) helps in analysing actual expenditures with that of budgeted expenditures. This enables the project team to improve future project estimates by refining the cost estimating techniques.
12. Project description describes changes in the original design that took place during the development of a project. It contains an overview of the dates of the projects that highlights turning points in the development and implementation of the system.

13.6 SUMMARY

- An essential part of the computer-based information system development is end-user training.
- Many training methods are being made use of in organizations presently, both instructor-led and self-paced. Most of them are aimed at providing the end-users with the required skills.
- The managers must then build trust within the organization by not only providing workplace security but also changing the attitude of the management.
- The task of designing the information system curricula and the training programmes can be given to the Electronic Data Processing (EDP) centre along with outside companies or academic institutions.
- Once high-quality, sincere and hard working employees are in place, the IT department managers must realize that these employees need to be continuously educated. Therefore, it is important to honour the commitments about the education of employees.
- Training courses can be successful if the managers see to it that the users focus on the advantages of their hard work in training, as well as attempt to prevent the negative feelings of the users toward the new system.
- Delayed trainings may also result in the delivery of problems at the help desk or other employees, which may be costlier thant the actual training.
- The training strategies should be determined by the trainer and the trainee. The analyst ensures that a person whose work is affected by the new information system is properly trained.
- Training also involves familiarization with run procedures, which involves working through the sequence of activities needed to use a new system on an ongoing basis.
- User training may involve equipment use, particularly in the case where, say, a microcomputer is in use and the individual involved is both operator and user.

NOTES

- User training must also instruct individuals in troubleshooting the system, determining whether a problem that arise is caused by the equipment or software or by something they have done in using the system.
- Data – handling activities receiving the most attention in user training are adding data (how to store new transactions), editing data (how to change previously stored data), formulating inquiries (finding specific records or getting responses to questions) and deleting records of data.
- Training method involves trainers as well as trainees who are required to meet at the same time, but not necessarily at the same place. Two broad categories of this type of training are: virtual and normal classrooms.
- The trainees involved in face-to-face training will have less interest and perceived learning than those who participate in a computer-based training.
- This method of training involves both trainers and trainees, who do not require to meet at the same time or at the same place.
- Conversion is a method of replacing the old software with a new one. A conversion plan includes a description of all those activities which must occur during the implementation of a new system.
- In parallel conversion, both the old and the new system are used simultaneously. This method has two advantages.
- First, the continued use of the old system provides a fallback when the new system fails. Any new information system, no matter how exhaustively it has been tested, may fail or have problems shortly after implementation. The old system provides business continuation capabilities while the new system is being modified.
- Second, this method serves as an ultimate test for the new system. Parallel conversion is typically conducted for at least two accounting cycles.
- In direct conversion, the new system is implemented and the old system is totally replaced. This method is cheap and fast, but is quite hazardous. The direct cutover method converts from the old to the new system abruptly. As mentioned earlier, testing never uncovers all problems. With the direct cutover.
- The phase-in method is used when it is not possible to install a new system throughout the organization at once.
- A project status report is generated every fortnight and discussed with the client project manager. An assessment of the results of the reviews and inspections confirms the progress.
- When new systems also involve new techniques or drastic changes in organization performance, this approach is preferred.
- There are different versions of the same software, whether language or database system.

NOTES

- To carry out a work where so many factors get involved, it is always better to follow a plan for conversion.
- This strategy is very important as it requires working on system hardware. All hardware may not support all kinds of software. More powerful microprocessors are developed now a days and a clear-cut strategy for the conversion of system hardware is to be decided. The conversion strategy should be such that the existing system does not stop functioning or need not be brought to a halt. For a particular period, it may be required to work in parallel and even make a comparison between these two systems.
- Hardware is not much varied as compared to software. Each hardware or peripheral comes with its driver software.
- In conversion strategy, strategy related to data conversion is described. All the factors related to assurance of data quality and controls in data conversion are to be clearly mentioned.
- For transferring data from the original system, a detailed description of procedures adopted for this purpose should be described. If machine-readable media is used for such transfer, their characteristics should be described.
- If a user decides to change the hardware platform, then such conversion, for example, from main frame to client-server or from mini frame to client-server platform may require re-engineering.
- The conversion plan includes a description of all the activities that must occur to implement the new system and put it into operation. It identifies the persons responsible for each activity and includes a timetable indicating when each activity will occur.
- The operating plan is checking of all arrangements. It includes reviewing conversion plans, verifying the delivery of equipment, software, forms, preparing the site and preparing the data and files.
- Analysts often work with vendor personnel to outline site – preparation guidelines. Due importance should be paid to electrical using, air conditioning needs, humidity controls, space requirements, etc.
- Generally, master files are manually generated. The number of records in the old master file of the system should correspond to the number of records in the new master file.
- In-process reviews are informal reviews intended to be held during the conduct of each SDLC phase. Informal review implies that there is little reporting to the customer on the results of the review.
- The scheduling of the reviews, while intentional and a part of the overall project plan, is rather flexible.

NOTES

- Phase end reviews are formal reviews that usually occur at the end of the SDLC phase or work period and establish the baselines for work in succeeding phases.
- A review of project management information, such as budget, schedule and risk are considered as important factors.
- The participants include the producer and software quality practitioner as well as the user or customer. Phase-end reviews are a primary means of keeping the user or customer aware of the progress and direction of the project.
- The PA is intended to assure that the full set of deliverables is an internally consistent set, i.e., the user manual is the correct one for this particular version of the software.
- The PA relies on the records for the delivered products. The software quality practitioner frequently is charged with the responsibility of conducting these two audits.
- Data from the PIR is intended for use by the software quality practitioner to help improve the software development process.
- A Post-Implementation Evaluation Review (PIER) enables you to verify that the project has achieved the desired outcome within specified time period and planned cost.
- A PIER (Post-Implementation Evaluation Review) enables you to evaluate and measure the value gained by the implementation of the project. PIER is conducted after implementation of a project to observe whether the implementation supports the strategic objectives.
- PIER helps in analysing actual expenditures with that of budgeted expenditures. This enables the project team to improve future project estimates by refining the cost estimating techniques.
- The post-implementation review process begins during the initial stages of the project development.
- Project team focuses on the changes and improvements that can be incorporated in the future projects.
- User staff focuses on the comparison of performance measures to the initial measurements and determine if the project realized the planned performance.
- Strategic management focuses on the benefits derived by the system. It is responsible for approving the project and ensures that the project has actually achieved the benefits.
- External users focus is on service. They participate to measure the benefits that apply to them.

NOTES

- Project description describes changes in the original design that took place during the development of a project. It contains an overview of the dates of the projects that highlights turning points in the development and implementation of the system.
- Provide feedback to the management about the success and failures of the system. It also includes the feedback from the user about the performance of the system.

13.7 KEY WORDS

- **Instructor-led:** It is a training method that involves trainers as well as trainees who are required to meet at the same time but not necessarily at the same place.
- **Conversion:** It is the process of replacing the old software with a new one.
- **Post-Implementation Evaluation Review (PIER):** A Post-Implementation Evaluation Review (PIER) enables you to verify that the project has achieved the desired outcome within specified time period and planned cost.
- **Strategic management:** Strategic management focuses on the benefits derived by the system. It is responsible for approving the project and ensures that the project has actually achieved the benefits.
- **Cost history:** Cost history provides an estimate of the total costs incurred in the development of the project by using the various cost schedule formats.

13.8 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. Explain the term training.
2. Define the significance of training systems operators.
3. Write the features of virtual classroom.
4. What are the features of multimedia training?
5. Give the definition of parallel conversion.
6. Write the types of conversion.
7. Define the term software conversion strategy.
8. Differentiate between conversion plan and operating plan.
9. Elaborate on the term strategic management.
10. Write the elements of a post implementation review.

NOTES

Long-Answer Questions

1. Briefly explain the terms training and training strategies with the help of appropriate examples.
2. To what extent training is important? Support your answer with the help of examples.
3. Compare the different conversion methods giving merits and demerits of each type.
4. Briefly explain the conversion strategy.
5. Describe the implementation phases in SDLC with the help of diagram.
6. Briefly explain the post implementation review process giving appropriate examples.
7. Explain the characteristic features of the following:
   a) Virtual classroom
   b) Normal classroom
   c) Multimedia
   d) Web-based
   e) Conversion plan
   f) Operating plan

13.9 FURTHER READINGS

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.

UNIT 14 SYSTEM MAINTENANCE

NOTES

Structure

14.0 Introduction
14.1 Objectives
14.2 System Maintenance
14.3 Hardware and Software Selection
14.3.1 Computer Hardware
14.3.2 Computer Software
14.4 Procedure for Hardware and Software Selection
14.5 Answers to Check Your Progress Questions
14.6 Summary
14.7 Key Words
14.8 Self Assessment Questions and Exercises
14.9 Further Readings

14.0 INTRODUCTION

A major element in building systems is selecting compatible Hardware & software. The kind of hardware & peripherals required is to be determined. The suitable software has to be selected. Comparisons are often made among different computer systems on the basis of actual performance data. Using benchmark data, generated by using synthetic programs, is more effective than simply comparing technical specifications. Software is classified as system software for controlling computer operations and application software for solving user oriented problems.

Successful implementation of system analysis and design requires a rigorous process of hardware and software selection. Hardware selection in the context of system analysis and design includes the determination of size and capacity requirements, such as internal memory size, processing speed, number of Input/ Output (I/O) channels, characteristics of display and communication components, availability of system supports and utility and type of auxiliary storage unit that can be attached. It also includes the comparison of various benchmarks, consideration of various financial aspects, and maintenance and support.

In this unit, you will study about the corrective, adaptive, hardware and software selection, computer hardware, computer software, and procedure of hardware and software selection.

14.1 OBJECTIVES

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

- Understand what system maintenance is

- Explain about the corrective and adaptive system maintenance
- Define about the hardware and software selection
- Discuss about the procedure of hardware and software selection
- Know about the evaluation process and financial considerations

NOTES

14.2 SYSTEM MAINTENANCE

Many factors directly or indirectly lead to high maintenance costs. A software maintenance framework is created to determine the affects of these factors on maintenance. This framework comprises user requirements, organizational and operational environments, maintenance process, maintenance personnel and the software product. These elements interact with each other based on three kinds of relationships, which are listed below:

- ***Relationship of Software Product and Environment:*** In this relationship, the software product changes according to the organizational and operational environment. However, it is necessary to accept only those changes which are useful for the software product.
- ***Relationship of the Software Product and User:*** In this relationship, the software product is modified according to the new user requirements. Hence, it is important to modify the software that is useful and acceptable to users after modification.
- ***Relationship of Software Product and Software Maintenance Team:*** In this relationship, the software maintenance team members act as mediators to keep track of the software product. In other words, the software maintenance team analyses the modifications in other elements of software maintenance framework to determine their effect on a software product. These elements include user requirements, organizational and operational environments, and the software maintenance process. All these elements affect the modifications in software and are responsible for maintaining software quality.

User Requirements

Generally, users have little knowledge of the software maintenance process due to which they can be unsupportive of the software maintenance team. Also, users may have some misconceptions, such as that software maintenance is like hardware maintenance, changing software is easy and changes cost too much and are time-consuming.

NOTES

If user requirements need major changes in the software, a lot of time may be consumed in implementing them. Similarly, users may opt for changes that are not according to the software standards or policies of a company. This situation creates a conflict between users and the software maintenance team.

To implement user requirements in a software, the following characteristics should be considered.

- ***Feasible:*** User requirements are feasible if the requested change is workable in the software system.
- ***Desirable:*** Before implementing new changes, it is important to consider whether the user modification request is necessary or not.
- ***Prioritized:*** In some cases, the user requirements may be both feasible and desirable. However, these requirements may not be of high priority at that time. In such a situation, the user requirements can be implemented at a later date.

Organizational and Operational Environment

The working of a software is affected by two kinds of environments, namely, organizational environment and operational environment. The **organizational environment** includes business rules, government policies, taxation policies, work patterns, and competition in the market. An organization has its own business rules and policies, which should be incorporated in the software maintenance process. The **operational environment** includes software systems, such as operating systems database systems and compilers and hardware systems such as processor, memory, peripherals.

In both the environments, scheduling of the maintenance process can create problems. The scheduling is affected by various factors, such as urgent requirement of the modified software, allocation of less amount of time to modify the software and lack of proper knowledge by the software maintenance team on how to implement user requirement in a software.

Maintenance Process

Changes are implemented in the software system by following the software maintenance process (also known as software maintenance life cycle). The facts of a maintenance process which affect the evolution of a software or contribute to high maintenance costs are as follows:

- ***Error Detection and Correction:*** It has been observed that error free software is virtually non-existent. That is, a software product tends to contain some kind of 'residual' errors. If these errors are uncovered at a later stage

of software development, they become more expensive to fix. The cost of fixing errors is even higher when errors are detected during the maintenance phase.

