20/10/2022

Date:
Page No:

Tutorial - 7

1. What do you mean by the term "design"? Define design methadology.

Ans: A design is a plan or specification for the construction of an object or system or for the implementation of an activity or process or the result of that plan or specification in the form of a prototype, product or process.

Design methodology refers to the development of a system or method for a unique situation.

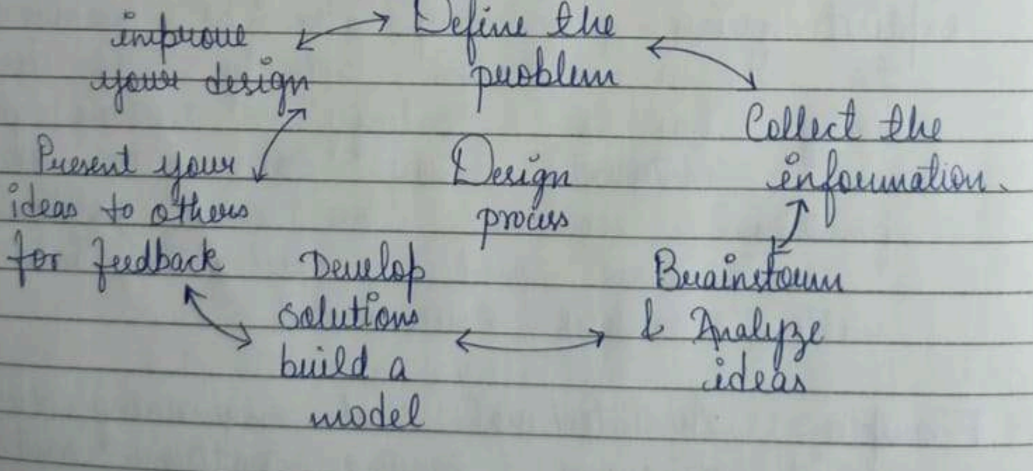
The verb to design expresses the process of developing a design. In some cases, the direct construction of an object without an explicit prior plan may also be considered to be a design activity. The design usually has to satisfy certain goals and constraints; may take into account functional, economic or socio - political considerations and is expected to interact with certain environment.

Examples: Architectural and engineering drawings circuit diagrams, sewing patterns and less

Date:
Page No:

tangible artifacts such as business process models.

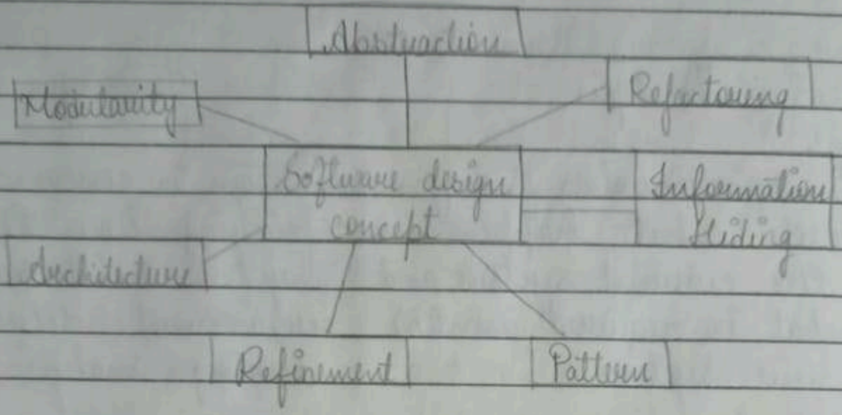
Today, the term design methodology is most often applied to technological fields in reference to web design, software or information system design. Method that promotes the rational thought implies the adoption of sequential structure from problem to solution that includes all the phases since the moment we enunciate the problem, collect information and begin the conceptual work until the solution. A set of techniques normally involving users as well as designers to be employed in creating new technologies.



improve
~~your design~~
Define the problem
Collect the information.
Present your ideas to others for feedback
Design process
Develop solutions build a model
Brainstorm & Analyze ideas

Date:
Page No:

3. Explain the design concepts.



i) Abstraction : It means to hide the details to reduce complexity and increase efficiency or quality. Different levels of abstraction are necessary and must be applied at each stage of design process so that any error that is present can be removed to increase the efficiency of software solution and to refine the software solution.

