
UNIT 7 SYSTEM DESIGN AND MODELING

Structure	Page Nos.
7.0 Introduction	31
7.1 Objectives	31
7.2 Logical and Physical Design	31
7.3 Process Modeling	36
7.3.1 Data Flow Diagrams	
7.4 Data Modeling	38
7.4.1 E-R Diagrams	
7.5 Process Specification Tools	39
7.5.1 Decision Tables	
7.5.2 Decision Trees	
7.5.3 Structured English Notation	
7.6 Data Dictionary	43
7.7 Summary	44
7.8 Solutions/Answers	44
7.9 Further Readings	47

7.0 INTRODUCTION

System Design is the specification or construction of a technical, computer based solution for the business requirements identified in system analysis phase. During design, system analysts convert the description of the recommended alternative solution into logical and then physical system specifications. S/he must design all aspects of the system from input and output screens to reports, databases, and computer processes. Databases are designed with the help of data modeling tools (for example, E-R diagrams) and computer processes are designed with the help of Structured English notation, Decision Trees and Decision Tables. Logical design is not tied to any specific hardware and software platform. The idea is to make sure that the system functions as intended. Logical design concentrates on the business aspects of the system. In physical design, logical design is converted to physical or technical specifications. During physical design, the analyst's team takes decisions regarding the programming language, database management system, hardware platform, operating system and networking environment to be used. At this stage, any new hardware or software can also be purchased. The final output of design phase is the system specifications in a form ready to be turned over to programmers and other system builders for construction (coding).

7.1 OBJECTIVES

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

- design major components of the system;
- identify tools for every component;
- to learn the process of using tools; and
- integrate component designs into a whole system specification.

7.2 LOGICAL AND PHYSICAL DESIGN

Logical Design is the phase of system development life cycle in which system analyst and user develops concrete understanding of the operation of the system. Figure 7.1 depicts various steps involved in the logical design. It includes the following steps:

- designing forms (hard copy and computer displays) and reports, which describe how data will appear to users in system inputs and outputs;

- designing interfaces and dialogues, which describe the pattern of interaction between users and software; and
- designing logical databases, which describe a standard structure for the database of a system that is easy to implement in a variety of database technologies.

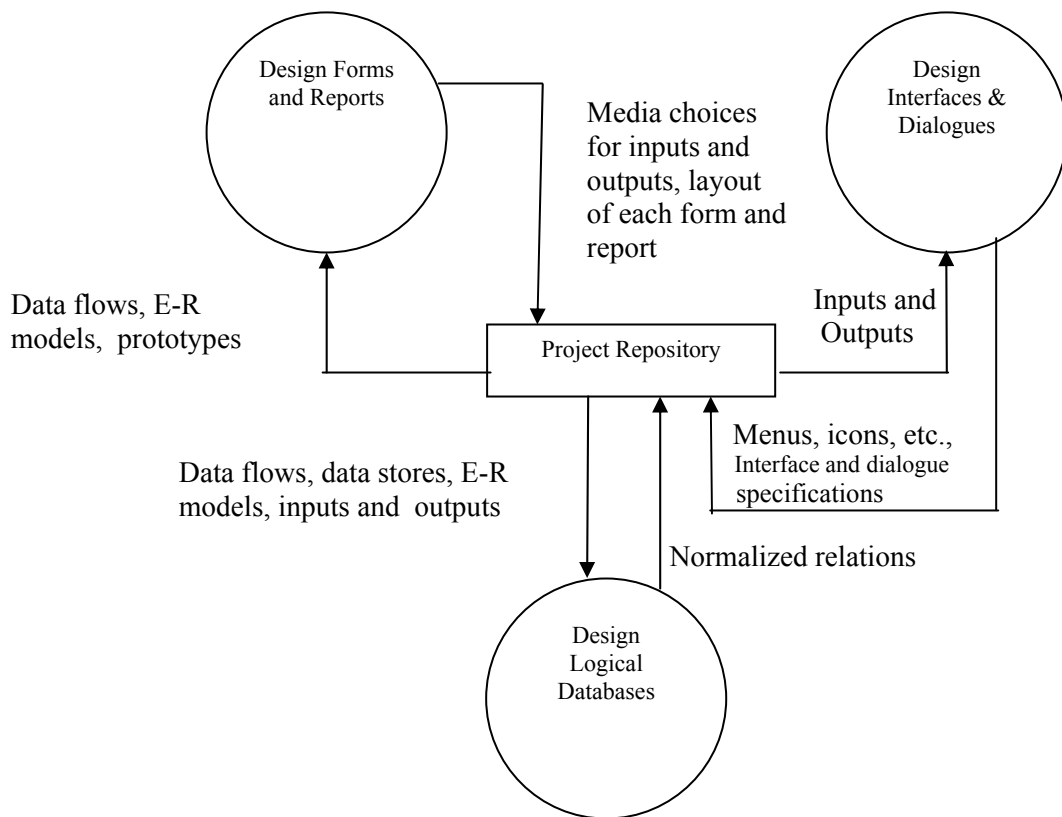


Figure 7.1: Steps in Logical Design

In logical design, all functional features of the system chosen for development in analysis are described independently of any computer platform. Logical design is tightly linked to previous system development phases, especially analysis. The three sub phases mentioned in the figure 7.1 are not necessarily sequential. The project dictionary or CASE repository becomes an active and evolving component of system development management during logical design. The complete logical design must ensure that each logical design element is consistent with others and satisfactory to the end user.

