Course: Software Project Management

Week 10: Software Quality Management (SQM)

Lecturer: Yimer Amedie (MSc.)

Addis Ababa Science and Technology University, Ethiopia

Contents

- Introduction
- Quality assurance and standards
- Quality Philosophies and Principles
- Software Quality Plan
- Quality Standards and Metrics
- Verification and Validation

Learning Outcomes

After completing this lesson, you will be able to:

- Explain the importance of quality assurance and standards
- Compare and contrast major quality philosophies.
- Develop a comprehensive Software Quality Plan (SQP).
- Interpret and apply relevant software quality standards and metrics.
- Differentiate between verification and validation techniques.

Introduction

- In Software Project Management,
  - Managing **scope**, **time**, **cost**, and **quality** is essential to delivering successful software projects.
  - Among these elements, **quality** stands out as a critical constraint, often fixed and non-negotiable.
  - Ensuring software quality is not just about meeting requirements, it directly impacts
    - **Customer satisfaction, system reliability, and**
    - **long-term project success**.

Introduction

- **Software Quality Management (SQM)** is therefore a core component of project management.
  - It involves systematically defining, planning, monitoring, and controlling quality throughout the project lifecycle.
  - By doing so, teams can ensure that the final product
    - **Meets both functional and non-functional requirements**
    - **adheres to standards, and**
    - **fulfills stakeholder expectations.**

What is Software Quality?

- Software quality refers to
  - The degree to which software meets specified requirements and customer expectations.
- Therefore, it refers to **how well** a software product
  - **Meets requirements**
  - **Satisfies user** and
  - **Is free from defects**.

What is Software Quality?

Meets Requirements

The software adheres to the defined specifications. This ensures that the software functions as intended and expected.

Satisfies User

The software effectively fulfills the user's needs. It provides a positive and useful experience for the intended audience.

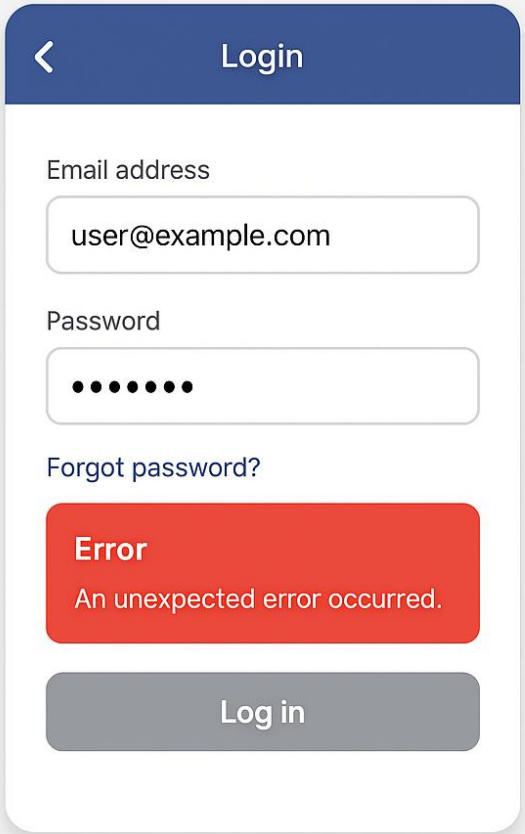
Free from Defects

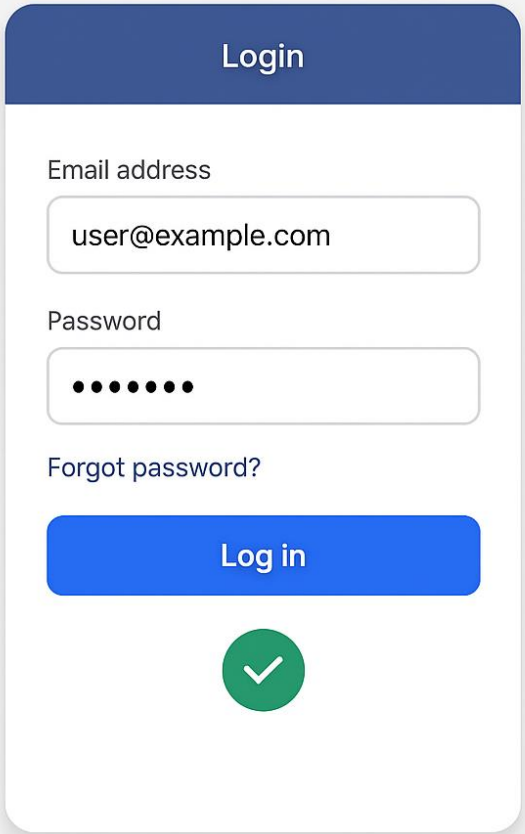
The software operates without errors or malfunctions. This ensures reliability and correctness in its performance.

