Systems Design

- **Systems design** is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements.
- Systems design could be seen as the application of systems theory to product development
- System design is the process of defining the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system.
- Systems design implies a systematic approach to the design of a system. It may take a bottom-up or top-down approach, but either way the process is systematic.

The Processes and Stages of System Design

- Information systems design is defined as those tasks that focus on the specification of a detailed computer-based solution. It is also called **physical design**.
- Hence, systems design focuses on the technical or implementation concerns of the
- information system.
- The purpose of design is to transform the requirements represented in analysis phase into physical design specifications that will guide system construction.
- The design specification is used to write computer programs during implementation phase.
- In other words, the design phase addresses how technology will be used in the new system.
- Design process represents a specific technical solution of the information system.
- The design phase is concerned with designing **files and databases**, **forms and reports**, **dialogues and interfaces** and **system and program structure**.

Based on the user requirements and the detailed analysis of the existing system, the new system must be designed. This is the phase of system designing. It is the most crucial phase in the developments of a system.

Stages of System Design:

1. **Preliminary or General Design**:
    - In the preliminary or general design, the features of the new system are specified.
    - The costs of implementing these features and the benefits to be derived are estimated.
    - If the project is still considered to be feasible, we move to the detailed design stage.
2. **Structured or Detailed Design**:
    - In the detailed design stage, computer oriented work begins in earnest.
    - At this stage, the design of the system becomes more structured.
    - Structure design is a blue print of a computer system solution to a given problem having the same components and inter-relationships among the same components as the original problem.
    - Input, output, databases, forms, codification schemes and processing specifications are drawn up in detail.

In the design stage, the programming language and the hardware and software platform in which the new system will run are also decided. There are several tools and techniques used for describing the system design of the system. These tools and techniques are:

- Flowchart
- Data flow diagram (DFD)
- Data dictionary
- Structured English
- Decision table
- Decision tree
- ER Diagram

The system design involves:

a) Defining precisely the required system output.
b) Determining the data requirement for producing the output.
c) Determining the medium and format of files and databases.
d) Devising processing methods and use of software to produce output.
e) Determine the methods of data capture and data input.
f) Designing Input forms
g) Designing Codification Schemes
h) Detailed manual procedures
i) Documenting the Design

Logical and Physical Design

Logical design

- The logical design of a system is an abstract representation of the data flows, inputs and outputs of the system.
- This is often conducted using a graphical tool.
- Logical design includes ER Diagrams i.e. Entity Relationship Diagrams.

Physical design

- The physical design relates to the actual input and output processes of the system.
- This explains about how data is input into a system, how it is verified/authenticated, how it is processed, and how it is displayed.

In Physical design, the following requirements about the system are decided.

1. Input requirement,
2. Output requirements,
3. Storage requirements,
4. Processing Requirements,
5. System control and backup or recovery.

Physical design is broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them.

Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Structured design

- **Structured design** is the art of designing the components of a system and the interrelationship between those components in the best possible way.
- Structured Design is one of the methods used to define the system that is going to be live in future.
- This is used to analyze the working of the system, the different modules(functional units of the system), interactions between them and the type of data they are using or sharing to perform their various tasks correctly.
- All this can be represented by the use of special type of diagrams, known as Structured Chart.

Modular Design

- When designing a system synthetically, the system could be designed by two broad ways.
- The first way would be to design the complete system using the known theories, and use the system, as it is designed, in the real conditions.
- An alternative way would be to design the different components of the system separately, and test each component in separate conditions.
- **Modular design**, or "modularity in design", is an approach that subdivides a system into smaller parts (modules) that can be independently created and then used in different systems to drive multiple functionalities.

Structure Chart

- A structure chart (module chart, hierarchy chart) is a graphic depiction of the decomposition of a problem. It is a tool to aid in software *design*.
- It is particularly helpful on large problems.
- A structure chart illustrates the partitioning of a problem into sub problems and shows the hierarchical relationships among the parts.
- A classic "organization chart" for a company is an example of a structure chart.

