

# **Course: Software Requirements Engineering**

## **Week 11: Requirements Validation**

**Lecturer: Yimer Amedie (MSc.)**

Addis Ababa Science and Technology University

May, 2026

# Contents



- Introduction
- Overview of requirement validation
- Validation vs verification
- Validation techniques and methods
- Common issues in requirements
- Best practices and case examples

**Figure 1.** *The software requirements*

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

# Learning Outcomes

After completing this lesson, you will be able to:

- Define the concept of requirement validation
- Explain the importance of validation in software projects
- Identify common validation techniques
- Analyze and detect requirement issues
- Apply validation practices in real scenarios

# Introduction

## From Requirements Analysis to Validation

- ✓ Requirements engineering lifecycle (Beatty, 2013)
  - ✓ **Steps: Elicitation → Analysis → Specify → Validation → Mgmt.**
  - ✓ Requirements may contain errors and gaps Need to confirm accuracy and completeness
  - ✓ Transition from defining to evaluating requirements
    - ✓ Leads to requirement validation phase

# What is Requirement Validation?



Ensures requirements reflect **user needs**



Focus on **correctness** and **completeness**



Answers:  
“Are we building the right system?”



# Importance of Requirement Validation



Ensures **alignment** with stakeholder needs and enhances **satisfaction**



**Reduces** costly errors, rework, and scope creep



**Improves system quality** through clearer, more complete requirements



Identifies **ambiguities** and **inconsistencies** early



Supports accurate **planning**, **testing**, and **validation**



# Validation Vs Verification

## VALIDATION

**Right system**

Are we building  
the right system?



**Focus:** Meeting user needs  
and solving the right problem

## VERIFICATION

**Built correctly**

Are we building  
the system right?



**Focus:** Conformance to  
requirements and specifications



## COMPLEMENTARY PROCESSES

Validation ensures we build the right system;  
Verification ensures we build the system right.

# Objectives of Validation



## CORRECTNESS

Ensures requirements accurately reflect stakeholder needs.



## COMPLETENESS

Ensures all necessary requirements are included.



## CONSISTENCY

Ensures requirements are uniform, non-conflicting, and coherent.



## FEASIBILITY

Ensures requirements are realistic, achievable, and within constraints.

# Characteristics of Valid Requirements



## CLEAR

Requirements are easy to understand and have a single, well-defined meaning.



## TESTABLE

Requirements can be verified through testing or other objective means.



## TRACEABLE

Requirements can be traced to their sources and related requirements throughout the lifecycle.



## UNAMBIGUOUS

Requirements are specified in a way that allows only one possible interpretation.

# Validation Process



## GOAL

Ensure requirements reflect stakeholder needs and are correct, complete, consistent, feasible, and testable.

# Stakeholder Involvement



Identify all relevant stakeholders



Engage users, clients, and sponsors



Encourage active participation



Capture diverse perspectives



Resolve expectation gaps



Ensure continuous communication

# Validation Techniques



**Figure 2.** *Requirements Validation Techniques*

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

# Requirement Review



**Figure 3.** *Requirements Review Validation Technique*

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

# Inspection

-  Formal review process
-  Defined roles (moderator, reviewer)
-  Systematic examination
-  Detect defects early
-  Documentation-based
-  Structured approach