What is Software Quality?

- **Characteristics/principles** of high-quality software:
  - **Functional correctness, Reliability, Usability, Maintainability, Performance, Security**
- **Dimensions:**
  - **Product quality**
    - Functionality, usability, efficiency.
  - **Process quality**
    - Development methods and practices

What is Software Quality?

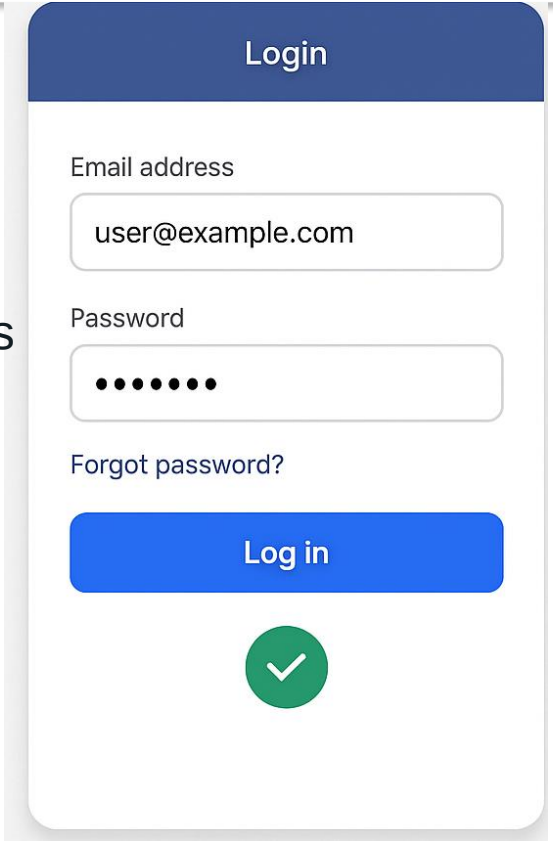




Why quality matters?

What is Software Quality?

- Why Software quality is Important?
  - **High-quality software**
    - Increases customer satisfaction
    - Reduces maintenance costs, minimize risks
    - Enhances brand reputation.
  - **Poor quality can lead to**
    - Financial loss
    - System failures, and
    - Legal issues.



Overview on SQM

- SQM is a framework of activities designed to ensure that software products meet quality standards.
- In project management, SQM is part of risk mitigation and scope control.
- It involves
  - Planning, assurance, and
  - Control processes to monitor and improve software quality.

Overview on SQM

- Components of SQM (PMI, 2013)(Bob Hughes, 2011) (Wiley, 2010)
  - **Quality Planning**: Identifying relevant quality standards.
  - **Quality Assurance (QA):** Implementing processes to meet those standards.
  - **Quality Control (QC):** Evaluating the actual product against standards.

Overview on SQM

- **Quality Planning**:
  - the process of identifying which quality standards are relevant and determining how to meet them.
  - It involves
    - setting quality objectives,
    - defining metrics, and
    - assigning responsibilities.

Overview on SQM

- **Quality Assurance (QA)**:
  - Focuses on improving development processes to prevent defects.
  - It is a proactive approach involving
    - Process audits
    - Training, and
    - Continuous improvement.
  - Supports project delivery timelines and stakeholder confidence.

Overview on SQM

- **Quality Assurance (QA)**: **Activities**
  - Defining coding standards or establish standard procedure.
  - Conducting process audits.
    - Reviewing compliance with defined processes.
  - Training development teams.
  - Reviewing requirements and design documents
    - Collaboratively identifying defects early.
  - Tied to cost and schedule management in projects.

