Course: Software Requirements Engineering

Week 8: Requirements' Prioritization & Negotiation

Lecturer: Yimer Amedie (MSc.)

Addis Ababa Science and Technology University

May, 2026

Contents

- Introduction
- Overview of Requirements Prioritization
- Prioritization Techniques
- Requirement Negotiations and Techniques
- Role of Feasibility
- Challenges and Best Practices

Figure 1. *The software requirements*
Note. Image generated using Sora by OpenAI (2026).

Learning Outcomes

After completing this lesson, you will be able to:

- Explain prioritization concepts
- Apply appropriate requirements prioritization techniques
- Evaluate different types of feasibility
- Resolve conflicts through negotiation
- Construct a prioritized requirements list

Introduction

Figure 2. *Why all requirements are not treated equally?*
Note. Image generated using ChatGpt by OpenAI (2026).

What is Requirements' Prioritization?

1. Ranking requirements by importance

2. Based on value and urgency

3. Supports planning decisions

4. Aligns with business goals

5. Continuous process

It answers (Beatty, 2013)

1. Which requirements should be **implemented first**
2. Which can be **deferred** or **discarded**.

Why Prioritization is Needed?

Figure 3. *Why requirements prioritization is needed?*
Note. Image generated using Sora by OpenAI (2026).

Why Prioritization is Needed?

If requirements are not prioritized (Beatty, 2013),

- ✓ Projects operate under limited time and resources
- ✓ Conflicting stakeholder needs
- ✓ High number of requirements
- ✓ Risk of scope creep
- ✓ Need for value delivery

Key Factors in Prioritization

Prioritization Techniques

MoSCoW method

Ranking techniques

Scoring models

Value vs effort matrix

Kano model

RICE Method

Risk-Based Prioritization

MoSCoW Techniques

M

Must-have requirements

Essential requirements that are non-negotiable.
The project cannot be delivered without them.

S

Should-have requirements

Important requirements that add significant value.
The project should include them if possible.

C

Could-have requirements

Desirable requirements that would be nice to have.
Included if there is time and budget.

W

Won't-have requirements

Requirements that are explicitly excluded
from the current project scope.

Simple and effective

Purpose: Helps prioritize requirements and manage stakeholder expectations by focusing on what matters most.

Example: Online Learning System (OLMS)

Scenario

- ✓ E-learning platform scenario
- ✓ Multiple stakeholders involved
- ✓ Diverse requirements collected
- ✓ Limited budget and timeline
- ✓ Need for prioritization

Stakeholders & Their Needs

- ➢ Students need easy access
- ➢ Instructors need content tools
- ➢ Admins need control features
- ➢ Owners focus on ROI
- ➢ IT team focuses on feasibility

Sample Requirements List

- ✓ User registration and login
- ✓ Video lecture streaming
- ✓ Assignment submission
- ✓ Real-time chat
- ✓ AI-based recommendations

MoSCoW – Must-have requirements

- ✓ Critical for system success
- ✓ Non-negotiable features
- ✓ System fails without them
- ✓ High priority delivery
- ✓ Basis for minimum viable product (MVP)

Example: OLMS

- ➢ **Must:** login, course access

MoSCoW – Should-have requirements

- ✓ Important but not critical
- ✓ Workarounds may exist
- ✓ High business value
- ✓ Lower urgency than Must-have
- ✓ Delivered if resources allow

Example: OLMS

- ➢ **Should:** assignments

MoSCoW – Could-have requirements

- ✓ Nice-to-have features
- ✓ Low impact on core system
- ✓ Improve user experience
- ✓ Lowest priority
- ✓ Often deferred

