

Course: Software Requirements Engineering

Week 14: Course Review

Lecturer: Yimer Amedie (MSc.)

Addis Ababa Science and Technology University

June, 2026

Contents

- Introduction
- Overview of Software Requirements Engineering concepts
- Requirements elicitation, analysis, and specification techniques
- Requirement modeling, validation, and verification methods
- Requirement management, traceability, and change control
- Agile requirements engineering and supporting tools

Learning Outcomes

After completing this lesson, you will be able to:

- Explain core concepts and importance of requirements engineering
- Apply techniques for eliciting and documenting requirements
- Analyze, model, and validate software requirements effectively
- Manage requirement changes and maintain traceability
- Evaluate agile and traditional approaches in requirements engineering

Introduction



Having a **problem**, we need **requirements** for a system that **solves** the **problem!!**

Figure 1. *The problem, requirements and solution concepts*
Note. Image generated using Sora by OpenAI (2026).

What is SRE?



Defines system needs and expectations



Foundation of successful software projects



Bridges business and technical teams



Reduces misunderstanding and rework



Supports project success and quality



Figure 2. *The problem, requirements and solution concepts*

Note. Image generated using ChatGPT by OpenAI (2026).

Importance of RE



Minimizes project failure risks



Reduces costly development errors



Improves stakeholder communication

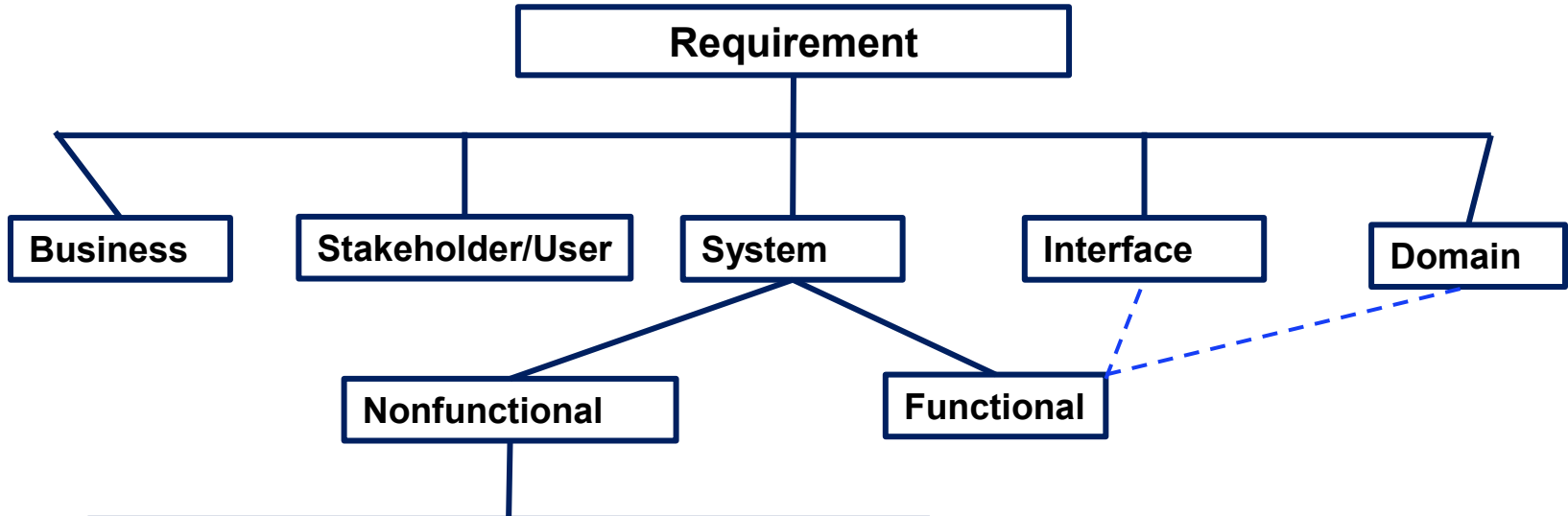


Enhances software quality assurance



Supports customer satisfaction

Types of Requirements



- ✓ Quality Attributes (-ilities)
- ✓ Constraints / Implementation
- ✓ Compliance / Business Rules
- ✓ Other / Cross-Cutting Requirements