Form Design

System inputs are designed through forms. The general principles for input design are:

1. Capture only variable data.
2. Do not capture data that can be calculated or stored in computer programs.
3. Use codes for appropriate attributes.
4. Include instructions for completing the form.
5. Data to enter should be sequenced.

Common GUI controls for Inputs

Text Box

It can allow for single or multiple lines of characters to be entered.

Radio Button

Radio buttons provide the user with an easy way to quickly identify and select a particular value from a value set. A radio button consists of a small circle and an associated textual description those responses to the value choice. Radio buttons normally appear in groups as one radio buttons per value choice.

Check Box

It consists of a square box followed by a textual description of the input field for which the user is to provide the Yes/No value.

List Box

A list box is a control that requires the user to select a data item's value from the list of possible choices. It is rectangular and contains one or more rows of possible data values. The values may appear as either a textual description or graphical representation.

Dropdown List

It is like a list box, but is intended to suggest the existence of hidden list of possible values for a data item.

Combination Box

It is also known as combo box. It combines the capabilities of a text box and list box. A combo box gives the user, the flexibility of entering a data item's value or selecting its value from a list.

Spin Box

A spin box is used to allow the user to make an input selection by using the buttons to navigate through a small set of meaningful choices.

The following steps are to be followed during the process of input design are given below:

1. Identify system inputs and review logical requirements.
2. Select appropriate GUI (Graphical User Interface) controls.
3. Design, validate and test inputs using some combination of:
 - i) Layout tools (e.g., hand sketches, printer/display layout chart, or CASE)
 - ii) Prototyping tools (e.g., spreadsheet, PC DBMS, 4GL)
4. If necessary, design the source document.

Reports Design

System outputs are designed through Reports. Outputs can be classified according to two characteristics:

- i) Their distribution inside or outside the organization and the people who read and use them; and
- ii) Their implementation method.

Types of outputs

- i) **Internal outputs:** Internal outputs are intended for the owners of the system and users within the organization. There are three sub-classes of internal outputs:
Detailed reports—Present information with little or no filtering or restrictions;
Summary reports—Categorize information for managers who do not want to go through details; and

Exception reports—Filter data before it is presented to the manager as information.

- ii) **External outputs:** These are intended for customers, suppliers, partners and regulatory agencies. They usually conclude or report on business transactions. Examples of external outputs are invoices, account statements, paycheques, course schedules, telephone bills, etc.

Implementation methods for outputs

The following are commonly used output formats.

- i) **Tabular output:** It presents information as rows and columns of text and numbers.
- ii) **Zoned output:** It places text and numbers into designated areas or boxes of a form or screen.
- iii) **Screen output:** It is the online display of information on a visual display device, such as CRT terminal or PC monitor.
- iv) **Graphic output:** It is the use of a picture to convey information in ways that demonstrate trends and relationships not easily seen in tabular output.

The following are various guidelines for output design:

- i) Computer outputs should be simple to read and interpret.
- ii) The timing of computer outputs is important.
- iii) The distribution of (or access to) computer outputs must be sufficient to assist all relevant system users.
- iv) The computer outputs must be acceptable to the system users who will receive them.

The steps to be followed during process of output design are given below:

- i) Identify system outputs and review logical requirements.
- ii) Specify physical output requirements.
- iii) As necessary, design any pre-printed external forms.
- iv) Design, validate and test outputs using some combination of:
 - a. Layout tools (e.g., hand sketches, printer/display layout chart, or CASE).
 - b. Prototyping tools (e.g., spreadsheet, PC DBMS, 4GL).
 - c. Code generating tools (e.g., report writer).

User interface design

User interface design is concerned with the dialogue between a user and the computer. It is concerned with everything from starting the system or logging into the system to the eventually presentation of desired inputs and outputs. The overall flow of screens and messages is called a dialogue.

The following are various guidelines for user interface design:

- i) The system user should always be aware of what to do next.
- ii) The screen should be formatted so that various types of information, instructions and messages always appear in the same general display area.
- iii) Messages, instructions or information should be displayed long enough to allow the system user to read them.
- iv) Use display attributes sparingly.
- v) Default values for fields and answers to be entered by the user should be specified.
- vi) A user should not be allowed to proceed without correcting an error.
- vii) The system user should never get an operating system message or fatal error.

- viii) If the user does something that could be catastrophic, the keyboard should be locked to prevent any further input, and an instruction to call the analyst or technical support should be displayed.

Logical Database Design

Data modeling is used for logical database design. A conceptual model of data used in an application is obtained by using an entity relationship model (E-R model). E-R model assists in designing relational databases. A relational database consists of a collection of relations relevant for a specified application. A relation is a table which depicts an entity set. Each column in the relation corresponds to an attribute of the entity. Each row contains a member of the entity set.

Normalization is a procedure used to transform a set of relations into another set which has some desirable properties. Normalization ensures that data in the database are not unnecessarily duplicated. It also ensures that addition and deletion of entity rows (or tuples) or change of individual attribute values do not lead to accidental loss of data or errors in database.