Example: OLMS

➢ **Could:** chat feature

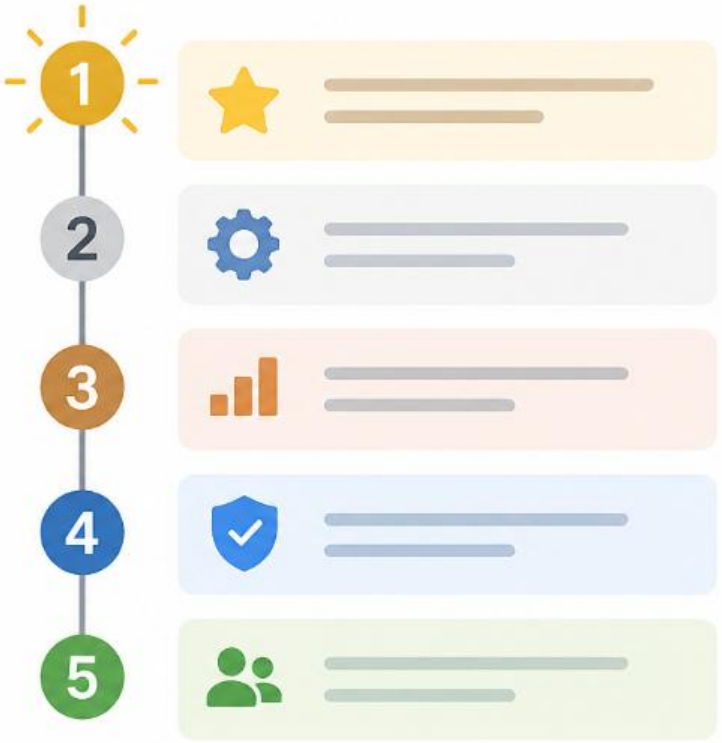
MoSCoW – Won’t-have requirements

- ✓ Out of current scope
- ✓ Agreed for exclusion
- ✓ Avoids scope creep
- ✓ May be future consideration
- ✓ Clarifies boundaries

Example: OLMS

➢ **Won’t:** AI recommendations

Ranking Techniques



- ✓ Order requirements by importance
- ✓ Simple and intuitive
- ✓ Direct comparison
- ✓ Suitable for small sets
- ✓ May lack objectivity

Scoring Model

- Assign numerical values
- Evaluate multiple criteria
- Weighted scoring approach
- More objective results
- Supports comparison

Value Vs Effort Matrix

1. Identify quick wins
2. High value, low effort first
3. Visual decision tool
4. Supports agile planning
5. Helps prioritize what delivers most value