- ***Difficulty in Capturing Change and Changing Requirements:*** Requirements and user problems become clear only when a system is in use. Also users may not be able to express their requirements in a form which is understandable to the analyst or programmer.
- ***Software Engineering Paradigm Shift:*** Older systems that were developed prior to the advent of structured programming techniques may be difficult to maintain.

Software Product

The software developed for users can be for general use or specific use. For example, Microsoft Office is a software application that is generic in nature and may be used by a wide number of people. Similarly, the pay roll system may be customized according to the needs of the organization. However, the problem occurs when a software is to be maintained. Generally, the aspects of a software product that contribute to the maintenance cost/challenge are:

- ***Difficulty of the Application Domain:*** The requirements of applications that have been widely used and well understood are less likely to undergo substantial modifications than those that have been recently developed.
- ***Inflexibility in Programs:*** While modifying a software, it should be checked for flexibility of change and reuse. This is because the inflexible software products are more prone to failures.
- ***Quality of the Documentation:*** Documentation is essential for understanding the requirements, software design, and how the requirements are to be implemented in the software code. The lack of up-to-date systems documentation affects maintenance productivity adversely.

Software Maintenance Team

The group of individuals who are involved in the maintenance of the software system are referred to as the software maintenance team which may or may not comprise the development team that 'built' the software. Often, a separate maintenance team (comprising analysts, designers and programmers) is formed to ensure that a system performs its functions properly. This team is employed, as it has been observed that generally developers do not keep documentation up-to-date, leading to the need for more individuals or resources to tackle a problem. This results in a long time gap between the time when a problem occurs and when it is fixed.

NOTES

Various functions performed by the software maintenance team are:

- Locating information in system documentation.
- Keeping system documentation up-to-date.
- Extending existing functions to accommodate new or changing requirements.
- Adding new functions to the system.
- Finding the source of system failures or problems.
- Managing changes to the system as they are made.

The aspects of a maintenance team that lead to high maintenance costs are:

- ***Staff Turnover:*** Generally, it is observed that when the staff turnover (ratio of number of individuals that leave the organization during a specified period of time) is high, the software maintenance is not performed properly. This is because employees who originally worked on software products are replaced by new personnel who spend a substantial proportion of the maintenance effort in understanding the system.
- ***Domain Expertise:*** Sometimes, the maintenance team may have little or no knowledge about the system domain and application domain they are working in. This problem is worsened if documentation is not maintained or is not up- to-date. All this may lead to delay in implementing the changes requested by the user.

Types of Software Maintenance

There are four types of maintenance, namely, corrective, adaptive, perfective and preventive. **Corrective** maintenance is concerned with fixing reported errors in the software. **Adaptive** maintenance means changing the software to some new environment, such as adapting a new version of an operating system. **Perfective** maintenance involves implementing new functional or non-functional requirements. **Preventive** maintenance involves implementing changes to prevent occurrence of errors. Figure 14.1 shows the distribution of types of maintenance by type and by percentage of time consumed.

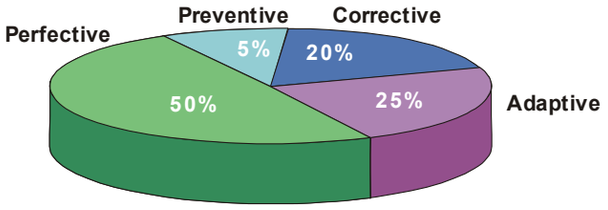


Fig. 14.1 *Types of Software Maintenance*

Corrective Maintenance

Corrective maintenance deals with the repair of faults or defects found in day-to-day system functions. A defect can result due to design errors, logic errors and

NOTES

coding errors. For example, design errors occur when changes made to the software are incorrect, incomplete, wrongly communicated or the change request is misunderstood. Similarly, logic errors result from invalid tests and conclusions, incorrect implementation of design specifications, faulty logic flow or incomplete test of data. All these errors, referred to as residual errors, prevent the software from conforming to its agreed specification. Note that the need for corrective maintenance is usually initiated by bug reports drawn by the users.

In the event of a system failure due to an error, actions are taken to restore operation of the software system. The approach in corrective maintenance is to locate the original specifications in order to determine what the system was originally designed to do. However, due to pressure from management, the maintenance team sometimes resorts to emergency fixes known as 'patching'. As shown in Figure 14.1, corrective maintenance accounts for 20% of all maintenance activities.

Adaptive Maintenance

Adaptive maintenance is the implementation of changes in a part of the system, which has been affected by a change that occurred in some other part of the system. Adaptive maintenance consists of adapting software to changes in the environment, such as the hardware or the operating system. The term environment in this context refers to the conditions and influences which act (from outside) on the system. For example, business rules, work patterns and government policies have a significant impact on the software system.

For example, a government policy uses a single 'European currency' will have a significant effect on the software system. An acceptance of this change will require banks in the various member countries to make significant changes in their software systems in order to accommodate this currency. As shown in Figure 14.1, adaptive maintenance accounts for 25% of all maintenance activities.

Perfective Maintenance

Perfective maintenance mainly deals with implementing new or changed user requirements. Perfective maintenance involves making functional enhancements to the system and activities to increase the system's performance even when the changes have not been suggested by faults. This includes enhancing both function and efficiency of the code and changes, insertions, deletions, modifications, extensions and enhancements made to a system to meet the evolving and/or expanding needs of the user.

Examples of perfective maintenance include modifying the payroll program to incorporate a new union settlement and adding a new report in the sales analysis system. As shown in Figure 14.1, perfective maintenance is the largest user of maintenance activities and accounts for 50 per cent of all maintenance activities.

NOTES

Preventive Maintenance

Preventive maintenance is the maintenance performed for the purpose of preventing problems before they occur. The objective is to make programs easier to understand. This helps in changing the software to improve its future maintainability or to provide a better base for future enhancements. Preventive maintenance includes activities that are aimed at increasing the system's maintainability, such as updating documentation, adding comments and improving the modular structure of the system. Examples of preventive change include restructuring and optimizing code.

Preventive software maintenance requires an investment of resources in the system. The return on that investment is often difficult for customers to appreciate, but as Lehman's second law states, the more a system is subjected to maintenance, the more its structure will degrade. This results in a system which is more difficult to understand and ultimately more expensive to change. As shown in Figure 14.1, preventive maintenance accounts for only 5 per cent of all maintenance activities.

Software Maintenance Life Cycle

As stated earlier, changes are implemented in the software system by following a software maintenance process, which is known as Software Maintenance Life Cycle (SMLC). This life cycle comprises seven phases, namely, problem identification, analysis, design, implementation, system testing, acceptance testing and delivery phase (refer Figure 14.2).

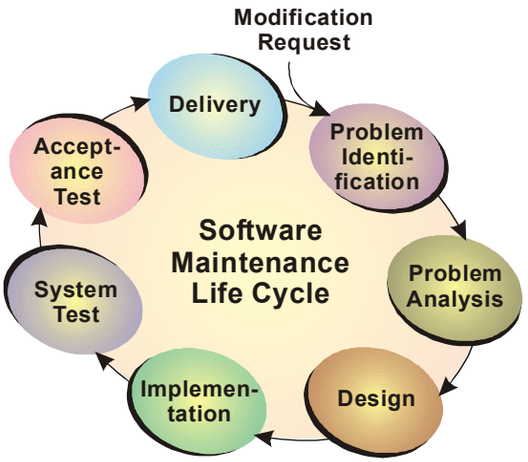


Fig. 14.2 *Phases of SMLC*

All the phases in SMLC evolve through attributes: **input, process, control** and **output**. All these attributes are listed in Table 14.1.

Table 14.1 *Attributes of SMLC*

Phases	Input	Process	Control	Output
Problem identification phase	• Modification request	• Assign change number • Classify modification request • Accept or reject change • Prioritize	• Uniquely identified modification request • Enter modification request in repository	• Validated modification request • Validated process determinations
Analysis phase	• Project document • Repository information • Validated modification request	• Feasibility analysis • Detailed analysis	• Conduct technical review • Verify test strategy • Verify whether the documentation is updated or not • Identify security issues	• Feasibility report • Detailed analysis report • Updated requirements • Preliminary modification list • Test strategy
Design phase	• Project document • Source code • Databases • Analysis phase output	• Create test cases • Revise requirements • Revise implementation plan	• Software inspections/ reviews • Verify design	• Revised modification list • Revised detailed analysis • Updated test plans
Implementation phase	• Source code • System documentation • Results of design phase	• Software code • Unit test • Test preparation review	• Software inspections/ review	• Updated software • Updated design documents • Updated test documents • Updated user documents • Test preparation review report
System test phase	• Updated software documentation • Test preparation review report • Updated system	• Functional test • Interface testing • Test preparation review	• Software code listings • Modification request • Test documentation	• Test system • Test reports
Acceptance test phase	• Test preparation review report • Fully integrated system • Acceptance test plans • Acceptance test cases • Acceptance test procedures	• Acceptance test • Interoperability test	• Acceptance test	• Acceptance test report
Delivery phase	• Tested/accepted system	• Installation • Training	• Version description document	• Version description document

NOTES

NOTES

Problem Identification Phase

In this phase, software modifications are identified, classified and assigned an initial priority ranking. Each Modification Request (MR) is evaluated to determine its classification and handling priority. To determine classification, types of maintenance (corrective, adaptive, perfective and preventive) are used.

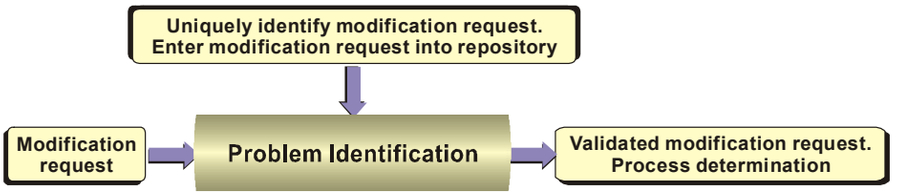


Fig. 14.3 *Problem Identification Phase*

As shown in Figure 14.3, the *input* attribute comprises modification request. If changes are required in the software, then the following set of activities are performed in the *process* attribute:

- Assigning an identification number.
- Classifying the type of maintenance.
- Analyzing the modification to determine whether to accept, reject or evaluate further.
- Estimating the modification size/magnitude.
- Prioritizing the modification.
- Assigning an MR to a block of modifications that are scheduled for implementation.

After the process attribute begins the *control* attribute which uniquely determines the identified MR and enters it into a repository. This is because there are different kinds of software maintenance requests and each modification required in a software system needs to be identified and stored according to its type, priority, and so on. Repository is a place of storage and consists of components, such as statement of problem or modification request, requirement evaluation, type of software maintenance, initial priority and an estimate of resources required in software maintenance. The *output* of this phase is a validated MR and the process determinations that are stored in a repository.

Problem Analysis Phase

The analysis phase uses the repository information and the validated MR, along with system and project documentation, to study the feasibility and scope of the modification and devise a preliminary plan for design, implementation, test and delivery. As shown in Figure 14.4, the input attribute comprises validated modification request, initial estimate of resources, project documentation and repository information.

NOTES

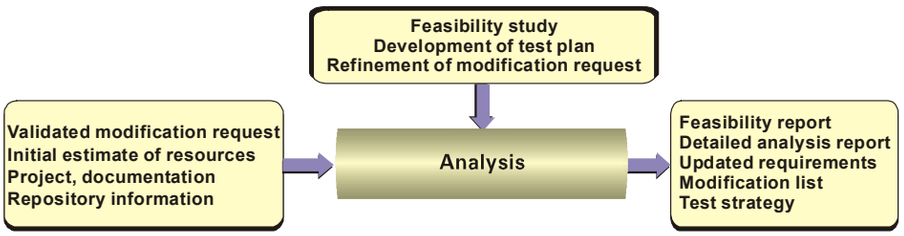


Fig. 14.4 *Problem Analysis Phase*

The process attribute comprises two components, namely, a feasibility analysis and a detailed analysis. A **feasibility analysis** is performed for an MR and a feasibility report is prepared, which contains the following information:

- Impact of the changes.
- Alternating solutions, including prototyping.
- Analysis of conversion requirements.
- Safety and security implications.
- Human factors.
- Short term and long term costs.

Detailed analysis serves the following purposes:

- Defining firm requirements for modification.
- Identifying the elements of modification.
- Identifying safety and security issues.
- Devising a test strategy.
- Developing an implementation plan.

