Software Engineering

T.E – B

Unit 5

Faculty Incharge: Ms. Drashti Shrimal

Q.1 What is Risk in SE? Explain its types.

"Tomorrow problems are today's risk." Hence, a clear definition of a "risk" is a problem that could cause some loss or threaten the progress of the project, but which has not happened yet.

These potential issues might harm cost, schedule or technical success of the project and the quality of our software device, or project team morale.

Risk Management is the system of identifying addressing and eliminating these problems before they can damage the project.

We need to differentiate risks, as potential issues, from the current problems of the project.

A software project can be concerned with a large variety of risks. In order to be adept to systematically identify the significant risks which might affect a software project, it is essential to classify risks into different classes. The project manager can then check which risks from each class are relevant to the project.

There are three main classifications of risks which can affect a software project:

1. Project risks
2. Technical risks
3. Business risks

1. Project risks: Project risks concern differ forms of budgetary, schedule, personnel, resource, and customer-related problems. A vital project risk is schedule slippage. Since the software is intangible, it is very tough to monitor and control a software project. It is very tough to control something which cannot be identified. For any manufacturing program, such as the manufacturing of cars, the plan executive can recognize the product taking shape.

2. Technical risks: Technical risks concern potential method, implementation, interfacing, testing, and maintenance issue. It also consists of an ambiguous specification, incomplete specification, changing specification, technical uncertainty, and technical obsolescence. Most technical risks appear due to the development team's insufficient knowledge about the project.

3. Business risks: This type of risks contain risks of building an excellent product that no one need, losing budgetary or personnel commitments, etc.

Other risks:

1. Known risks: Those risks that can be uncovered after careful assessment of the project program, the business and technical environment in which the plan is being developed, and more reliable data sources (e.g., unrealistic delivery date)

2. Predictable risks: Those risks that are hypothesized from previous project experience (e.g., past turnover)

3. Unpredictable risks: Those risks that can and do occur, but are extremely tough to identify in advance.

Risk Identification:

Identifying risk is one of most important or essential and initial steps in risk management process. By chance, if failure occurs in identifying any specific or particular risk, then all other steps that are involved in risk management will not be implemented for that particular risk. For identifying risk, project team should review scope of program, estimate cost, schedule, technical maturity, parameters of key performance, etc. To manage risk, project team or organization are needed to know about what risks it faces, and then to evaluate them.

Methods for Identifying Risks:

1. **Checklist Analysis –** Checklist Analysis is type of technique generally used to identify or find risks and manage it. The checklist is basically developed by listing items, steps, or even tasks and is then further analyzed against criteria to just identify and determine if procedure is completed correctly or not.
2. **Brainstorming –** This technique provides and gives free and open approach that usually encourages each and everyone on project team to participate. It also results in greater sense of ownership of project risk, and team generally committed to managing risk for given time period of project.
3. **SWOT Analysis –** Strengths-Weaknesses-Opportunities-Threat (SWOT) is very technique and helpful for identifying risks within greater organization context. It is

generally used as planning tool for analyzing business, its resources, and also its environment simply by looking at internal strengths and weaknesses and opportunities and threats in external environment.

4. **Flowchart Method –** This method allows for dynamic process to be diagrammatically represented in paper. This method is generally used to represent activities of process graphically and sequentially to simply identify the risk.

Risk Assessment:

Risk assessment simply means to describe the overall process or method to identify risk and problem factors that might cause harm. It is actually a systematic examination of a task or project that you perform to simply identify significant risks, problems, hazards, and then to find out control measures that you will take to reduce risk. The best approach is to prepare a set of questions that can be answered by project managers in order to assess overall project risks.

These questions are shown below:

- Will the project get proper support from the customer manager?
- Are end-users committed to software that has been produced?
- Is there a clear understanding of the requirements?
- Is there an active involvement of customers in the requirement definition?

Thus, the number of negative answers to these questions represents the severity of the impact of risk on the overall project. It is not about creating or making a large number of work papers, but rather simply identify and find out measures to control risks in your workplace.

Q.2 Explain Principle of Risk Management.

1. **Global Perspective:** In this, we review the bigger system description, design, and implementation. We look at the chance and the impact the risk is going to have.
2. **Take a forward-looking view:** Consider the threat which may appear in the future and create future plans for directing the next events.
3. **Open Communication:** This is to allow the free flow of communications between the client and the team members so that they have certainty about the risks.

4. **Integrated management:** In this method risk management is made an integral part of project management.
5. **Continuous process:** In this phase, the risks are tracked continuously throughout the risk management paradigm.

Q.3 Explain the RMM model.

A risk management technique is usually seen in the software Project plan. This can be divided into Risk Mitigation, Monitoring, and Management Plan (RMMM). In this plan, all works are done as part of risk analysis. As part of the overall project plan project manager generally uses this RMMM plan.

In some software teams, risk is documented with the help of a Risk Information Sheet (RIS). This RIS is controlled by using a database system for easier management of information i.e creation, priority ordering, searching, and other analysis. After documentation of RMMM and start of a project, risk mitigation and monitoring steps will start.

Risk Mitigation:
It is an activity used to avoid problems (Risk Avoidance).
Steps for mitigating the risks as follows:
1. Finding out the risk.
2. Removing causes that are the reason for risk creation.
3. Controlling the corresponding documents from time to time.
4. Conducting timely reviews to speed up the work.

Risk Monitoring:
It is an activity used for project tracking.
It has the following primary objectives as follows.
1. To check if predicted risks occur or not.
2. To ensure proper application of risk aversion steps defined for risk.
3. To collect data for future risk analysis.
4. To allocate what problems are caused by which risks throughout the project.

Risk Management and planning:
It assumes that the mitigation activity failed and the risk is a reality. This task is done by Project manager when risk becomes reality and causes severe problems. If the project manager effectively uses project mitigation to remove risks successfully then it is easier to manage the risks. This shows that the response that will be taken for each risk by a manager. The main objective of the risk management plan is the risk register. This risk register describes and focuses on the predicted threats to a software project.

Example:
Let us understand RMMM with the help of an example of high staff turnover.

Risk Mitigation:
To mitigate this risk, project management must develop a strategy for reducing turnover. The possible steps to be taken are:

- Meet the current staff to determine causes for turnover (e.g., poor working conditions, low pay, competitive job market).
- Mitigate those causes that are under our control before the project starts.
- Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave.
- Organize project teams so that information about each development activity is widely dispersed.
- Define documentation standards and establish mechanisms to ensure that documents are developed in a timely manner.
- Assign a backup staff member for every critical technologist.

Risk Monitoring:
As the project proceeds, risk monitoring activities commence. The project manager monitors factors that may provide an indication of whether the risk is becoming more or less likely. In the case of high staff turnover, the following factors can be monitored:

- General attitude of team members based on project pressures.
- Interpersonal relationships among team members.
- Potential problems with compensation and benefits.
- The availability of jobs within the company and outside it.

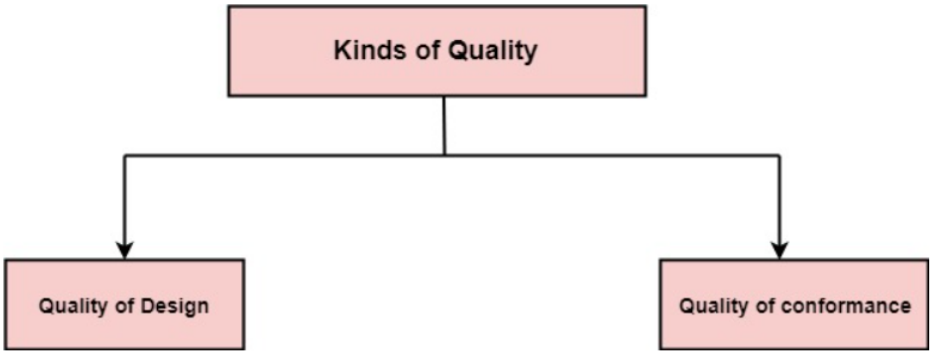
Risk Management:
Risk management and contingency planning assumes that mitigation efforts have failed and that the risk has become a reality. Continuing the example, the project is well underway, and a number of people announce that they will be leaving. If the mitigation strategy has been followed, backup is available, information is documented, and knowledge has been dispersed across the team. In addition, the project manager may temporarily refocus resources (and readjust the project schedule) to those functions that are fully staffed, enabling newcomers who must be added to the team to "get up to the speed**".**

Drawbacks of RMMM:
- It incurs additional project costs.
- It takes additional time.
- For larger projects, implementing an RMMM may itself turn out to be another tedious project.

- RMMM does not guarantee a risk-free project, infact, risks may also come up after the project is delivered.

Q.4 Explain Software Quality Assurance.

Quality defines to any measurable characteristics such as correctness, maintainability, portability, testability, usability, reliability, efficiency, integrity, reusability, and interoperability.



Quality of Design: Quality of Design refers to the characteristics that designers specify for an item. The grade of materials, tolerances, and performance specifications that all contribute to the quality of design.

Quality of conformance: Quality of conformance is the degree to which the design specifications are followed during manufacturing. Greater the degree of conformance, the higher is the level of quality of conformance.

Software Quality: Software Quality is defined as the conformance to explicitly state functional and performance requirements, explicitly documented development standards, and inherent characteristics that are expected of all professionally developed software.

Software Quality Assurance (SQA) is simply a way to assure quality in the software. It is the set of activities which ensure processes, procedures as well as standards are suitable for the project and implemented correctly.
Software Quality Assurance is a process which works parallel to development of software. It focuses on improving the process of development of software so that problems can be prevented before they become a major issue. Software Quality Assurance is a kind of Umbrella activity that is applied throughout the software process.

Major Software Quality Assurance Activities:

1. **SQA Management Plan:**
   Make a plan for how you will carry out the sqa throughout the project. Think about which

set of software engineering activities are the best for project. Check level of sqa team skills.

2. **Set The Check Points:**
   SQA team should set checkpoints. Evaluate the performance of the project on the basis of collected data on different check points.

3. **Multi testing Strategy:**
   Do not depend on a single testing approach. When you have a lot of testing approaches available use them.

4. **Measure Change Impact:**
   The changes for making the correction of an error sometimes re introduces more errors keep the measure of impact of change on project. Reset the new change to change check the compatibility of this fix with whole project.

5. **Manage Good Relations:**
   In the working environment managing good relations with other teams involved in the project development is mandatory. Bad relation of sqa team with programmers team will impact directly and badly on project. Don't play politics.

Benefits of Software Quality Assurance (SQA):

1. SQA produces high quality software.
2. High quality application saves time and cost.
3. SQA is beneficial for better reliability.
4. SQA is beneficial in the condition of no maintenance for a long time.
5. High quality commercial software increase market share of company.
6. Improving the process of creating software.
7. Improves the quality of the software.

Disadvantage of SQA:
There are a number of disadvantages of quality assurance. Some of them include adding more resources, employing more workers to help maintain quality and so much more.

Q.5 FTR in SE.

Formal Technical Review (FTR) is a software quality control activity performed by software engineers.

Objectives of formal technical review (FTR):
Some of these are:

- Useful to uncover error in logic, function and implementation for any representation of the software.
- The purpose of FTR is to verify that the software meets specified requirements.
- To ensure that software is represented according to predefined standards.

- It helps to review the uniformity in software that is development in a uniform manner.
- To makes the project more manageable.

In addition, the purpose of FTR is to enable junior engineer to observe the analysis, design, coding and testing approach more closely. FTR also works to promote back up and continuity become familiar with parts of software they might not have seen otherwise. Actually, FTR is a class of reviews that include walkthroughs, inspections, round robin reviews and other small group technical assessments of software. Each FTR is conducted as meeting and is considered successful only if it is properly planned, controlled and attended.

Q.6 Walkthroughs in SE.

In the walkthrough, the code or document is read by the author, and others who are present in the meeting can note down the important points or can write notes on the defects and can give suggestions about them. The walkthrough is an informal way of testing, no formal authority is been involved in this testing.

As there is an informal way of testing involved so there is no need for a moderator while performing a walkthrough. We can call a walkthrough an open-ended discussion, it does not focus on the documentation. Defect tracking is one of the challenging tasks in the walkthrough.

Following are some of the objectives of the walkthrough.

- To detect defects in developed software products.
- To fully understand and learn the development of software products.
- To properly explain and discuss the information present in the document.
- To verify the validity of the proposed system.
- To give suggestions and report them appropriately with new solutions and ideas.
- To provide an early “proof of concept”.

Q.7 SCM in Software Engineering.

Whenever a software is build, there is always scope for improvement and those improvements brings changes in picture. Changes may be required to modify or update any existing solution or to create a new solution for a problem.
Requirements keeps on changing on daily basis and so we need to keep on upgrading our systems based on the current requirements and needs to meet desired outputs.
Changes should be analyzed before they are made to the existing system, recorded before they are implemented, reported to have details of before and after, and controlled in a manner that will improve quality and reduce error.
This is where the need of System Configuration Management comes. **System Configuration Management (SCM)** is an arrangement of exercises which controls change by recognizing the items for change, setting up connections between those things, making/characterizing

instruments for overseeing diverse variants, controlling the changes being executed in the current framework, inspecting and revealing/reporting on the changes made.

It is essential to control the changes in light of the fact that if the changes are not checked legitimately then they may wind up undermining a well-run programming. In this way, SCM is a fundamental piece of all project management activities.