**Figure 4.** *Inspection Technique*

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

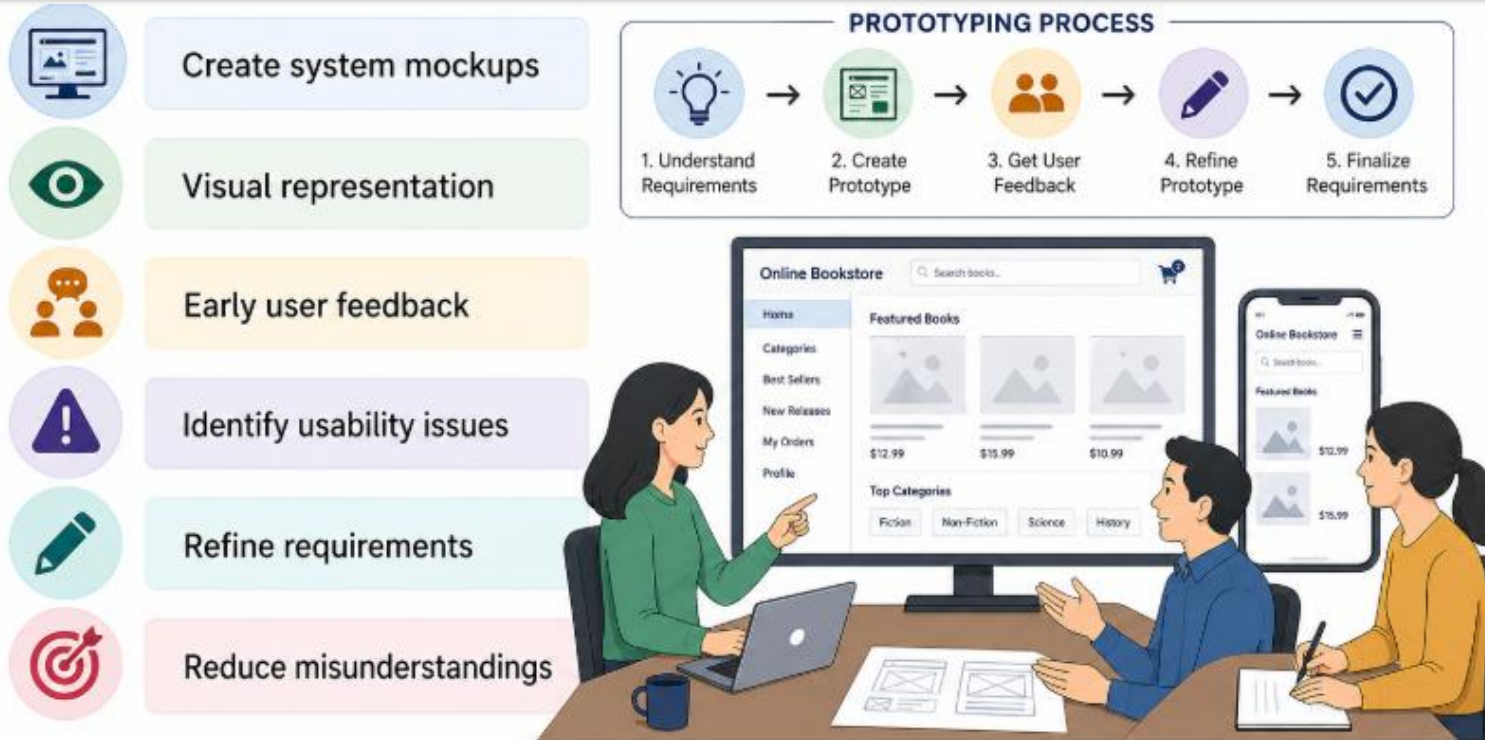
# Walkthrough



**Figure 5.** *Walkthrough Validation Technique*

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

# Prototyping



**Figure 6.** *Prototyping Technique*

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

# Test Case Generation



Derive test cases from requirements



Ensure testability



Identify missing requirements



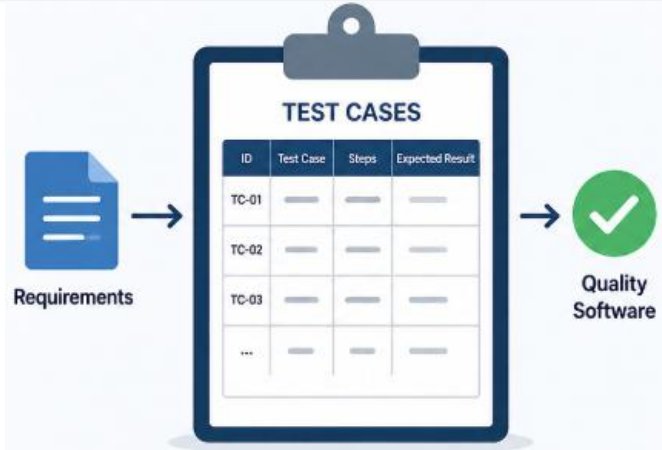
Validate expected outcomes



Improve requirement clarity



Support QA process



# Test Case Generation – The Process

## TEST CASE GENERATION PROCESS



1. Review Requirements



2. Identify Scenarios & Conditions



3. Create Test Cases



4. Review & Refine



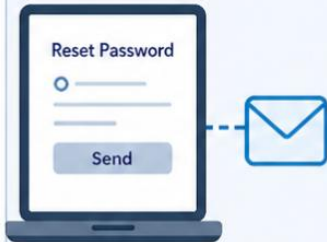
5. Use in QA Process

# Test Case Generation – Example

## EXAMPLE: REQUIREMENT TO TEST CASES

### REQUIREMENT

User shall be able to reset password using registered email.



Test Case ID	Test Scenario	Test Data	Expected Result
TC-001	Reset password with valid email	user@example.com	Password reset link sent to email
TC-002	Reset password with invalid email	invalid@abc.com	Error message: Email not found
TC-003	Reset password with empty email	(empty)	Validation message: Email is required
TC-004	Reset password link expiry	Expired link	Error message: Link has expired

# Simulation and Modeling



Represents system behavior using models



Simulates real-world scenarios before development



Helps visualize complex requirements



Detects logical and functional errors early



Useful for performance and interaction analysis



Supports stakeholder understanding and feedback

# Automated Validation Tools



Requirement management tools (e.g., JIRA, DOORS)



Automated consistency checking



Traceability matrix generation



