Type: MCQ

Q1. Software Engineering describes (0.5)

1. **Systematic and Cost Effective techniques for Software Development.
2. Industrial Engineering
3. Clinical Engineering
4. None of the other alternatives

Q2. In a Build and Fix Style (0.5)

1. **A program is quickly developed without making any specification plan or design.
2. Code is not written.
3. Program is not written.
4. None of the other alternatives.

Q3. Feedback paths are available in (0.5)

1. Classical Water Fall Model.
2. **Iterative Waterfall Model.
3. Requirements.
4. None of the other alternatives.

Q4. Development of GUI and handling technical issues for projects that are not clear to development team is done by (0.5)

1. V model.
2. Classical Waterfall Model.
3. **Prototype Model.
4. None of the other alternatives.

Q5. One of the goals of RAD model is (0.5)

1. Producing Bad Software Producs.
2. Producing Un satisfactory requirements for Making Software Products.
3. **To decrease the time taken and cost incurred to develop the software systems.
4. None of the other alternatives.

Q6. The final outcome of the Requirement Analysis and Requiremnet Specification phase is (0.5)

1. **SRS document.
2. Feasibility Study only.
3. Wrong Software Testing.
4. None of the other alternatives.

Q7. Each High Level Requirement (0.5)

1. Is not required for the Software Product.

2. **Characterises a way of system usage by some user to perform some meaningful peace of work.
3. Is not a part of SRS document.
4. None of the other alternatives.

Q8. Measure of Functional Strength is (0.5)

1. Content Coupling.
2. **Cohesion.
3. Stamp Coupling.
4. None of the other alternatives.

Q9. Degree of Coupling between two modules depends on (0.5)

1. Logical Cohesion.
2. **Interface Complexity.
3. Procedural Cohesion.
4. None of the other alternatives.

Q10. In Content Coupling there is (0.5)

1. **Sharing Code
2. No Code.
3. Only data.
4. None of the other alternatives.

Type: DES

Q11. A module has different functions. Illustrate how do you show cohesion and coupling in terms of their strengths. (4)

Soln: Types of Cohesion and Coupling.(2+2)

Q12. A library information system has three requirements, New Member, Renewal and Cancel Membership. Give the Decision Tree and Decision Table. (4)

Soln: Decision Tree and Decision Table (2 + 2)

Q13. Describe the drawbacks of the Exploratory style of Software Development. (3)

Soln: Drawbacks (0.5 * 6 or 1*3)

Q14. With the help of an example explain the concept of Abstraction in Software Engineering. (3)

Soln: Example: Countries(3 M)

Q15. Which software development model is most suitable for developing software for railway signalling systems to manage train movements and ensure safety on railway tracks, and why is this model a good fit for such an application? (3)

Q16. The GUI part of a software system is almost always developed using the prototyping model. Justify the statement with enough reasons. (3)

15)Which software development model is most suitable for developing software for railway signalling systems to manage train movements and ensure safety on railway tracks, and why is this model a good fit for such an application?

Answer Scheme:

The choice of a software development model for a project involving railway signalling systems is crucial due to the high level of complexity, criticality, and safety requirements associated with the domain. In this context, the V-Model (Validation and Verification Model) is the most suitable development model for the following reasons: 1M

- Emphasis on Validation and Verification: The V-Model places a strong emphasis on verification and validation activities at each stage of the development process. This aligns well with the criticality of railway signalling systems, where safety and reliability are utmost important.
- Early Detection of Issues: By following the V-Model, potential issues, including safety-critical defects and deviations from requirements, are detected early in the development process. This enables timely corrections and reduces the cost and risk of addressing issues at later stages.
- Traceability: The V-Model enforces strong traceability between requirements and the verification activities. In railway signaling systems, traceability is essential for ensuring that each safety-critical requirement is implemented correctly and validated thoroughly.
- Risk Mitigation: The V-Model allows for early identification and mitigation of risks associated with railway signaling systems. This is crucial in an environment where safety is a top priority, as it minimizes the likelihood of system failures.

2M

16) The GUI part of a software system is almost always developed using the prototyping model. Justify the statement with enough reasons.

The statement that the GUI (Graphical User Interface) part of a software system is almost always developed using the prototyping model can be justified with several reasons:

User-Centric Design: Prototyping allows for a user-centric design approach. GUI is the primary point of interaction between a user and the software. Developing the GUI through prototyping ensures that the end-users' needs and expectations are considered and incorporated into the design from an early stage.

