SOFTWARE ENGINEERING & PROJECT MANAGEMENT (SEPM)

MODULE 4 – Risk Management, SCM & SQA

Based on Mumbai University PYQs (2023–2025)

This document contains detailed descriptive answers for ALL Module 4 questions from the uploaded SEPM question bank.

Covered Questions:

1. Explain Risk Management in detail.
2. Explain RMMM Plan.
3. Explain Software Configuration Management (SCM).
4. Explain SCM Activities.
5. Explain Software Quality Assurance (SQA).
6. Explain Formal Technical Reviews (FTR).
7. Explain Software Reliability.
8. Explain ISO Quality Standards.
9. Explain Risk Identification and Risk Projection.
10. Explain Software Quality Factors.
11. Explain Software Reviews.
12. Explain Reverse Engineering and Reengineering.

Q1. Explain Risk Management in Detail.

Introduction

Software projects are exposed to various types of risks during development. These risks may affect:

- Cost
- Schedule
- Quality
- Performance
- Project success

Risk Management helps organizations identify and control risks before they become major problems.

Definition of Risk

A risk is an uncertain event or condition that may negatively affect a software project.

Definition of Risk Management

Risk Management is the process of:

- Identifying risks
- Analyzing risks
- Prioritizing risks
- Reducing risks
- Monitoring risks

throughout the software development lifecycle.

Objectives of Risk Management

1. Reduce project failures.
2. Improve project planning.
3. Reduce uncertainties.
4. Minimize financial losses.
5. Improve software quality.
6. Ensure successful project completion.

Types of Risks in Software Engineering

1. Project Risks

Risks affecting project schedule and resources.

Examples

- Budget problems
- Schedule delays
- Resource shortages

---

2. Technical Risks

Risks related to technology and implementation.

Examples

- Design issues
- Integration problems
- Hardware failures

---

3. Business Risks

Risks affecting business operations.

Examples

- Market competition
- Product failure
- Customer dissatisfaction

Risk Management Process

The Risk Management process consists of the following activities:

```
Risk Identification → Risk Analysis → Risk Prioritization → Risk Mitigation → Risk Monitoring
```

---

1. Risk Identification

Possible risks are identified during project planning.

Sources of Risks

- Requirement changes
- Technology changes
- Lack of skilled developers
- Poor communication

- Hardware problems

Techniques for Risk Identification

1. Brainstorming
2. Interviews
3. Questionnaires
4. Historical Data Analysis
5. Checklists

---

2. Risk Analysis

Risk Analysis estimates:

- Probability of occurrence
- Impact of risk

Risk Exposure Formula

```
Risk Exposure = Probability × Loss
```

Example

Probability of server failure = 0.4 Loss due to failure = Rs 50,000

```
Risk Exposure = 0.4 × 50,000
              = Rs 20,000
```

---

3. Risk Prioritization

Risks are ranked according to severity.

Risk Priority Table

Risk	Probability	Impact	Priority
Budget Risk	High	High	Very High
Hardware Failure	Medium	High	High
Requirement Change	High	Medium	High

---

4. Risk Mitigation

Preventive actions are taken to reduce risks.

Examples

Risk	Mitigation Strategy
Staff Shortage	Hire additional developers
Requirement Changes	Frequent customer meetings
Server Failure	Backup systems

---

5. Risk Monitoring

Risks are continuously monitored during development.

Advantages of Risk Management

1. Better project control.
2. Reduced uncertainties.
3. Improved planning.
4. Reduced project failure.
5. Better decision-making.

Disadvantages of Risk Management

1. Time-consuming.
2. Additional cost.
3. Difficult for small projects.

Conclusion

Risk Management is essential for successful software project execution. It helps organizations reduce uncertainties and improve project quality.

Q2. Explain RMMM Plan.

Introduction

RMMM stands for:

Risk Mitigation, Monitoring and Management

RMMM Plan is prepared to handle project risks effectively.

Objectives of RMMM Plan

1. Reduce risks.
2. Monitor project risks.
3. Define corrective actions.
4. Improve project success rate.

Components of RMMM Plan

1. Risk Mitigation

Preventive actions taken before risk occurs.

Example

Providing developer training to reduce technical risk.

---

2. Risk Monitoring

Continuous observation of risks.

Activities

- Track identified risks
- Monitor risk indicators
- Update risk status

---

3. Risk Management

Actions taken when risks actually occur.

Example

Switching to backup servers during hardware failure.

RMMM Plan Table Example

Risk	Probability	Impact	Mitigation
Staff Turnover	High	High	Backup staffing
Requirement Changes	Medium	High	Customer meetings
Budget Risk	Medium	Medium	Cost monitoring

Advantages of RMMM Plan

1. Better risk handling.
2. Reduced project failure.
3. Improved planning.
4. Better resource utilization.

Conclusion

RMMM Plan helps organizations manage project risks systematically and efficiently.

Q3. Explain Software Configuration Management (SCM).

Introduction

Software systems continuously evolve during development and maintenance.

Changes in software must be controlled properly.

SCM helps manage these changes systematically.

Definition of SCM

Software Configuration Management is the process of:

- Identifying software components
- Controlling changes
- Managing software versions
- Maintaining software integrity

throughout the software lifecycle.

Objectives of SCM

1. Control software changes.
2. Maintain software consistency.
3. Prevent unauthorized modifications.
4. Improve team coordination.
5. Maintain software versions.

Need for SCM

SCM is needed because:

- Multiple developers work simultaneously.
- Software changes frequently.
- Old versions must be maintained.
- Errors may occur during updates.