- The top of the chart is a box representing the entire problem, the bottom of the chart shows a number of boxes representing the less complicated sub problems.
- A structure chart is NOT a flowchart. It has nothing to do with the logical sequence of tasks.
- It does NOT show the order in which tasks are performed.
- It does NOT illustrate an algorithm.

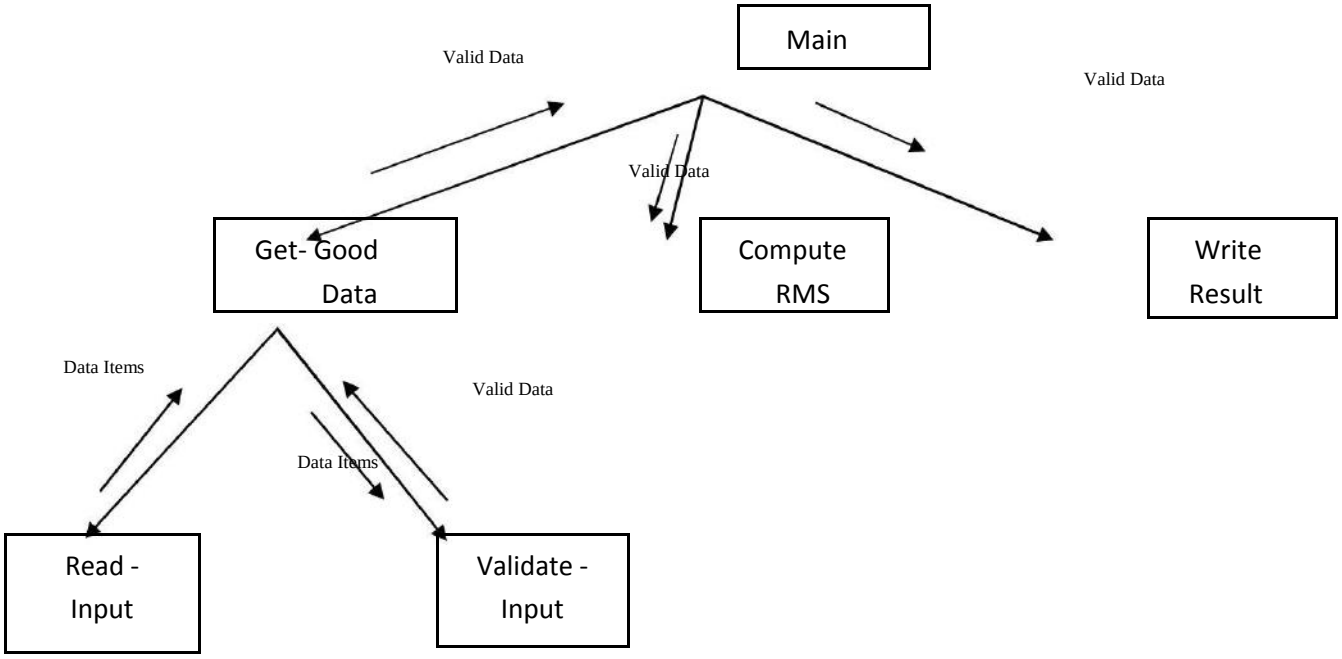


fig. Structure Chart

Functional Strength

- Functional strength of a module is the boundness within it. It is the strength by which a module is bounded.
- A module having high cohesion and low coupling is said to be functionally independent of other modules.

Cohesion

- Cohesion is a measure of functional strength of a module.
- A module having high cohesion and low coupling is said to be functionally independent of other modules.
- By the term functional independence, we mean that a cohesive module performs a single task or function.
- A functionally independent module has minimal interaction with other modules.