Stakeholders in RE



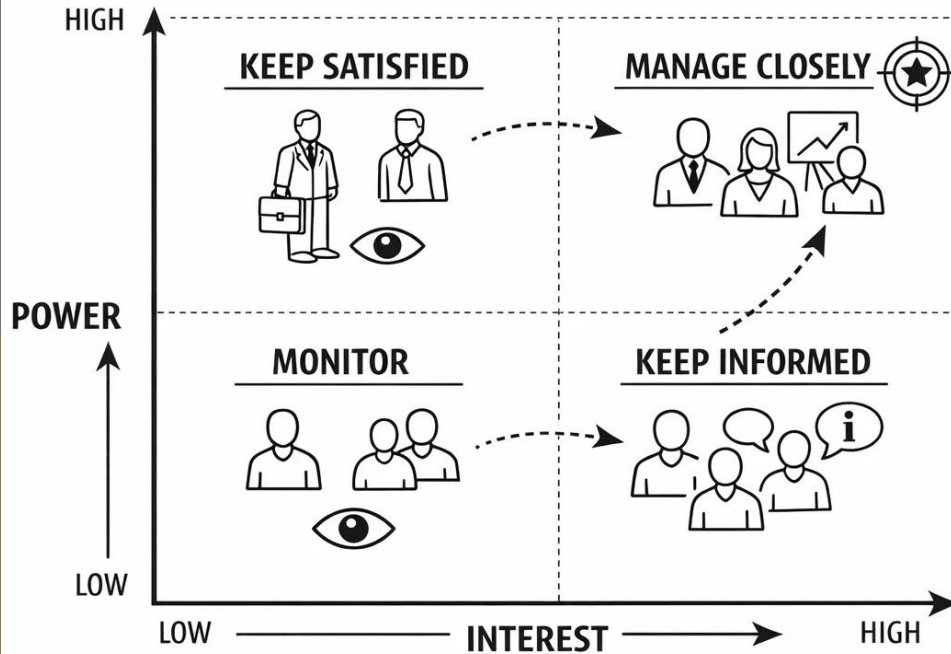
Figure 3. Stakeholders in requirement engineering

Note. Image generated using ChatGPT by OpenAI (2026).

Stakeholders – Types

- ✓ **Primary:** Directly use the system.
Examples: End users, Customers, Operators
- ✓ **Secondary:** They support or maintain the system but may not use it daily.
Examples: System administrators, Maintenance teams, IT support staff
- ✓ **External:** People or organizations outside the company but affected by the system.
Example: Government regulators, Suppliers, Business partners

