DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING Software Engineering

Third Year CSE(Sem:I)

2 marks Questions and Answers

UNIT 1

1. **What are software myths**

Answer: Management myths: We already have a book that's full of standards and procedures for building software. Won't that provide my people with everything they need to know?
If we get behind schedule, we can add more programmers and catch up (sometimes called the "Mongolian horde" concept).
If I decide to outsource the software project to a third party, I can just relax and let that firm build it
Customer myths: A general statement of objectives is sufficient to begin writing programs—we can fill in the details later
Software requirements continually change, but change can be easily accommodated because software is flexible.
Practitioner's myths: Once we write the program and get it to work, our job is done
Until I get the program "running" I have no way of assessing its quality
The only deliverable work product for a successful project is the working program.
Software engineering will make us create voluminous and unnecessary documentation and will invariably slow us down.

2. **What is water fall model?**

Ans: The waterfall model, sometimes called the classic life cycle, suggests a systematic, sequential approach6 to software development that begins with customer specification of requirements and progresses through planning, modeling, construction, and deployment, culminating in ongoing support of the completed software

3. **What is spiral model?**

Ans: The spiral development model is a risk-driven process model generator that is used to guide multi-stakeholder concurrent engineering of software intensive systems. It has two main distinguishing features. One is a cyclic approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk. The other is a set of anchor point milestones for ensuring stakeholder commitment to feasible and mutually satisfactory system solutions.

4. **What are the steps in component based Model?**

Ans: The component-based development model incorporates the following steps

1. Available component-based products are researched and evaluated for the application domain in question.
2. Component integration issues are considered.
3. A software architecture is designed to accommodate the components.
4. Components are integrated into the architecture.

5. Comprehensive testing is conducted to ensure proper functionality.

5 **What are the phases in unified process?**

Ans: The inception phase: of the UP encompasses both customer communication and planning activities. By collaborating with stakeholders, business requirements for the software are identified; a rough architecture for the system is proposed; and a plan for the iterative, incremental nature of the ensuing project is developed.

The elaboration phase : encompasses the communication and modeling activities of the generic process model

The construction phase: of the UP is identical to the construction activity defined for the generic software process.

The transition phase: of the UP encompasses the latter stages of the generic construction activity and the first part of the generic deployment (delivery and feedback) activity.

The production phase: of the UP coincides with the deployment activity of the generic process.

6. What is Personal Software Process (PSP)and Team software process(TSP)?

Ans: The Personal Software Process (PSP) emphasizes personal measurement of both the work product that is produced and the resultant quality of the work product. In addition PSP makes the practitioner responsible for project planning (e.g., estimating and scheduling) and empowers the practitioner to control the quality of all software work products that are developed.

The goal of TSP is to build a "selfdirected" project team that organizes itself to produce high-quality software.

7. Give human factors for agile process

Ans: Competence. In an agile development (as well as software engineering) context, "competence" encompasses innate talent, specific software-related skills, and overall knowledge of the process that the team has chosen to apply.

Self-organization. In the context of agile development, self-organization implies three things: (1) the agile team organizes itself for the work to be done, (2) the team organizes the process to best accommodate its local environment, (3) the team organizes the work schedule to best achieve delivery of the software increment

Collaboration. Software engineering (regardless of process) is about assessing, analyzing, and using information that is communicated to the software team; creating information that will help all stakeholders understand the work of the team.

8 **What is process Assesment?**

Ans: The process should be assessed to ensure that it meets a set of basic process criteria that have been shown to be essential for successful software engineering.

Many different assessment options are available:

SCAMPI(standard CMMI Assessment method for Process improvement)--- assessed for CMMI standards compliance

CBA IPI---(CMM Based Appraisal for internal process Improvement)

SPICE --- assessed for ISO/IEC15504 compliance

ISO 9001:2000 --- assessed for ISO 9001 compliance

9 **Define software Engineering? Give the characteristics of software engineering**

Ans: the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software Software is developed or engineered not manufactured.

Software doesn't' t wear out

Although industry is running towards component based software is custom build.