Visualization of Design: Prototyping provides a visual representation of the GUI, which is essential for stakeholders, including users and developers, to visualize how the software will look and feel. This visual aspect is crucial for design feedback and validation.

Rapid Iteration: Prototyping enables rapid iteration and quick feedback gathering. It allows designers and users to experiment with various interface elements, layouts, and interactions, making it easier to refine the design and user experience.

Early User Testing: Prototyping facilitates early user testing. Users can interact with a working prototype to provide feedback on the usability and user-friendliness of the GUI. This early feedback can help identify and address design flaws and usability issues before extensive development takes place.

Reduced Risk: Developing the GUI in isolation without user involvement can lead to misaligned expectations and potential rework. Prototyping helps reduce the risk of building a final GUI that users find unsatisfactory or challenging to use, leading to costly changes later in the development process.

Improved Communication: Prototypes serve as a communication tool between designers, developers, and stakeholders. They provide a common reference point for discussing design choices, which can help avoid misunderstandings and misinterpretations.

Flexibility in Design Choices: GUIs often require careful consideration of aesthetics, layout, and user flow. Prototyping allows for flexibility in experimenting with different design choices, such as colors, fonts, and visual elements, to find the most effective and visually appealing solution.

Alignment with User Needs: Users' needs and preferences may evolve or change during the development process. Prototyping allows for real-time adjustments to the GUI based on changing requirements or user feedback, ensuring that the software remains aligned with user needs.

Cost-Efficiency: Fixing design issues in the early stages of development is far more cost-effective than making changes in later stages. Prototyping helps identify and address design issues early, reducing the cost of rework and redesign.

Enhanced User Satisfaction: A well-designed GUI, developed with user input and validated through prototyping, leads to a more satisfying user experience. Users are more likely to adopt and appreciate software that meets their expectations and usability preferences.

Q17. List all the basic activities of the software development process of Extreme Programming Model? (3)

Q18. Explain why the spiral life cycle model is considered to be a meta model. (2)

17) Explain why the spiral life cycle model is considered to be a meta model. (2 M)

Ans: The spiral life cycle model is considered to be a meta model, because the spiral model subsumes all other development models. a single loop spiral actually represents the waterfall model. The spiral model uses a prototyping approach by first building a prototype before the actual product development starts. Also, the spiral model can be considered as supporting the evolutionary model —

the iterations along the spiral can be considered as evolutionary levels through which the complete system is built. This enables the developer to understand and resolve the risks at each evolutionary level (i.e., iteration along the spiral). The spiral model uses prototyping as a risk reduction mechanism and also retains the systematic step-wise approach of the waterfall model.

Reason for meta model – 0.5 M; 3 different models mention – 3X 0.5 = 1.5 M

Total- 2 Marks

18)List all the basic activities of the software development process of Extreme Programming Model?(3M)

Ans. The basic activities of the software development process of Extreme Programming Model are:
Coding: XP argues that code is the crucial part o f any system development process, since without code it is not possible to have a working system. Therefore, utmost care and attention need to be placed on coding activity. However, the concept of code as used in XP has a slightly different meaning from what is traditionally understood. For example, coding activity includes drawing diagrams (modelling) that will be transformed to code, scripting a web-based system, and choosing among several alternative solutions.
Testing: XP places high importance on testing and considers it be the primary means for developing a fault-free software.
Listening: The developers need to carefully listen to the customers if they have to develop a good quality software. Programmers may not necessarily be having an in-depth knowledge of the the specific domain of the system under development. On the other hand, customers usually have this domain knowledge. Therefore, for the programmers to properly understand what the functionality of the system should be, they have to listen to the customer.
Designing: Without proper design, a system implementation becomes too complex and the dependencies within the system become too numerous and it becomes very difficult to comprehend the solution, and thereby making maintenance prohibitively expensive. A good design should result in elimination of complex dependencies within a system. Thus, effective use of a suitable design technique is emphasised.
Feedback: It espouses the wisdom: “A system staying out of users is trouble waiting to happen”. It recognises the importance of user feedback in understanding the exact customer requirements. The time that elapses between the development of a version and collection of feedback on it is critical to learning and making changes. It argues that frequent contact with the customer makes the development effective.
Simplicity: A corner-stone of XP is based on the principle: “build something simple that will work today, rather than trying to build something that would take time and yet may never be used”. This in essence means that attention should be focused on specific features that are immediately needed and making them work, rather than devoting time and energy on speculations about future requirements.
Any 6 basic activities – 6 X0.5 = 3 Marks