The steps to be followed during physical design are given below:

- Designing physical files and databases — describes how data will be stored and accessed in secondary computer memory and how the quality of data will be ensured.
- Designing system and program structure — describes the various programs and program modules that correspond to data flow diagrams and other documentation developed in earlier phases of lifecycle.
- Designing distributed processing strategies — describes how your system will make data and processing available to users on computer networks within the capabilities of existing computer networks.

Figure 7.2 depicts various steps involved in physical design.

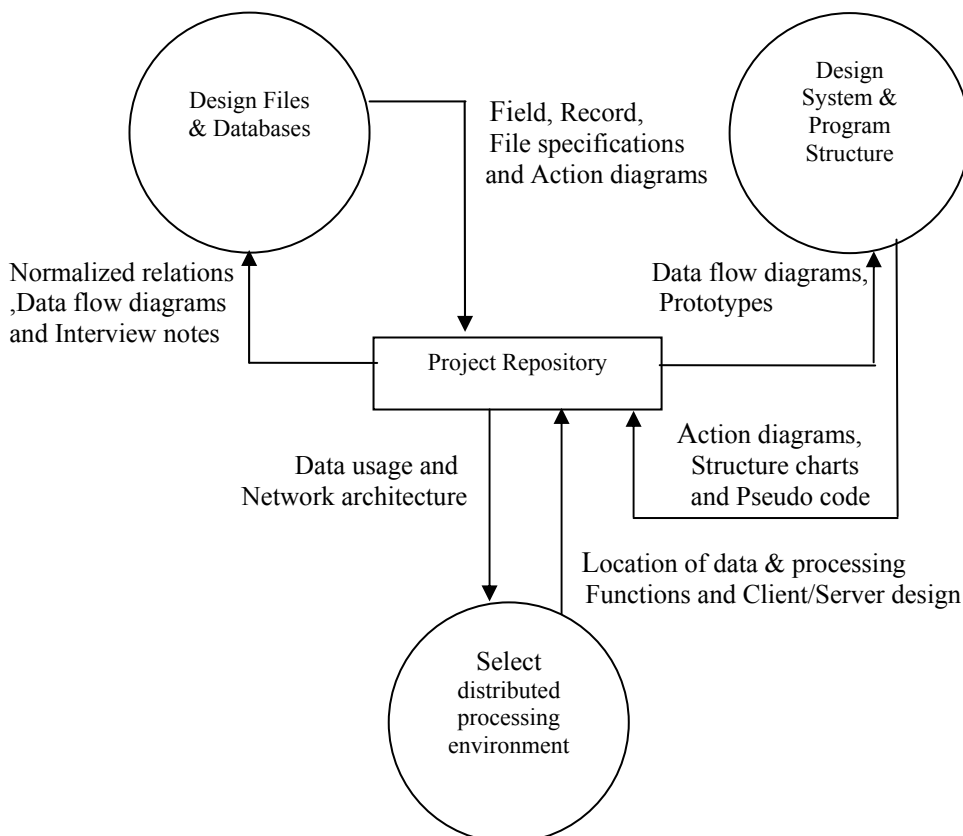


Figure 7.2: Steps in Physical Design

7.3 PROCESS MODELING

Process modeling involves graphically representing the functions or processes, which capture, manipulate, store and distribute data between a system and its environment and between components within a system. A common form of a process model is **data flow diagram**. It represents the system overview.

7.3.1 Data Flow Diagrams

A DFD can be categorized in the following forms:

Context diagram: An overview of an organizational system that shows the system boundaries, external entities that interact with the system and the major information flows between the entities and the system. In this diagram, a single process represents the whole system.

First level DFD: A data flow diagram that represents a system's major processes, data flows, and data stores at a high level of detail.

Functional decomposition diagram: Functional decomposition is an iterative process of breaking the description of a system down into finer and finer detail which create a set of charts in which one process on a given chart is explained in greater detail on another chart. In this diagram, sub-processes of first level DFD are explained in detail.

There is no limit on the number of levels of Data Flow Diagrams that can be drawn. It depends on the project at hand.

The following are various components of a Data Flow Diagram:

1. Process

During a process, the input data is acted upon by various instructions whose result is transformed data. The transformed data may be stored or distributed. When modeling the data processing of a system, it doesn't matter whether the process is performed manually or by a computer. People, procedures or devices can be used as processes that use or produce (transform) data.

The notation (given by Yourdon) for process is



2. Data Flow

Data moves in a specific direction from a point of origin to point of destination in the form of a document, letter, telephone call or virtually any other medium. The data flow is a "packet" of data.

The notation (given by Yourdon) for data flow is



3. Source or sink of data

The origin and /or destination of data some times referred to as external entities. These external entities may be people, programs, organization or other entities that interact with the system but are outside its boundaries. The term source and sink are interchangeable with origin and destination.

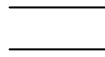
The notation (given by Yourdon) for source or sink is



4. Data store

A data store is data at rest, which may take the form of many different physical representations. They are referenced by a process in the system. The data store may reference computerized or non-computerized devices.

Notation (given by Yourdon) for data store is



Rules for drawing a data flow diagram:

1. For process:
 - i. No process can have only outputs.
 - ii. No process can have only inputs.
 - iii. A process has a verb phrase label.
2. For Data Store:
 - i. Data cannot move directly from one data store to another data store. Data must be moved through a process.
 - ii. Data cannot move directly from an outside source to data store. Data must be moved through a process that receives data from the source and places it into the data store.
 - iii. Data cannot move directly to an outside sink from a data store. Data must be moved through a process.