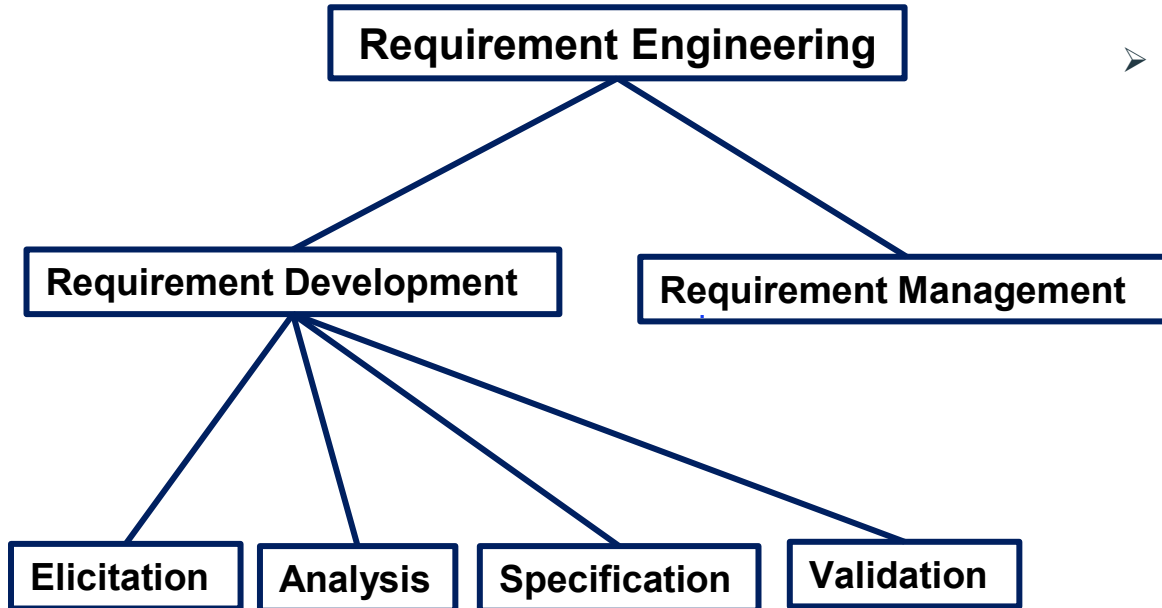
Stakeholders – Power-Interest Grid



The **Power-Interest Grid** is a stakeholder analysis **tool** used to

- ✓ **classify** stakeholders based on their **level of power and interest** in a project,
- ✓ helping project managers decide how to **manage** and **communicate** with them.

Components of RE



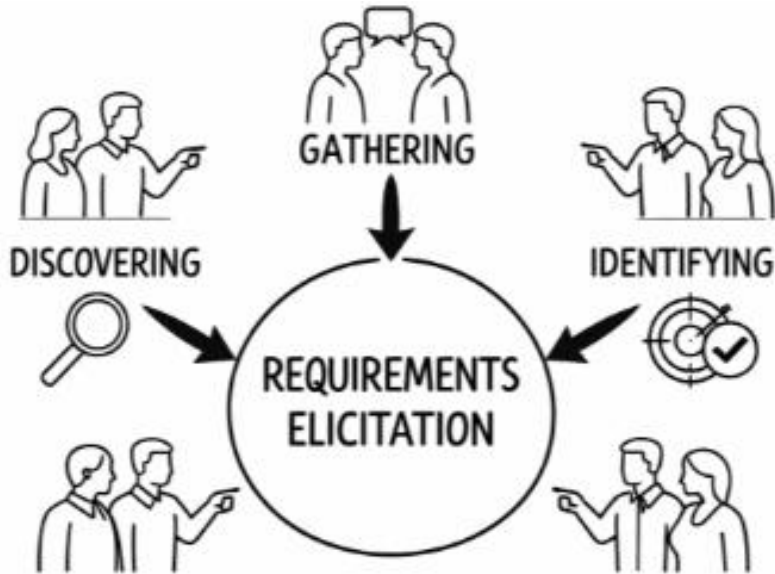
- Whether the approach is:
 - ✓ Pure waterfall
 - ✓ Phased
 - ✓ Iterative/Incremental
 - ✓ Agile or some hybrid

Requirements Lifecycle



**Requirements Engineering:
Process**

Overview of Elicitation



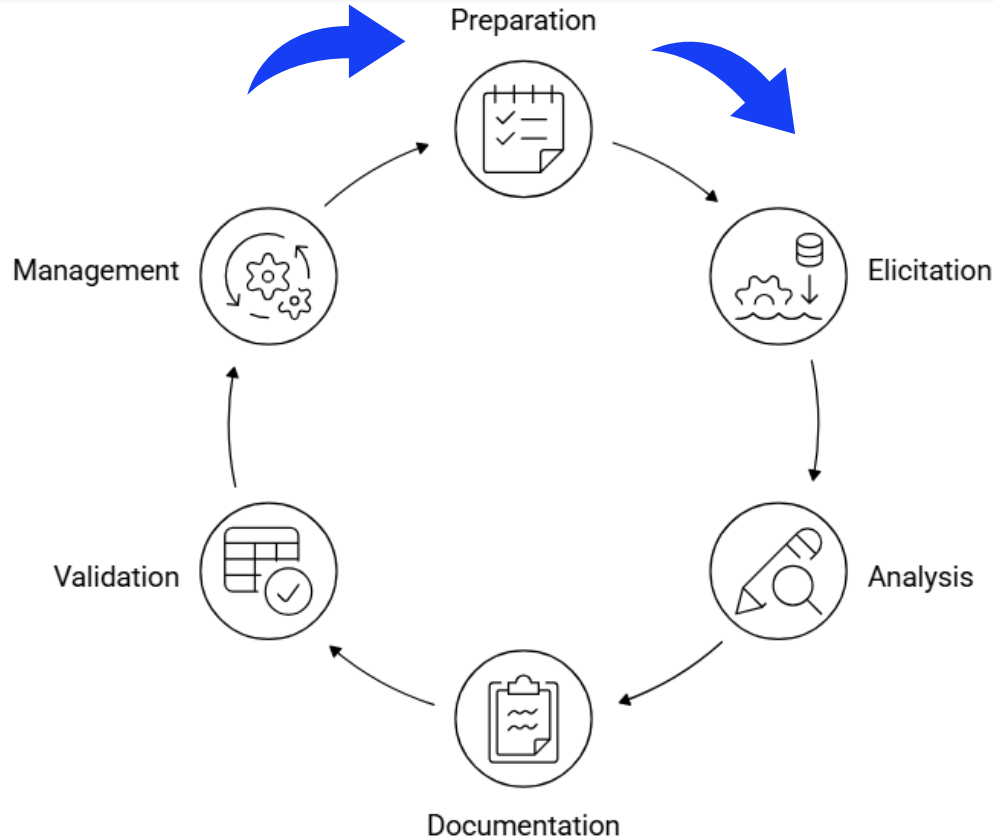
- ✓ The process of **discovering**, **gathering**, and **identifying** the needs and expectations of stakeholders for a software system (Beatty, 2013) (Chemuturi, 2013)