The control attribute of this phase concentrates on review of the project documentation and review of the modification request to evaluate technical and economic feasibility. In addition, this attribute verifies that project documentation is updated according to the analysis of modification request. The time and resource estimates are verified for their accuracy in control attribute. The output of this phase comprises elements, such as feasibility report for modification request, detailed analysis report, updated requirements, modification list and test strategy.

Design Phase

In the design phase, all current system and project documentation, existing software and databases and the output of the analysis phase will be used to design the modification in the system. As shown in Figure 14.5, the input attribute comprises outputs produced by analysis phase, system and project documentation, existing source code, comments and databases.

NOTES

The *process* attribute for design comprises the following steps:

1. Identifying affected software modules.
2. Modifying software module documentation (like data flow diagrams and program design language).
3. Creating test cases for the new design, including safety and security issues.
4. Creating regression tests.
5. Identifying documentation (system/user) to update requirements.
6. Updating modification list.

In the design phase, *control* attribute performs the following functions:

- Conducting software inspection of the design in compliance with IEEE standard.
- Verifying that the new design/requirement is documented.
- Verifying the inclusion of new design material including safety and security issues.
- Verifying that the appropriate test documentation has been updated.

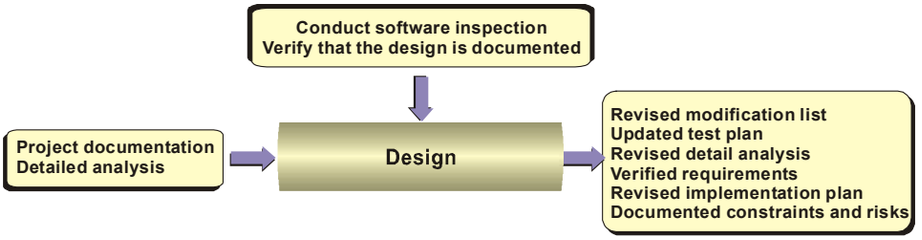


Fig. 14.5 *Design Phase*

The *output* of this phase is a revised modification list, updated test plans, revised detailed analysis, verified user requirements, list of constraints and risks in implementation. The output provides information of detailed analysis of user requirements along with the implementation plan. This plan lists the tasks to be implemented in the implementation phase of SMLC. The design test plans are prepared according to the updated design. Generally, the quality factors that are considered before developing the software design include flexibility of design and reusability of the software code.

Implementation Phase

The implementation phase is concerned with making actual modifications in the software code, adding new features that support the specification of present software and finally installing the modified software. As shown in Figure 14.6, the

input attribute comprises output of the design phase, the current source code and the updated project and system documentation.

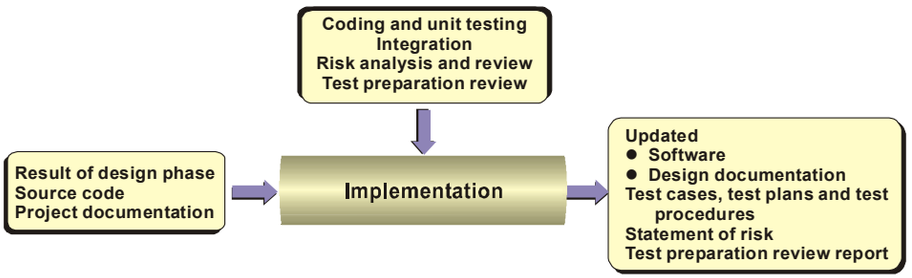


Fig. 14.6 *Implementation Phase*

The *process* attribute comprises the following subprocesses:

- ***Coding and Unit Testing:*** The user requirements as identified in design phase are implemented in the form of a software code. After the software code is written, unit testing on individual modules is performed.
- ***Integration:*** When the software code is written and all modules are tested, the modified software is integrated with the system. After integration of the software system, integration test and regression test are performed. These tests are conducted to identify the impact on the functionality, performance, usability, and security of the modified software system. Impacts that lead to serious errors, such as failure of software system, are recorded so that they can be removed after identifying and understanding the source of error.
- ***Risk Analysis and Review:*** In the implementation phase, the risk analysis and review are carried out periodically rather than on completion of this phase. Risks that commonly occur are risk for data failure, provision of back up and so on.
- ***Test Preparation Review:*** This review is conducted to evaluate the review for system test in accordance with IEEE standard.

The *control* attribute performs the following functions:

- Conducting software inspections of the code in compliance with IEEE standard.
- Ensuring that unit and integration testing are performed and properly documented.
- Ensuring that test documentation, such as test plan and test cases are either updated or created.
- Verifying that the new software is placed under Supply Chain Management (SCM) control.
- Verifying that the training and technical documentation have been updated.

NOTES

NOTES

The *output* of this phase is in the form of updated (modified) software according to modification request, updated design documentation, updated test documentation, updated user documentation and updated training material.

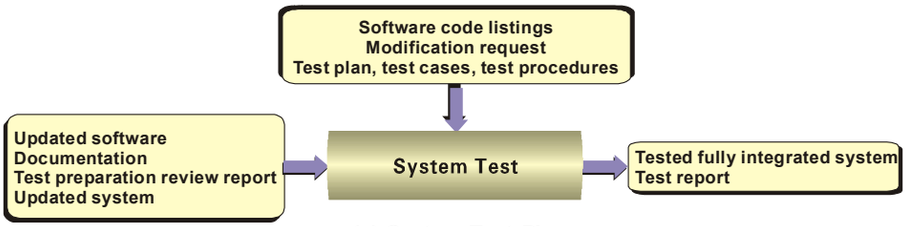
System Test Phase

Regression testing (a type of system testing) is performed on the modified system to validate that the modified code does not introduce faults which did not exist prior to the maintenance activity. As shown in Figure 14.7(a), the *input* attribute comprises the test readiness review report, documentation (system test plans and system test cases), user manuals, design and updated system.

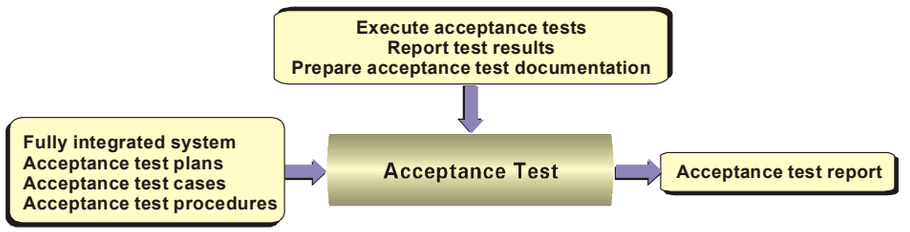
In the *process* attribute, various tests, such as system functional test, interface testing, and regression testing are performed on a fully integrated system. In the *control* attribute, system test is conducted by a separate test function. The test function is required to record the output of a system test and determine the status of the criterion that has been established in the test plan for system testing. This information is essential to proceed to the next phase, which is acceptance testing. The *output* of this phase includes the tested and integrated system along with the test report.

Acceptance Test Phase

Acceptance testing is performed on the fully integrated system by the user or by a third party specified by the user. The objective is to detect errors and verify that the software features are according to the requirements stated in the modification request. As shown in Figure 14.7 (b), the *input* attribute comprises the test readiness review report, fully integrated system, acceptance test plans, acceptance test cases, and acceptance test procedures.



(a) System Test Phase



(b) Acceptance Test Phase

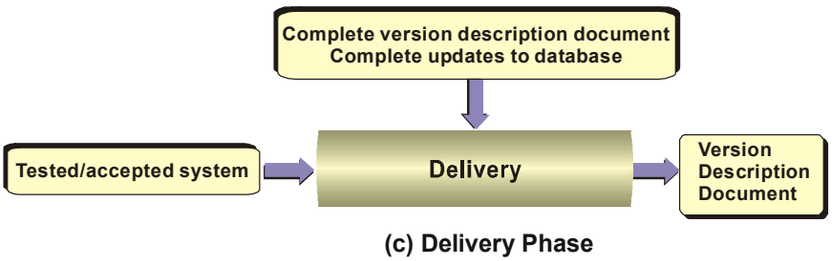


Fig. 14.7 *Phases in SMLC*

The *process* attribute includes the procedures for performing acceptance test at functional level and testing the system to determine its performance and operation. The *control* attribute performs the following functions:

- Executing acceptance tests.
- Conducting functional audit.
- Reporting test results for the Functional Configuration Audit (FCA).
- Placing the acceptance test documentation under SCM control.

The *output* of acceptance test phase comprises an FCA report and the acceptance test report.

Delivery Phase

The delivery phase is concerned with the delivery of a modified software system to the user. In addition, users are provided with a proper documentation consisting of manuals and help files that describe the operation of the software along with its hardware specifications. As shown in Figure 14.8 (c), the input comprises a fully tested version of the system.

The *process* attribute for the delivery phase comprises the following steps:

1. Conducting a Physical Configuration Audit (PCA).
2. Notifying the users.
3. Developing an archival version of the system for back up.
4. Performing installation and training at the users' end.

The *control* attribute performs the functions listed hereunder:

- Arranging and documenting a PCA.
- Providing system materials for access to users including replication and distribution.
- Providing a complete Version of Description Document (VDD).
- Placing contents of the delivery under SCM control.

The *output* of this phase is a version description document and a PCA report.

A project needs to be evaluated in order to know whether the project is a success or a failure.

NOTES

NOTES

Allocation of Resources

Allocation of resources and their management is important for a project where resources have to be shared among concurrent projects. An organization has a pool of different types of experts, such as system analysts, program writers, database designers, system administrators, network administrators and so on and their services are required simultaneously in a number of other projects. In such a situation, the program manager is concerned about the optimal use of specialist staff. When a project is planned, at the stage of allocation of resources, the program manager makes a **'Project Sharing Resource Chart'** as shown in Figure 14.8.

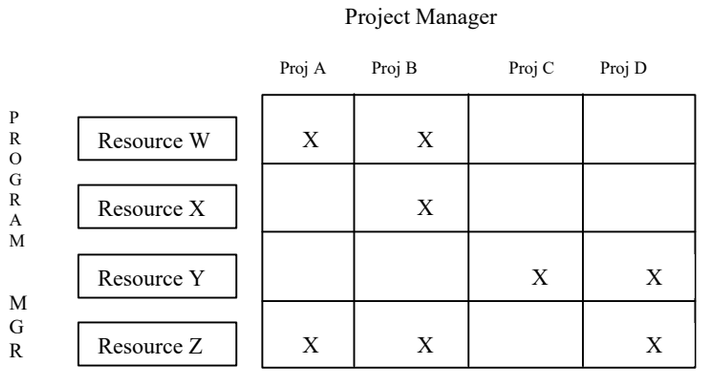


Fig. 14.8 *Project Sharing Resource Chart*

Some activities in the project may have to be delayed until the requisite technical staff is relieved from other projects. During the actual execution of a project, activities can take longer or at times, less time as against the planned schedule.

Evaluation Techniques, Cost Benefit Analysis and Risk Evaluation

The feasibility of every project must be evaluated. There are three major factors on the basis of which a project is evaluated. These are as follows:

(i) ***Technical feasibility or technical assessment*** of the proposed system is to evaluate the required functionality with the available hardware and software infrastructure. Certain lives on organization imposes constraints on the proposed system within the existing framework of hardware and software infrastructure. In this case, the constraint would influence the cost of the solution which must be taken into account while doing Cost Benefit (CB) analysis.

(ii) ***Cost-Benefit analysis*** is a widely used popular method wherein we add all the values of benefits of a course of action and then subtract the value of all the costs associated with it. It helps in reducing the time and effort consumed on working out solutions to financial problems which may finally be of no use.

The costs as categorized here are fixed costs and recurring costs. Recurring costs are those that are ongoing. A specific time period needs to be calculated in which the benefits will be realized. This time period, also known as *payback period* is the time taken to repay costs.

Fixed costs remain the same irrespective of whether the project is running or closed.

How to use the Tool?

CB analysis is computed using mostly financial costs and financial benefits. For example, in case of the simple cost-benefit ratio to be computed for a road scheme, the financial benefit of better transport links is subtracted from the cost of building the road. Intangible costs and benefits, such as environmental damage and shorter travel time, are not taken into account.

A different approach in this regard would be to use intangible costs and benefits to make the CB model even more efficient. For example, should the value of a historic water meadow be worth ₹ 35,000 or ₹ 5,50,000 because of its environmental importance? What is the value of stress-free travel to work in the morning?

CB analysis is usually simple in case of small projects but it becomes an extremely complex and sophisticated art in case of large scale projects or industries.

'Payback time' is also known as 'break-even point' and is at times viewed because the benefits are to be derived from a project. For example, in case of an organization that wants to buy new machinery, the break-even point is obtained graphically. Costs and income are plotted on the graph of quantity of output as against the costs of input. The break-even point is obtained where the two lines meet or cross.