10 **What are the umbrella activities in a software development life cycle**

The umbrella activities in a software development life cycle process include the following:

- Software Project Management.
- Formal Technical Reviews.
- Software Quality Assurance.
- Software Configuration Management.
- Re-usability Management.
- Risk Management.
- Measurement and Metrics.

UNIT 2

1. What are requirement engineering tasks?

Ans: Inception—Establish a basic understanding of the problem and the nature of the solution.

- Elicitation—Draw out the requirements from stakeholders.
- Elaboration—Create an analysis model that represents information, functional, and behavioral aspects of the requirements.
- Negotiation—Agree on a deliverable system that is realistic for developers and customers.
- Specification—Describe the requirements formally or informally.
- Validation—Review the requirement specification for errors, ambiguities, omissions, and conflicts.
- Requirements management—Manage changing requirements.

2. **What is Critical Path Method (CPM)**

Ans: CPM: Shows the minimum amount of time it will take to complete a project

- Reveals those activities that are most critical to completing the project on time

- Real time (actual time): estimated amount of time required for the activity to be completed
- Available time: amount of time available in the schedule for the activity's completion
- Slack time: the difference between the available time and the real time for that activity
- Critical path: the slack at every node is zero
  - can be more than one in a project schedule
- Slack time = available time – real time
  = latest start time – earliest start time

3 What are risk management activities?

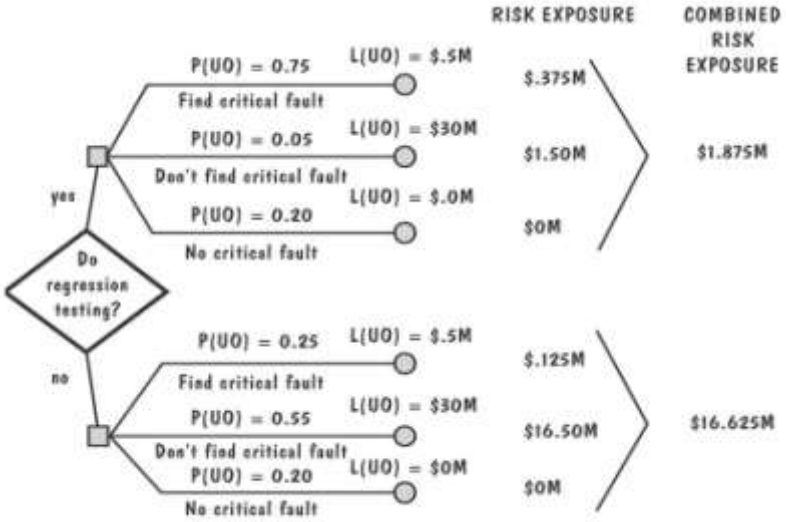
Ans: Three strategies for risk reduction

a. *Avoiding the risk*: change requirements for performance or functionality
b. *Transferring the risk*: transfer to other system, or buy insurance
c. *Assuming the risk*: accept and control it