Classification of cohesion

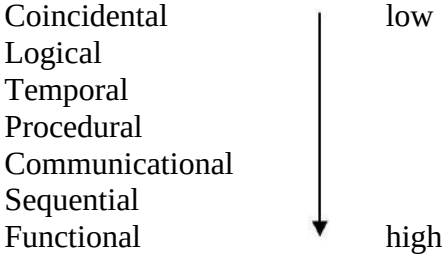


Fig**.** Classification of cohesion

Coincidental cohesion:

- A module is said to have coincidental cohesion, if it performs a set of tasks that relate to each other very loosely, if at all.
- In this case, the module contains a random collection of functions.
- It is likely that the functions have been put in the module out of pure coincidence without any thought or design.
- For example, in a transaction processing system (TPS), the get-input, print-error, and summarize-members functions are grouped into one module.
- The grouping does not have any relevance to the structure of the problem.

Logical cohesion:

- A module is said to be logically cohesive, if all elements of the module perform similar operations, e.g. error handling, data input, data output, etc.
- An example of logical cohesion is the case where a set of print functions generating different output reports are arranged into a single module.

Temporal cohesion:

- When a module contains functions that are related by the fact that all the functions must be executed in the same time span, the module is said to exhibit temporal cohesion.
- The set of functions responsible for initialization, start-up, shutdown of some process, etc. exhibit temporal cohesion.

Procedural cohesion:

- A module is said to possess procedural cohesion, if the set of functions of the module are all part of a procedure (algorithm) in which certain sequence of steps have to be carried out for achieving an objective, e.g. the algorithm for decoding a message.

Communicational cohesion:

- A module is said to have communicational cohesion, if all functions of the module refer to or update the same data structure, e.g. the set of functions defined on an array or a stack.

Sequential cohesion:

- A module is said to possess sequential cohesion, if the elements of a module form the parts of sequence, where the output from one element of the sequence is input to the next.

- For example, in a TPS, the get-input, validate-input, sort-input functions are grouped into one module.

Functional cohesion:

- Functional cohesion is said to exist, if different elements of a module cooperate to achieve a single function.
- For example, a module containing all the functions required to manage employees' pay-roll exhibits functional cohesion.
- Suppose a module exhibits functional cohesion and we are asked to describe what the module does, then we would be able to describe it using a single sentence.

Coupling

- Coupling between two modules is a measure of the degree of interdependence or interaction between the two modules.
- A module having high cohesion and low coupling is said to be functionally independent of other modules.
- If two modules interchange large amounts of data, then they are highly interdependent.
- The degree of coupling between two modules depends on their interface complexity.
- The interface complexity is basically determined by the number of types of parameters that are interchanged while invoking the functions of the module.

Classification of Coupling

Even if there are no techniques to precisely and quantitatively estimate the coupling between two modules, classification of the different types of coupling will help to quantitatively estimate the degree of coupling between two modules. Five types of coupling can occur between any two modules. This is shown in fig.

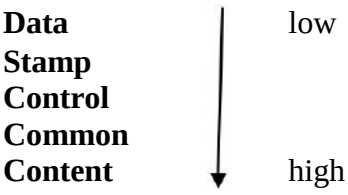


Fig. Classification of coupling

Data coupling:

- Two modules are data coupled, if they communicate through a parameter.
- An example is an elementary data item passed as a parameter between two modules, e.g. an integer, a float, a character, etc.

Stamp coupling:

- Two modules are stamp coupled, if they communicate using a composite data item such as a record in PASCAL or a structure in C.

Control coupling:

- Control coupling exists between two modules, if data from one module is used to direct the order of instructions execution in another.

- An example of control coupling is a flag set in one module and tested in another module.

Common coupling:

- Two modules are common coupled, if they share data through some global data items.

Content coupling:

- Content coupling exists between two modules, if they share code, e.g. a branch from one module into another module.

Database Design

Data: It is a collection of facts. It can be numbers, words, measurements, observations etc.

Information: It is a meaningful data.

Database: A database is a collection of data that is organized so that its contents can easily be accessed, managed and updated.

A database is a collection of logically related records.

DBMS: A DBMS is a collection of programs that enables you to store, modify and extrat information from a database.