Thus, it is the most common way of carrying out an economic assessment of the proposed system. Here, comparison is made between the development and operation of the system keeping in view the benefits when the product becomes operational. It is assessed if the estimated income/benefits exceed the estimated cost. This CB analysis comprises the following two steps:

1. All costs and benefits are identified and estimated for carrying out the project and operating the delivered operation. The main components, such as development cost, operating cost and benefits (direct and indirect) are expected to occur from the new system.
2. These costs and benefits are expressed in common units. Following this, the net benefit (***Net Benefit = Total Benefit – Total Cost***) is evaluated. Each cost and benefit is expressed in one common unit, i.e., in money.

If the proposal shows that the benefits exceed the costs, then it is considered for further development and is now known as *candidate proposal*. ***Cash flow***

NOTES

forecasting is an important technique of cost-benefit evaluation. It indicates the point at which the expenditure and income will take place (refer Figure 14.9).

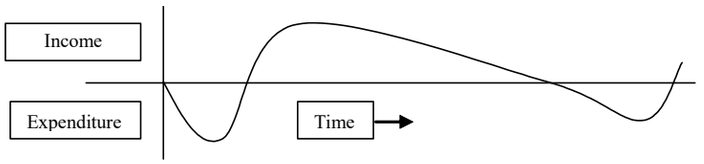


Fig. 14.9 *Cash Flow Forecasting*

At the initial stage, money is required to be spent on activities, such as installation of hardware, software, wages of staff and so on. Such expenditure cannot be avoided until the income has been received. Table 14.2 summarizes a cash flow forecast table for four projects. Here, it is assumed that the cash flow takes place at the end of each year:

Table 14.2 *Cash Flow Forecast Table*

Year	Project 1	Project 2	Project 3	Project 4
0	–80,000	–1,000,000	–200,000	–125,000
1	11,000	100,000	80,000	35,000
2	13,000	300,000	35,000	35,000
3	13,000	200,000	35,000	35,000
4	25,000	250,000	35,000	35,000
5	150,000	350,000	35,000	80,000
Net Profit	132,000	200,000	20,000	95,000

Here, negative values represent expenditure and positive values represent income. The following are some common methods of comparing projects:

1. **Net Profit =Total Cost – Total Income**

   In Table 14.2, Project 2 has the greatest net profit, but it requires large investment. In all the projects, the income rises in the end. We get more income with less investment in Project 1 than in Project 3.

2. **Payback period = Time taken to break even or pay back the initial investment**

   The project with the shortest payback period is chosen because the organization wishes to minimize the project's 'in debt' time.

3. **Return on Investment (ROI) = Accounting Rate of Return (ARR) = (Average Annual Profit / Total Investment) × 100**

   ROI (Return On Investment) is also known as Accounting Rate of Return (ARR). The ROI concept suffers from two disadvantages: (i) It does

not take into account the timing of cash flow. (ii) This rate of return bears no relationship with the interest rates offered or charged by banks.

4. **Net Present Value (NPV) = Value in Year t / $(1+r)^t$**

Here 'r' is the discount rate, expressed in decimal value and 't' is the number of years into the future that the cash flow occurs. This calculation takes into account the timing of cash flow. It is obvious that receiving ₹ 100 today is better than receiving it a year later because its value after one year will be less. For example, the present value of ₹ 100 can become ₹ 91 after a year. This means that the future income is discounted by 10 per cent and we would need an extra 10 per cent to make the wait for a year worthwhile. The annual rate by which we discount the future earnings is known as *discount rate* (10 per cent in this example). Similarly, receiving ₹ 100 in two years' time at this rate will have a present value of approximately ₹ 83. Alternatively, we can say that if we invest ₹ 83 today, then an interest rate of 10 per cent would get us ₹ 100 in 2 years.

Another figure worth considering here is the **Internal Rate of Return (IRR).** It is calculated as the percentage discount rate that would produce an NPV of zero.

NOTES

Points to Remember

Some points to remember for using cost-benefits tool are as follows:

- Cost-benefit analysis is a widely used, popular tool which is relatively easy to use and helps in deciding if a change needs to be made.
- For using the tool, first you work out expenses for the change. Then calculate the benefit that you will derive from it.
- The next step is to compute the time it will take for the benefits to repay the cost, i.e., when the costs or benefits are paid or received over a span of time, work out the time it will take for the benefits to repay the costs.
- Financial cost and benefits are taken into account for carrying out the cost-benefit analysis. However, we can include intangible items in the analysis. If this is the case, then we must estimate a value for these which inevitably bring out an element of subjectivity into the process.

Degree of Risk Involved

Risk evaluation is a technique of assessing the risk associated with a project because we are concerned that the project might not meet the objectives due to the risk involved. Here, we identify risk and quantify its potential effects. We construct a project risk matrix utilizing a checklist of possible risks and classify each risk according to its relative importance and likelihood. Table 14.3 summarizes the structure of summaries of basic project risk matrix. The importance and likelihood are evaluated as high (H), medium (M) and low(L) or exceedingly unlikely(–).

NOTES

Table 14.3 Project Risk Matrix

Risk	Importance	Likelihood
The Software that never completed/delivered	H	-
Project cancelled after design stage	H	-
Late delivery of software	M	M
Development budget exceeded < 20%	L	M
Development budget exceeded > 20%	M	L
Maintenance cost that is higher than estimated	L	L

When the project is relatively risky, it is a common practice to use a higher discount rate to calculate net present value.

Check Your Progress

1. Why software maintenance framework is created?
2. What does operational environment include?
3. Write the various functions of software maintenance team.
4. Explain the types of maintenance.
5. Define the concept of perfective maintenance.
6. Why regression testing is performed?
7. What is in process attribute?

14.3 HARDWARE AND SOFTWARE SELECTION

The hardware and software selection is the very crucial and important phase in System Development Life Cycle (SDLC). The success rate of system design and implementation phase depends on this factor. The SDLC phase works out with vendor selection and benchmarking schemes. Selecting software is a crucial task to determine the issues of objectives, goals, needs, plans, etc. Basically, the development team mainly focuses through the swamping of software functions. The information-based system finalizes the information requirements to check out the vendor. The developed system is not complete until software is installed properly, tested and up-and-running. The hardware and software selection plays an important role in SDLC because of data security and system maintenance. The categories of hardware and software are considered as per functional requirements. The selection evaluates basically software vendors, implementers, Application Service Providers (ASP), technology trends and impact for the organization for which complete system is being developed. The complete scenario is explained with the help of comprehensive case studies. If a company plans to select and implement the LAN (Local Area Network) setting for the organization, the various types of software is

NOTES

delivered to the computer as per selection. The performance and acceptance criteria play a business scale for successful implementation of system. For this, system analysts and vendors are quite aware for taking step-wise decision. It is also necessary for both to be sure that the selected software is compatible with specific server that fulfils the requirements. If company wants that selected software is to be available on each station then the proper arrangements are implemented by analysts, such as check the costing factor to install all the systems of the specified Website. The basic task is to determine what software is required and plan for hardware acquisition.

The software selection is not easy task because selections should always be reflective of the company's mission. It is believed that no guarantee is preferred for any software and it is in the commercial circulation performs the best of its specific types. It is also crucial to select hardware equipments because it is hard to know whether increased memory in hardware is able to drive the software configurations. The more memory requires on a computer hard drive or server. Most software packages will list the minimum memory and hardware requirements to run the software properly; be sure to check these requirements before a purchase. The servers are installed to enhance and renovate the system design. For this, system analyst keeps in the mind about memory capacity and other hardware system limitations when selecting vendors and software packages.

Hardware Selection

The hardware selection is essential task in the phase of system designing because system analyst checks thoroughly the comprehensive platform for developed system. For example, the system is to developed for online air-ticket booking system then analyst must pay the attention of various Web tools, enhanced browsers, etc. The system application server software requires the hardware and software requirements. There must be 35MB free space for temporary files and directories and 250MB free space is required for installation. If you remove the application server from system unit in which software is installed, maximum space is required to install a new version. The various ports are used for default domain. Some of the default ports are required for developed systems. The port 8080 is required for HTTP, port 3920 is required for SSL with mutual authentication, port 4848 is required for administration server etc. Firewall software must be installed with system application server software because most of the ports are disabled by default. Software needs an installation program for available ports that can enable the ports for programs. The hardware and software requirements during the system installation phase enables networking capabilities and fast internet connection and that are required to access an HTTP server. It simulates the Web page accessing. The following Table 10.4 shows the supported platform refers to support operating system.

NOTES

Table 14.4 *Platform for Selected Language*

Operating system	Minimum memory	Recommended memory	Minimum disk space	Recommended Minimum disk space
Sun Solaris 9, 10 (SPARC) Solaris 9, 10(x86)	512MB	512MB	250MB free	500MB free
64-bit Sun Solaris 10 SPARC, x86)	512MB	512MB	250MB free	500MB free
Red Hat Enterprise Linux 3.0 U1, 4.0	512MB	1GB	250MB free	500MB free
SuSE Linux Enterprise Server 10SP1	512MB	1GB	250MB free	500MB free
64-bit SuSE Linux Enterprise server 10SP1	512MB	1GB	250MB free	500MB free
Ubuntu Linux, Hardy Release	512MB	1GB	250MB free	500MB free
AIX 5.2, 5.3	512MB	1GB	250MB free	500MB free
Windows Server 2003 Windows XP Pro Windows Vista	1GB	2GB	500MB free	1GB free
Macintosh (Intel, Power)	512MB	512MB	250MB free	500MB free

In Table 14.4, Solaris 9, 10(x86, SPARC) use the Solaris patch cluster that supports security features and runtime environment for programs and applets. The connectivity requirements contain 64-bit hardware communication of TCP/IP (Transmission Control Protocol/Internet Protocol). The TCP/IP for IPv4 and IPv6 protocols are released with software program so that programs and applets make connection easily with Internet. The hardware contains RAM, Field Programmable-Gate Array (FPGA) and ROM (Read Only Memory) is tagged with host controller in which host system contains finalizer, scheduler, clock, idler and memory. The drivers and databases need to get the hardware and software compatibility requirements. The configuration of application server contains one database/driver as shown in the below table.

Software Selection

For online system, the software selection is crucial task because you can not use Foxpro database for back-end support instead you can use MySQL or Oracle for back-end database. In the same way, ASP.NET or PHP (Hypertext Preprocessor) scripting language is required for that. The word processors, computer language programs, notepad, text-pad, language editors, various software, such as Java, C++, Ada, other scripting languages, Web development tools, Oracle, etc. must be installed in the system to write, edit, debug the coding. The application server is designed to support the connectivity of additional DBMS (Database Management System) that corresponds to system driver. Table 14.5 shows the requirement of supported database servers for driver.

NOTES

Table 14.5 Supported Database Server

Vendor of JDBC Driver	Supported Database Server
Derby Network Client	Derby 10.2
DataDirect 3.6.x, 3.7.x	Oracle 10g Oracle 9i MySQL 2005
MySQL Connector	MySQL 5.0
Oracle 10G	MySQL 5.0
PostGres	8.2x
Oracle OCI	Oracle 9i
IBM DB2	DB2 9.1

The output files can be displayed in proper display unit. The system unit must have 1024 pixels×768 pixels. The header and navigation area needs 760 pixels×570 pixels for navigation and header areas that provide complete visibility for graphics and pictures. The development environment for various programs provides runtime environment that is necessary to Java programs. It facilitates plug-in for browsers that linked with the programs that are to be fetched by browsers. This software toolkit comes in two types, such as a precompiled binary and a source package. The Web development tools must be installed in windows machine therefore, windows OS is required. Table 14.6 shows the software and hardware requirement for this release.