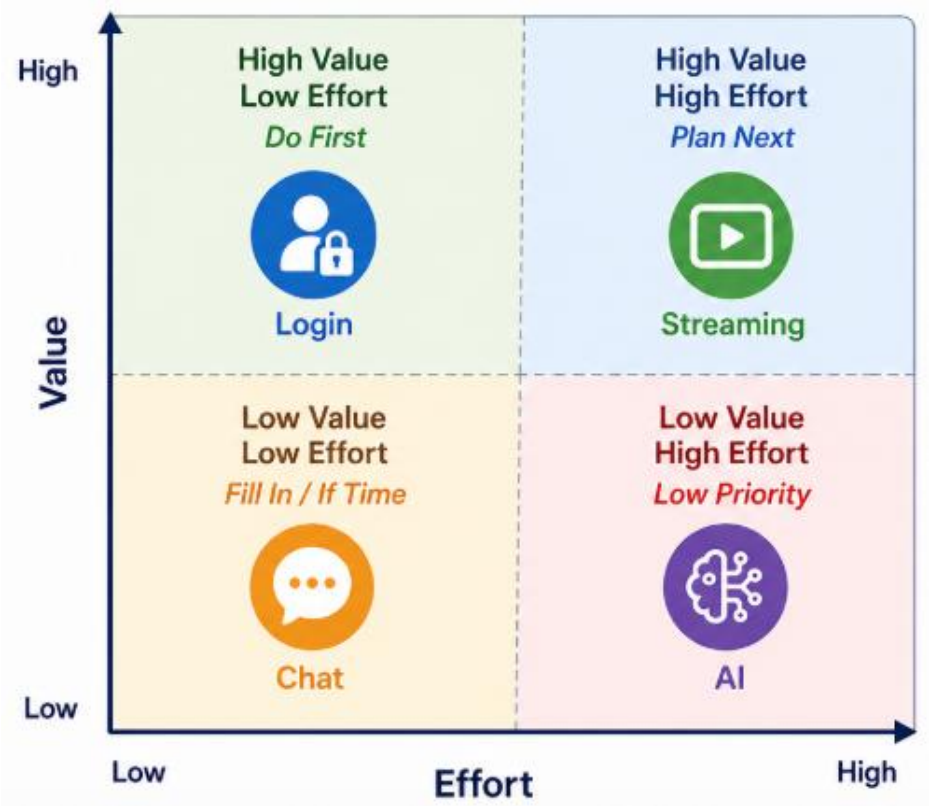
Value Vs Effort Matrix – Example – OLMS

Login
High value,
low effort

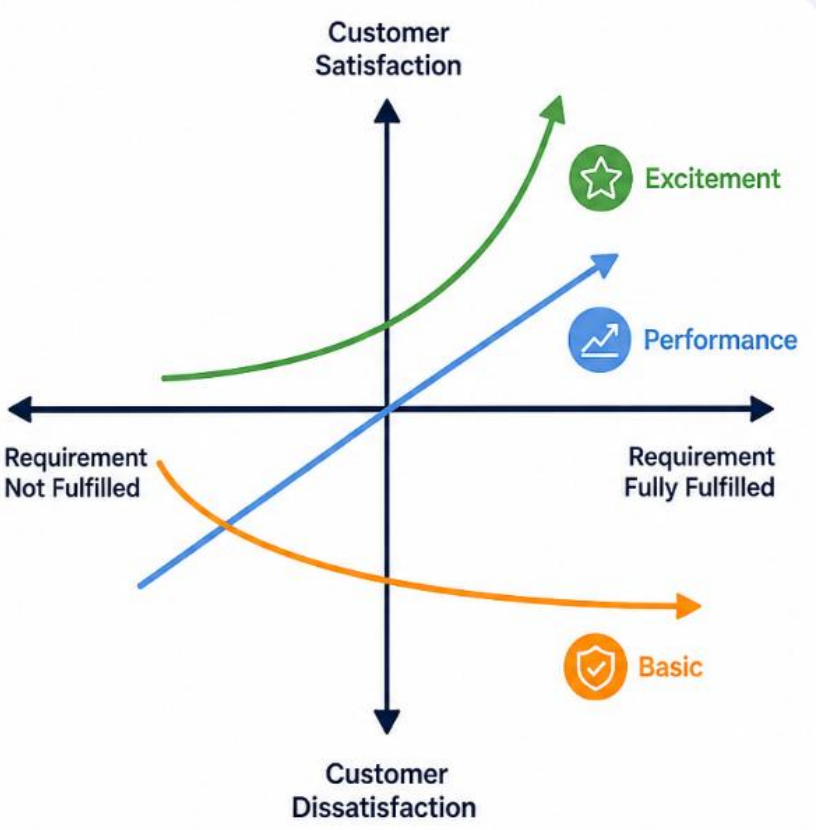
Streaming
High value,
high effort

Chat
Medium value
(Medium effort)

AI
High effort,
low priority



Kano model



- ✓ Focus on customer satisfaction
- ✓ Basic, performance, excitement
- ✓ Differentiates feature types
- ✓ Improves product quality
- ✓ Customer-centric approach

RICE Method

R — REACH

- How many users or stakeholders will be affected?
- Measured over a time period (e.g., users per month)

Example:
5,000 users per month

I — IMPACT

- How much value will it deliver per user?
- Common scale:

3	= Massive
2	= High
1	= Medium
0,5	= Low
0.25	= Minimal

C — CONFIDENCE

- How sure are you about your estimates?
- Expressed as a percentage (e.g., 80% = 0.8)