A data store has a noun phrase label.
3. For source/sink:
 - i. Data cannot move directly from a source to a sink. It must be moved by a process
 - ii. A source/sink has a noun phrase label
4. For data flow:
 - i. A data flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read operation before an update.
 - ii. A data flow cannot go directly back to the same process it leaves. There must be at least one other process which handles the data flow, produces some other data flow and returns the original data flow to the beginning process.
 - iii. A data flow to a data store means update (delete or change).
 - iv. A data flow from a data store means retrieve or use.
 - v. A data flow has a noun phrase label.

Check Your Progress 1

1. Develop a context level DFD and First level DFD for the hospital pharmacy system describe in the following case study: “The pharmacy at Sanjeevni Hospital fills medical prescriptions for all patients and distributes these medications to the nurse stations responsible for the patient’s care. Medical prescriptions are written by doctors and sent to the pharmacy. A pharmacy technician reviews the prescriptions and sends them to the appropriate pharmacy

station. At each station, a pharmacist reviews the order, checks the patient file to determine the appropriateness of the prescriptions and fills the order. If the pharmacist does not fill the order, the prescribing doctor is contacted to discuss the situation. In this case the order may ultimately be filled or the doctor may write another prescription, depending on the outcome of the discussion. Once filled, a prescription level is generated listing the patient's name, the drug type and dosage, an expiration date and any special instruction. The level is placed on the drug container and the order is sent to the appropriate nurse station. The patient's admission number, the drug type, and the cost of the prescription are then sending to the billing department".

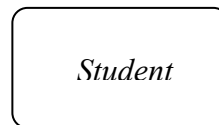
7.4 DATA MODELING

It is a technique for organizing and documenting a system's data. Data modeling is sometimes called database modeling because a data model is eventually implemented as database. It is also some times called information modeling. The tool for data modeling is entity relationship diagram.

7.4.1 ER Diagram

It depicts data in terms of entities and relationships described by the data. Martin gives the following notations for the components of ERD.

1. **Entities:** An entity is something about which the business needs to store data. An entity is a class of persons, places, objects, events or concepts about which we need to capture and store data. An entity instance is a single occurrence of an entity. The notation is given below:



Student is the name of entity.

2. **Attribute:** An attribute is a descriptive property or characteristic of an entity. Synonyms include element, property and field.
A compound attribute is one that actually consists of other attributes. It is also known as a composite attribute. An attribute "Address" is the example of compound attribute as shown in the following illustration.
3. **Relationships:** A relationship is a natural business association that exists between one or more entities. The relationship may represent an event that links the entities.

The following are some important terms related to ER diagrams:

Cardinality defines the minimum and maximum number of occurrences of one entity that may be related to a single occurrence of the other entity. Because all relationships are bidirectional, cardinality must be defined in both directions for every relationship. Figure 7.4 depicts various types of cardinality.

Degree: The degree of a relationship is the number of entities that participate in the relationship.

Recursive relationship: A relationship that exists between different instances of the same entity is called recursive relationship. Figure 7.3 depicts recursive relationship between the instances of the *Course* entity.

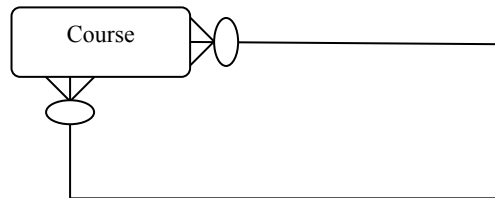


Figure 7.3: Example of Recursive relationship

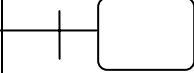
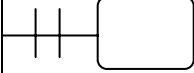
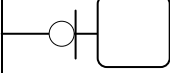
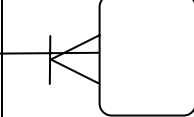
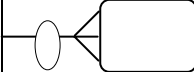
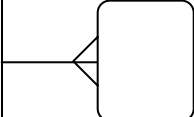
Cardinality Interpretation	Minimum Instances	Maximum Instances	Graphic Notation
Exactly one (One and only one)	1	1	 or 
Zero or one	0	1	
One or more	1	Many (>1)	
Zero, one or more	0	Many (>1)	
More than one	>1	>1	

Figure 7.4: Different types of cardinality

Check Your Progress 2

1. Draw an E-R diagram for the following case study:
 “In a purchasing department at one company, each purchase request is assigned to a “case worker” within the purchasing department. This caseworker follows the purchase request through the entire purchasing process and acts as the sole contact person with the person or unit buying the goods or services. The purchasing department refers to its fellow employees buying goods and services as “customers”. The purchasing process is such that purchase request must go to vendors. The product or service can simply be bought from any approved vendor, but the purchase request must still be approved by the purchasing department.”

7.5 PROCESS SPECIFICATION TOOLS

Processes in Data Flow Diagrams represent required tasks that are performed by the system. These tasks are performed in accordance with business policies and procedures.

Policy

A policy is a set of rules that govern some task or function in the business. Policies consist of rules that can often be translated into computer programs. Systems Analyst

along with representatives from the policy making organization can accurately convey those rules to the computer programmer for programming purposes.

