Unit – 04: Software Modeling and Design

- Translating Requirement model into design model:
- Data Modelling.
- Fundamental Design Concepts (Abstraction, Information hiding, Structure, Modularity).
- Software Design Approaches:
  - Function Oriented Design (Data flow Diagrams, Data Dictionary, Decision Trees a Tables).
  - Object-oriented design approach.
- Cohesion and coupling.
- Structured flowcharts.
- Decision Tables.
- Basic UI design

Questions to be discussed:

1. What do you understand by software design?
2. Discuss the difference between object oriented and function oriented design.
3. What is difference between a flow chart and a structure chart?
4. Differentiate between cohesion and coupling.
5. Write short notes on:
   a. Data flow diagram
   b. Data dictionary

What is software model?

- Software models are ways of expressing a software design.
- The models specify the stages and order of a process.
- A software process model is an abstraction of the software development process.

What is a Data Model?

- Good data allows organizations to establish baselines and goals to keep moving forward.
- A Data Model is this abstract model that allows the further building of conceptual models and to set relationships between data items.

What is Data Modeling?

- Data Modeling is the process of creating data models.
- It is the process of simplifying the data model by applying certain formal techniques.
- It conceptually represents data with diagrams, symbols, or text to visualize the interrelation.
- It involves expressing data and information through text and symbols.
- The data model provides the blueprint for building a new database application.

What do you understand by Software Design?

- Software design is the process of defining software solutions to one or more sets of problems.
- It deals with representing the client's requirement, as described in SRS document.
- It is a mechanism to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.
- One of the main components of software design is the software requirements analysis (SRA).
- The software design phase is the first step in SDLC.

Fundamental Design Concepts:

- The software design concept simply means the idea or principle behind the design.
- It describes how you plan to solve the problem and how you will design software.
- There are many concepts of software design and some of them are given below:
  - Abstraction
  - Information hiding
  - Structure
  - Modularity

Abstraction - hide Irrelevant data

- Abstraction means to hide the details to reduce complexity and increases efficiency or quality.

Information Hiding:

- Information hiding means to hide the information so that it cannot be accessed by an unwanted party.

Structure- design a structure of something

- Architecture simply means a technique to design a structure of something.

Modularity- subdivide the system

- Modularity means dividing the project into smaller parts to reduce the complexity of the project.

Software Design Approaches:

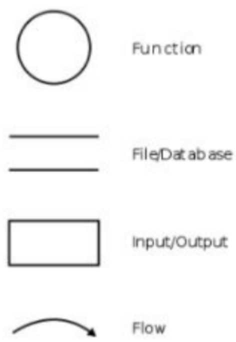
There are two main approaches to software design:

1. Function-Oriented Approach and
2. Object-Oriented Approach.

Difference between Function Oriented Design and Object Oriented Design:

Function-Oriented Design	Object-Oriented Design
The basic abstractions, which are given to the user, are real world functions.	The basic abstractions are not the real world functions but real world entities are represented.
Functions are grouped together by which a higher level function is obtained.	Function are grouped together since the classes are associated with their methods.
It is a top down approach.	It is a bottom up approach.
Begins by considering the use case diagrams.	Begins by identifying objects and classes.
Here, we decompose in function/procedure level.	We decompose in class level.
This approach is mainly used for computation sensitive application.	This approach is mainly used for evolving system which mimics a business.

Data Flow Diagram:



- DFD stands for Data Flow Diagram.
- DFD is a graphical representation using a standard set of symbols and notations.
- The flow of data of a system or a process is represented by DFD.
- It uses defined symbols like rectangles, circles and arrows etc. to show data inputs, outputs, storage points and the routes between each destination.

Data Dictionaries:

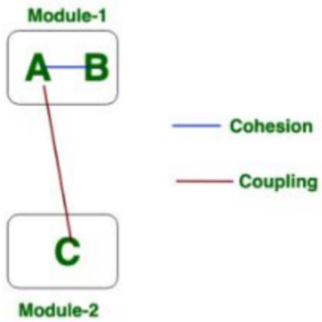
- A data dictionary is a set of files that includes a database's metadata.
- It consists all the data items appearing in DFD.
- The data dictionary hold records about other objects in the database.
- The data dictionary is an essential component of any relational database.
- Typically, only database administrators interact with the data dictionary.
- Data Dictionary is the major component in the structured analysis model of the system.