) Modularity: It means dividing the system or project into smaller parts to reduce the complexity of system or project. Subdividing a system into smaller parts so that these parts can be created independently and then use these parts in different systems to perform different functions.

iii)
iv)
v)
vi)
vii)
4.
Ans:
2.

Date:
Page No:

iii) Architecture: It is a technique to design a structure of something. It focuses on various elements and data of structure.

iv) Refinement: It means to refine something to remove any impurities if present and increase the quality.

v) Pattern: It means a repeated form or design in which same shape is repeated several times to form a pattern.

vi) Information Hiding: It means to hide the information so that it cannot be accessed by an unwanted party.

vii) Refactoring: It means reconstructing something in such a way that it does not affect the behaviour of any other feature

4. Write short note on design principle.

Ans: 1. Should not suffer from "Tunnel Vision" - It should not focus on completing or achieving aim but on other effects also.

2. Traceable to analysis model - It should satisfy all requirements that software requires

Date:
Page No:

9. Review to discuss errors - Overall evaluation should be done to check if there is any error present or if it can be minimized.

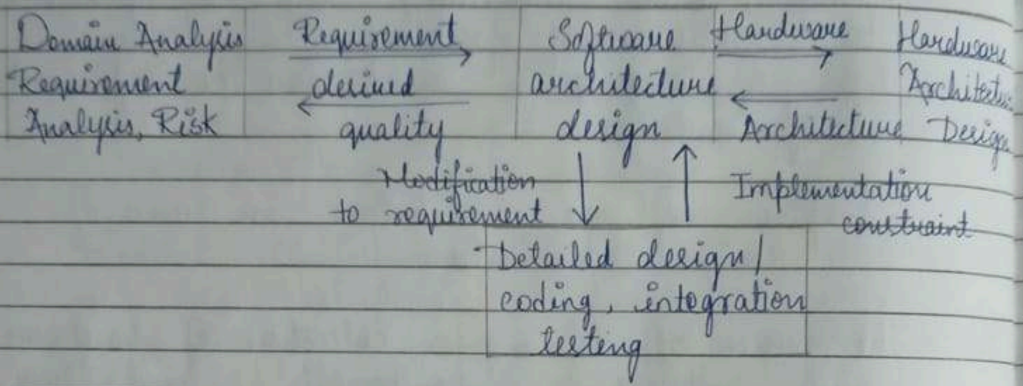
10. Design is not coding & coding is not design - Design means describing logic of program to solve any problem & coding is type of language used for the implementation of design.

5. Explain software architecture in terms of design specification.

Ans: The process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system. The software that is built for computer based systems can exhibit one of these many architectural styles.

- A set of components (eg. a database, computational modules) that will perform a function required by the system.
- The set of connectors will help in coordination, communication and cooperation between the components.
- Components that how components can be

integrated to form the system.
- Semantic models that help the designer to understand the overall properties of the system.

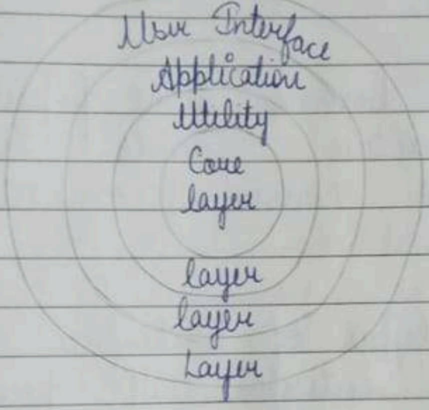

Domain Analysis Requirement Analysis, Risk
Requirement
desired quality
Software architecture design
Hardware
Architecture
Hardware Architecture Design
Modification to requirement
Implementation constraint
Detailed design/ coding, integration testing

6. What do you mean by architectural style? Give brief introduction to

1. Data center : Data store will reside at centre of this architecture & is accessed frequently by other components that update, add, delete or modify data present within store. The data centered architecture promote integrability. Existing components can be changed and new client components can be added to architecture without permission or concern of other clients.