Version control and change tracking



Error and duplication detection



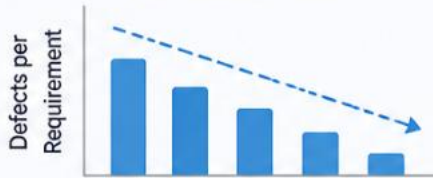
Improve validation efficiency

# Validation Metrics



## Defect density in requirements

Number of defects found per requirement.



## Requirement coverage percentage

Percentage of requirements that are validated.



## Number of issues identified

Total number of issues identified during validation.



## Validation completion rate

Percentage of validation activities completed.



# Validation Metrics ... cont'd



## Review effectiveness rate

Percentage of reviews that identified issues.



## Quality improvement indicators

Measure improvement in requirement quality over time.



These metrics help track validation effectiveness, identify areas for improvement, and ensure high-quality requirements.

# Handling Ambiguity



01

Avoid vague terms (e.g., “fast,” “user-friendly”)



02

Use precise and measurable language



03

Define clear acceptance criteria



04

Engage stakeholders for clarification



05

Use standard templates

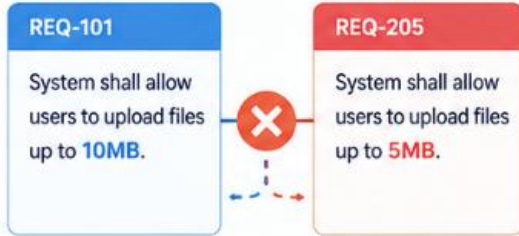



06

Apply examples and scenarios

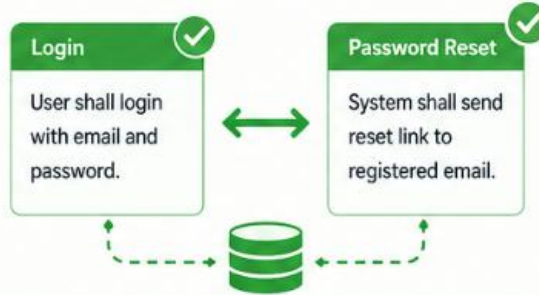
# Detecting Inconsistencies


## 01 Identify conflicting requirements



 Spot contradictions and conflicts in requirements.

## 02 Cross-check related requirements



 Ensure related requirements are consistent and aligned.

## 03 Use validation checklists




 Apply standard checklists to catch inconsistencies early.

# Detecting Inconsistencies ... cont'd


## 04 Conduct review sessions



 Collaborate with stakeholders to resolve discrepancies.


## 05 Apply automated tools



 Leverage tools to automatically detect issues and anomalies.

## 06 Ensure logical alignment



 Confirm requirements support goals and system behavior logically.