Elicitation transforms *ideas and expectations* into **clear, usable requirements**

Figure 4. *Concept of requirement elicitation*

Note. Image generated using Sora by OpenAI (2026).

Stages of Elicitation



Elicitation Techniques



Interviews



Workshops



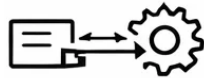
Brainstorming



Observation



**Interface
Analysis**



System



User

Focus Group



Prototyping



**Questionnaires
(Surveys)**



**Document
Analysis**



Classification of Techniques

- ✓ Individual techniques
- ✓ Group techniques
- ✓ Observational techniques
- ✓ Creative techniques
- ✓ Facilitation-based techniques

Facilitated Activities	Independent Activities
Interviews, prototyping	Document analysis
Workshops, focus group	System interface analysis
Observations, brainstorming	User interface analysis
Questionnaires	

Requirements Analysis



Examines requirement consistency



Identifies conflicts and ambiguities



Evaluates feasibility and constraints



Prioritizes critical requirements



Improves requirement quality



Figure 5. Requirements Analysis concepts

Note. Image generated using Sora by OpenAI (2026).

Requirement Prioritization



Identifies high-value requirements



Supports resource allocation decisions



Helps manage limited budgets



Reduces unnecessary development effort



Aligns software with business goals

PRIORITIZED REQUIREMENTS			
Priority	Requirement	Business Value	Effort
High	=====	★★★★★	●
High	=====	★★★★☆	●
Medium	=====	★★★☆☆	●
Low	=====	★★☆☆☆	●
Low	=====	★☆☆☆☆	●



Figure 6. Requirements Prioritization concepts

Note. Image generated using Sora by OpenAI (2026).

Prioritization Techniques



MoSCoW method



Ranking techniques



Scoring models



Value vs effort matrix



Kano model



RICE Method



**Risk-Based
Prioritization**

Requirement Modeling

Turning Requirements into Visual Understanding



01 Represents requirements visually

Transforms textual requirements into visual models like use case diagrams, activity diagrams, class diagrams, etc.



05 Reduces requirement ambiguity

Eliminates vague or incomplete statements by clarifying relationships, constraints, and business rules.



02 Simplifies complex system understanding

Breaks down complex systems into clear structures and relationships that are easier to grasp.



03 Improves stakeholder communication

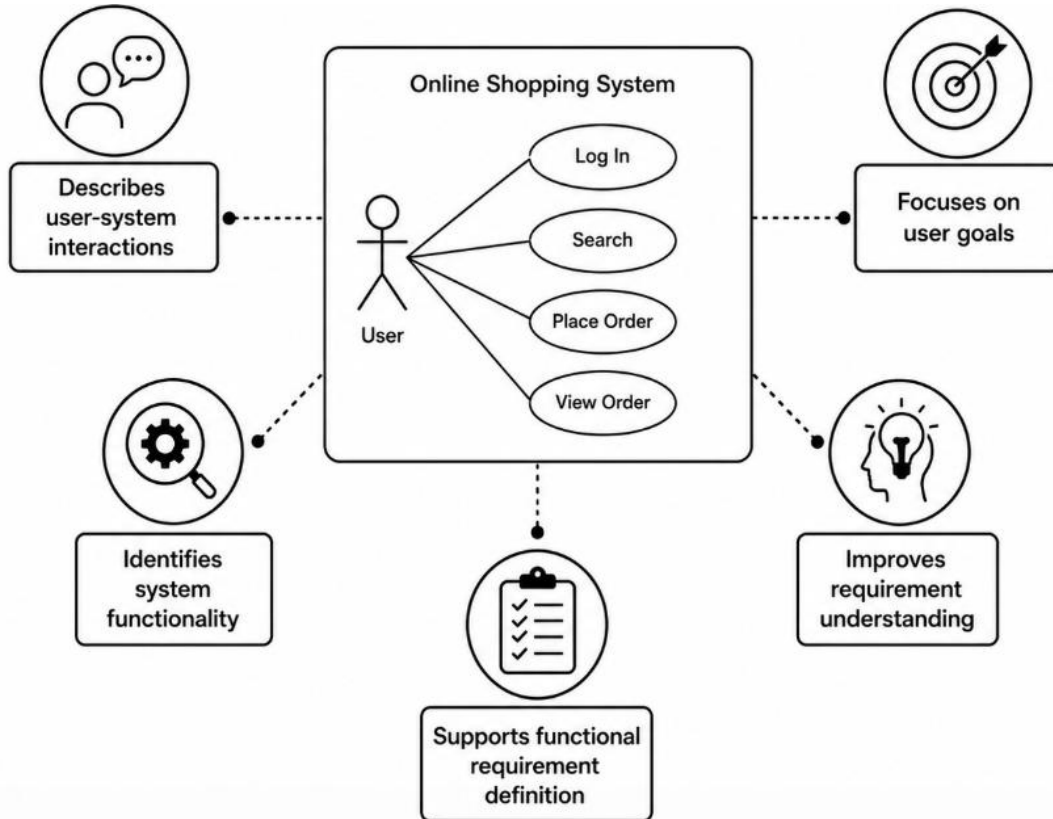
Provides a common visual language that helps all stakeholders align and share the same understanding.



04 Supports system analysis activities

Helps in impact analysis, traceability, validation, and identifying missing or conflicting requirements.

Use Case Modeling



Software Requirement Specification (SRS)

-  Documents system requirements formally
-  Defines project scope clearly
-  Serves as development reference
-  Supports testing and validation
-  Reduces requirement misunderstanding



Figure 7. *What is Software Requirements Specification?*

Note. Image generated using Sora by OpenAI (2026).

Characteristics of Good Requirements

1

Clear and understandable



2

Complete and consistent



3

Testable and measurable



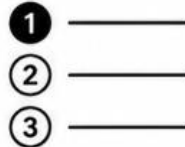
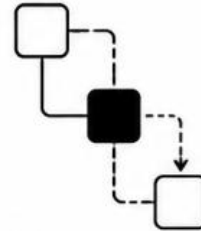
4

Feasible and realistic

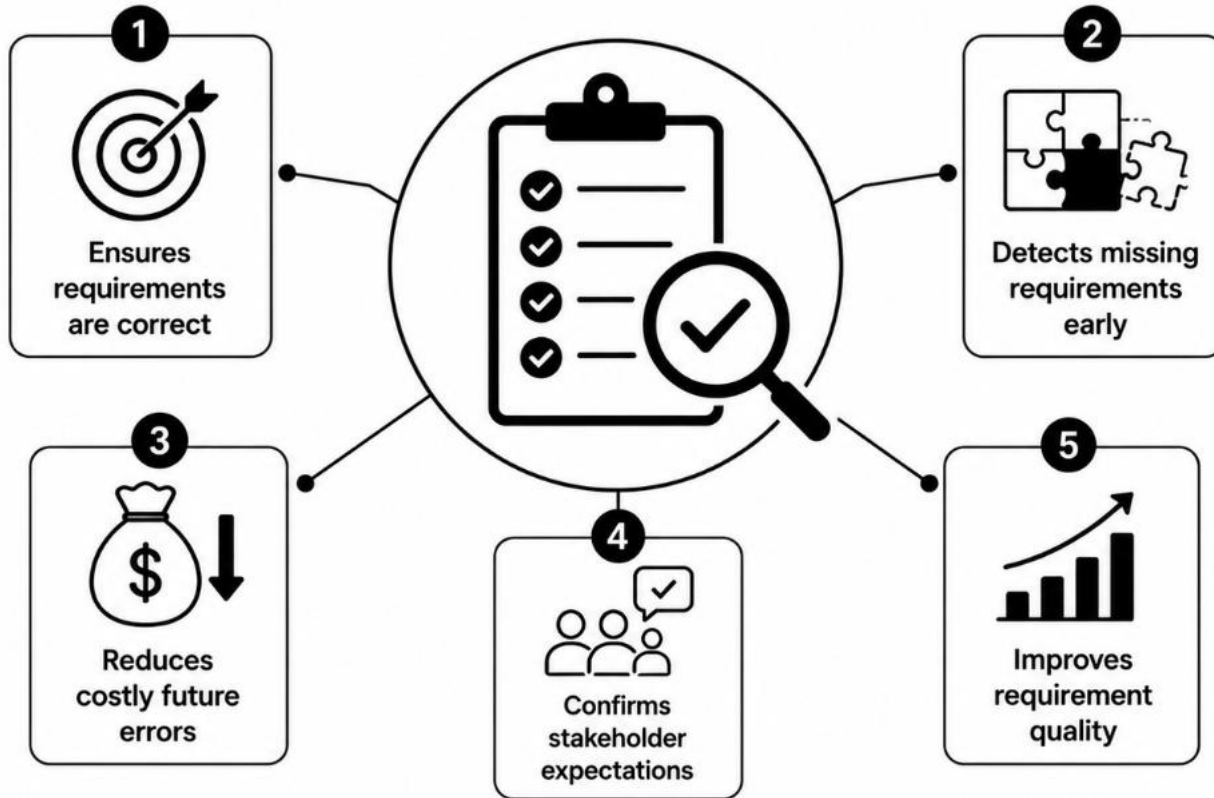


5

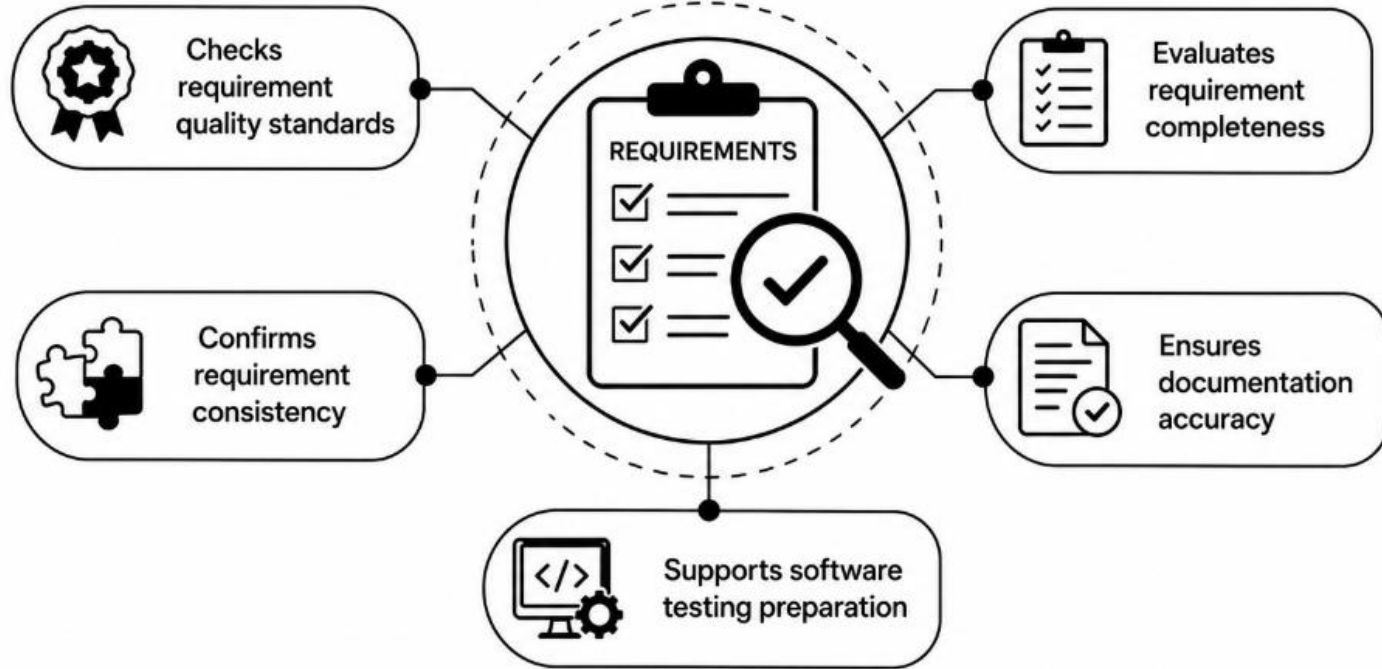
Traceable and prioritized



Requirement Validation



Requirement Verification








Requirement Traceability

Requirement traceability is the ability to track requirements throughout the software lifecycle