Date:
Page No:

instruction set progressively. At outer layer, components will receive user interface operation & at inner layer components will perform operating system interfacing. Intermediate layers to utility services and application software functions.



Date:
Page No:

UI design principles :

① Structure :
⇒ Design should organize the user interface purpose based on precise, consistent models that are apparent & recognizable to users putting related things together and separating unrelated things, differentiating dissimilar things.

② Simplicity :
⇒ The design should make the simple, common task easy, communicating clearly and directly in user's language.

③ Visibility :
⇒ It should make all required options and materials for a given function visible without distracting the user with redundant data.

④ Feedback :
The design should keep users informed of actions or interpretation, change of state or condition and bugs or expectations that are relevant and of interest to the user through clear, concise and unambiguous language familiar to users.

Date:
Page No:

directed towards performing a single task are contained in the component.

⇒ COUPLING :
- It is an indication of interconnection b/w modules in a structure of software.
- It is also the indication of the relationships b/w modules.
- Usually low coupling is good for software.
- It is the measure of degree of independence b/w the modules.
- A good software will have low coupling.

5. Describe types of coupling.

Types of coupling

1. Data Coupling :
If the dependency between the modules is based on the fact that they communicate by passing only data, then the modules are said to be data coupled.

2. Stamp Coupling :
• The complete data structure is passed from one module to another module. Thus it involves tramp data.
• It may be necessary due to efficiency factors

Date:
Page No:

A functional cohesion performs the task and functions. It is an ideal situation.

(2) Sequential cohesion:
An element outputs some data that become the input for other element i.e data flow b/w the parts. It occurs naturally in functional programming language.

(3) Communicational cohesion:
Two elements operate on the same input data or contribute towards the same output data

(4) Procedural cohesion: Elements of procedural cohesion ensures the order of execution. Actions are still weakly connected and unlikely to be reusable.

(5) Temporal cohesion: The elements are related by their timing involved. A module connected with temporal cohesion all the tasks must be executed in the same time span.

(6) Logical cohesion:
The elements are logically related and not functionally. A component reads

Date:
Page No:

input from tape, disk and networks. All the code for these functions is in the same component operations are related, but the functions are significantly different.

⑦ Coincidental Cohesion:
The elements are not related. The elements have no conceptual relationship other than location in source code. It is accidental and the worst form of cohesion.

Date:
Page No:

this choice made by the insightful designer.

3. Control Coupling:
- If the modules communicate by passing control information then it is control coupled.
- It can be bad if the parameters indicate completely different behaviour & good if parameter allow factoring & reuse of functionality.

4. External Coupling:
- The modules depend on other modules, external to the software being developed on to a particular type of hardware.

5. Common Coupling:
- The modules have shared data such as global data structure.
- The changes in global data mean tracking back to all modules which access that data to evaluate the effect of the change.

6. Describe levels of cohesion:

① Functional cohesion:
Every essential element for a single computation is contained in the component.

Date:
Page No:

information hiding is an important aspect to the abstraction of software.
- We use software module to implement information hiding.
- The info contained in the modules should be hidden from the rest of the software system outside of module.

Describe the 2 quantative criteria of functional independence.

- The functional independence is the concept of separation & related to the concept of modularity, abstraction & info hiding.
- The functional independence is accessed using two criteria i.e cohesion & coupling.

⇒ COHESION:
- It is an extension of info hiding concept.
- A cohesive module performs a single task and it requires small interaction with the other components in other part of program.
- It is indication of relationship within module
- It is measure of degree to which the elements of the module are functionally related
- It is degree to which all elements

Date:
Page No:

4.	It has tight coupling issues & low interactivity between various modules.	It has high interactivity between various modules
5.	It is based on decomposition approach.	It is based on composition approach.
6.	It may not be possible to break the problem into set of smaller problems.	Sometimes it is difficult to identify overall functionality of system in initial stages.

3. What do you understand by information hiding?

Ans: ⇒ Information hiding for programmers is executed to prevent system design change.
⇒ If design decisions are hidden, certain program code cannot be modified or changed
⇒ It is usually done for internally changeable code which is sometimes especially designed not to be exposed.
⇒ change resilience of classes & ease of use by client objects are two by products of hidden data.
⇒ It is an important aspect of modularity and if you recall the definition of abstraction,

Date:
Page No:

5) Tolerance:
The design should be flexible and tolerant, decreasing the cost of errors while also preventing bugs whenever possible by tolerating varied inputs and sequences by interpreting all reasonable actions.

2. Differentiate between top-down & bottom-up design.

	Top-Down Model	Bottom-Up Model
1.	It's focus is on breaking the bigger problem into smaller one & then repeat the process with each problem.	It's focus is on identifying & resolving smallest problem & then interpreting them together to solve bigger problem.
2.	It is mainly used by structural languages like C, Fortran etc.	It is mainly used by object-oriented language like C++, Java etc.
3.	It has high ratio of redundancy as the size of project increases.	It is better suited as it ensures minimum data redundancy & focus on re-usability.

4. It h
issu
inte
vari

5. It
dec
a

6. It
to
pu

3. U

Ans: ⇒

⇒

⇒ It

⇒

⇒

Date:
Page No:

Tutorial - 8

1. What is UI? What are UI design principle?

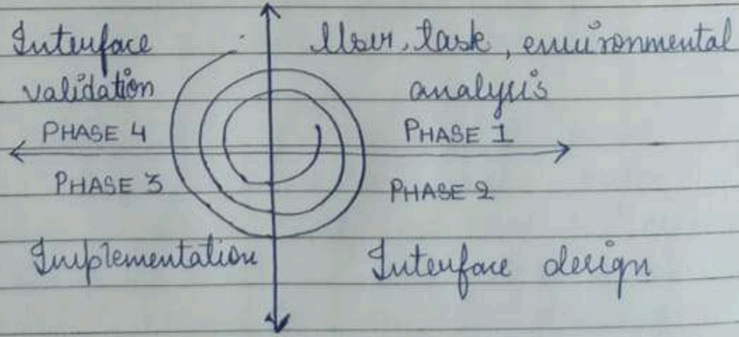
UI is the front end application view to which user interacts in order to use the software.

The software becomes more popular if it's UI is attractive, simple to use, responsive in short time, consistent on all interface screens.

There are 2 types of interfaces:

① Command line interface: It provides a command prompt, where the user types command and feeds to the system.

② Graphical User interface: It provides the simple interactive interface to interact with the system.

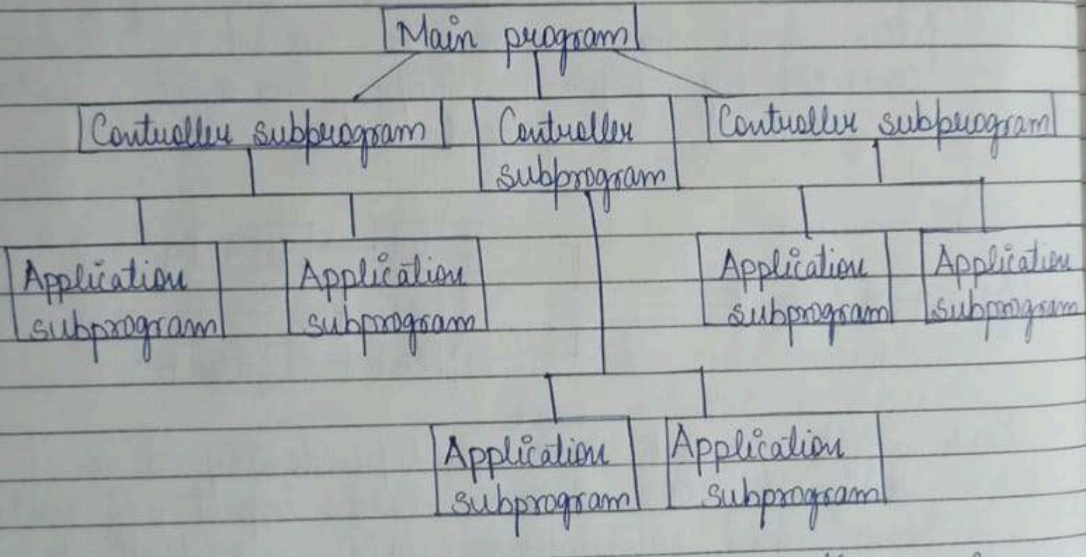


Date:
Page No:

that is easy to scale and modify.