Processes involved in SCM – Configuration management provides a disciplined environment for smooth control of work products. It involves the following activities:

1. **Identification and Establishment –** Identifying the configuration items from products that compose baselines at given points in time (a baseline is a set of mutually consistent Configuration Items, which has been formally reviewed and agreed upon, and serves as the basis of further development). Establishing relationship among items, creating a mechanism to manage multiple level of control and procedure for change management system.
2. **Version control –** Creating versions/specifications of the existing product to build new
3. **Change control –** Controlling changes to Configuration items (CI).
   A change request (CR) is submitted and evaluated to assess technical merit, potential side effects, overall impact on other configuration objects and system functions, and the projected cost of the change. The results of the evaluation are presented as a change report, which is used by a change control board (CCB) —a person or group who makes a final decision on the status and priority of the change. An engineering change Request (ECR) is generated for each approved change. Also CCB notifies the developer in case the change is rejected with proper reason. The ECR describes the change to be made, the constraints that must be respected, and the criteria for review and audit. The object to be changed is "checked out" of the project database, the change is made, and then the object is tested again. The object is then "checked in" to the database and appropriate version control mechanisms are used to create the next version of the software.
   4. **Configuration auditing –** A software configuration audit complements the formal technical review of the process and product. It focuses on the technical correctness of the configuration object that has been modified.
   5. **Reporting –** Providing accurate status and current configuration data to developers, tester, end users, customers and stakeholders through admin guides, user guides, FAQs, Release notes, Memos, Installation Guide, Configuration guide etc .

System Configuration Management (SCM) is a software engineering practice that focuses on managing the configuration of software systems and ensuring that software components are properly controlled, tracked, and stored. It is a critical aspect of software development, as it helps to ensure that changes made to a software system are properly coordinated and that the system is always in a known and stable state.

SCM involves a set of processes and tools that help to manage the different components of a software system, including source code, documentation, and other assets. It enables teams to

track changes made to the software system, identify when and why changes were made, and manage the integration of these changes into the final product.

The main advantages of SCM are:

1. Improved productivity and efficiency by reducing the time and effort required to manage software changes.
2. Reduced risk of errors and defects by ensuring that all changes are properly tested and validated.
3. Increased collaboration and communication among team members by providing a central repository for software artifacts.
4. Improved quality and stability of software systems by ensuring that all changes are properly controlled and managed.

The main disadvantages of SCM are:

1. Increased complexity and overhead, particularly in large software systems.
2. Difficulty in managing dependencies and ensuring that all changes are properly integrated.
3. Potential for conflicts and delays, particularly in large development teams with multiple contributors.

Q.8 Software Maintenance and its types.

Software maintenance refers to the process of modifying and updating a software system after it has been delivered to the customer. This can include fixing bugs, adding new features, improving performance, or updating the software to work with new hardware or software systems. The goal of software maintenance is to keep the software system working correctly, efficiently, and securely, and to ensure that it continues to meet the needs of the users.

There are several key aspects of software maintenance, including:

1. Bug fixing: The process of finding and fixing errors and problems in the software.
2. Enhancements: The process of adding new features or improving existing features to meet the evolving needs of the users.
3. Performance optimization: The process of improving the speed, efficiency, and reliability of the software.
4. Porting and migration: The process of adapting the software to run on new hardware or software platforms.
5. Re-engineering: The process of improving the design and architecture of the software to make it more maintainable and scalable.
6. Documentation: The process of creating, updating, and maintaining the documentation for the software, including user manuals, technical specifications, and design documents.

Software maintenance is a critical part of the software development life cycle and is necessary to ensure that the software continues to meet the needs of the users over time. It is also important to consider the cost and effort required for software maintenance when planning and developing a software system.

Software maintenance is the process of modifying a software system after it has been delivered to the customer. The goal of maintenance is to improve the system's functionality, performance, and reliability and to adapt it to changing requirements and environments.

There are several types of software maintenance, including:

- **Corrective maintenance:** This involves fixing errors and bugs in the software system.
- **Adaptive maintenance:** This involves modifying the software system to adapt it to changes in the environment, such as changes in hardware or software.
- **Perfective maintenance**: This involves improving the functionality, performance, and reliability of the software system.
- **Preventive maintenance:** This involves taking measures to prevent future problems, such as updating documentation, reviewing and testing the system, and implementing preventive measures such as backups.
- Software maintenance is a continuous process that occurs throughout the entire life cycle of the software system. It is important to have a well-defined maintenance process in place, which includes testing and validation, version control, and communication with stakeholders.

Need for Maintenance –
Software Maintenance must be performed in order to:
- Correct faults.
- Improve the design.
- Implement enhancements.
- Interface with other systems.
- Accommodate programs so that different hardware, software, system features, and telecommunications facilities can be used.
- Migrate legacy software.
- Retire software.
- Requirement of user changes.
- Run the code fast

Lecture Takeaway:

1. **State and explain Risks with Risk Identification methods.**
2. **Explain Risk Management process/ RMM model.**
3. **Explain software maintenance with its types.**
4. **State characteristics/ principles of risks.**