Example:
80%
= 0.8

E — EFFORT

- How much work is required?
- Usually measured in person-weeks or person-months

Example:
6 person-weeks of effort

RICE Method … cont'd

$$\text{RICE Score} = \frac{Reach \times Impact \times Confidence}{Effort}$$

Requirement	Reach	Impact	Confidence	Effort	RICE Score
Course access	200	2	0.8	4	80
Video streaming	150	3	0.7	8	39
Live chat	100	1	0.9	2	45

➔ **Course access** should be prioritized first.

For **confidence**, teams typically use three specific percentages to score confidence, these options are (Microsoft, 2024):

✓ **100% = high, 80% = medium, and 50% = low**

RICE Method … cont'd

Requirement Negotiation

Process of resolving conflicts among stakeholders and reaching agreement on requirements.

Resolving requirement conflicts

Identify and resolve conflicts, overlaps, and contradictions among requirements.

Achieving stakeholder agreement

Facilitate discussions and reach consensus among all relevant stakeholders.

Balancing interests

Balance business value, constraints, risk, cost, and user needs.

Key part of analysis

Essential activity in requirements analysis to ensure clarity, feasibility, and value.

Continuous process

Occurs throughout the project as needs evolve and new information emerges.

Sources of Conflict

Figure 4. *Source of conflicts in software requirements*
Note. Image generated using Sora by OpenAI (2026).