Overview on SQM

- **Quality Assurance (QA)**:
  - Prevention Over Detection
  - Process-Oriented Approach
  - Continuous improvement
  - Product and Process Quality
  - Objective evaluation
  - Standards and compliance
  - Stakeholder involvement
  - Early and frequent testing

Overview on SQM

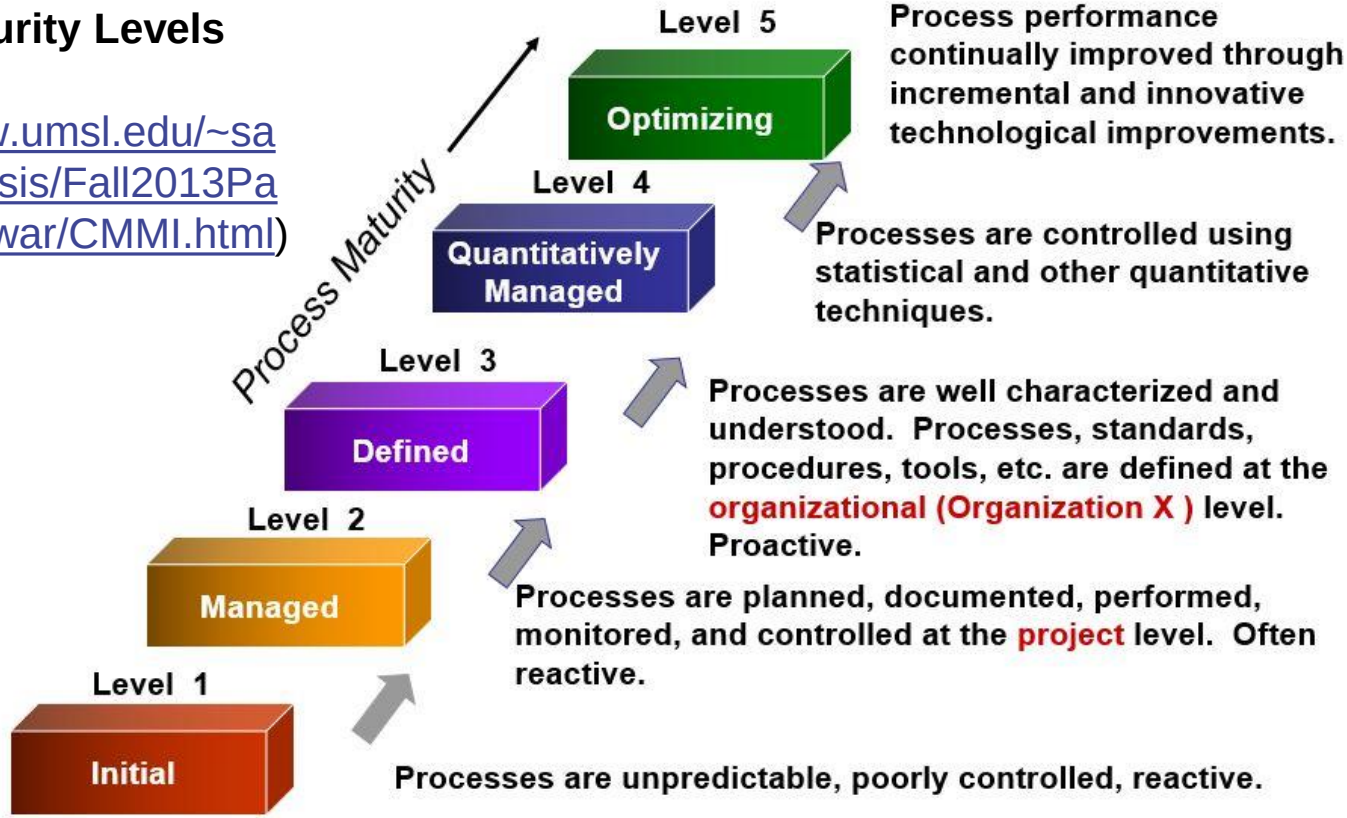
- **Quality Assurance (QA)**: **Standards**
  - **ISO 9001**
    - A general quality management standard applicable across industries.
    - Focuses on **customer satisfaction, process improvement**, and **documentation**.
    - Useful for large software projects with contractual quality clauses.
  - **IEEE 730: Software quality Assurance plans (SQAP)**

Overview on SQM

- **Quality Assurance (QA)**: **Standards**
  - **Capability Maturity Model Integration (CMMI)**
    - Defines process improvement levels.
    - **5 levels of maturity:**
      - **Initial, Managed, Defined, Quantitatively Managed, Optimizing**.
    - Higher maturity levels help manage risks and improve project performance.

