Unit-3

Concept of Software Design

Software design is a mechanism to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.

- It allows the s/w engineer to create the model of the s/w that is to be developed.
- The s/w designed phase is the first step in SDLC, which moves the concentration from the problem domain to the solution domain.

General task involved in design Process

1. Design the overall the system processes
2. Segmenting the system into smaller modules
3. Designing the database structure
4. Documenting the system design.

Objectives

1. Correctness:- Software design should be correct as per requirement. It should correctly implement all the functionality of the system.
2. Completeness:- The design should have all components like data structure modules, and external interfaces etc.
3. Efficiency:- Resources should be used efficiently by the program.
4. Flexibility:- Able to modify on changing needs.
5. Consistency:- There should not be any inconsistency in the design.
6. Maintainability:- The design should be so simple so that it can be easily maintainable by other designers.

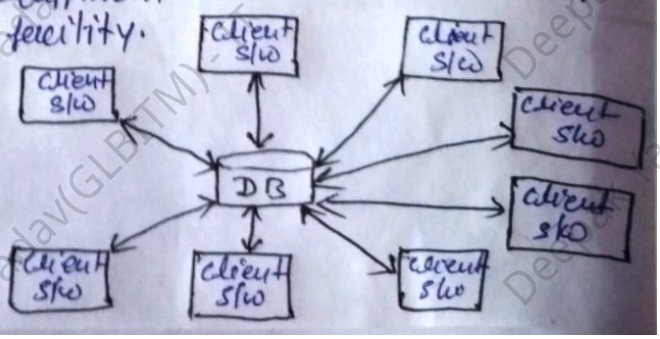
The software design concept simply means the idea or principle behind the design. It describes how you plan to solve the problem of designing s/w, the logic, or thinking behind how you will design s/w. It allows the s/w engineer to create the model of the system or s/w or product that is to be developed or built.

Architectural Design

- The process of defining a collection of h/w and s/w components and their interfaces to establish the framework for the development for a computer system.
- The o/p of this design process is a description of the s/w ~~process~~ architecture.
- Architectural design is an early stage of the system design process.
- It represents the link b/w specification and design processes and is often carried out in parallel with some specification activities.
- A set of components (e.g: a database, computational modules) that will perform a function required by the system.
- The set of connectors will help in coordination, communication and cooperation b/w the components.
- Conditions that how components can be integrated to form the system.

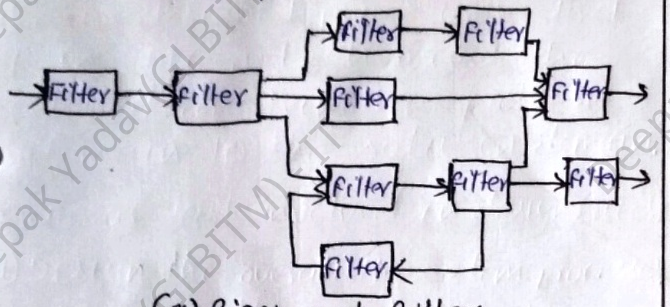
Data Centered Architecture:-

- Data centered Architecture is a layered process which provides architectural guidelines in data center development.
- Data centered Architecture is also known as Database Centric Architecture.
- This architecture is the physical and logical layout of the resources and equipment within a data center facility.

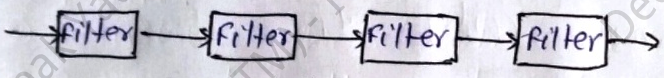


Data Flow architecture:-

- This kind of architecture is used when i/p data to be transformed into output data through a series of computational manipulatives components.
- The figure represents pipe-and filter architecture since it uses pipe and filter and it has a set of components called filters connected by pipes.
- Pipes are used to transmit data from one component to the next.



(a) Pipes and filter



(b) batch sequential.

- The interconnection should be simple to avoid costly maintenance.
- Two modules are directly connected if one module can call the other module.

Modularity has several key benefits.

- Testing & Debugging :- It becomes easy to test each component in isolation by using a mocking or isolation framework.
- Reusability :-
- Extensibility :- Your s/w now runs as a set of independent components connected by an abstraction layer.

Advantages of modularization.

- Smaller components are easier to maintain
- Program can be divided based on functional aspects.
- Easier because programmer only focus on an small simple problem rather than a large complex problem.
- Testing and debugging easier.
- Easy to isolate bugs and easy to fix it.

Procedural Design

- Procedural design used to model programs that have an obvious flow of data from i/p to o/p.
- It represents the architecture of a program as a set of interacting processes that pass data from one to another.
- Procedural design is also called components design. It is completely based on process and control specification.
- The "state" transition diagram" of the requirement analysis model is also used in components design.
- Components design is usually done after user interface design.

Modularization

- Modularity is design refers to the splitting of a large software system into smaller connected modules.
- Modules are interconnected through their interfaces.