Data Dictionary Notations tables:

Notations	Meaning
X = a+b	X consists data elements a and b.
X = [a/b]	X consists of either elements a or b.
X = a X	X consists of optimal data elements a.
X = y[a]	X consists of y or more events of data element a
X = [a] z	X consists of z or less events of data element a
X = y [a] z	X consists of some events of data elements between y and z.

Cohesion and coupling:

- Coupling and Cohesion are two key concepts in software engineering.
- It is used to measure the quality of a software system's design.



Coupling:

- Coupling represents the relationships between modules.
- High coupling means modules are closely connected & changes in module may affect the other modules.
- Low coupling means modules are independent & changes in module have little impact on other modules.
- Loosely coupling gives the best software.

Cohesion:

- Cohesion represents the relationship within a module.
- High cohesion means that elements are closely related and focused on a single purpose.
- Low cohesion means that elements are loosely related and serve multiple purposes.
- High cohesion gives the best software.

Difference between cohesion and coupling:

Cohesion	Coupling
Cohesion is the concept of intra-module.	Coupling is the concept of inter-module.
It represents the relationship within a module.	It represents the relationships between modules.
Increasing cohesion is good for software.	Increasing coupling is avoided for software.
It represents functional strength of modules.	It represents the independence among modules.
Highly cohesive gives the best software.	Loosely coupling gives the best software.
In cohesion, the module focuses on a single thing.	In coupling, modules connected to other modules.
Cohesion is created between the same module.	Coupling is created between two different modules.
There are Six types of Cohesion 1. Functional Cohesion. 2. Procedural Cohesion. 3. Temporal Cohesion. 4. Sequential Cohesion. 5. Layer Cohesion. 6. Communication Cohesion.	There are Six types of Coupling 1. Common Coupling. 2. External Coupling. 3. Control Coupling. 4. Stamp Coupling. 5. Data Coupling 6. Content Coupling.

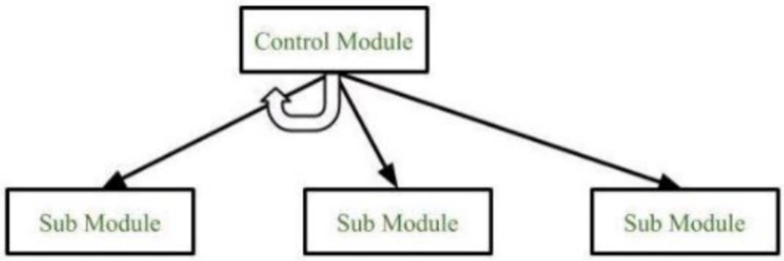
Decision table:

- It is a visual representation for specifying which actions to perform depending on given conditions.
- The information represented in decision tables can also be represented as decision trees.
- It is also represented in programming language using if-then-else and switch-case statements.
- A decision table is a good way to settle with different combination inputs with their corresponding outputs and is also called a cause-effect table.

Email (condition1)	T	T	F	F
Password (condition2)	T	F	T	F
Expected Result (Action)	Account Page	Incorrect password	Incorrect email	Incorrect email

Structure Chart:

- Structure Chart represent hierarchical structure of modules.
- It breaks down the entire system into lowest functional modules.
- Which describe functions & sub-functions of each module of a system to a greater detail.
- Modules at top level called modules at low level.
- Components are read from top to bottom and left to right.



Basic UI Design:

- UI design stands for User interface design.
- It is the process to build interfaces in software focusing on looks or style.
- Designers aim to create interfaces which users find easy to use and pleasurable.
- UI design refers to graphical user interfaces and other forms—e.g., voice-controlled interfaces.
- It is the front-end application view to which user interacts in order to use the software.
- The software becomes more popular if its user interface is:
  - Attractive
  - Simple to use
  - Responsive in short time
  - Clear to understand
  - Consistent on all interface screens

User Interface Design Process: