B.TECH. DEGREE EXAMINATION, JAN 2023

Fifth semester

Computer Science and Engineering

SOFTWARE ENGINEERING

(2013-14 Regulation)

PART-A

1. **Define software engineering. What are its applications?**
   Software engineering is a detailed study of engineering to the design, development and maintenance of software.

2. **What are the advantages of waterfall model?**
   1. Simple and easy to understand.
   2. It works well for smaller projects where requirements are very well understood.
   3. Clearly defined stages.
   4. Easy to arrange tasks.

3. **Differentiate risk management and scheduling?**

Risk management	Scheduling
risk management is defined as the process of identifying	Schedule Management is the process of defining project tasks .
monitoring and managing potential risks	durations, dependencies, and assigned resources in order to complete the project
minimize the negative impact	Monitoring and reporting on the schedule to ensure the project is delivered on time.

4. **Write short notes on organization and team structure?**

Organization Structure:

The organization structure is classified into two types

1. Functional organization
2. Project organization

Team Structure:

Problems of different complexities and sizes require different team structures:

- Chief programmer: suitable for routine work.
- Democratic: Small teams doing R&D type work

5. **Mention few characteristics of a good software design?**

- Correctness.
- Understand ability

- Efficiency
- Maintainability

6. List out the difference between coupling and cohesion.

Coupling	Cohesion
Coupling is the degree of interdependence between the modules.	Cohesion is the degree to which the elements inside a module belong together
Two modules highly coupled	A module with high cohesion contains elements that are tight related to each other
Low coupling among them work	Low cohesion if it contains unrelated elements
Loose coupling reflects the higher quality of software design	Highly cohesive modules reflect higher quality of software design

7. Define patterns.

Design pattern are commonly accepted solutions to some problems that recur during designing different application.

8. What are the advantages of UML models?

- Provides standard for software development.
- Reducing of costs to develop diagrams of UML using supporting tools.
- Development time is reduced.

9. What are the steps followed in testing?

1. Requirement Analysis.
2. Test Planning.
3. Test Case Designing and Development.
4. Test Environment Setup.
5. Test Execution.
6. Test Closure.

10. List out the levels of testing?

- Unit Testing.
- Integration Testing.
- System Testing.
- Acceptance Testing.

PART-B

UNIT-I

11. Discuss on evaluation and impact of software development projects.

Software development is a continuous process wherein we constantly improve existing functionality or add new features. However, every change introduced to a

Product might have an impact on a particular part of the product or even on the entire Product. And the more changes we make to a product, the more difficult it becomes to track their consequences.

Impact analysis is a software testing approach that helps you define all risks associated with any kind of changes made to the product under test.

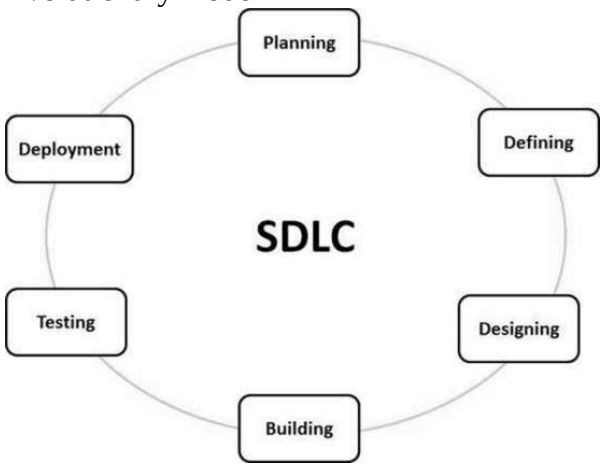
It's best to perform an impact analysis whenever:

- there's a request for a change to the product
- there are changes in product requirements
- there are changes to current modules or features
- you plan to implement new modules or features

1. Dependency impact analysis
2. Experiential impact analysis.
3. Traceability impact analysis

12. Explain in detail about software life cycle models.

- SDLC is the acronym of Software Development Life Cycle.
- It is also called as Software development process.
- The software development life cycle (SDLC) is a framework defining tasks performed at each step in the software development process.
  - Waterfall Model
  - Iterative Model
  - Spiral Model
  - Prototyping model
  - Evolutionary mode



UNIT-II

13. Write short notes on COCOMO model.

Constructive COst estimation MOdel (COCOMO) was proposed by Boehm[1981]. COCOMO prescribes a three stage process for project estimation.

In the first stage, an initial estimate is arrived at.

- Over the next two stages, the initial estimate is refined to arrive at a more accurate estimate.
- COCOMO uses both single and multivariable estimation models at different stages of estimation. The three stages of COCOMO estimation technique are
  - Basic COCOMO,
  - Intermediate COCOMO, and
  - Complete COCOMO.

14. Explain the software configuration management and what the steps to be followed for requirement gathering and analysis.

Introduction:

- Goal of software project management is to enable a group of engineers to work efficiently towards successful completion of a software project.
- Many software projects fail due to faulty project management practices. Hence it is important to learn different aspects of software project management.

Responsibility of project managers

The responsibilities of the software project manager are as follows

- Project proposal writing,
- Project cost estimation,
- Scheduling,
- Project staffing,
- Project monitoring and control,
- Software configuration management,
- Risk management,
- Managerial report writing and presentations, etc.

Steps to be followed for requirement gathering and analysis

- Identify the relevant stakeholders.
- Establish project goals and objectives.
- Elicit requirements from stakeholders.
- Document the requirements.
- Confirm the requirements.
- Prioritize the requirements.

UNIT-III

15. List out the characteristics of a good software design and differentiate coupling and cohesion.

Characteristics of a good software design

- Correctness: A good design should correctly implement all the functionalities identified in the SRS document.
- Understand ability: A good design is easily understandable.
- Efficiency: It should be efficient.

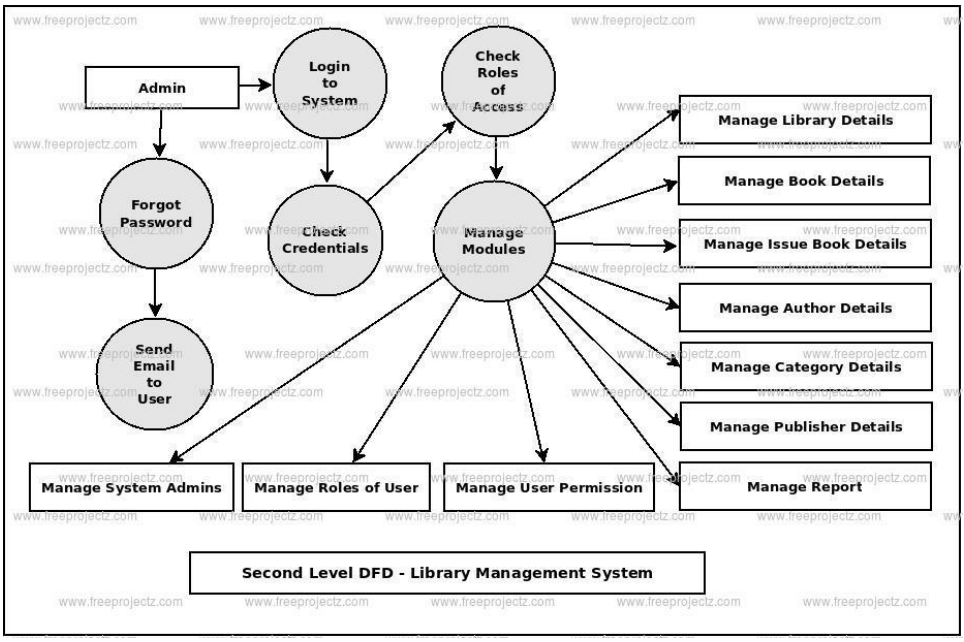
- Maintainability: It should be easily amenable to change

Differentiate coupling and cohesion.

Coupling	Cohesion
Coupling is the degree of interdependence between the modules.	Cohesion is the degree to which the elements inside a module belong together
Two modules highly coupled	A module with high cohesion contains elements that are tight related to each other
Low coupling among them work	Low cohesion if it contains unrelated elements
Loose coupling reflects the higher quality of software design	Highly cohesive modules reflect higher quality of software design

16. Sketch the data flow diagram of library management system.

- The DFD (also known as the bubble chart) is a simple graphical formalism that can be used to represent a system in terms of the input data to the system.
- Various processing carried out on these data, and the output data generated by the system.
- The main reason why the DFD technique is so popular is probably because of the fact that DFD is a very simple formalism.
- It is simple to understand and use.
- A DFD model is uses a very limited number of primitive symbols to represent the functions performed by a system and the data flow among these functions with a set of high level functions that a system performs.
- A DFD model hierarchically represents various sub-functions. External entity Process Data store output Data flow



(Diagram based upon their own knowledge)

UNIT-IV

17. Differentiate use case model and class diagram explain with any one of the example.

use case model	class diagram
Behavioral diagram	Structural Diagram
Describes functional requirements	Describes the structure of a system
By mentioning that who will perform what function, and which kind of association is exist between functions (use cases)	By showing the system's classes, their attributes, functions, relationships among objects.
Communicates, Includes, Extends, Generalization	Inheritance, Association, Aggregation, Composition, Dependency, Realization
Specify the behavior (what system will do?)	Specify the internal structure that will help to complete a functionality mentioned in use cases
Use cases can be denoted both by Only denoted by visual textual(i.e. use case description) and representation visual representation (i.e. use case diagram)	Only denoted by visual representation
Only summarizes relationships between use cases, actors, and systems does not show the order in which steps are performed to achieve the goals of classes executes each use case	Only summarizes relationships between classes of the system does not show the order in which classes executes.

(Diagram was based on student's example)

18. Write short notes on object oriented analysis and design methodology.

AN OBJECT-ORINETED ANALYSIS AND DESIGN METHODOLOGY

- The results of the analysis activities are be redefined into a design model through several iterations.

The Unified Process

- The unified process is an extensible framework which needs to be customized for specific types of projects.
- The two main characteristics of the unified process are
  - Use case driven
  - Iterative
- Case-driven implies that use cases of the system are considered to be the central and most important view.
- The use case view should be the first one to be constructed and should be refined iteratively into an implementation

- The use case model is the central model. All models that constructed in the subsequent design activities must conform to the use case model
- The unified process involves iterating over the following four phases as follows:

1. **Inception:**
   - During this phase, the scope of the project is defined and prototypes may be developed to form a clear idea about the project.
2. **Elaboration:**
   - The functional and the non-functional requirements are captured
3. **Construction:**
   - Analysis, design and implementation activities are carried out.
   - Full text descriptions of use cases are written during the construction phase and each use case is taken up for the start of a new iteration.
   - System features are implemented in a series of short iterations
   - Each iteration results in an executable release of the software
4. **Transition:**
   - The product is installed in the user's environment and maintained.

Overview of the OOAD Methodology

- The use case model is developed first.

UNIT-IV

19. Explain in detail about the various levels of testing with examples.

Software Testing:

- Testing is a process of executing a program with the intent of finding error.
- Software Testing are contains two types:
  - Manual Testing and,
  - Automation Testing

Manual Testing:

- Manual testing is the process of manually testing software for defects.
- It requires a tester to play the role of an end user, and use most of all features of the application to ensure correct behaviour
- To ensure completeness of testing, the tester often follows a written test plan that leads them through a set of important test cases.

Testing:

Testing is a process of executing a program with the intent of finding error

- **Unit Testing:**

  - It concentrates on each unit (Module, Component…) of the software as implemented in source code.
  - During this stage they conduct program level testing, with the help of the WBT techniques.

- **Integration Testing:**
  - ❖ Putting the modules together and construction of software architecture.
  - ❖ There are two types of approaches to conduct Integration Testing:
    - ♣ Top-down Approach
    - ♣ Bottom-up approach.

20. Differentiate black box testing and white box testing.

S. No.	Black Box Testing	White Box Testing
1.	It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it.	It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software.
2.	Implementation of code is not needed for black box testing.	Code implementation is necessary for white box testing.
3.	It is mostly done by software testers.	It is mostly done by software developers.
4.	No knowledge of implementation is needed.	Knowledge of implementation is required.
5.	It can be referred to as outer or external software testing.	It is the inner or the internal software testing.
6.	It is a functional test of the software.	It is a structural test of the software.
7.	This testing can be initiated based on the requirement specifications document.	This type of testing of software is started after a detail design document.
8.	No knowledge of programming is required.	It is mandatory to have knowledge of programming.
9.	It is the behavior testing of the software.	It is the logic testing of the software.

S. No.	Black Box Testing	White Box Testing
10.	It is applicable to the higher levels of testing of software.	It is generally applicable to the lower levels of software testing.
11.	It is also called closed testing.	It is also called as clear box testing.
12.	It is least time consuming.	It is most time consuming.
13.	It is not suitable or preferred for algorithm testing.	It is suitable for algorithm testing.
14.	Can be done by trial and error ways and methods.	Data domains along with inner or internal boundaries can be better tested.
15.	**Example:** Search something on google by using keywords	**Example:** By input to check and verify loops
16.	**Black-box test design techniques-** • Decision table testing • All-pairs testing • Equivalence partitioning • Error guessing	**White-box test design techniques-** • Control flow testing • Data flow testing • Branch testing
17.	**Types of Black Box Testing:** • Functional Testing • Non-functional testing • Regression Testing	**Types of White Box Testing:** • Path Testing • Loop Testing • Condition testing
18.	It is less exhaustive as compared to white box testing.	It is comparatively more exhaustive than black box testing.