Table: A table is a two-dimentional structure made up of rows (tuple, records) and columns (attributes, fileds). For example:

StudentID	Activity	Fee
100	Skiing	200
150	Swimming	50
175	Cricket	500
200	Swimming	50

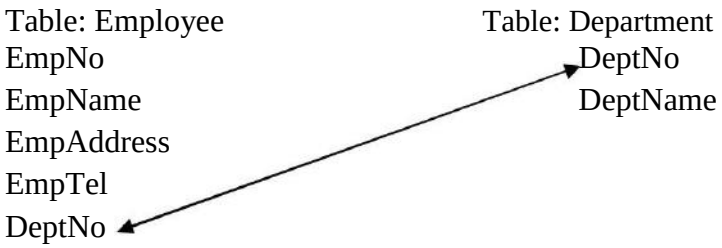
Primary Key: It is an attribute or a collection of attributes whose value(s) uniquely identify each row in a relation. A primary key should be minimal. It should not contain unneccessary attributes.

Composite Keys: A table can only have one primary key. But sometimes the primary key can be made up of several fields.

Candidate Key: Sometimes, there is more than one possible choice; each possible choice is called a candidate key.

Foreign Key: A foreign key is an attribute or a collection of attributes whose vlaue are intended to match the primary key of some related record (usually in a different table).

Example:



File Organizations

- A file is a collection of data, usually stored on disk.
- As a logical entity, a file enables you to divide your data into meaningful groups, for example, you can use one file to hold all of a company's product information and another to hold all of its personnel information.
- As a physical entity, a file should be considered in terms of its organization.

Types of Files

Sequential File

- A sequential file is one in which the individual records can only be accessed sequentially, that is, in the same order as they were originally written to the file.
- New records are always added to the end of the file.

Relative File

- A relative file is a file in which each record is identified by its ordinal position within the file (record 1, record 2 and so on).
- This means that records can be accessed randomly as well as sequentially.
- For sequential access, you simply execute a READ or WRITE statement to access the next record in the file.
- For random access, you must define a data-item as the relative key and then specify, in the data-item, the ordinal number of the record that you want to READ or WRITE.
- Because records can be accessed randomly, access to relative files is fast.

Indexed File

- An indexed file is a file in which each record includes a primary key.
- To distinguish one record from another, the value of the primary key must be unique for each record.
- Records can then be accessed randomly by specifying the value of the record's primary key. Indexed file records can also be accessed sequentially.
- As well as a primary key, indexed files can contain one or more additional keys known as alternate keys.
- The value of a record's alternate key(s) does not have to be unique.

Normalization

- It is a process that helps analysts or database designers to design table structures for an application.
- Normalization is to attempt to reduce redundant table data to the very minimum. Normalization is a technique that:
  - ✓ decomposes data into two-dimensional tables

- ✓ Eliminates any relationship in which table data does fully depend upon the primary key of a record.
- ✓ Eliminates any relationship that

contains dependencies.

Table: EmpProj

Field	Key	Type
Project No.	…	
Project Name	…	
EmpNo.	…	1-n
EmpName	…	1-n
Rate Category	…	1-n
Houly Rate	…	1-n

Project No.	**Project Name**	**EmpNo.**	**EmpName**	**Rate Category**	**Hourly Rate**
101	LMIS	E101	Shyam	A	500
101	LMIS	E102	Sita	B	400
102	HMIS	E203	Hari	A	500
102	HMIS	E204	Santi	B	400

First Normal Form

When a table is decomposed into two-dimensional tables with all repeating with groups of data eliminated, the table is said to be in its first normal form.

A table is said to be in first normal form if:

- There are no repeating groups
- All the key attributes are defined
- All attributes are dependent on a primary key

Table: EmpProj

Field	Key
Project No.	Primary Key
Project Name	…

EmpNo.	Primary Key
EmpName	…
Rate Category	…
Houly Rate	…

Second normal Form

A table is said to be in its second normal form when each record in the table is in the first normal form and each column in the record is fully depends on its primary key.

A table is in its second normal form if:

- It is in first normal form
- It includes no partial dependencies (Where an attribute is dependent on only a part of a primary key).