- ✓ Links related project artifacts
- ✓ Connects requirements and testing
- ✓ Supports impact analysis
- ✓ Helps manage requirement changes
- ✓ Improves accountability and consistency

Requirement Change Management

-  Handles evolving requirements systematically
-  Evaluates change impact carefully
-  Prevents uncontrolled scope growth
-  Supports stakeholder approval process
-  Maintains project stability



Requirement Change Management Process



01

Change request submission

Stakeholders submit a change request clearly describing the need, reason and impact.



02

Change evaluation and analysis

The change is analyzed for impact on scope, time, cost, quality and risks.



03

Approval or rejection decision

The change control board or authorized personnel approve or reject the request.



04

Implementation of approved changes

Approved changes are implemented as per the agreed plan and necessary updates are made.



05

Documentation and monitoring

All changes are documented and their status is monitored to ensure effectiveness and compliance.

Requirement Documentation Techniques



Organizes requirements systematically



Improves communication clarity



Supports stakeholder understanding



Reduces ambiguity in requirements



Enhances project consistency



Requirement Conflicts and Resolution



Conflicts arise among stakeholders



Different priorities create disagreements



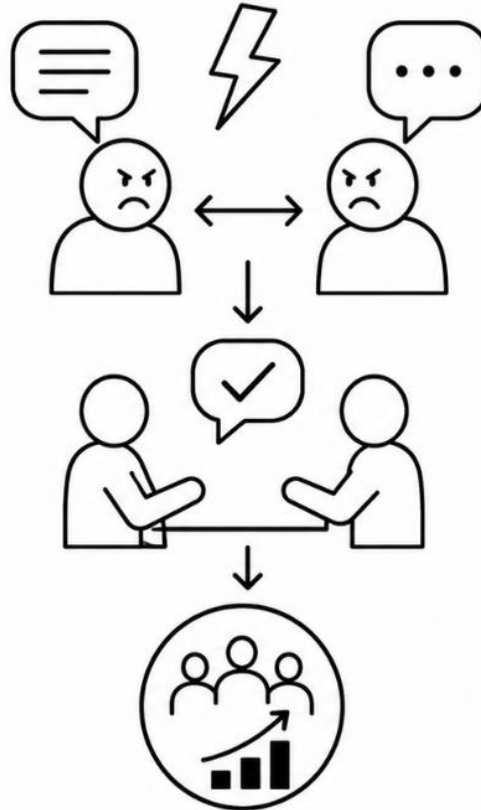
Negotiation helps resolve issues



Trade-offs balance competing needs



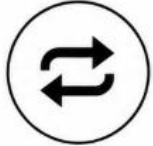
Consensus improves project success



Requirement Risks and Challenges



Incomplete requirements cause problems



Scope changes affect schedules



Communication gaps create misunderstandings



Stakeholder involvement may be limited



Poor requirements increase project risk

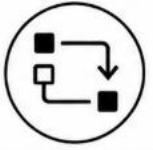
Requirement Management Tools



Support requirement documentation



Enable collaboration among teams



Improve traceability management



Assist requirement version control



Enhance project visibility

Agile Requirements Engineering



Supports iterative requirement development



Encourages continuous stakeholder feedback



Handles changing requirements quickly



Focuses on customer value



Promotes collaboration and flexibility

Summary

This lecture:

- ✓ Reviewed complete course concepts
- ✓ Connected major requirement topics
- ✓ Reinforced practical understanding
- ✓ Prepared for assessment and practice
- ✓ Ready for real-world application

References

1. Beatty, K. W. (2013). Software Requirements (3rd ed.). Washington: Microsoft Press.
2. Chemuturi, M. (2013). Requirements Engineering and Management for Software Development Projects. New York Heidelberg Dordrecht London: Springer. doi:10.1007/978-1-4614-5377-2

Thank You!