Table 14.6 Component and Requirement for System Development

Component	Requirement
Operating System	Solaris 10 on SPARC, x86 and x64 based systems that includes whole root local and sparse root zones.
	Red Hat Enterprise Linux 5.0 Server, 32 and 64-bit update versions
	Red Hat Enterprise Linux 3 and 4 update versions
	Advanced Server (32-bit and 64-bit versions) and Enterprise Server (32-bit and 64-bit versions)
	Windows OS
	Windows 2000 Advanced Server, Data Center Server version SP4 on x86
	Windows 2003 Standard (32-bit and 32-bit versions), Enterprise (32-bit and 32-bit versions), Data Center Server (32-bit version) on x86 and x64 based systems
	Windows XP Professional SP2 on x86 based systems
J2SE platform	J2SE platform 6.0, 5.0 Update 9 (HP-UX: 1.5.0.03), 1.4.2 Update 11 and Update 5.0 and Update 12 (as of Java Enterprise System 5 update 1)

NOTES

Directory Server	Directory server represents Access Manager information tree. Directory Server Enterprise Edition 6.1 (compatible with Access Manager P7.1 Patch 1), Sun Java system directory Server 6.0 and Sun Java system directory server 5.2 2005Q4. Access manager identity repository contains Directory Server enterprise edition 6.1 and 6.2 compatible with Access manager 7.1 Patch 1, Sun Java system directory server 5.2 and 6.0 and Microsoft active directory.
Web Containers	Sun Java system Web server 7.0 and 7.0 update 1. On supported platform/OS combinations you may elect to run the Web server instance in a 64 bit JVM. Support platforms are required as Web containers in Java for example, Solaris 9/SPARC, Solaris 10/ SPARC, Solaris 10/AMD64, Red Hat AS or ES 3.0/AMD64, Red Hat AS or ES 4.0/AMD64.
RAM	The RAM capacity contains basic testing of system performance as 512 Mbytes. The actual deployment should be 1 Gbyte for threads and Access Manager can be SDK, HTTP server and other internals.
DISK SPACE	The disk space would be of 512 Mbytes that is suitable for Access Manager and Java associated applications.

Browsers

The requirement of supporting Web server for various applications is explained by Table 14.7.

Table 14.7 *Supported Web Browser for System Application*

Browser	Version
Mozilla	1.7.12
Internet Explorer	6.0 Service Pack 2, 7.0
Firefox	1.5.x, 2.x
Netscape	8.0.4, 8.1, 9.0, 9.0.x

Table 14.8 shows the list of browsers that are supported by the software released by the software development process.

Table 14.8 *Supported Browsers*

Browser	Platform
Firefox 1.0.7, 1.5, 2.0 and latest versions	Windows XP Windows 2000 Solaris OS, versions 9 and 10 Red Hat Linux 3 and 4 Mac OS X
Internet Explorer	Windows XP
Internet Explorer 6.0 SP2	Windows XP
Internet Explorer 6.0 SP2	Windows 2000
Mozilla 1.7.12	Solaris OS [Versions 9 and 10], Windows XP, Windows 2000, Red Hat Linux 3 and 4, Mac OS X
Netscape Communicator 8.0.4	Windows XP, Windows 2000
Netscape Communicator 7.1	Solaris OS, versions 9 and 10 along with latest versions.

NOTES

The minimum hardware requirements for developing the Web environment are as follows:

- **Web application server:** At least 2 GHz processor, 1 GB RAM and 30GB capacity is required.
- **Database server:** At least 2 GHz processor, 1 GB RAM, 100 GB RAID 4 + disk capacity are required.
- **File-server:** At least 2 GHz processor, 512 MB RAM, RAID 5 storage with appropriate disk space are required.
- **Application cluster:** Application-dependent sizing are required.

The minimum software requirements for developing the Web environment are as follows:

- FTP server (required for server-side):
- IIS (Windows)
- FTP (AIX/LINUX available as operating system component)

Browsers (one client-side browser required)

The list of client-side browser for code development environment is as follows:

- Internet Explorer 6.0 SP2 or higher
- Netscape 6.2 or higher
- Mozilla 5.0 or higher
- Firefox 1.5 or higher
- Web application server
- IBM WebSphere 6.0.2.9
- Apache Tomcat 5.0.28

Relational Database (one server-side RDBMS is required):

The list of relational database in software development is as follows:

- IBM (Internatinoal Business Machines) DB2 Universal Database Enterprise Server Edition (ESE) V8.2, with FixPak 3 and Oracle 10g, MySQL engine.
- One file vault server is required on server side.
- FLEXlm license server 10.8 or higher version is required on server side.

The optional software application might be required before the installation and deployment of SimManager in which lightweight directory access protocol (LDAP) compatible directory service, such as active directory and Netscape directory service. The following additional software applications may be added during or after the installation and deployment of SimManager to enhance performance or extend capabilities:

- Load balancing system.

NOTES

- Queue management system, i.e., Log Structured File (LSF) or other analysis manager, and.
- Product data management (PDM) integration.

The required applications are run either on server side computers or client side computers. The batch versions of MSC Nastran or MSC Patran are frequently required to execute the MSC.Software provided by demonstration portals. The Table 14.9 shows the software requirements on server side environment:

Table 14.9 *Required Software on Server Side*

Software	Description	Required/Optional
Java SDK 1.4.1	Java Interpreter	Required
Apache Ant 1.5	Language build tool	Required
Apache Tomcat 4.0.6	Web application server	One web server is required
IBM WebSphere 5.1 with IBM HTTP Server	Web application server	One web server is required
Oracle 8.1.7 Enterprise Edition	Relational database server	One database server is required
MS SQL Server 2000	Relational database server (2000)	One database server is required
FlexLM 9.2	License Server	Required for Licensing

Table 14.10 shows the optional software requirements on server-side environment:

Table 14.10 *Software, Description and Status of Requirement*

Software	Description	Required/Optional
Log structured file (LSF) Server	Queuing system Server	One server is required for per LSF configuration.
LSF Client	Queuing system Client	One server is required for per client host.
External Application (Patran, Nastran, BlowSim)	External applications for processing data	Nastran is required for demonstration portals.
LaTex	Tool to customize reports and returns output to PDF format.	Required for report generation.

14.3.1 Computer Hardware

Hardware is the physical characteristic of computers, telecommunications and other devices. Hardware not only includes the computer system but also the cables, connectors, power supply units, media equipment, peripheral devices, such as the keyboard, mouse, audio speakers and printers. Selecting the accurate hardware is very significant and essential technological decisions. Analyse the requirements first and then recommend the most cost-effective solutions. Various brands of computer hardware equipments are available in the market and on Internet. Select the specific type after analysing the properties of different configurations.

While acquiring hardware, must keep in mind the requirements of the organization and the 'software' to be used as it is the heart of the system. Hence, while selecting the specific computer systems first evaluate the software properties and specifications that it must be efficient and effective for satisfying the overall

requirements of an organization and the 'hardware' acquired too must be compatible to support the selected software.

Determining Hardware Needs

NOTES

Sources of needs can be either external or internal. It is important to consult users and computer center personnel for determining the needs to successfully implement the acquired hardware. For this, attributes of needs can be categorized as mandatory and desirable (Table 14.11).

Mandatory attribute: Mandatory attribute defines that the hardware to be acquired must possess. Vendors who can not offer products along with all these typical attributes can be removed from considerations.

Desirable attribute: This attribute defines that the hardware to be acquired may or may not be striking to the buyer.

The difficult part of this process is to categorize the hardware to be acquired as per the needs of an organization into the technical mandatory and desirable attributes.

Table 14.11 *Mandatory and Desirable Attributes*

Possible Mandatory Attributes	Possible Desirable Attributes
Hardware Compatibility	Field Upgradability
Software Compatibility	Purchase Price
Speed	Capacity
Maintenance	Waste Heat
Delivery Date	Free Trial Period
Training	Operating Cost
Installation Time	Electric Power Needs
Guarantees and Warranty	Associated Hardware
Documentation	Cabling Constraints

Your IT strategy should define the generic type of hardware that your business needs, and a timetable for its acquisition. Consider upgrade costs, warranties, what level of support you want and the maintenance contract with your supplier. These will affect pricing and ongoing costs. You can acquire hardware in a number of different ways, for example by buying, leasing or renting it. Many small businesses buy their desktop and server hardware outright. Use our interactive tool to find out which computer equipment you should buy for your business. You can buy hardware directly from the original manufacturer, usually via their Website or over the telephone, or through a retail channel. The direct route can be cost-effective for a small number of PCs at a time. To get the best from this route you must have a clearly defined specification for your needs. Choose a standard offering that is close to, but better equipped, than your minimum requirements. You can upgrade memory, for example, later on if you need to.

Another option is to hire a consultant to help you refine your requirements, place your shopping list with various different suppliers and see what sort of deals you are offered. You can also buy printers directly from the manufacturer, but

NOTES

consider the likely running costs before choosing a particular type. Ink-jet printers are often priced cheaply, but the ink cartridges are expensive and may have to be replaced often. It may be better to purchase a more expensive laser-based printer and share it between the staff using your network. The lower running costs will quickly cover the additional capital cost if you use colour a lot.

For PCs, servers and printers, you will need network infrastructure, such as network switches and all the wiring needed to connect equipment. You can obtain this type of hardware from most of the major catalogue based IT vendors who carry a large range of network equipment. It is important to balance how much you spend on infrastructure and how much on PCs. Make sure to include this in your IT strategy and plan any network upgrades so that they are implemented at the right time, not in a piecemeal fashion.

14.3.2 Computer Software

Software selection is a critical aspect of system development. The search starts with the software, followed by the hardware. There are two ways of acquiring software: custom – made or "Off – the –shelf" packages. Today's trend is toward purchasing packages, which represent roughly 10 percent of what it costs to develop the same in house. In addition to reduced cost, there are other advantages:

1. A good package can get the system running in a matter of days rather than the weeks or months required for "home-grown" packages.
2. MIS personnel are released for other projects.
3. Packages are generally reliable and perform according to stated documentation.
4. Minimum risks are usually associated with large – scale systems and programming efforts.
5. Delays in completing software projects in house often occur because programmers quit in midstream.
6. It is difficult to predict the cost of "home-grown" software.
7. The user has a change of seeing how well the package performs before purchasing it.

There are drawbacks, however, to software packages:

1. The package may not meet user requirements adequately.
2. Extensive modification of a package usually results in loss of the vendor's support.
3. The methodology for package evaluation and selection is often poorly defined. The result is a haphazard review based on a faulty process or questionable selection criteria.
4. For first – time software package users, the overall expectation from a package is often unclear and ill defined.

It can be seen, then, that the quality of a software package cannot be determined by price alone. A systematic review is crucial.

Criteria for Software Selection

NOTES

Prior to selecting the software the project team must set up criteria for selection.

Selection criteria fall into the categories described here.

Reliability

It is the probability that the software will execute for a specified time period without a failure, weighted by the cost to the user of each failure encountered. It relates to the ease of recovery and ability to give consistent results. Reliability is particularly important to the professional user. For example, a pharmacist relies on past files on patients when filling prescriptions. Information accuracy is crucial.

Hardware may become inoperative because of design errors, manufacturing errors, or deterioration caused by heat, humidity, friction, and the like. In contrast, software does not fail or wear out. Any reliability problems are attributable to errors introduced during the production process. Furthermore, whereas hardware failure is based largely on random failures, software reliability is based on predestined errors.

Although reliable software is a desirable goal, limited progress has been made toward improving it in the last decade. The fact of unreliable software had led to the practice of securing maintenance agreements after the package is in operation. In a sense, unreliability is rewarded.

Software reliability brings up the concept of modularity, or the ease with which a package can be modified. This depends on whether the package was originally designed as a package or was retrofitted after its original development for single installation use. A package with a high degree of modularity has the capacity to operate in many machine configurations and perhaps across manufacturers' product lines.

With modularity come expandability, which emphasizes the sensitivity of a software package to handle an increased volume of transaction or to integrate with other programs.

The following questions should be considered:

1. Is there room for expanding the master file?
2. How easily can additional fields and files be added?
3. Are there errors a user can make that will ring down the system?
4. How much of the system becomes unusable when a part of it fails?
5. What are the recovery capabilities?

NOTES

Functionality

It is a definition of the facilities, performance, and other factors that the user requires in the finished product. All such information comes from the user. The following are key questions to consider:

1. Do the input transactions, files, and reports contain the necessary data elements?
2. Are all the necessary computations and processing performed according to specifications?

Capacity

Capacity refers to the capability of the software package to handle the user's requirements for size of files, number of data elements, volume of transactions and reports and number of occurrences of data elements. All limitations should be checked.

Flexibility

It is a measure of the effort required to modify an operational program. One feature of flexibility is adaptability, which is a measure of the ease of extending the product.

Usability

This criterion refers to the effort required to operate, prepare the input, and interpret the output of a program. Additional points to be considered are portability and understand ability. Portability refers to the ability of the software to be used on different hardware and operating systems. Understand ability means that the purpose of the product is clear to the evaluator and that the package is clearly and simply written, is free of noise, and contains sufficient references to readily available documents so that the reader can comprehend advance contents.

Security

It is a measure of the likelihood that a system's user can accidentally or intentionally access or destroy unauthorized data. A key question is how well one can control access of software or data file? Control provides system integrity.