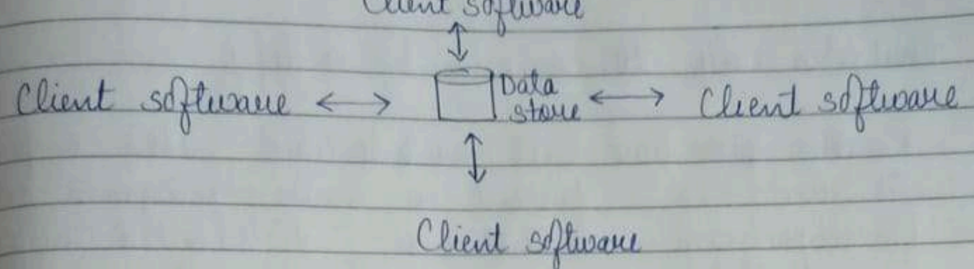
1. Remote procedure call architecture - This component is used to present in main program or subprogram architecture distributed among multiple computers on network.

2. Main program or sub program architecture - Main program contains no of subprograms that can invoke other components.

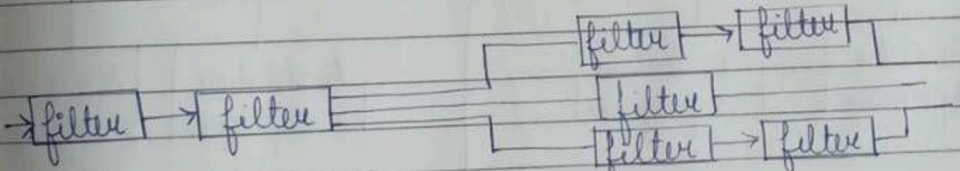


4. Layered Architecture - A no of different layers are defined with each layer performing well defined set of operations. Each layer will do some operations that become closer to machine

Date:
Page No:



2. Data Flow: It is used when input data to be transformed into output data through series of computational manipulative components. It represents pipe & filter architecture since it uses both pipes and filter & it has components called filters connected by pipes. Pipes transmit data from component to next.



Each filter will ~~look~~ work independently & is designed to take data input of certain form and produces data output to next filter of specified form. The filters don't require any knowledge of the working of neighbouring filters.

3. Call and return: It is used to create a program

Date:
Page No:

to develop high quality product.

Should not "reinvent the wheel" - It should not waste time or effort in creating things that already exist. Due to this, overall development will get increased

Minimize intellectual distance - Reduce the gap b/w real world problems & software solutions for that problem.

Exhibit Uniformity and Integration - It should be uniform throughout the process without any change. Integration means it should mix or combine all parts of software.

Accommodate change - The software should adjust to change that is required to be done as per user's need.

7. Degrade gently - It should work properly even if an error occurs during execution.

8. Assessed or quality - During the evaluation the quality of design needs to be checked & focused on.

Date:
Page No:

Differentiate between analysis and design.

Analysis	Design
Analysis involves finding out what happens in the current system and what is required in the new system.	Design is concerned with specifying how the new system will satisfy the requirement determined during analysis.
Understanding, investigating and modelling requirements.	How the system will be constructed without actually building it. The aim is to produce the best possible solution that satisfies the requirement.
There is one answer and you need to find it. The answer to the question/problem is same when asked again.	There are multiple answers and you need to find the most desirable one. The solution will probably be different the next time.
A small model	A large model
Describe structure of real world applications. Focus on requirement of system.	Describe structure of proposed software system. Focus on software structure that implements the system