Procedures

Procedures put the policies into action. Policies are implemented by procedures. Procedures represent the executable instructions in a computer program.

There are tools with the help of whom specification for policies can be created. They are Decision Table, Decision Tree and Software Engineering notations.

7.5.1 Decision Tables

Decision Table is very useful for specifying complex policies and decision-making rules. Figure 7.5 depicts a Decision table.

The following are various components of a Decision table:

Condition stubs: This portion of table describes the conditions or factors that will affect the decision or policy making of the organisation.

Action stubs: This portion describes the possible policy actions or decisions in the form of statements.

Rule: Rules describe which actions are to be taken under a specific combination of conditions.

Decision tables use a standard format and handle combinations of conditions in a very concise manner. Decision table also provides technique for identifying policy incompleteness and contradictions.

Rules									
Process Name		1	2	3	4	5	6	7	8
Conditions	_____	X	—

Actions	_____	—	X	?

X— Action (condition is true)

— Condition is irrelevant for this rule

? — Unknown rule

Figure 7.5: An example Decision Table

7.5.2 Decision Trees

Decision tree is a diagram that represents conditions and actions sequentially, and thus shows which conditions to consider first, and so on. It is also a method of showing the relationship each condition and permissible subsequent actions. The diagram resembles branches on a tree.

The root of the tree is the starting point of the decision sequence. The particular branch to be followed depends on the conditions that exist and decision that will be made. Progression from the left to right along a particular branch is the result of making a series of decisions. Following each decision point is the next set of decisions to be considered. The nodes of the tree thus represent conditions and indicate that a determination must be made about which condition exists before the next path can be chosen. Figure 7.6 depicts a Decision tree.

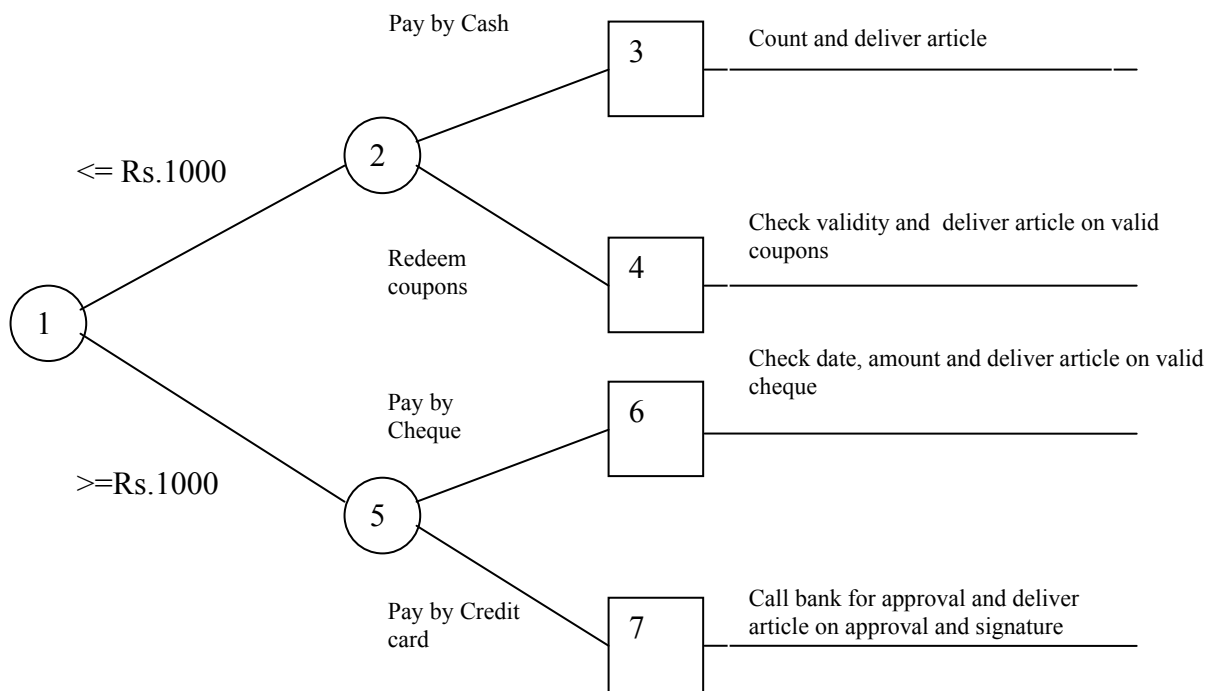


Figure 7.6: An Example Decision tree for a Customer Bill Payment System

7.5.3 Structured English Notation

It is a tool for describing process. There are three valid constructs in this notation. . They are:

1. A sequence of single declarative statements.
2. The selection of one or more declarative statements based on a decision, e.g., if-then-else, switch, case.
3. The repetition of one or more declarative statements, e.g., looping constructs such as do-while, for-do

The following are the guidelines usage of Structured English notation:

1. Avoid computer programming language verbs such a move, open or close.
2. The statement used in the Structured English Notation should always specify the formula to be used.

Structured English notation is based on the principles of structured programming. Process specification logic consists of a combination of sequences of one or more imperative sentences with decision and repetition constructs.