SCM Activities

1. Configuration Identification

Software components are identified.

Examples

- Source code
- Documents
- Test cases

---

2. Version Control

Different software versions are maintained.

Example Tools

- Git
- GitHub
- SVN

---

3. Change Control

Changes are approved before implementation.

---

4. Configuration Status Accounting

Tracks software configuration details.

---

5. Configuration Auditing

Ensures software follows standards.

SCM Process Diagram

```
Configuration Identification
            ↓
Version Control
            ↓
Change Control
            ↓
Status Accounting
            ↓
Configuration Audit
```

Advantages of SCM

1. Better version management.
2. Improved collaboration.
3. Reduced development errors.
4. Easier maintenance.
5. Better project control.

Disadvantages of SCM

1. Additional management effort.
2. Requires training.
3. Increased documentation.

Conclusion

SCM is essential for controlling software changes and maintaining software quality throughout development.

Q4. Explain Software Quality Assurance (SQA).

Introduction

Quality is one of the most important aspects of software development.

Software Quality Assurance ensures that software meets required standards.

Definition of SQA

Software Quality Assurance is a planned and systematic set of activities used to ensure software quality.

Objectives of SQA

1. Improve software quality.
2. Prevent defects.
3. Ensure standards compliance.
4. Improve customer satisfaction.
5. Reduce maintenance cost.

SQA Activities

1. Quality Planning

Quality standards are defined.

---

2. Reviews and Audits

Software documents and code are reviewed.

---

3. Testing

Testing is performed to identify defects.

4. Configuration Management

Software changes are controlled.

---

5. Defect Tracking

Errors are recorded and corrected.

SQA Process Diagram

```
Quality Planning → Reviews → Testing → Audits → Improvement
```

Advantages of SQA

1. Better software quality.
2. Reduced defects.
3. Improved reliability.
4. Increased customer satisfaction.
5. Reduced maintenance cost.

Disadvantages of SQA

1. Additional cost.
2. Time-consuming.
3. Requires skilled professionals.

Conclusion

SQA helps organizations develop reliable, maintainable, and high-quality software systems.

Q5. Explain Formal Technical Reviews (FTR).

Introduction

Formal Technical Review is a software quality control activity used to identify defects before testing.

Objectives of FTR

1. Detect defects.
2. Ensure standards compliance.
3. Improve software quality.
4. Improve maintainability.
5. Reduce rework cost.

Participants in FTR

Participant	Responsibility
Moderator	Conducts review
Author	Explains work product
Reviewer	Identifies defects
Recorder	Records review findings

FTR Process

```
Planning → Preparation → Review Meeting → Rework → Follow-up
```

Review Guidelines

1. Review product, not developer.
2. Keep meeting short.

3. Prepare before meeting.
4. Focus on defect detection.

Advantages of FTR

1. Early defect detection.
2. Reduced testing effort.
3. Improved software quality.
4. Better communication.

Conclusion

FTR is an effective defect prevention technique that improves software quality significantly.

Q6. Explain Software Reliability.

Definition

Software Reliability is the probability that software performs correctly without failure for a specified period.

Factors Affecting Reliability

1. Defect density
2. Software complexity
3. Testing quality
4. Operating environment

Reliability Metrics

1. MTTF

Mean Time To Failure

2. MTTR

Mean Time To Repair

3. Availability

```
Availability = MTTF / (MTTF + MTTR)
```

Importance of Reliability

1. Improves customer trust.
2. Reduces failures.
3. Improves safety.
4. Reduces maintenance cost.

Conclusion

Software reliability is essential for developing dependable and high-quality software systems.

Q7. Explain ISO Quality Standards.

Introduction

ISO stands for:

International Organization for Standardization

ISO quality standards help organizations maintain software quality.

ISO 9001

ISO 9001 is the most popular quality management standard.

Objectives of ISO Standards

1. Improve quality.
2. Standardize processes.
3. Improve customer satisfaction.
4. Continuous improvement.

Advantages of ISO Standards

1. Better quality management.
2. Improved reputation.
3. Increased customer trust.
4. Better process control.

Conclusion

ISO standards help organizations maintain consistent software quality and improve customer satisfaction.

Q8. Explain Software Quality Factors.

Introduction

Software Quality Factors define characteristics that determine software quality.

McCall’s Software Quality Factors

Product Operation Factors

1. Correctness
2. Reliability
3. Efficiency
4. Integrity
5. Usability

Product Revision Factors

1. Maintainability
2. Flexibility
3. Testability

Product Transition Factors

1. Portability
2. Reusability
3. Interoperability

Advantages

1. Improves software quality.
2. Better maintainability.
3. Increased reliability.
4. Improved usability.

Conclusion

Software Quality Factors help developers evaluate and improve software quality.

Q9. Explain Reverse Engineering and Reengineering.

Reverse Engineering

Reverse Engineering analyzes existing software to understand its design and functionality.

Objectives

1. Understand old software.
2. Recover design information.

3. Improve documentation.

Reengineering

Software Reengineering modifies existing software to improve quality and maintainability.

Activities in Reengineering

1. Inventory Analysis
2. Document Restructuring
3. Code Restructuring
4. Data Restructuring
5. Forward Engineering

Difference Between Reverse Engineering and Reengineering

Reverse Engineering	Reengineering
Understands software	Improves software
Analysis process	Modification process
No major changes	Major improvements

Advantages

1. Improved maintainability.
2. Reduced redevelopment cost.
3. Better software understanding.

Conclusion

Reverse Engineering and Reengineering help organizations improve legacy software systems.

END OF MODULE 4