Design structure charts

- A structure chart (SC) in software engineering and organizational theory is a chart which shows the breakdown of a system to its lowest manageable levels.
- They are used in structural programming to arrange modules into a tree. Each module is represented by a box, which contains the module's name.
- The tree shows the relationships b/w modules, showing data transfer b/w the models. Data being passed from module that needs to be processed.
- Structure chart is a chart derived from a Data Flow Diagram. It represents the system in more details than DFD. It breaks down the entire system into lowest functional modules, describes functions and sub-functions of each modules of the system to a greater details than DFD.

Pseudo Codes

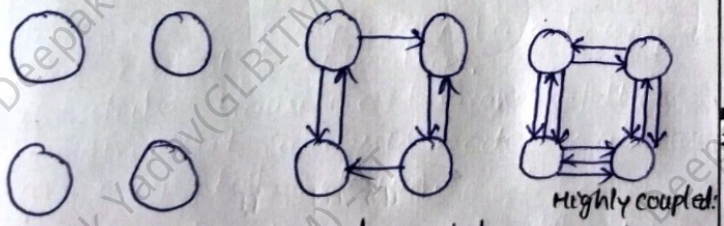
- Pseudo code uses the vocabulary of one languages. i.e English & syntax of another. i.e, any structural programming languages.
- Pseudo code is a ~~programming~~ combination of algo. written in a ~~syn~~ simple language and programming language statements.
- Using Pseudo code, the designer design describes system characteristics structured by keywords such as IF-THEN-ELSE, while-do and end.

Advantages of Pseudo Codes

- Converting a Pseudo code to a programming language is much easier as compared to converting to flowchart or decision table.
- It is easier to modify whenever programmer modification are necessary.
- Take less time to writing as compared to drawing flow chart.
- Done easily on word processor.

#Coupling

- The coupling is the degree of independence b/w s/w modules.
- Two modules that are tightly coupled are strongly dependent on each other.
- However, two modules ~~that~~ are loosely coupled are not dependent on each other.
- Uncoupled modules have no independence at all within them.

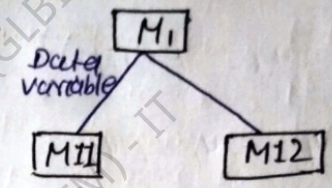


Uncoupled: no dependencies

Loosely coupled: Some dependencies

- A good design is the one that has loco coupling.
- Coupling is measured by the number of relations b/w the modules.
- That is, the coupling increases as the no. of calls b/w modules increase. ~~Ess~~ Thus it can be said that a design with high coupling will have more errors.

① Data coupling:- when data of one module is passed to another module, this is called data coupling.



② Stamp coupling: Two modules are stamp coupled if they communicate using composite data items such as structure, objects etc.

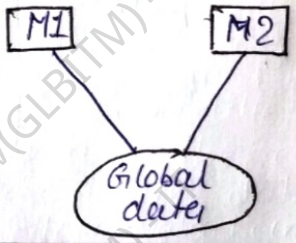
when the module passes entire structure to another module, they are said to be stamp coupled.

Ex:- variables in C or object in C++

③ Control coupling:- Control coupling exists among two modules if ~~they~~ data from one module is used to direct the structure of instruction execution in another.

④ External coupling:- external coupling arises when two modules share an externally imposed data format, communication protocols or device interface. This is related to communication to external tools and devices.

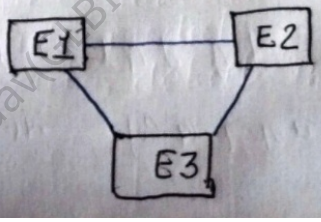
⑤ Common coupling:- Two modules are common coupled if they share information through some global data items.



#Cohesion

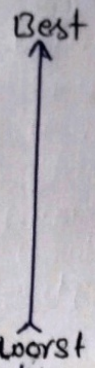
Cohesion show the relationship within the module.

Cohesion is an ordinal type of measurement and is generally described as "high cohesion" or "low cohesion".



Types of cohesion

① Functional cohesion
② Sequential cohesion
③ Communication cohesion
④ Procedural cohesion
⑤ Temporal cohesion
⑥ Logical cohesion
⑦ Coincidental cohesion



① Function cohesion:- Functional cohesion is said to exist if the different elements of a module, cooperate to ~~activate~~ acheive a single function.

② Sequential cohesion:- If the element of a module form the components of the sequence, where the output from one component of the sequence is i/p to the ~~te~~ next.

③ Communicational cohesion:- If all tasks of the module refer to or update the same data structure e.g. the set of functions defined on an array or a stack.

④ Procedural cohesion:- If the set of purpose of the module are all parts of a procedure in which particular sequence of steps has to be carried ~~to~~ out for acheiving a goal. e.g. the algorithm for decoding a message.

⑤ Temporal cohesion:- When a module includes functions that are associated by the fact that all the methods must be executed in the same time.