**Figure 7.** *Detecting inconsistency during requirements validation*

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

# Missing Requirements



## Identify requirement gaps

Analyze requirements to find missing or incomplete information.



## Use stakeholder interviews

Engage stakeholders to discover hidden needs and expectations.



## Apply scenario-based analysis

Use real-world scenarios to uncover missing requirements.



## Utilize checklists and templates

Leverage proven checklists and templates to ensure comprehensive coverage.



## Review business processes

Examine current processes to identify overlooked requirements.



## Perform iterative validation

Continuously validate and refine requirements to close gaps.



# Conflict Resolution



**Figure 8.** Conflict resolution during requirements validation

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

# Conflict Resolution ... cont'd

## 04 Use trade-off analysis

Option	Cost (\$)	Quality	Features	Time
Option A	Low	High	Low	Fast
Option B	Medium	Medium	Medium	Medium
Option C	High	High	High	Slow



Evaluate options based on various factors and trade-offs.

## 05 Prioritize requirements



Focus on what delivers the most value to achieve project goals.

## 06 Achieve consensus

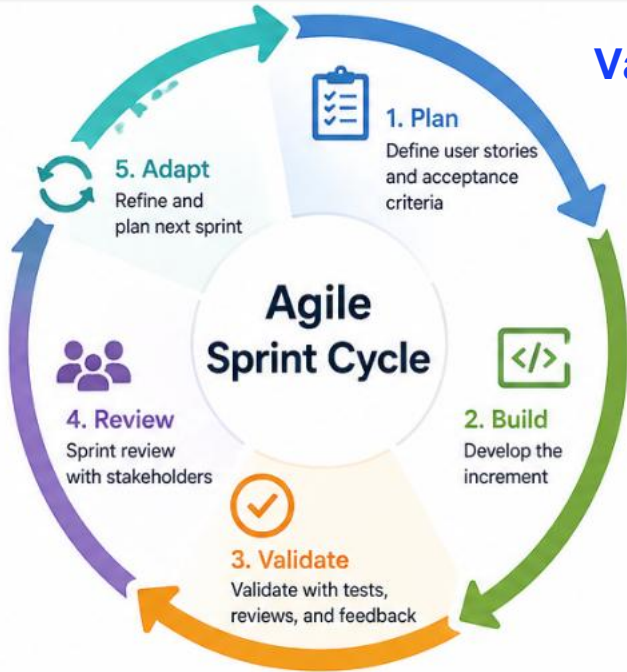


Reach a mutually acceptable solution and move forward together.

**Figure 9.** Conflict resolution during requirements validation  
**Note.** Image generated using ChatGPT by OpenAI (2026).

# Validation in Agile

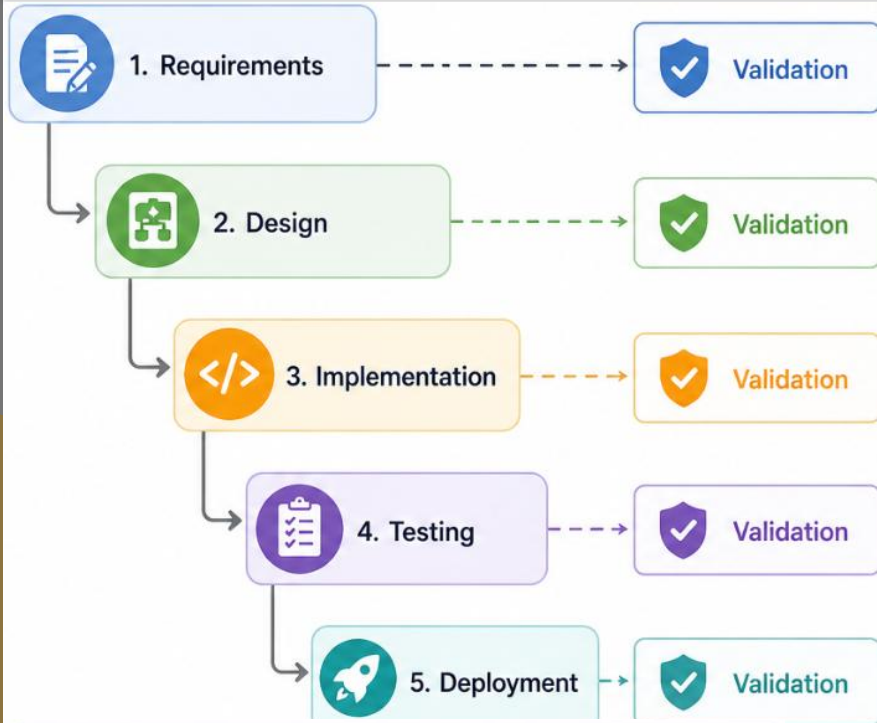
Validating early and often to deliver value with confidence



- ✓ Continuous validation
- ✓ User stories and acceptance criteria
- ✓ Sprint reviews
- ✓ Frequent stakeholder feedback
- ✓ Iterative refinement
- ✓ Rapid adaptation



# Validation in Waterfall



- ✓ Sequential validation phase
- ✓ Formal requirement reviews
- ✓ Detailed documentation
- ✓ Limited stakeholder feedback later
- ✓ Structured validation approach
- ✓ Risk of late error detection



Validation happens at the end of each phase, making changes costly and time-consuming.

# Role of Analysis



**Figure 10.** *Challenges of requirements validation*

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

# Role of Stakeholders



**Figure 11.** Role of stakeholders during requirements validation

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

# Common Challenges

01

## AMBIGUITY

"The system should be fast and user-friendly."



Vague or unclear requirements lead to different interpretations.



### IMPACT

Misunderstandings and incorrect implementations

02

## CONFLICTS

"The system must be secure."

"The system must allow full access for users."



Conflicting requirements create confusion and difficult trade-offs.



### IMPACT

Rework, delays, and unsatisfied stakeholders

**Figure 12.** Challenges of requirements validation

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

# Common Challenges ... cont'd

03

## CHANGING REQUIREMENTS



### IMPACT

Scope creep, rework, and schedule slippage

04

## LIMITED INVOLVEMENT



### IMPACT

Incomplete requirements and poor user satisfaction

05

## TIME CONSTRAINTS



### IMPACT

Defects, rework, and delivery risks

**Figure 13.** Challenges of requirements validation

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



Addressing these challenges early leads to better requirements, smoother development, and successful project outcomes.



# Best Practices



**Figure 14.** *Best Practices during requirements validation*

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

# Example – Online Learning System

## 1 Sample Requirement

“Students should easily access course materials”



## 2 Validation Issues Identified



**Ambiguity**  
 (“easily” unclear)



**Missing requirement**  
 (mobile access not specified)



**Inconsistency**  
 (conflicts with security login steps)

- ✓ **Students**
- ✓ **Instructors**
- ✓ **Admin**

## 3 Validation Techniques Applied



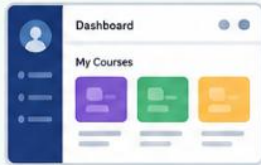
Review & walkthrough  
with stakeholders



Prototype of  
student dashboard



Test case creation  
(access within 2 clicks)



## 4 Refined Requirement



“Students can access enrolled  
course materials **within 2 clicks**  
via web and mobile platforms  
after secure login”

## 5 Outcome



Clear



Testable



Consistent



Validated  
Requirement



# Summary



Critical  
for  
success



Prevents  
costly  
failures



Continuous  
activity



Requires  
collaboration



Improves  
system  
quality

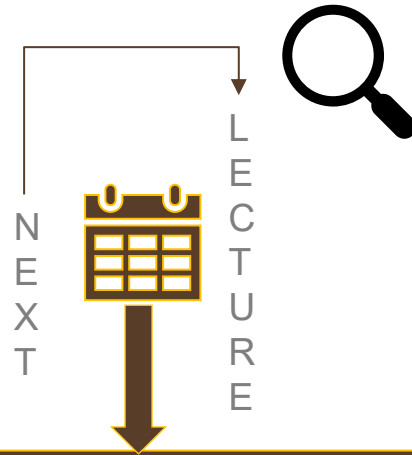


Leads to  
better  
outcomes

# References

1. Beatty, K. W. (2013). Software Requirements (3rd ed.). Washington: Microsoft Press.

# Thank You!



**Requirement Change Management**