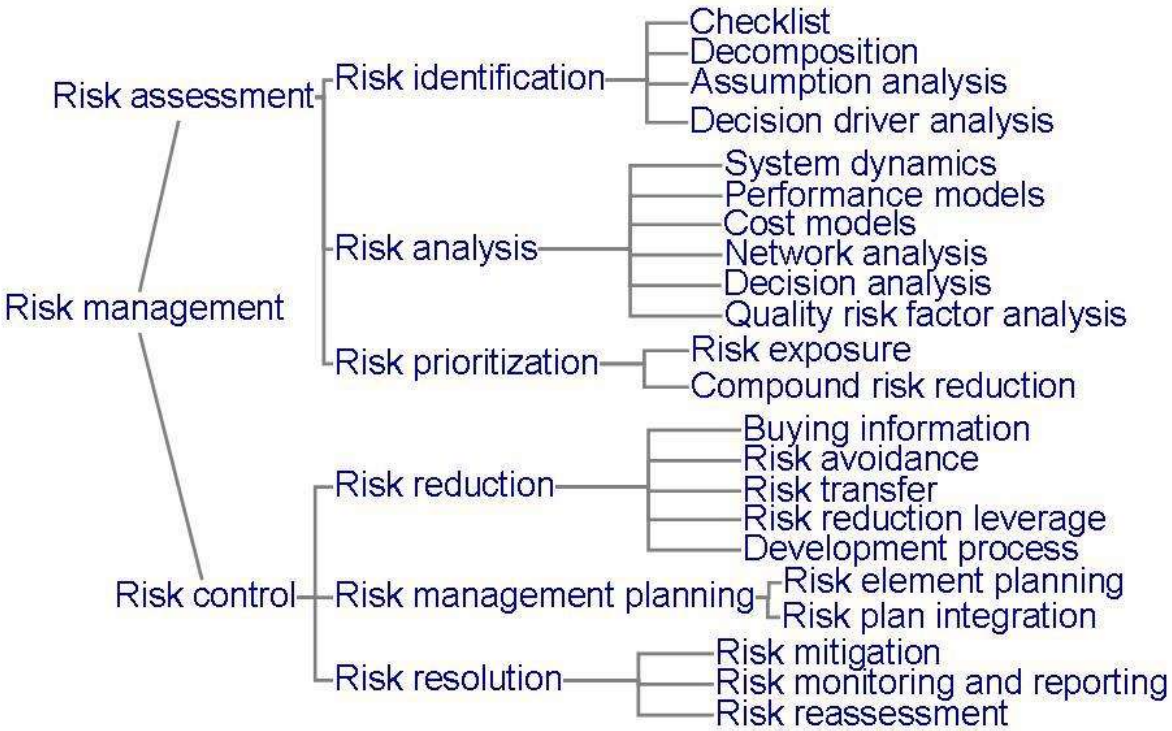
Cost of reducing risk

d. *Risk leverage* = (risk exposure before reduction) – (risk exposure after reduction) / (cost of risk reduction

4 **Give an example of risk exposure.**



4 Give the risk management activities



6 Give the equations for Bailey-Basili technique.

Ans: Minimize standard error estimate to produce an equation such as $E = 5.5 + 0.73S^{1.16}$

Adjust initial estimate based on the difference ratio

a. If *R* is the ratio between the actual effort, *E*, and the predicted effort, *E',* then the effort adjustment is defined as

b. $ER_{adj} = R - 1$ if $R > 1$

$= 1 - 1/R$ if $R < 1$

- Adjust the initial effort estimate E_{adj}
  - $E_{adj} = (1 + ER_{adj})E$ if $R > 1$

$= E/(1 + ER_{adj})$ if $R < 1$

6 **What vis an usecase?**

Ans: A use-case scenario is a story about how someone or something external to the software (known as an actor) interacts with the system.

Each scenario answers the following questions:

Who is the primary actor, the secondary actor(s)?

a. What are the actor's goals?
b. What preconditions should exist before the story begins?
c. What main tasks or functions are performed by the actor?
d. What exceptions might be considered as the story is described?
e. What variations in the actor's interaction are possible?
f. What system information will the actor acquire, produce, or change?

7 What is analysis patterns?

Ans: Pattern name: Captures the essence of the pattern.

Intent: What the pattern accomplishes or represents.

Forces and context: External issues (forces) that affect how the pattern is used, and external issues resolved when the pattern is applied.

Solution: How the pattern is applied to solve the problem; emphasizes structural and behavioral issues.

Consequences: What happens when the pattern is applied; what trade-offs exist during its application.

Design: How the pattern can be achieved via known design patterns.

Known uses: Examples of uses within actual systems.

Related patterns: Patterns related to the named pattern because

(1) they are commonly used with the named pattern;
(2) they are structurally similar to the named pattern;they are a variation of the named pattern

8 Give some vValidating Requirements

Ans: Is each requirement consistent with the objective of the system?

Have all requirements been specified at the proper level of abstraction?

Is the requirement really necessary?

Is each requirement bounded and unambiguous?

Does each requirement have attribution?