Performance

It is a measure of the capacity of the software package to do what it is expected to do. This criterion focuses on throughput, or how effectively a package performs under peak loads. Each package should be evaluated for acceptance on the user's system. The language in which a package is written and the operating system are additional performance considerations. If we plan to modify or extend a package, it is easier if it is written in a language that is commonly known to programmers. Likewise, if the package run only under a disk operating system and the installation is under a full operating system, then either the package will have to be upgraded to

the larger operating system or the system downgraded to handle the package as is. In either case, the change could be costly and counterproductive.

Serviceability

This criterion focuses on documentation and vendor support. Complete documentation is critical for software enhancement. It includes a narrative description of the system, system logic and logic instructions. Vendor support assures the user adequate technical support for software installation, enhancements, and maintenance, the user should determine how much on – site technical assistance is provided by the vendor, especially during the first few weeks after the installation. The user expects on – site training and support as part of most commercial packages. It is vital to inquire about the amount of training provided. The user may require training at several levels—clerical, operations, programming, and management.

Ownership

Who owns the software once it is "sold" to the user? Most of the standard license agreement forms essentially lease the software to the user for an indefinite time. The user does not "Own" it, which means that the source code is inaccessible for modification, except by the vendor. Many users enter into an escrow arrangement whereby the vendor deposits code to the user if the vendor goes out of business or is unable to perform the services specified in the license.

In acquiring software, several questions should be asked:

1. What rights to the software is the user buying?
2. Can the user sell or modify the software?
3. If the vendor is modifying the package especially for the user, can the vendor sell it to other within the same industry the user is in?
4. What restrictions are there to copying the software or documentation?

Minimal costs

Cost is a major consideration in deciding between in – house and vendor software. Cost – conscious users consider the following points:

1. Development and conversion costs.
2. Delivery schedule.
3. Cost and frequency of software modifications.
4. Usable life span of the package.

14.4 PROCEDURE FOR HARDWARE AND SOFTWARE SELECTION

The systems come with hardware, software and support. Today, selecting a system is a serious and time-consuming business.

NOTES

NOTES

There are several factors to consider prior to system selection:

1. Define the system capabilities that make sense for business. Computers have proven valuable to business in the following areas:
   - Cost reduction includes reduction of the inventory, savings on space and improved ability to predict business trends
   - Cost avoidance includes early detection of problems and ability to expand operations without adding clerical help.
   - Improved service emphasizes quick availability of information to customers, improved accuracy and fast turnaround
   - Improved profit reflects the bottom line of the business and its ability to keep receivables within reason.
2. Specify the magnitude of the problem, that is, clarify whether selections consist of a few peripherals or major decision concerning the mainframes.
3. Assess the competence of the in-house staff. This involves determining the expertise needed in areas such as telecommunications and data base design. Acquiring a computer often results in securing temporary help for conversion. Planning for this help is extremely important.
4. Consider hardware and software as a package. This approach ensures compatibility. In fact, software should be considered first, because often the user secures the hardware and then wonders what software is available for it.
5. Develop a schedule for the selection process. Maintaining a schedule helps keeps the project under control.
6. Provide user indoctrination. This is crucial, especially for first-time users. Selling the system to the user staff, providing adequate training, and preparing an environment a conductive to implementation are pre-requisites for system acquisition.

Major phases in selection

The selection process should be viewed as a project, and a project team should be organized with management support. In larger projects, the team includes one or more user representatives, an analyst and EDP auditor, and a consultant. Several steps make up the selection process.

There are six phases of selection process:

1. Requirements Analysis
2. System Specifications
3. Request For Proposal (RFP)
4. Evaluation and Validation

5. Vendor Selection
6. Post-Installation Review

NOTES

Requirements analysis

The first step in selection is understanding the user's requirements within the framework of the organization's objectives and the environment in which the system is being installed. Consideration is given to the user's resources as well as to finances.

In selecting software, the user must decide whether to develop it in house, hire a software company or contract programmer to create it, or simply acquire it from a software house. The choice is logically made after the user has clearly defined the requirements expected of the software. Therefore, requirements analysis sets the tone for software selection.

System Specifications

Failure to specify system requirements before the final selection almost always results in a faulty acquisition. The specifications should delineate the user's requirements and allow room for bids from various vendors. They must reflect the actual applications to be handled by the system and include system objectives, flowcharts, input-output requirements, file structure and cost. The specifications must also describe each aspect of the system clearly, consistently and completely.

Request for Proposal

After the requirements analysis and system specifications have been determined, a request for proposal is drafted and sent to selected vendors for bidding. Bids submitted are based on discussions with vendors. At a minimum, the RFP should include the following

1. Complete statement of the system specifications, programming language, price range, terms and time frame.
2. Request for vendor's responsibilities for conversion, training and maintenance.
3. Warranties and terms of license or contractual limitations.
4. Request for financial statement of vendor.
5. Size of staff available for system support.

Evaluation and validation

The evaluation phase ranks vendor proposals and determines the best suited to the user's needs. It looks into items such as price, availability and technical support. System validation ensures that the vendor can match his/her claims, system performance. True validation is obtained verified by having each system demonstrated. An outside consultant can be employed for consulting purpose.

NOTES

Vendor selection

This step determines the winner – the vendor with the best combination of reputation, reliability, service record, training, delivery time, lease finance terms and conversion schedule. Initially a decision is made which vendor to contact. The sources available to check on vendors include the following

1. Users
2. Software Houses
3. Trade Associations
4. Universities
5. Publications/Journals
6. Vendor Software Lists
7. Vendor Referral Directories
8. Published Directories
9. Consultants
10. Industry Contacts

Post- installation Review

Sometime after the package is installed, a system evaluation is made to determine how closely the new system conforms to plan. System specifications and user requirements are audited to pinpoint and correct any differences

Software selection

Software selection is a critical aspect for system development. There are 2 ways of acquiring the software.

- Custom -made
- Packages

Criteria for Software selection

Reliability – It is the probability that the software will executed in a specific period of time without any failures. It is important to the professional user. It brings up the concept of modularity, or the ease which a package can be modified.

Functionality – It is the definition of the facilities, performance and other factors that the user requires in the finished product.

Capacity – Capacity refers to the capability of the software package to handle the user's requirements for size of files, number of data elements, and reports. All limitations should be checked.

NOTES

Flexibility – It is a measure of effort required to modify an operational program. One feature of flexibility is adaptability.

Usability – This criteria refers to the effort required to operate, prepare the input, and interpret the output of a program. Additional points considered here are portability and understand ability. Portability refers to the ability of the software to be used. Understand ability is the purpose of the product.

Security – It is a measure of the likelihood that a system's user can accidentally or intentionally access or destroy unauthorized data.

Performance – It is a measure of the capacity of the software package to do what it is expected to do. This criteria focuses on throughput or how effectively a package performs under peak load.

Serviceability –This criteria focuses on documentation and vendor support.

Ownership – Who owns the software, and to consider whether he has the right to access the software, or he can sell or modify the software.

Minimal costs – Cost is a major consideration in deciding between in-house and vendor software.

Evaluation process

There are three process for evaluating hardware and software.

1. **Benchmark programs:** It is a sample program for evaluating different computers and their software. It is necessary because computers often uses the same instructions, words of memory or machine cycle to solve a problem. Benchmarking includes the following:
   - Determination of the minimum hardware.
   - An acceptance test
   - Testing in an ideal environment to determine the timings and in the normal environment to determine its influence on other programs.
2. **Experience of other users:** Benchmarking only validates vendors' claims. Experience of other users with the same system software is essential.
3. **Product reference manuals:** These evaluate a system's capability. These reports elaborate on computer products, services and prices.

Evaluation of proposals

After all proposals are evaluated, the final vendor is selected using any of the 3 methods

1. **Adhoc** refers to the user's inclination to favor one vendor over others.

NOTES

2. **Scoring**. In this method the characteristics of each system are listed and score is given in relation to the maximum point rating. Then each proposal is rated according to its characteristics.
3. **Cost value approach.** In this method a dollar credit method is applied to the proposal that meets the user's desirable characteristics. This credit is subtracted from the vendor's quoted price. The proposal with the lowest price is selected.

Performance evaluation

Hardware selection requires an analysis on the following criteria

1. System availability
2. Compatibility
3. Cost
4. Performance
5. Uptime
6. Support
7. Usability

For the software evaluation, the following are considered

1. The programming language and its suitability to the applications
2. Ease of installation and training
3. Extent of enhancements to be made prior to installation.

In addition to hardware and software evaluation, the quality of the vendor's should be examined. Considerations to ensure vendor quality are as follows

1. Backup
2. Conversion
3. Maintenance
4. System development

Financial Considerations in Software Selection

The financial consideration works with various software development projects which is associated mainly with elapsed time for estimating the total cost. It sometimes involves benchmarking tool to measure the selection of various types of software packages and applications. The financial aspects in software selection review refer to the potential software suppliers maintenance cost. Basically, many companies tend to focus only on the advance software technologies in the network area. The software's potential product functionality and cost are the two major approaches.

NOTES

These two factors are crucial and important factors in selection process of software. Let us take a small example to clarify the complete scenario. Let there be a business plan involving PowerPoint business software. The product services are delivered to the organization in the PowerPoint presentation. For this, the efficiency of software package and the cost of the software application put great impact in selection process. But, one disadvantage of this mechanism is that such type of relative productivity of software selection investigates the use of data development analysis as a method of benchmarking. This scenario articulates the following principles:

- What is the annual subsidy for purchasing this software application?
- What is the financial self sufficiency?

Usually, a procedure is used for this as step wise selection that leads to cost recovery process if software fails to fulfill the requirement. Let us take an example of sample company known as DataSoft Company. The word benchmarking itself is a generic term and it can be understood with the help of case studies. The DataSoft Company plans to expand the previous setup. For this, it includes application modules for the aspects of finance and human resources. Now, the company involves Enterprise Resource Planning (ERP) to its system that is integrated with RDBMS. Basically, ERP is a software infrastructure that manages almost various parts of company or business. This complete system interacts with software purchasing, manufacturing, sales and customer services for the DataSoft Company. The Datasoft Company considers the system requirements, such as ERP addresses, like client-server distributed architecture, object oriented programming software and RDBMS supported software. This company is being tagged with ERP that allows selecting latest technologies, like Electronic Fund Transfer (EFT), Electronic Data Interchange (EDI), video conferencing, Internet, intranet, report generation and data mining. The DataSoft Company checks the requirement of various types of software to spread its planning. Generally, it is difficult to maintain a customized software package therefore some leading software companies design ERP software which especially checks the financial strategy of an organization to implement the following functionality:

- Which advance software technologies will be implemented across the departments in DataSoft Company?
- What is the least cost for better project management?
- Which vendor is more beneficial to innovate the complete system?

For this, the famous ERP vendors, such as Oracle, PeopleSoft, International Business Machines (IBM), Systems Applications and Products (SAP), etc., can be used to check the financial obligations, such as number of systems, suitable applications for the organization, pricing cost, maintenance cost, training of staffs, cost to implement the advance technologies, etc. These vendors are expensive.

NOTES

The DataSoft follows the Understand, Simplify, Automate (USA) process to innovate the system. In fact, the software development projects are basically known for inaccuracies to elapsed time and total cost estimates. The benchmarking tools are measured to review the functionality and cost of the additional areas, such as checking the software undergo to marketing and promotion schemes so that they would be available on reduced cost. The company considers any opportunity that generates an endowment as well as revenue so that the total cost would be maintained. The ongoing costs are incurred in the DataSoft if the project is going to be completed and implemented. Therefore, the financial consideration in software selection involves the following types of cost:

- **Training Costs:** These include the cost associated with training the user about the use of the new system and procedures.
- **Documentation and User Form Cost:** This includes cost associated with preparing documents and manuals needed at the time of implementation of the project.
- **Hardware Cost:** This includes cost associated with user owned computers, printers, communications devices and security devices.
- **Software License or Acquisition Costs:** These include cost associated with software licensing and cost incurred in locating mainframe and minicomputer on multiple sites.

The importance of cost effectiveness in buying software is taken in broad perspective because low cost of product might result in high cost along the software's life as per demand. For example, a Document Management System (DMS) might be the cheapest one but if it is installed to set up a virtual office its cost might be increased as per need. According to the business point of view an Adoption and Learning Curve (ALC) is involved with every new types of software purchasing because it is integrated with the present workable software. The complex software adopts resistance. The shorter learning curve trains better to a new user, for example collaboration software allows sharing the Outlook data. Let there be software 'shelfware' which is never used with system installation because it is expensive so if this software is collaborated with Outlook, a new concept of 'software as a service' is hosted. System software provides documented help engine to the company. Training cost for staffs is also considered as a vast expense for the enhanced technology whether it is on Webinars, i.e., online seminars or sometimes offline seminars. The company might offer paid training. The software maintenance cost impacts on the performance and adoptability which plays a great role in buying a software. The uptime factor in selecting software also covers the service level agreement. Efforts to constantly improve upon the software underline a commitment to providing quality service. The above mentioned overall factors conduct software selection projects. The software system includes ERP, payroll

or Human Resource (HR), Customer Relationship Management (CRM), retailing, billing, etc. The selection process is considered a straightforward technique . This process includes an embedded software evaluation form on a comparison spreadsheet. It lists lots of the criteria that are common to all software applications and which the company should be checking for suitability. It also includes fields that allow you to enter your own details for each of the software applications the company is considering. It can be compared or evaluated with pre-existing ERP systems, business systems, management systems and accounting or payroll software applications. The software comparison spreadsheet can be used for any of these software packages. A detailed description of software evaluation methodology helps the company with the software selection project as summarized in Table 14.12.

