Software Quality Assurance - UNIT 4 & UNIT 5 Notes

UNIT 4: SOFTWARE QUALITY MANAGEMENT & METRICS

1. Project Process Control

Introduction:

Project process control ensures project activities stay on track with timelines, resources, risk management, and quality goals. It enables early detection of issues and timely corrective actions.

Detailed Explanation:

Project control activities include progress control, risk management control, schedule control, resource control, and internal/external participant control. Techniques like milestone tracking, reports, periodic assessments, and resource monitoring help maintain project stability.

Conclusion:

Effective process control ensures projects meet goals within time, cost, and quality constraints.

2. Computerized Tools for Quality Management

Introduction:

Computerized tools streamline and automate quality assurance activities across project management, testing, version control, CI/CD, and documentation.

Detailed Explanation:

Examples include JIRA, Trello, Redmine (tracking tools), SonarQube (code analysis), Git (version control), Jenkins (automation), and JMeter (performance testing). Tools improve efficiency, transparency, and error reduction.

Conclusion:

Choosing the right tools enhances quality management efficiency and accuracy.

3. Software Quality Metrics

Software Quality Assurance - UNIT 4 & UNIT 5 Notes

Introduction:

Metrics objectively measure software processes and product quality to guide improvements.

Detailed Explanation:

Metrics are classified into process metrics (development efficiency) and product metrics (product quality). Examples: defect density, test coverage, review efficiency. Metrics support quality control, project management, and risk mitigation.

Conclusion:

Effective metric use leads to continuous quality improvement.

4. Cost of Software Quality (CoSQ)

Introduction:

CoSQ measures the total cost associated with ensuring software quality.

Detailed Explanation:

It includes Prevention Costs (training), Appraisal Costs (testing), Internal Failure Costs (rework), and External Failure Costs (customer complaints). Investing more in prevention and appraisal reduces failure costs.

Conclusion:

Tracking CoSQ helps balance quality improvement investments and cost efficiency.

5. Classical Quality Cost Model

Introduction:

The classical model divides quality costs into four categories for analysis.

Detailed Explanation:

Prevention, Appraisal, Internal Failure, and External Failure costs are tracked. Emphasizes investing in prevention/appraisal to avoid high failure costs.

Conclusion:

Helps organizations optimize resource allocation for quality assurance.

6. Extended Quality Cost Model

Introduction:

Extended model adds Measurement and Improvement Costs for modern quality practices.

Detailed Explanation:

Measurement costs cover data gathering/analysis, and improvement costs cover process enhancements. Focus on proactive quality culture beyond defect fixing.

Conclusion:

Supports continuous process improvement and future-proofing software quality.

7. Application of Cost Models

Introduction:

Applies cost models practically for budgeting and improvement.

Detailed Explanation:

Involves planning, recording quality-related costs, analyzing patterns, and driving improvements.

Conclusion:

Application ensures better quality management, reduced defects, and efficient resource use.

UNIT 5: STANDARDS, CERTIFICATIONS & ASSESSMENTS

Software Quality Assurance - UNIT 4 & UNIT 5 Notes

1. Quality Management Standards

Introduction:

Quality standards like ISO 9001 ensure products and services consistently meet customer requirements.

Detailed Explanation:

Principles include customer focus, leadership, engagement, process approach, and continual improvement. Software standards include ISO/IEC 12207, ISO/IEC 15504, and IEEE 1012/1028.

Conclusion:

Standards enable global recognition, efficiency, and quality assurance.

2. Capability Maturity Models (CMM and CMMI)

Introduction:

CMM/CMMI guide process improvement in software development.

Detailed Explanation:

CMM defines 5 maturity levels; CMMI integrates multiple CMMs into one model with continuous/staged representations, applied across software, systems, and services.

Conclusion:

Following CMMI ensures disciplined, predictable, and improving processes.

3. Evolution of CMM

Introduction:

CMM evolved to address broader quality needs.

Detailed Explanation:

Variants like SE-CMM, T-CMM, SSE-CMM, and P-CMM emerged. Later unified as CMMI for better integration and application across industries.

Conclusion:

Evolution broadened quality focus and integration possibilities.

4. CMMI Structure and Process Areas

Introduction:

CMMI structure provides organized process improvement steps.

Detailed Explanation:

Staged and continuous representations; key process areas like Project Planning, Risk Management, Quantitative Management, and Innovation Deployment.

Conclusion:

Following CMMI structure ensures systematic quality growth.

5. Bootstrap Methodology

Introduction:

Bootstrap is a lightweight European quality improvement framework.

Detailed Explanation:

Combines ISO and CMM concepts, uses graded assessments, and focuses on gradual improvements, especially for SMEs.

Conclusion:

Enables small organizations to efficiently improve processes.

6. SPICE Project (ISO/IEC 15504)

Introduction:

SPICE standardizes software process assessments globally.

Detailed Explanation:

Defines Process Reference Model, Assessment Model, and Measurement Framework (Level 0-5). Encourages systematic evaluation and improvement.

Conclusion:

SPICE ensures global process quality comparability.

7. Software Project Process Standards (IEEE 1012 & IEEE 1028)

Introduction:

Process standards regulate verification, validation, and review activities.

Detailed Explanation:

IEEE 1012 focuses on V&V, ensuring products meet requirements; IEEE 1028 defines review types (inspections, walkthroughs, audits).

Conclusion:

Structured reviews and V&V improve quality and reliability.

8. Department Management Responsibilities

Introduction:

Department managers operationalize SQA principles.

Detailed Explanation:

Plan SQA activities, monitor compliance, manage project risks, allocate resources, and liaise with top management. Ensure departmental quality alignment with organizational goals.

Conclusion:

Their role ensures consistent process execution and product quality.

9. Project Management Responsibilities

Introduction:

Project managers drive quality within projects.

Detailed Explanation:

Prepare project/quality plans, oversee reviews/testing, manage staffing/training, handle customer communication, and intervene in schedule/resource deviations.

Conclusion:

Project managers are key to balancing quality, cost, and timelines effectively.