Overview on SQM

CMMI Maturity Levels
(Source: https://www.umsl.edu/~sauterv/analysis/Fall2013Papers/Lanjewar/CMMI.html)



Overview on SQM

- **Quality Assurance (QA)**: **Roles in SDLC**
  - QA is integrated into each phase of the Software Development Life Cycle.
  - From **requirements** to **maintenance**
    - QA ensures standards and
    - Best practices are followed.
  - SDLC alignment is a key responsibility of project managers.

Overview on SQM

- **Quality Control (QC)**:
  - It involves operational techniques and activities to fulfill quality requirements.
    - Ensures the final product meets requirements before release
  - It includes activities like
    - Testing (Unit, integration, system, regression, UAT)
    - Code inspections, and
    - Reviews to find and fix defects.
  - QC results influence project acceptance and closure.

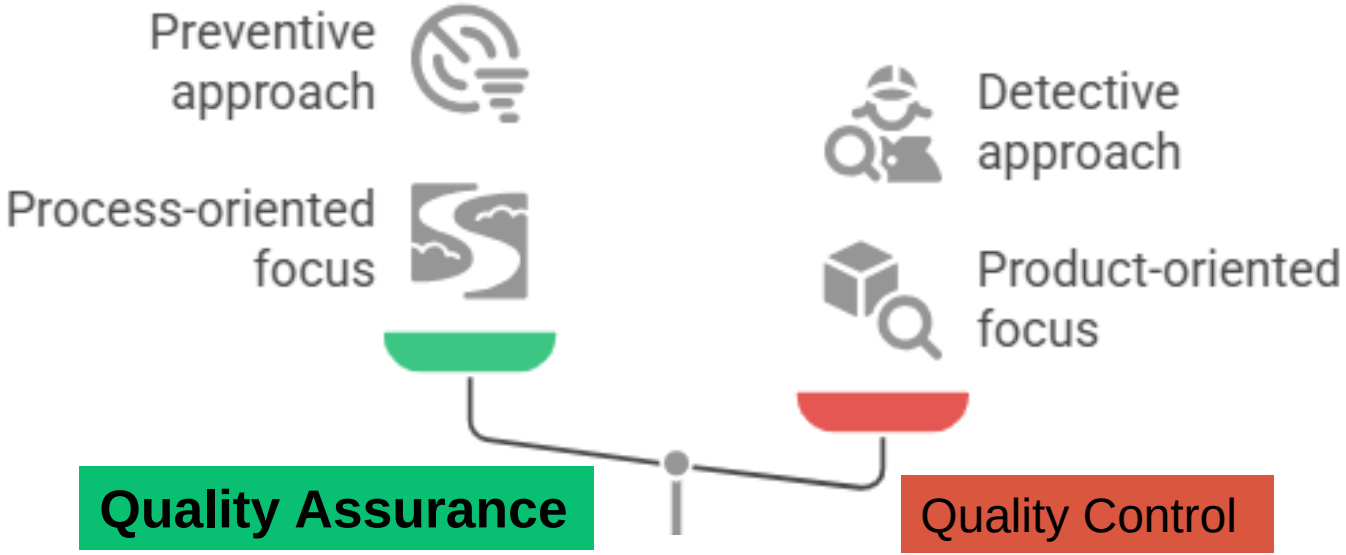
Overview on SQM - Processes

Planning
Establishing quality objectives and strategies

Control
Monitoring and adjusting processes to maintain quality

Assurance
Implementing processes to meet quality standards

Overview on SQM – QA Vs QC



Balancing the QA and QC is necessary for project success.

Quality Philosophies and Principles

- Quality is a fundamental philosophy that shapes the entire project lifecycle
- Software quality is not just about only testing
  - It’s a mindset embedded in development culture.
- Quality philosophies encompass a range of principles and practices aimed at ensuring that
  - Products meet or exceed customer expectations.

Quality Philosophies and Principles

- Software quality philosophies are guiding principles and frameworks that help organizations produce high-quality software.
- These philosophies influence how teams
  - Plan, develop, test, and maintain software systems.
- They originate from both software engineering and general quality management
  - including manufacturing principles adapted for software.