Table 14.12 *Software Selection Methodology*

Considered Criteria	Function
Requirement analysis	Software selection efficiently gathers a minimal cost for company's system requirement. After preparing the report, software vendors are analysed to select the applications.
Business process review	Software selection is used to implement the enhanced technology. This section deals with the document and demonstration script.
Software research	The company spends hours on the phone and interacts with the software vendors to get the information of the key criteria.
Software demonstration	A demonstration script is used to show how the company's requirements are compared with the software vendors to track the evaluation and this mechanism later helps to communicate to select the software.

The selection process is overviewed with the organization's time and energy. As a result, the cost of selected software packages and applications puts a great impact to take risk in the whole selecting process. The online auction software supports a wide range of platforms that support databases to the existing standards in the organization. It eliminates the need to adopt new standard and technology for the auction system. It helps in reducing the amount of training required in implementing the system. The created Web site for online auction system also involves the cost of maintenance. The marketing group and Web group contracts the overall look to a design company. Instead, the content administrator easily adds the auction data to RDBMS. If the auction system is more sophisticated, the cost would be added on extra to the system. No doubt, it allows automation either to the system's database or through an openended Application Programming Interface (API). Once the design of Web site is approved the initial small groups are made to view the design's visual effectiveness, the general abilities and the level of user friendliness. After approving the testing of design, the functionality of Web site is tested that includes the manual and automated process. The component of the workflow moves from inventory entry to fulfillment which should be tested

NOTES

thoroughly. All reports created to analyse the system bidding patterns and trends should also be run for examining the suitability at this time. The system should also be tested under expected peak user loads and data volumes. Screen below displays the deployment cost plan in system selection. At the last phase in online auction system implementation, the initial marketing plan is pushed. A huge cost is involved in marketing the complete system (here online auction system). In marketing push and promote activities, it is generally advisable to bid the regular auction schedules. Automated e-mail systems are also implemented to allow the people to make the auction as a regular part of business practices. Train technical resources, install and configure software, design and implement Web site, design and implement supporting interfaces include design phase whereas system testing includes implementation and testing phase. In testing phase, user acceptance/beta testing is done. Market the site and Go Live options include the production phase in system development life cycle as shown in Figure 14.10.

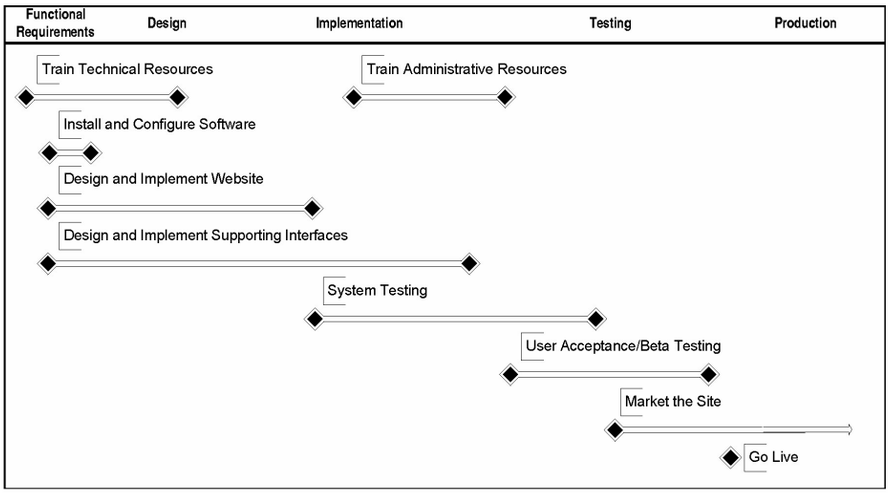


Fig. 14.10 *Testing Phase of Software*

The auction software allows notifications without ongoing marketing cost. It is also very crucial phase that the auction software analyses the bidding patterns and trends to make subsequent auctions more effective. For example, registering the trading partners can communicate with the system after paying a nominal charge and entice into a new level of association with the company. The automated software allows for efficient communication with large groups of partners thus ensuring the company achieves its goals within its own budget. This virtual market place offers the business advantages in the following way:

- A rapid return on investment.
- Competitive bidding incorporates maximized revenues.
- A bidding database is created in one-to-one marketing.
- Sales channel conflict is minimized.
- Administrative cost is reduced.

NOTES

Overall, the software selection process becomes a necessary tool to address the increasing needs for openness, fairness, accountability and impartiality while facing a technology selection. The ideal software selected has following features:

- Cost to publicize the company's requirements.
- Cost to evaluate the available system and interchanged system.
- Cost to implement the compared system providing best matching option.

Every system requires security features to prevent unauthorized use and access of information related to the system. Audit control protects a system from external security breaches and internal fraud.

System Security

System security can be divided into different categories according to security issues, which are related to the file structure, data structure and access procedures of system. Following are the various system security issues:

- **System Reliability:** Refers to the technical innovation and procedures applied to the hardware and operating system to protect system from damage.
- **System Integrity:** Refers to the proper working process of hardware, software, appropriate physical security and safety from external security threat.
- **Data Integrity:** Refers that data should be same as its original form and should not be disclosed, altered or destroyed due to unsecure access.
- **System Privacy:** Refers to the rights of the users or organization to determine which information should be shared with or accepted from other and how can an organization protect their information system from unfair use.
- **Protection Against Viruses:** Refers to the methodes that can be used to protect system from virus infection. For this you can use diffrernt anti virus softwares to protect system.

Threats to System Security

A procedure of system security makes sure that the facility is physically secure and provides a recovery capability. But system security can be damaged due to external threats. Various external threats are as follows:

- Errors generated due to people making mistakes
- Unauthorized users
- Natural disasters
- Physical attack

Risk Analysis

The risk analysis helps to analyse the probability of problem occurrence, cost of each possibility of system damage and the preventive measures to include as part

of security plan. Two key factors of risk analysis are the value of potential loss and the probability of loss.

Control Measures

NOTES

After determining system security issues and risk associated with system security you need to identify the control measures that can be internal and external. Following are the various control measures:

- **Identifying the Users:** Helps to detremine the user of system with their defined access right.
- **Access Control:** Helps in controlling the system access. Various activities can be performed to control the access of system, such as encoded card system and encryption method.

Protection Against Virus

A virus refers to a computer program which enters in a computer without the knowledge of the user. In a virus infected file, virus is hidden inside a program contained in the file. You can protect your system against viruses using anti virus software and recovery method.

Antivirus Software

Antivirus software contains computer programs that identify and eliminate computer viruses from computer. Following methods are used by anti virus software to detect viruses in a computer as follows:

- Scanning all the files available in a computer to search for viruses.
- Detecting suspicious behaviour of any program in a computer which might be a virus.

Recovery Methods

Recovery methods are used to recover the data infected by virus in a computer. A computer infected with virus is unsafe to use without reinstalling the operating system. The various recovery methods are as follows:

- Operating system reinstallation helps recover the data infected by virus.
- Virus removal tools are provided by antivirus software which helps recover the data.

System Audit

System audit refers to a technical assessment of a system or application. Personal computers, servers and mainframes are the example of system whereas database and Web services are the examples of applications. System audit is performed using two methods, automated and manual. Automated system audit uses software

for automated assessment. This system audit performs various activities, such as generating audit reports or monitor and report changes made to various files stored in the system. Manual system audit includes various activities, such as interviewing staff, reviewing system and security vulnerability scans.

Implementation and Evaluation

Evaluation is an assessment that refers to design, implementation and results of completed or on-going project, program and policy. It should be systematic and objective. Key criteria to be used in evaluating the system are relevance, fulfillment of objectives, developmental efficiency, effectiveness, impact and sustainability. Evaluation should provide credible and useful information to enable the incorporation of lessons learned into the decision-making process (recipients and donors). Evaluation begins with target audience and system assessment. The baseline of the proposed project will help you to measure the effect that the delivered system has on your audience. The outcome of your program has no value if you do not know the target audience (such as admin side users, common users or database users). Process evaluation can find problems in the developed system. It includes an assessment of the staff, budget review, and how well the system has been doing overall. The following are the objectives of system evaluation:

- To know how to use different methods of evaluation.
- Be able to match your evaluation methods with your program objectives.
- To know where to apply methods for evaluation in the different stages of system development.

Impact evaluation can tell if the program has a short term effect on the behaviour, knowledge and attitudes of the target audience. It also measures the extent to which you have met your objectives. Outcome evaluation looks to see if the long-term program goals have been met. Evaluation helps you to:

- See whether program objectives have been met.
- Document the strengths and weaknesses of the program.
- Maintain data for keeping good financial records.
- Improve staff member skills in planning, conducting and evaluating activities.
- Identify hypotheses about behaviour for future evaluation.

As with the project selection process, the real criteria used to assess the projects will be varied by the organization. An important project evaluation method that is widely used for assessing information systems development projects is known as value chain analysis. Performance evaluation is determined by value chain analysis including technical feasibility, cost-benefit analysis and risk evaluation. They

collectively analyse the services which are used to determine where value is added and costs are incurred in the project management phase. Table 14.13 summarizes the objectives and result of evaluation process.

NOTES

Table 14.13 *Objectives and Result of Evaluation Process*

Objective	Result	Evaluation
Program Objectives	Changes in methods, modules and programs (input design, output design and screen design)	What is the outcome?
Bahavioural Objective	Changes in behavioural adaption	What is the impact? Can a new interface be designed?
Learner Objectives	Changes in knowledge and skill of particular domain, such as application development side, database designing side or networking side	Is there the requisite change in application?
Process Objective	Adherence to timeline tasks and efficient use of resources	Is the system working?

Check Your Progress

8. Explain the term hardware selection.
9. What are the main hardware requirements for developing a Web environment?
10. Define the term mandatory attribute.
11. Name the various phase for selection process.
12. Write the function of auction software.
13. What is risk analysis? Name the two key factors of risk analysis.
14. What is virus?

14.5 ANSWERS TO CHECK YOUR PROGRESS QUESTIONS

1. A software maintenance framework is created to determine the affects of these factors on maintenance. This framework comprises user requirements, organizational and operational environments, maintenance process, maintenance personnel and the software product.
2. The operational environment includes software systems, such as operating systems database systems and compilers and hardware systems such as processor, memory, peripherals.

3. Various functions performed by the software maintenance team are:
   - Locating information in system documentation.
   - Keeping system documentation up-to-date.
   - Extending existing functions to accommodate new or changing requirements.
   - Adding new functions to the system.
   - Finding the source of system failures or problems.
   - Managing changes to the system as they are made.
4. There are four types of maintenance, namely, corrective, adaptive, perfective and preventive. Corrective maintenance is concerned with fixing reported errors in the software. Adaptive maintenance means changing the software to some new environment, such as adapting a new version of an operating system. Perfective maintenance involves implementing new functional or non-functional requirements. Preventive maintenance involves implementing changes to prevent occurrence of errors.
5. Perfective maintenance mainly deals with implementing new or changed user requirements. Perfective maintenance involves making functional enhancements to the system and activities to increase the system's performance even when the changes have not been suggested by faults.
6. Regression testing (a type of system testing) is performed on the modified system to validate that the modified code does not introduce faults which did not exist prior to the maintenance activity.
7. The process attribute includes the procedures for performing acceptance test at functional level and testing the system to determine its performance and operation.
8. The hardware selection is essential task in the phase of system designing because system analyst checks thoroughly the comprehensive platform for developed system.
9. The minimum hardware requirements for developing the Web environment are as follows:
   - Web application server: At least 2 GHz processor, 1 GB RAM and 30GB capacity is required.
   - Database server: At least 2 GHz processor, 1 GB RAM, 100 GB RAID 4 + disk capacity are required.
   - File-server: At least 2 GHz processor, 512 MB RAM, RAID 5 storage with appropriate disk space are required.
   - Application cluster: Application-dependent sizing are required.

