**SYSTEM DESIGN**

**6.1. INTRODUCTION**

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer’s goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word “Quality”. Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer’s view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

**6.2. NORMALIZATION**

It is a process of converting a relation to a standard form. The process is used to handle the problems that can arise due to data redundancy i.e. repetition of data in the database, maintain data integrity as well as handling problems that can arise due to insertion, updation, deletion anomalies.

Decomposing is the process of splitting relations into multiple relations to eliminate anomalies and maintain anomalies and maintain data integrity. To do this we use normal forms or rules for structuring relation.

**Insertion anomaly**: Inability to add data to the database due to absence of other data.

**Deletion anomaly**: Unintended loss of data due to deletion of other data.

**Update anomaly**: Data inconsistency resulting from data redundancy and partial update

**Normal Forms**: These are the rules for structuring relations that eliminate anomalies.

**FIRST NORMAL FORM**:

A relation is said to be in first normal form if the values in the relation are atomic for every attribute in the relation. By this we mean simply that no attribute value can be a set of values or, as it is sometimes expressed, a repeating group.

**SECOND NORMAL FORM**:

A relation is said to be in second Normal form is it is in first normal form and it should satisfy any one of the following rules.

1. Primary key is a not a composite primary key
2. No non key attributes are present
3. Every non key attribute is fully functionally dependent on full set of primary key.

**THIRD NORMAL FORM**:

A relation is said to be in third normal form if their exits no transitive dependencies.

**Transitive Dependency**: If two non key attributes depend on each other as well as on the primary key then they are said to be transitively dependent.

The above normalization principles were applied to decompose the data in multiple tables thereby making the data to be maintained in a consistent state.

**6.3. DATA DICTIONARY**

**Database Tables (Data Dictionary):** After careful analysis the system has identified to be presented with the following database tables:

**empdetails**

Name Null? Type

----------------------------------------------------- -------- ----------------

EMPID NUMBER

KEY VARCHAR2(10)

AC\_STATUS VARCHAR2(1)

DOJ VARCHAR2(10)

USERID VARCHAR2(25)

**address book**

Name Null? Type

----------------------------------------------------- -------- ----------------

NAME VARCHAR2(40)

USERID VARCHAR2(20)

DOB VARCHAR2(14)

EMAIL VARCHAR2(40)

PHONE VARCHAR2(16)

CITY VARCHAR2(30)

OUSERID VARCHAR2(25)

**advices**

Name Null? Type

----------------------------------------------------- -------- ----------------

NAME VARCHAR2(50)

PHONE VARCHAR2(14)

EMPID VARCHAR2(6)

ADDRESS VARCHAR2(40)

MESSAGE VARCHAR2(500)

**clientonline**

Name Null? Type

----------------------------------------------------- -------- ----------------

USERID VARCHAR2(25)

ROOMNAME VARCHAR2(20)

DATEOFLOG VARCHAR2(25)

TIMEOFLOG VARCHAR2(25)

**mail**

Name Null? Type

----------------------------------------------------- -------- ----------------

USERID VARCHAR2(25)

WHOSEND VARCHAR2(25)

MESSAGE LONG

TIMING VARCHAR2(50)

READ CHAR(1)

SUBJECT VARCHAR2(25)

**password**

Name Null? Type

----------------------------------------------------- -------- ----------------

USERID VARCHAR2(25)

PWD VARCHAR2(20)

BAN VARCHAR2(1)

**roomdetails**

Name Null? Type

----------------------------------------------------- -------- ----------------

ROOMNAME VARCHAR2(25)

RIN VARCHAR2(20)

MODERATOR VARCHAR2(25)

**userdetails**

Name Null? Type

----------------------------------------------------- -------- ----------------

USERID VARCHAR2(25)

NAME VARCHAR2(60)

ADDRESS VARCHAR2(120)

ZIP VARCHAR2(6)

PHONE VARCHAR2(12)

EMAIL VARCHAR2(50)

SEX VARCHAR2(1)

EDUCATION VARCHAR2(30)

COUNTRY VARCHAR2(25)

CITY VARCHAR2(20)

STATE VARCHAR2(25)

DOB VARCHAR2(12)

**security**

Name Null? Type

----------------------------------------------------- -------- ----------------

USERID VARCHAR2(25)

QUES VARCHAR2(100)

ANS VARCHAR2(50)

**6.4. E – R DIAGRAMS**

* + The relation upon the system is structure through a conceptual ER-Diagram, which not only specifics the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.
  + The Entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object descriptions.
  + The set of primary components that are identified by the ERD are
  + Data object
  + Relationships
  + Attributes
  + Various types of indicators.