Quality Philosophies and Principles

- Different philosophies guide how teams approach quality, from **prevention** to **continuous improvement**.
  - **Zero Defects Philosophy (Philip Crosby)**
  - **Continuous Improvement (Kaizen & Lean)**
  - **Total Quality Management (TQM)**
  - **Prevention Over Inspection (Deming’s Influence)**
  - **Fitness for Use (Joseph Juran)**
  - **Agile, quality by design**

Quality Philosophies and Principles

Zero Defects Philosophy

- Do it right the first time, aim for zero defects.

Quality Philosophies and Principles

Zero Defects Philosophy

- **Principles:**
  - Quality is defined by requirements.
  - Prevention is better than inspection.
  - Aim for zero defects, not an acceptable number.
  - Quality is a management responsibility.
- **Application in Software:**
  - Emphasizes defect prevention through requirements clarity, proper planning, and early validation.

Quality Philosophies and Principles

- **Application of Quality Philosophies in Software Projects**

Software Quality Standards

- Standards provide a framework for quality and consistency.
- They help organizations benchmark and improve software processes.
- Ensuring compliance with standards is a part of project scope management.
- The standards for software quality:
  - **IEEE 829**
  - **ISO/IEC 25010**

Software Quality Standards

- **IEEE 829**
  - IEEE Provides detailed software engineering standards.
  - **IEEE 829: Test Documentation**
  - Enhances professionalism.
- **ISO/IEC 25010**
  - A software-specific standard defining quality models.
  - Important for defining quality criteria in project scope documents.

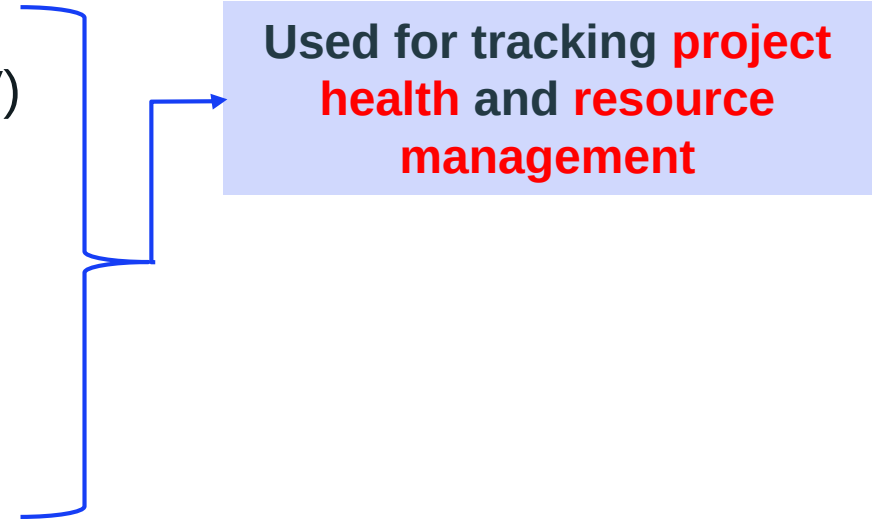
Software Quality Standards

Software Quality Metrics

- Software metrics are quantitative measures used to assess the quality, performance, and progress of software development.
  - Should be reliable, relevant, and easy to interpret.
  - Project managers use them to track quality performance.
  - It can be categorized as
    - **Process Metrics (Development Efficiency)**
    - **Product Metrics (Code & Quality)**
    - **Maintenance Metrics (Post-Release)**
    - **Project Metrics (Management & Planning)**

Software Quality Metrics

- **Project Metrics**
  - These help track budget, schedule, team performance, and risk management.
    - Schedule variance (SV)
    - Cost Variance (CV)
    - Bug resolution time
    - Team velocity (Agile)
    - Resource utilization



Software Quality Metrics

- **Project Metrics**

Metric	Description	Ideal Value
Schedule Variance (SV)	(Earned Value – Planned Value).	SV ≥ 0 (on/before schedule).
Cost Variance (CV)	(Earned Value – Actual Cost).	CV ≥ 0 (under/on budget).
Bug Resolution Time	Average time to fix a reported bug.	Shorter is better (hours/days).
Team Satisfaction	Survey-based metric on developer happiness.	High scores = better retention.

Software Quality Metrics

- **Effective use of Metrics**
  - Track progress and identify areas for improvement.
  - Avoid “metrics for the sake of metrics.”
  - Support earned value analysis and quality KPIs.