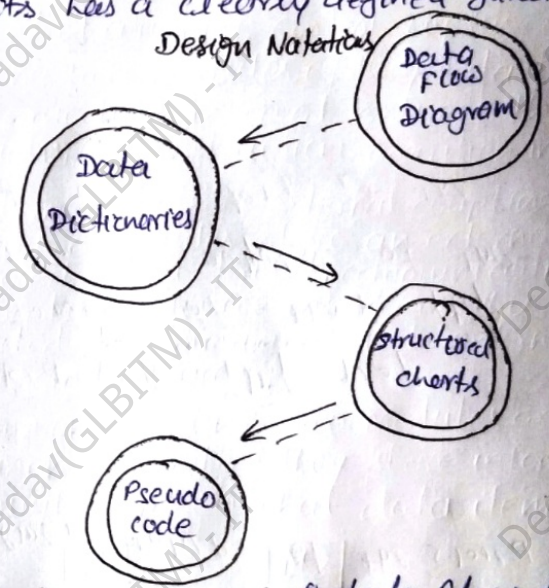
⑥ Logical cohesion:- If ~~the~~ all the elements of the module perform a similar operation, for example Error handling, data i/p and data o/p etc.

⑦ Coincidental cohesion:- If it performs a set of tasks that are associated with each other very loosely, if at all.

Coupling	cohesion
• Coupling is also called Inter-Module Binding.	• Cohesion is also called Intra-Module Binding.
• Coupling shows the relationships b/w modules.	• Cohesion shows the relationship within the module.
• In coupling, modules are linked to the other modules.	• In cohesion, the module focuses on a single thing.

Function Oriented Design

Function Oriented Design is an approach to s/w design design where the design is decomposed into a set of interacting units where each units has a clearly defined function.



• Data Flow Diagram:- A data flow diagram maps out the flow of information for any process or system.

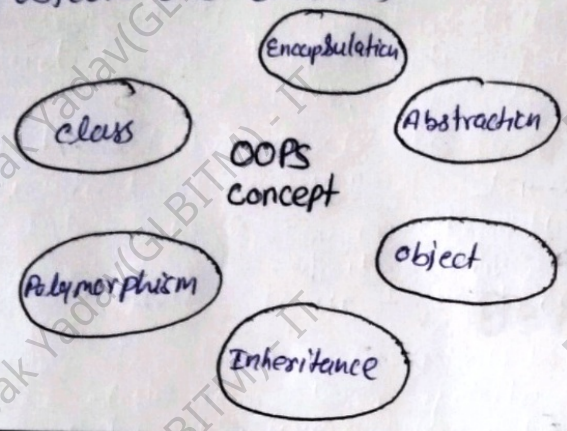
• Data Dictionaries:- Data dictionaries are simply repositors to store information about all data items defined in DFDs. At the requirement stage, data ~~structure~~ dictionaries contain data items.

• Structure charts:- Components are read in from top to bottom and left to right. When a module calls another, it views the called module as black box, passing required parameters and receiving results.

• Pseudo code:- It uses keywords and indentation Pseudo codes are used as replacement for flow charts. It decreases the amount of docu. required.

Object Oriented Design

- Object oriented design is the process of planning a system of interacting objects for the purpose of solving a s/w problem. It is one approach to s/w design.
- In the object-oriented design method, the system is viewed as a collection of objects (i.e. entities)



Measurement

- A s/w metric is a measure of s/w characteristic which are measurable or countable. s/w metrics are valuable for many reasons, including measuring s/w performance, planning work items, measuring productivity, and many other uses.
- within the s/w development process, many metrics are that are all connected. s/w metrics are similar to the four functions of management.

Need of s/w ~~Management~~ Measurement:

- s/w is measured to:-
- Create the quality of the current product.
- Anticipate future qualities of the product or process.
- Enhance the quality of a product or process.
- Regulate the state of the project in relation to budget and schedule.

Top-Down Approach	Bottom-Up Approach
① In this approach we focus on breaking up the problem into small parts.	① In bottom up approach we solve smaller problem and integrate it as a whole & complete the sol".
② Mainly used by structured programming language such as COBOL, C, etc.	② Mainly used by object oriented programming language such as C++, C#, Python.
③ Each part is programmed separately therefore certain redundancy	③ Redundancy is minimized by using data encapsulation and data hiding.
④ In this the communication is less among modules	④ In this module must have communication.
⑤ It is used in debugging module documentation, etc.	⑤ It is basically used in testing.
⑥ In top-down approach decomposition takeplace	⑥ In bottom up approach composition take place.
⑦ In this, top function of system might be hard to identify.	⑦ In this sometimes we can not build a program from the piece we have started.
⑧ In this, implementation details may differ.	⑧ This is not natural for people to assemble.

Metrics

A metrics is a measurement of the level that any impute belongs to a system product or process.

There are 4 functions related to s/w metrics.

① Planning:-
② Organizing
③ Controlling
④ Improving.

Types of s/w metrics:-

① Product Metrics:- Product metrics are used to evaluate the state of the product, tracing risks and undercovering prospective problem areas. The ability of team to control quality is evaluated.

② Process Metrics:- Process metrics pay particular attention on enhancing the long term process of the team or organization.

③ Project metrics:-
- The project matrix describes the project characteristics and execution process.
- No. of s/w developer.
- Staffing pattern over the life cycle of s/w.
- cost and schedule.
- Productivity.