Consider the following:

1. An imperative sentence usually consists of an imperative verb followed by the contents of one or more data stores on which the verb operates.
For example, add PERSONS-SALARY TO TOTAL SALARY
2. In imperative sentences, verbs such as “process”, “handle” or “operate” should not be used.
3. Verbs should define precise activities such as “add” or compute average etc.
4. Adjectives that have no precise meaning such as “some” or “few” should also not be used in imperative sentences, because they cannot be used later to develop programs.
5. Boolean and arithmetic operations can be used in imperative statements. Table 7.1 lists the arithmetic and Boolean operators.

Table 7.1: Arithmetic and Boolean Operators

Arithmetic	Boolean
Multiply (*)	AND
Divide (/)	OR
Add (+)	NOT
Subtract (-)	Greater Than (>)
Exponentiate (**)	Less Than (<)
	Less Than or Equal To (<=)
	Greater Than or Equal to (>=)
	Equals to (=)
	Not equal to (≠)

6. Structured English logic:

Structured English uses certain keywords to group imperative sentences and define decision branches and iterations. These keywords are:

(BEGIN, END), (REPEAT, UNTIL), (IF, THEN, ELSE), (DO, WHILE), FOR, CASE, etc.

7. Grouping imperative sentences:

1) **Sequence Construct**

A sequence of imperative statements can be grouped by enclosing them with BEGIN and END keywords

2) **Decision (Selection)**

- a) A structure, which allows a choice between two groups of imperative sentences. The key words IF, THEN and ELSE are used in this structure. If a condition is 'true', then group 1 sentences are executed. If it is false, then group 2 sentences are executed.
- b) A structure which allows a choice between any number of groups of imperative sentences. The keywords CASE and OF are used in this structure. The value of a variable is first computed. The group of sentences that are selected for execution depends on that value.

3) **Repetition**

This structure shows two ways of specifying iterations in structured English.

- a) One way is to use the WHILE...DO structure. Here, the condition is tested before a set of sentences is processed.

Alternative to WHILE...DO is FOR structure.

- b) REPEAT...UNTIL structure. Here, a group of sentences is executed first then the condition is tested. So, in this structure, the group of sentences are executed at least once.

Table 7.2 depicts the criteria to be used for deciding the notation among Structured English, Decision Tables and Decision Trees. Table 7.3 depicts the criteria to be used for deciding the notation among Decision Tables and Decision Trees.

Table 7.2: Criteria for deciding the notation to be used

Criteria	Structured English	Decision Tables	Decision Trees
Determining condition & actions	2	3	1
Transforming conditions & actions into sequence	1	2	1
Checking consistency & completeness	2	1	1

Table 7.3: Criteria for deciding the notation to be used between Decision tables and Decision trees

Criteria	Decision Tables	Decision Trees
Portraying complex logic	Best	Worst
Portraying simple problems	Worst	Best
Making decisions	Worst	Best
More compact	Best	Worst
Easier to manipulate	Best	Worst

Check Your Progress 3

1. Draw a decision table for following policy statement:

“A bank offers two types of savings accounts, regular rate and split rate. The regular rate account pays dividends on the account balance at the end of each quarter. Funds withdrawn during the quarter earn no dividends. There is no minimum balance on the regular account. Regular rate account may be insured. Insured account gets 5.75 percent annual interest. Uninsured regular rate accounts get 6.00 percent annual interest.

For split rate accounts, dividends are paid monthly on the average daily balance for that month. Daily balances go up and down in accordance with deposits and withdrawals. The average daily balance is determined by adding each days closing balance and dividing this sum by the number of days in the month.

If the balance dropped below Rs.2000/- during month then no dividend is paid. So, if the average daily balance is less than Rs.2000/-, then no dividend is paid. Otherwise, if the average daily balance is Rs.2000/- or more, then an interest of 6% per annum is paid on the first Rs.5000/-, 6.5% on the next Rs.20000/- and 7% on funds over Rs.25000/-. There is no insurance on split rate.”

2. Draw a Decision Tree for the policy statement stated above in question no.1.

7.6 DATA DICTIONARY

A Data Dictionary consists of data about data. The major elements of data dictionary are data flows, data stores and processes. The data dictionary stores details and descriptions of these elements. It does not consist of actual data in the database. But, DBMS cannot access data in database with out accessing data dictionary.

If analysts want to know the other names by which a data item is referenced in the system or where it is used in the system, they should be able to find the answers in properly developed data dictionary. Data dictionaries are hidden from users so that data in it is not tampered.

Analysts use data dictionaries for the following reasons:

1. To manage the detail in large systems.
2. To communicate a common meaning for all system elements.
3. To document the features of the system.

4. To facilitate analysis of the details in order to evaluate characteristics and determine changes that should be made to the system.
5. To locate errors and omissions in the system.

The dictionary contains two types of descriptions for the data flowing through the system: Data elements and Data structures. Data elements are grouped together to make up a data structure.

Data elements are recorded in data dictionary at the fundamental data level. Each item is identified by a data name, description, alias and length and has specific values that are permissible for it in the system.

A data structure is a set of data items that are related to one another and then collectively describe a component in the system. Data is arranged in accordance with one of the relationships namely sequence, selection, iteration and optional relationship.

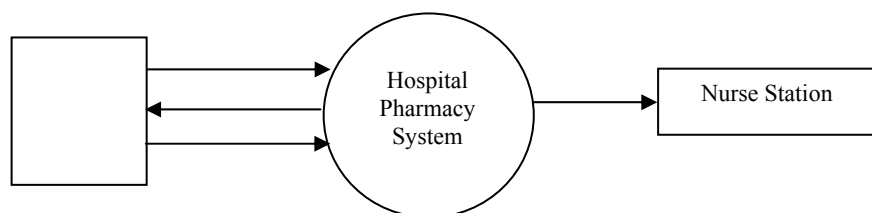
7.7 SUMMARY

Design is the phase that precedes coding. All the components of system are designed logically. Then physical aspects of the system are designed. Form design, report design, database design and program design etc. are designed with the help of GUI controls, process modeling tools, data modeling tools and process specification tools. These tools reduce the complexity of the design process. A Data Dictionary is data about data. These elements centre around data and the way they are structured to meet user requirements and organization's needs.

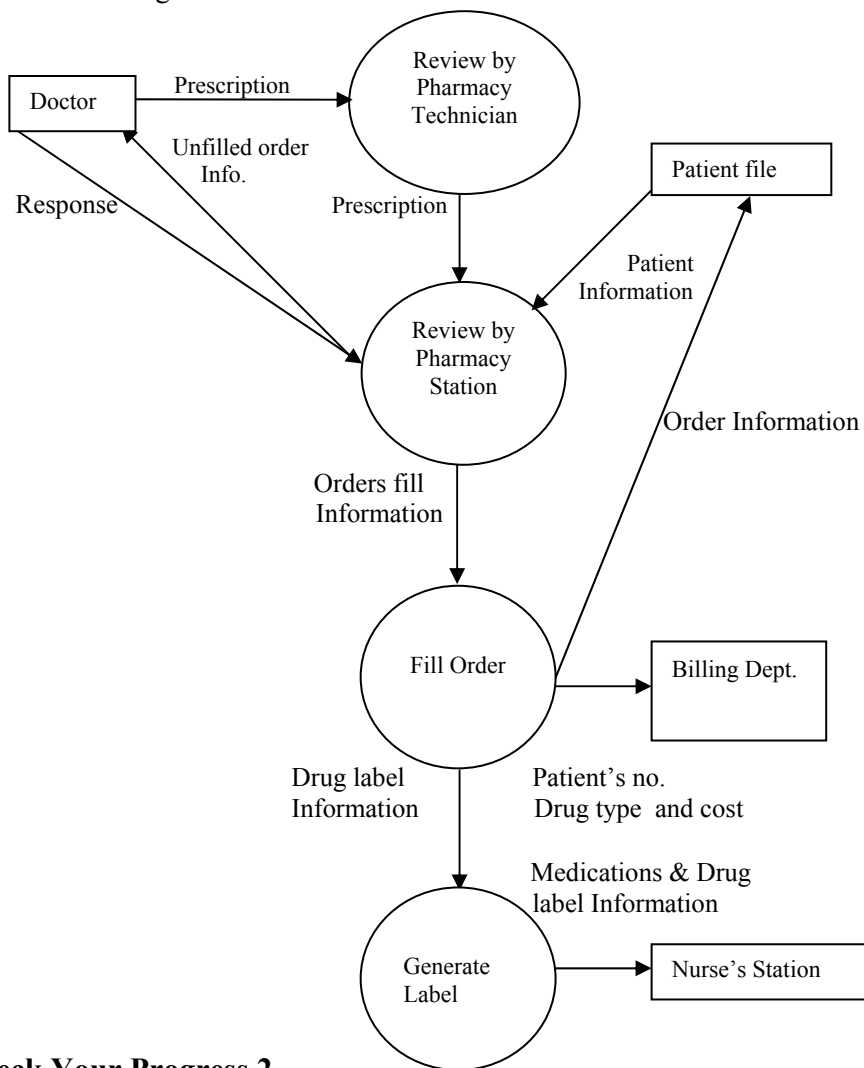
7.8 SOLUTIONS/ANSWERS

Check Your Progress 1

1. The following is the context level DFD:

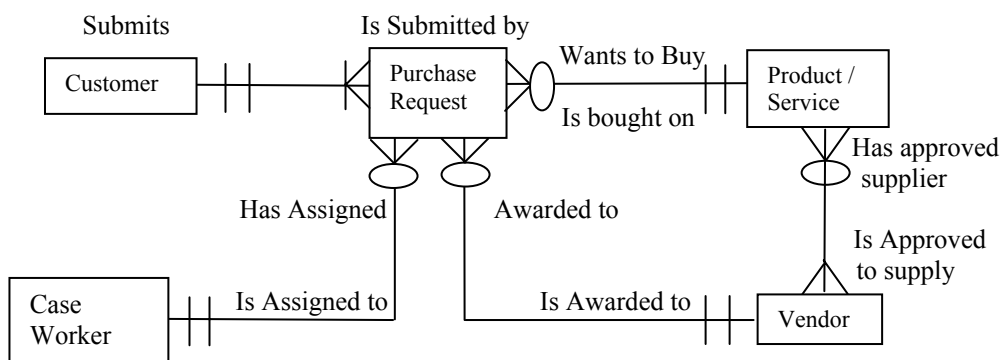


The following is the first level DFD:



Check Your Progress 2

1. The following is the ER diagram:



Check Your Progress 3

1. We have to identify conditions and values.

Data Elements Condition

1. Account type
2. Insurance

Values

R = Regular
S = split
Y = Yes
N = No

3. Balance Dropped below Rs.2000/- during month?

Y = Yes

Steps for Decision Table

To identify conditions (Data Elements) and their values.