Strategic Metrics

Align with goals for valuable insights

Misleading Metrics

Lack clear objectives and context

Software Verification & Validation

Delivering high-quality software that satisfies both the ***stakeholders' requirements*** and the ***end users' expectations.***

(PMI, 2013)((Bob Hughes, 2011))

(Wiley, 2010)

Software Verification

- **Verification** **Are we building the product right?**
  - Ensures the product is built correctly according to specifications.
  - Early-stage activity in development.
  - Project phase-level quality checks.
- **Techniques**:
  - Walkthroughs
  - Peer reviews
  - Formal inspections

  Static
  - Fit well in milestone-based reviews in project schedules.

Software Validation

- **Validation** **Are we building the right product?**
  - Ensures the product meets user expectations and requirements.
  - Often involves actual testing against business needs.
  - Validates deliverables during project acceptance.
- **Techniques**:
  - System testing
  - Integration testing
  - End-to-end testing

  Dynamic
  - Managed as key activities in quality assurance project plan.

Software Validation

- **Validation: UAT (User Acceptance Testing)**
  - UAT is performed by end-users to validate the system.
  - It confirms that business requirements are met.
  - Final step before project delivery and closure.
  - Process of UAT
    - Identify test scenarios and users.
    - Develop test scripts.
    - Execute and log results.
    - Report issues and retest.

Software Validation

- **Validation: UAT (User Acceptance Testing)**
  - UAT sign-off is a formal part of project completion.
  - UAT often reviewed by project sponsors and clients.
  - Types:
    - Alpha Testing: Internal staff.
    - Beta Testing: External user feedback.
    - Contract Acceptance: Per contract terms.
    - Regulatory Testing: Compliance focus.

Software V&V

Verification, Validation & UAT

Type	Focus	Performed By	Purpose
Verification	Specifications	Developers/ QA Team	Ensure the product is built correctly (according to design/specs)
Validation	Requirements	Testers	Ensure the product meets user needs and expectations
UAT	Business Use Cases	End-users / Clients	Confirm the product is ready for release and acceptable for business use

Software Quality Plan

- A document that shapes how an organization will ensure software quality.
- It outlines how quality will be
  - Defined
  - Measured
  - Assured, and
  - Controlled throughout a software project's life cycle
- Tailored to project needs, stakeholders, and risk levels.

Should be developed during project planning phase.

Software Quality Plan

Key contents, SQMP ➔PMI/PMBoK

1. Introduction
2. Quality management approach
3. Quality requirements & standards
4. Quality objectives
5. QA activities
6. QC activities
7. Metrics and measurements
8. Verification and Validation plan
9. Tools, techniques & resources
10. Change control and defect management
11. Risk Management related to quality
12. Reporting and communication
13. Continuous improvement process

Software Quality Plan

Key contents, SQAP ➔ IEEE 730

1. Purpose
2. Reference documents
3. Management (organization, task & Responsibility)
4. Documentation
5. Standards, practices, conventions, & metrics
6. Reviews and audits
7. Test
8. Problem reporting and corrective action
9. Tools, techniques & methodologies
10. Code control
11. Media control
12. Supplier control
13. Records collection, maintenance, and retention
14. Training
15. Risk management
16. SQA plan approval

Summary

- Software Quality refers to:
  - How well the software meets
    - requirements, user needs, and expectations.
  - How the software is reliable, maintainable, secure, efficient, and usable.
- Quality refers to both aspects
  - **Functional:**
    - Does it do what it should?
  - **Non functional:**
    - Is it fast, secure, easy to use?

Summary

- The components of SQM are:
  - **Quality Planning:** setting quality goals and processes.
  - **Quality Assurance (QA):** ensuring processes prevent defects.
  - **Quality Control (QC):** identifying defects (mainly through testing).
  - **Process Improvement:** refining how software is built and maintained.
- QA sets up the framework to make sure quality is built into the product from the beginning,
- While QC checks if the final product meets the specified requirements.
- QA helps avoid the need for excessive QC.

References

1. *PMI. (2013). A Guide to the Project Management Body of Knowledge (5th ed.). PMI, Inc.*
2. *Wiley. (2010). Quality Software Project Management. Futrell and Shafer.*
3. *Bob Hughes, M. C. (2011). Software Project Management (5th ed.). McGraw-Hill.*