The primary purpose of the ERD is to represent data objects and their relationships.

**ER-Diagram**

**ADMINISTRATOR:**

**ADMINISTRATOR**

**CLIENT**

**CLIENT:**

**6.5. DATA FLOW DIAGRAMS**

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD’S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consist a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a “bubble Chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

**DFD SYMBOLS:**

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data

Process that transforms data flow

Source or Destination of data

Data flow

Data Store

**CONSTRUCTING A DFD:**

Several rules of thumb are used in drawing DFD’S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numbered.
4. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

**SAILENT FEATURES OF DFD’S**

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

**TYPES OF DATA FLOW DIAGRAMS**

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

**CURRENT PHYSICAL:**

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

**CURRENT LOGICAL:**

The physical aspects at the system are removed as much as possible so that the current system is reduced to its essence to the data and the processors that transform them regardless of actual physical form.

**NEW LOGICAL**:

This is exactly like a current logical model if the user were completely happy with the user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

**NEW PHYSICAL:**

The new physical represents only the physical implementation of the new system.

**RULES GOVERNING THE DFD’S**

**PROCESS**

1. No process can have only outputs.
2. No process can have only inputs. If an object has only inputs than it must be a sink.
3. A process has a verb phrase label.

**DATA STORE**

1. Data cannot move directly from one data store to another data store, a process must move data.
2. Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
3. A data store has a noun phrase label.

**SOURCE OR SINK**

The origin and /or destination of data

1. Data cannot move direly from a source to sink it must be moved by a process
2. A source and /or sink has a noun phrase land

**DATA FLOW**

1. 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 before an update. The later is usually indicated however by two separate arrows since these happen at different type.
2. A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
3. A data flow cannot go directly back to the same process it leads. There must be at least one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
4. A Data flow to a data store means update (delete or change).
5. A data Flow from a data store means retrieve or use.

A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package

Architectural View of Global Communication Media:

Admin

Client

Client

Moderator

I Level DFD for Admin

Admin

Login Information

Verify Admin ID, Password

Return Message

I Level DFD for Client

Client

Login Information

Verify UserID

Password

Return Message

Storing Info

Invalid User

Valid User

II Level DFD for Client

Client

Password

Password

User Details

Invalid User

Userid

User Info

UserId&Pwd

Security

Hint Q&A

Valid User

Client Online

Room

Name

Room Details

Offline

Room Creation

Save Offline

Logout

Room table

Client online

Room table

Save online messages

Client

Password

User details

Addresses

II Level DFD for Admin

Admin

Password

Password

Userid

Password

Emp Details

Sitemap:

**MAIN**

**ADMIN**

**SITEMAP**

**SITEMAP**

**SITEMAP**

**SITEMAP**

**SUGGESTIONS**

**CLIENT**

**SUGGESTIONS**

**HELP**

**Change Password**

**Contacts**

**Conference**

**Statistics**

**Offline**

**Contacts**

**User**

**Maintenance**

**Contacts**

**Conference**

**Offline**

**Change Password**

**View Profiles of Others**

**Check User Status**

**Search Contact**

**View Profiles of Others**

**Add Employee**

**Check suggestions**

**6.6. ACTIVITY DIAGRAMS**

Admin Login

Verify Registration

Add Products

View Products

Search Products

Verify Requests

**6.7. USE CASE DIAGRAMS**

Administrator Use Case Diagram

ADMINISTRATOR

Create

Room

Add Client

Delete Client

Conference

CLIENT

MODERATOR

Or CLIENT

Login Verification

<<Extends>>

Password Verification

p

**6.8. SEQUENCE DIAGRAMS**

Administrator Sequence Diagram

Admin

Login

Add Employee

Check Sugge

stions

User Maintenance

Ener Id,Password

Verify

Succses

Add Employee Details

Succses

Check Suggestions

Succses

User Maintenance

Succses

Ban user

Delete

user

**6.9. CLASS DIAGRAMS**

DataManipulator

Login

InputScreen

Report

Menu

GUIComponent

Add Employee

DataStore

options

conference

Check Suggestion

Search

View status

<<instantiates>>

<<uses>>

Options

user maintence

**State Diagram:**

**( id,pwd )**

**authenticated**

Adminr Menu

**Validation**

unauthenticated

**( valid )**

**( invalid )**

**( username,pwd)**

Adminr Menu

**Validation**

unauthenticated

**( valid )**

**( invalid )**

**( username,pwd)**