Table: Emp

Field	Key
EmpNo.	Primary Key
EmpName	…
Rate Category	…
Houly Rate	…

Table: Proj

Field	Key
Project No.	Primary Key
Project Name	…

Third Normal Form

Table data is said to be in third normal form when all transitive depencies are removed from this data.

The table is in third normal form if;

- It is in second normal form
- It contains no transitive dependencies.
- A, B, C are three columns in table.

If C is related to B and If B is related to A, then C is indirectly related to A.

This is when transitive dependency exists.

Table: EmpProj

Field	Key
Project No.	Primary Key
Employee No.	Primary Key

Table: Emp

Field	Key
EmpNo.	Primary Key
EmpName	…
Rate Category	…

Table: Rate

Field	Key
Rate Category	Primary Key
Houly Rate	…

Table: Proj

Field	Key
Project No.	Primary Key
Project Name	…

Input and Output Design

Input Design

Input design has six main objectives:
1. Select suitable input and data entry method
2. Reduce input volume
3. Design attractive data entry screens
4. Use validation checks to reduce input errors
5. Design required source documents
6. Develop effective input controls

There are two main data entry methods; batch and online input.

Batch input

· Data entry is performed on a specified time schedule
· Collection (batch) of data is input at one time

Online input

· Data is validated and available immediately
· Source data automation
· Combines online data entry with online data capture
· Uses magnetic data strips and swipe scanners
· Common examples: ATMS, point-of-sale terminals, bar code readers, patient ID bracelets, libraries

Screen Design

When designing data entry screens, form filling is the most effective method of online data entry since it resembles filling a paper-based form on the screen. Input data can be grouped into three:

· Data that must be entered by the user
· Data generated by the system
· Data calculated by the system

Some screen design guidelines are:

1. Restrict user access to screen locations where data is entered
2. Provide a descriptive caption for every field
3. Display a sample format if a user must enter values in a specific format
4. Require an ending keystroke in every field
5. Do not require leading zeros for numeric fields
6. Do not require trailing zeros
7. Display default values
8. Use default values for constant entries
9. Display a list of acceptable values for fields
10. Provide a way to leave the data entry screen without entering the current record
11. Provide the opportunity to confirm to confirm the accuracy of input data
12. Provide for movement among fields in a standard order or any chosen order
13. Design the screen form layout to match that of the source documents
14. Allow users to add, change, delete, and view records
15. Provide for users to search for specific information

Form layout guidelines are:

- Allow sufficient space
- Offer clear instructions
- Provide logical organization
- Use captions effectively

Output Design

Output design involves important questions, such as:

- What is the purpose of the output?
- Who the information, why is it needed, and how will it be used?
- What specific information will be included?
- Will the output be printed, viewed on-screen, or both?
- When will the information be provided, and how often must it be updated?
- Do security or confidentiality issues exist?

Report Design

Printed output is highly visible. Reports, like any other element of the user-computer interface, should be attractive, professional, and easy to use. Systems analysts should realize that managers sometimes judge an entire project by the quality of the reports they receive.

Reports must include information that the user needs, and too much (or too little) information presents problems. There are three main types of reports:

- Detail reports
- Exception reports
- Summary reports

Detail Reports:

- Provide the most information
- At least one line of output is produced for each record processed
- Usually shows totals and subtotals
- Detail reports can be quite lengthy

Exception Reports:

- Show only records that meet a specific condition
- Useful when particular information is required
- Useful when the user wants information only on records that might require action, but does not need the details
- Special parameter queries can be used to select only the records that meet specified conditions

Summary Reports

- Show only subtotals and totals but not the supporting details
- Useful for upper-level managers who do not require extensive detail

Reports are an important way of delivering information to users, so users must be involved in the design process:

- All report designs should be approved in advance
- Submit each design as it is completed
- Preparing a prototype with sample data is most useful

Printed reports must be attractive, professional, and easy to read. Good report design, like any other aspect of the user interface, requires effort and attention to detail. The main elements in report design are:

- Page heading lines
- Column heading lines
- Column heading alignment
- Column spacing
- Field order
- Grouping detail lines