1. To determine the maximum number of rules. The maximum number of rules in a decision table is calculated by multiplying the number of values for each condition data element.

Example: Condition 1 offers two values

Condition 1 offers two values

Condition 1 offers two values

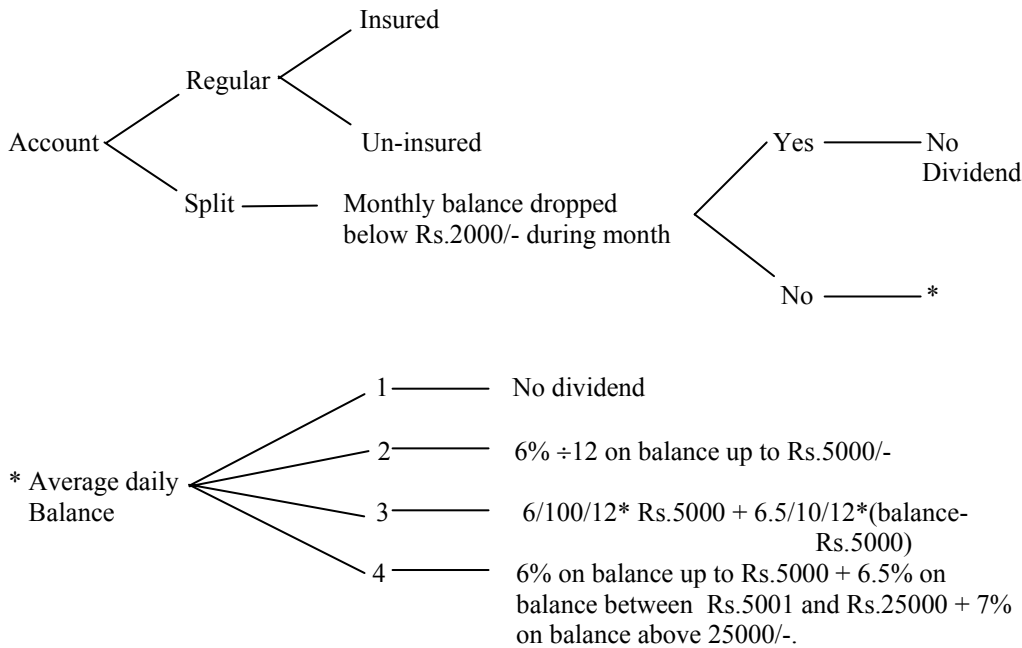
then, total number of rules = $2 \times 2 \times 2 = 8$

2. To identify the possible actions. It means to identify each independent action to be taken for the decision or policy.
3. To enter all possible rules and record all conditions and actions in their respective places in the decision table.
4. The way of defining rules:
 - a. For the first condition, to alternate its possible values Example: If Y and N are two possible values and there are 8 rules in the table then we will define these rules as:
YN YN YN YN
 - b. For condition two,
size of pattern that repeats in step (a) i.e. 2
 \Rightarrow YY NN YY NN
 - c. For condition three,
size of pattern that repeats in step (b) i.e., 4 so the possible combination is:
YYYY NNNN
5. After arranging all conditions, actions and rules
Now action will be taken according to rule i.e.
 - X : Action (correct rule)
 - : Irrelevant or indifference symbol
 - ? : Unknown rule
6. To verify the policy. Analyst can resolve any rules for which the actions are not specified. Analyst can also resolve apparent contradictions such as one rule with two possible actions.
8. Finally, Analysts must simplify the decision table:
 - * To eliminate impossible rules.
 - * The rules can be consolidated into a single rule for indifferent conditions where indifferent condition is a condition whose values do not affect the decision and always result in the same action.

The resultant Decision Table is given below:

Process Name		Dividend rate Rules						
		1	2	3	4	5	6	7
Conditions	Account type	R	R	S	S	S	S	S
	Insurance	Y	N	--	--	N	N	N
	Balance dropped below Rs.2000/- during month	--	--	Y	N	N	N	N
	Average daily balance	--	--	--	1	2	3	4
Actions	Pay no dividend			X	X			
	5.75% ÷ 4 quarterly dividend on entire balance.	X						
	6.000% ÷ 4 quarters		X					
	6.000% ÷ 12 monthly dividend on balance up to Rs.5000/-					X	X	X
	6.500% ÷ 12 monthly dividend on balance between Rs.5001/- to Rs.20000/-						X	X
	7.000% ÷ 12 monthly dividend on a balance of above Rs.25000/-							X

2.



7.9 FURTHER READINGS

Jeffrey L. Whitten, Lonnie D. Bentley and Kevin C. Dittman; *Systems Analysis and Design Methods*; Tata McGraw Hill Publishing Company Limited; Fifth Edition; 2000.

Jeffrey A. Hoffer, Joey F. George and Joseph S. Valacich; *Modern Systems Analysis and Design*; Pearson Education Publishing Company Limited; Third Edition; 2001.

V. Rajaraman; *Analysis and Design of Information Systems*; Prentice-Hall of India Private Limited; Second Edition.

Reference Websites

<http://www.rspa.com>

<http://www.ieee.org>