Do any requirements conflict with other requirements?

Is each requirement achievable in the system's technical environment?

Is each requirement testable, once implemented?

9 What is Risk ? Give 3 strategies to avoid risk?

Ans: Risk is an unwanted event that has negative consequences.

Three strategies for risk reduction

Avoiding the risk: change requirements for performance or functionality.

Transferring the risk: transfer to other system, or buy insurance.

Assuming the risk: accept and control it.

10 What is project schedule.

Ans: Describes the software-development cycle for a particular project by enumerating the phases or stages of the project breaking each phase into discrete tasks or activities to be completed Portrays the interactions among the activities and estimates the times that each task or activity will take.

UNIT 3

1. What are the elements of analysis model?

- Scenario Based Elements
  - Use cases (text & diagrams), Activity, Swim lane
- Flow Oriented Elements
  - Data flow, Control flow diagrams, Processing narratives
- Class Based Elements
  - Class diagrams, CRC models, collaboration diagrams
- Behavioral Elements
  - State diagrams, sequence diagrams

2 How to allocate responsibilities to classes:

- System intelligence should be distributed across classes to best address the needs of the problem
- Each responsibility should be stated as generally as possible (high enough in hierarchy)
- Information and the behavior related to it should reside within the same class (encapsulation)
- Information about one thing should be localized with a single class, not distributed across multiple classes.
- Responsibilities should be shared among related classes, when appropriate.

3 Give the steps for Reviewing the CRC Model

- All participants in the review (of the CRC model) are given a subset of the CRC model index cards.
  - Cards that collaborate should be separated (i.e., no reviewer should have two cards that collaborate).
- All use-case scenarios (and corresponding use-case diagrams) should be organized into categories.
- The review leader reads the use-case deliberately.
  - As the review leader comes to a named object, she passes a token to the person holding the corresponding class index card.
- When the token is passed, the holder of the class card is asked to describe the responsibilities noted on the card.

- The group determines whether one (or more) of the responsibilities satisfies the use-case requirement.

- **4 Give the class types in building analysis model.**
- Entity classes, also called model or business classes, are extracted directly from the statement of the problem (e.g., FloorPlan and Sensor). (persist)
- Boundary classes are used to create the interface (e.g., interactive screen or printed reports) that the user sees and interacts with as the software is used. (like cameraWindow)
- Controller classes manage a "unit of work" from start to finish (design time). That is, controller classes can be designed to manage
  - the creation or update of entity objects;
  - the instantiation of boundary objects as they obtain information from entity objects;
  - complex communication between sets of objects;

5 What is the Purpose of Design?

- Design is where customer requirements, business needs, and technical considerations all come together in the formulation of a product or system
- The design model provides detail about the software data structures, architecture, interfaces, and components
- The design model can be assessed for quality and be improved before code is generated and tests are conducted
  - Does the design contain errors, inconsistencies, or omissions?
  - Are there better design alternatives?
  - Can the design be implemented within the constraints, schedule, and cost that have been established?

6 Give the design quality guidelines?

1) A design should exhibit an architecture that
   a) Has been created using recognizable architectural styles or patterns
   b) Is composed of components that exhibit good design characteristics
   c) Can be implemented in an evolutionary fashion, thereby facilitating implementation and testing
2) A design should be modular; that is, the software should be logically partitioned into elements or subsystems
3) A design should contain distinct representations of data, architecture, interfaces, and components
4) A design should lead to data structures that are appropriate for the classes to be implemented and are drawn from recognizable data patterns.

7 **Give the design concepts?**

8 Abstraction

9 Procedural abstraction – a sequence of instructions that have a specific and limited function

10 Data abstraction – a named collection of data that describes a data object

11 Architecture

12 The overall structure of the software and the ways in which the structure provides conceptual integrity for a system

13 Consists of components, connectors, and the relationship between them

14 Patterns

15 A design structure that solves a particular design problem within a specific context

16 It provides a description that enables a designer to determine whether the pattern is applicable, whether the pattern can be reused, and whether the pattern can serve as a guide for developing similar patterns