NOTES

10. Mendatory attribute defines that the hardware to be acquired must possess. Vendors who can not offer products along with all these typical attributes can be removed from considerations.
11. There are six phases of selection process:
    1. Requirements Analysis
    2. System Specifications
    3. Request For Proposal (RFP)
    4. Evaluation and Validation
    5. Vendor Selection
    6. Post-Installation Review
12. The auction software allows notifications without ongoing marketing cost. It is also very crucial phase that the auction software analyses the bidding patterns and trends to make subsequent auctions more effective.
13. The risk analysis helps to analyse the probability of problem occurrence, cost of each possibility of system damage and the preventive measures to include as part of security plan. Two key factors of risk analysis are the value of potential loss and the probability of loss.
14. A virus refers to a computer program which enters in a computer without the knowledge of the user. In a virus infected file, virus is hidden inside a program contained in the file.

14.6 SUMMARY

- A software maintenance framework is created to determine the affects of these factors on maintenance. This framework comprises user requirements, organizational and operational environments, maintenance process, maintenance personnel and the software product.
- If user requirements need major changes in the software, a lot of time may be consumed in implementing them. Similarly, users may opt for changes that are not according to the software standards or policies of a company.
- User requirements are feasible if the requested change is workable in the software system.
- The operational environment includes software systems, such as operating systems database systems and compilers and hardware systems such as processor, memory, peripherals.

NOTES

- In both the environments, scheduling of the maintenance process can create problems.
- Changes are implemented in the software system by following the software maintenance process (also known as software maintenance life cycle).
- Requirements and user problems become clear only when a system is in use. Also users may not be able to express their requirements in a form which is understandable to the analyst or programmer.
- There are four types of maintenance, namely, corrective, adaptive, perfective and preventive. Corrective maintenance is concerned with fixing reported errors in the software. Adaptive maintenance means changing the software to some new environment, such as adapting a new version of an operating system. Perfective maintenance involves implementing new functional or non-functional requirements. Preventive maintenance involves implementing changes to prevent occurrence of errors.
- Corrective maintenance deals with the repair of faults or defects found in day-to-day system functions.
- Logic errors result from invalid tests and conclusions, incorrect implementation of design specifications, faulty logic flow or incomplete test of data. All these errors, referred to as residual errors, prevent the software from conforming to its agreed specification. Note that the need for corrective maintenance is usually initiated by bug reports drawn by the users.
- Perfective maintenance mainly deals with implementing new or changed user requirements. Perfective maintenance involves making functional enhancements to the system and activities to increase the system's performance even when the changes have not been suggested by faults.
- Preventive maintenance is the maintenance performed for the purpose of preventing problems before they occur. The objective is to make programs easier to understand.
- Preventive software maintenance requires an investment of resources in the system.
- As stated earlier, changes are implemented in the software system by following a software maintenance process, which is known as Software Maintenance Life Cycle (SMLC).
- After the process attribute begins the control attribute which uniquely determines the identified MR and enters it into a repository.
- The analysis phase uses the repository information and the validated MR, along with system and project documentation, to study the feasibility and scope of the modification and devise a preliminary plan for design, implementation, test and delivery.

NOTES

- The control attribute of this phase concentrates on review of the project documentation and review of the modification request to evaluate technical and economic feasibility. In addition, this attribute verifies that project documentation is updated according to the analysis of modification request.
- In the design phase, all current system and project documentation, existing software and databases and the output of the analysis phase will be used to design the modification in the system.
- The implementation phase is concerned with making actual modifications in the software code, adding new features that support the specification of present software and finally installing the modified software.
- The user requirements as identified in design phase are implemented in the form of a software code. After the software code is written, unit testing on individual modules is performed.
- When the software code is written and all modules are tested, the modified software is integrated with the system. After integration of the software system, integration test and regression test are performed. These tests are conducted to identify the impact on the functionality, performance, usability, and security of the modified software system.
- Conducting software inspections of the code in compliance with IEEE standard.
- Ensuring that test documentation, such as test plan and test cases are either updated or created.
- Verifying that the new software is placed under Supply Chain Management (SCM) control.
- Regression testing (a type of system testing) is performed on the modified system to validate that the modified code does not introduce faults which did not exist prior to the maintenance activity.
- In the process attribute, various tests, such as system functional test, interface testing, and regression testing are performed on a fully integrated system. In the *control* attribute, system test is conducted by a separate test function.
- The process attribute includes the procedures for performing acceptance test at functional level and testing the system to determine its performance and operation.
- The delivery phase is concerned with the delivery of a modified software system to the user. In addition, users are provided with a proper documentation consisting of manuals and help files that describe the operation of the software along with its hardware specifications.

NOTES

- Allocation of resources and their management is important for a project where resources have to be shared among concurrent projects.
- Technical feasibility or technical assessment of the proposed system is to evaluate the required functionality with the available hardware and software infrastructure.
- Cost-Benefit Analysis or CBA is a widely used popular method wherein we add all the values of benefits of a course of action and then subtract the value of all the costs associated with it.
- CB analysis is computed using mostly financial costs and financial benefits.
- CB analysis is usually simple in case of small projects but it becomes an extremely complex and sophisticated art in case of large scale projects or industries.
- Costs and income are plotted on the graph of quantity of output as against the costs of input.
- All costs and benefits are identified and estimated for carrying out the project and operating the delivered operation. The main components, such as development cost, operating cost and benefits (direct and indirect) are expected to occur from the new system.
- Cost-benefit analysis is a widely used, popular tool which is relatively easy to use and helps in deciding if a change needs to be made.
- For using the tool, first you work out expenses for the change. Then calculate the benefit that you will derive from it.
- Financial cost and benefits are taken into account for carrying out the cost-benefit analysis. However, we can include intangible items in the analysis. If this is the case, then we must estimate a value for these which inevitably bring out an element of subjectivity into the process.
- The hardware and software selection is the very crucial and important phase in System Development Life Cycle (SDLC). The success rate of system design and implementation phase depends on this factor.
- The information-based system finalizes the information requirements to check out the vendor. The developed system is not complete until software is installed properly, tested and up-and-running. The hardware and software selection plays an important role in SDLC because of data security and system maintenance.
- The hardware selection is essential task in the phase of system designing because system analyst checks thoroughly the comprehensive platform for developed system.

NOTES

- The system application server software requires the hardware and software requirements.
- The hardware and software requirements during the system installation phase enables networking capabilities and fast internet connection and that are required to access an HTTP server.
- The optional software application might be required before the installation and deployment of SimManager in which Lightweight Directory Access Protocol (LDAP) compatible directory service, such as active directory and Netscape directory service.
- Attribute defines that the hardware to be acquired must possess. Vendors who can not offer products along with all these typical attributes can be removed from considerations.
- Your IT strategy should define the generic type of hardware that your business needs, and a timetable for its acquisition. Consider upgrade costs, warranties, what level of support you want and the maintenance contract with your supplier. These will affect pricing and ongoing costs.
- For PCs, servers and printers, you will need network infrastructure, such as network switches and all the wiring needed to connect equipment.
- Software selection is a critical aspect of system development. The search starts with the software, followed by the hardware. There are two ways of acquiring software: custom – made or "Off – the –shelf" packages.
- Hardware may become inoperative because of design errors, manufacturing errors, or deterioration caused by heat, humidity, friction, and the like. In contrast, software does not fail or wear out.
- Software reliability brings up the concept of modularity, or the ease with which a package can be modified. This depends on whether the package was originally designed as a package or was retrofitted after its original development for single installation use.
- Specify the magnitude of the problem, that is, clarify whether selections consist of a few peripherals or major decision concerning the mainframes.
- Assess the competence of the in-house staff. This involves determining the expertise needed in areas such as telecommunications and data base design. Acquiring a computer often results in securing temporary help for conversion. Planning for this help is extremely important.
- Consider hardware and software as a package. This approach ensures compatibility. In fact, software should be considered first, because often the user secures the hardware and then wonders what software is available for it.

NOTES

- Develop a schedule for the selection process. Maintaining a schedule helps keeps the project under control.
- Provide user indoctrination. This is crucial, especially for first-time users. Selling the system to the user staff, providing adequate training, and preparing an environment a conductive to implementation are pre- requisites for system acquisition.
- The first step in selection is understanding the user's requirements within the framework of the organization's objectives and the environment in which the system is being installed.
- The evaluation phase ranks vendor proposals and determines the best suited to the user's needs. It looks into items such as price, availability and technical support. System validation ensures that the vendor can match his/her claims, system performance.
- Sometime after the package is installed, a system evaluation is made to determine how closely the new system conforms to plan.
- It is the probability that the software will executed in a specific period of time without any failures. It is important to the professional user. It brings up the concept of modularity, or the ease which a package can be modified.
- The financial consideration works with various software development projects which is associated mainly with elapsed time for estimating the total cost.
- The financial aspects in software selection review refer to the potential software suppliers maintenance cost.
- Usually, a procedure is used for this as step wise selection that leads to cost recovery process if software fails to fulfill the requirement.
- The Datasoft Company considers the system requirements, such as ERP addresses, like client-server distributed architecture, object oriented programming software and RDBMS supported software.
- The auction software allows notifications without ongoing marketing cost. It is also very crucial phase that the auction software analyses the bidding patterns and trends to make subsequent auctions more effective.
- System security can be divided into different categories according to security issues, which are related to the file structure, data structure and access procedures of system.
- The risk analysis helps to analyse the probability of problem occurrence, cost of each possibility of system damage and the preventive measures to include as part of security plan. Two key factors of risk analysis are the value of potential loss and the probability of loss.

NOTES

- A virus refers to a computer program which enters in a computer without the knowledge of the user. In a virus infected file, virus is hidden inside a program contained in the file.
- System audit refers to a technical assessment of a system or application. Personal computers, servers and mainframes are the example of system whereas database and Web services are the examples of applications.
- Evaluation is an assessment that refers to design, implementation and results of completed or on-going project, program and policy.
- Impact evaluation can tell if the program has a short term effect on the behaviour, knowledge and attitudes of the target audience.
- An important project evaluation method that is widely used for assessing information systems development projects is known as value chain analysis.

14.7 KEY WORDS

- **Feasible:** User requirements are feasible if the requested changes is workable in the software system.
- **Adaptive maintenance:** The implementation of changes in a part of the System.
- **Benchmarking:** Benchmarking with reference to system analysis and design is a performance tool that involves the continuous, systematic evaluation and comparison of the capabilities of an organization with that of others.
- **Semiconductor memory:** Also known as integrated-circuit memory, large-scale integrated memory, memory chip, semiconductor storage and transistor memory, it is a device used for storing digital information fabricated by using integrated circuit technology.
- **Virus:** A computer program that can replicate itself and spread from one computer to another.

14.8 SELF ASSESSMENT QUESTIONS AND EXERCISES

Short-Answer Questions

1. What is software maintenance?
2. Define the concept of corrective maintenance.

3. Explain the term adaptive maintenance.
4. Write the feature of preventive maintenance.
5. What is problem analysis phase?
6. Give the definition of allocation resources.
7. Apart from Netscape 6.2, what are the other major client-side browsers used for code development?
8. Define the term desirable attribute.
9. State about the computer software.
10. Elaborate on the term requirement analysis.
11. Explain the term vendor selection.
12. State about the system security.
13. Elaborate on the term risk analysis.
14. What does system audit refers to?

NOTES

Long-Answer Questions

1. Discuss about the types of software maintenance with the help of diagram.
2. Illustrate software maintenance life cycle with the help of diagram.
3. Both hardware and software are equally important in selection. Support your answer with the help of examples.
4. What are the criteria for selection of appropriate hardware and software for system analysis and design.
5. Differentiate between computer hardware and software giving examples.
6. Briefly explain the procedure for hardware and selection and major phase of selection.
7. Describe the criteria for software selection and evaluation process with the help of examples.
8. Explain the characterisitc features of the following:
   a) User requirement
   b) Organizational and operational environment
   c) Allocation resource
   d) Implementation and evaluation

14.9 FURTHER READINGS

NOTES

Award, Elias M. 1991. *System Analysis and Design*. New Delhi: Galgotia Publications Pvt. Ltd.

Senn, James A. 1989. *Analysis and Design of Information Systems*. New York: McGraw Hill.

Hawryszkiewycz, Igor T. 2001. *Introduction to Systems Analysis and Design*, 5th Edition. New Jersey: Prentice Hall.

Rajaraman, V. 2011. *Analysis and Design of Information Systems*, 2nd Edition. New Delhi: PHI Learning Pvt. Ltd.

Whitten, Jeffrey L. and Lonnie D. Bentley. 2007. *Systems Analysis and Design Methods*, 7th Edition. New York: McGraw Hill.