Negotiation Techniques

Trade-off analysis

Prioritization methods

Stakeholder discussions

Compromise strategies

Win-win approach

Example - OLMS

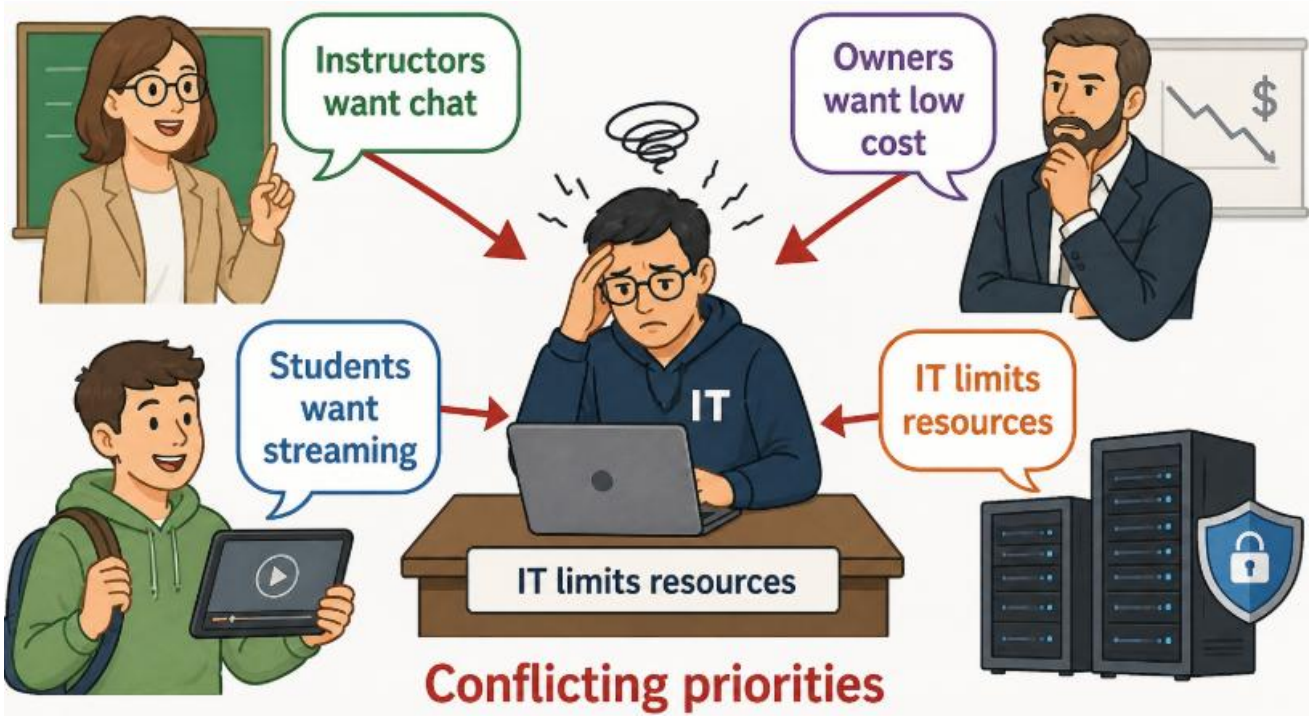


Figure 5. *Example of conflict scenario*
Note. Image generated using Sora by OpenAI (2026).

Negotiation in Action

Discuss stakeholder needs

Analyze trade-offs

Focus on value

Agree on compromises

Document decisions

For example: OLMS

- ✓ Delaying the chat feature to focus on video streaming.
- ✓ Leads to better decisions and stronger collaboration.

Role of Feasibility

Determines practicality

Evaluates technical, economic, operational and schedule feasibility to determine what is achievable.

Guides prioritization

Helps focus on requirements that are feasible and deliver the most value.

Reduces risks

Identifies potential issues early and helps avoid costly rework or failure.

Supports decisions

Provides insights that help stakeholders make informed and confident decisions.

Ensures realistic planning

Ensures plans are based on what is realistic in terms of time, cost, resources and capabilities.

Feasibility ensures we build the right solution, the right way, at the right time.

Types of Feasibility

Technical feasibility

Economic feasibility

Operational feasibility

Legal feasibility

Schedule feasibility

Technical Feasibility

- ✓ Availability of technology
- ✓ System compatibility
- ✓ Required skills and expertise
- ✓ Infrastructure readiness
- ✓ Integration complexity

Economic Feasibility

- ✓ Cost-benefit analysis
- ✓ Budget constraints
- ✓ Return on investment
- ✓ Development and maintenance cost
- ✓ Financial risks

Operational Feasibility

Figure 6. *Legal & Compliance Feasibility*
Note. Image generated using Sora by OpenAI (2026).

Legal & Compliance Feasibility

Figure 7. *Legal & Compliance Feasibility*
Note. Image generated using Sora by OpenAI (2026).

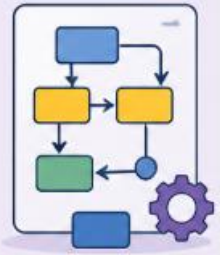
Schedule Feasibility

Time constraints

Project deadlines

Resource availability

Task dependencies



Delivery milestones

Ensuring the project schedule is realistic, achievable, and aligned with goals, resources, and constraints.

Feasibility Consideration – Example – OLMS

- ✓ Streaming requires bandwidth
- ✓ Chat needs real-time support
- ✓ AI needs advanced tools
- ✓ Budget constraints apply
- ✓ Feasibility guides decisions

Final Prioritized List

- ✓ **Phase 1:** login, streaming
- ✓ **Phase 2:** assignments
- ✓ **Phase 3:** chat
- ✓ **Future:** AI features
- ✓ **Incremental** delivery

Decision-Making in Prioritization

Data-driven decisions

Stakeholder input

Use of models

Trade-off analysis

Transparent process

Role of Stakeholders in Prioritization

Challenges in Prioritization

Conflicting stakeholder views

Changing requirements

Limited resources

Lack of clear criteria

Bias in decision-making

Challenges in Negotiations

Communication gaps

Power imbalance

Emotional conflicts

Resistance to compromise

Lack of trust

Best Practices

Figure 8. *Best Practices in requirement prioritization*
Note. Image generated using Sora by OpenAI (2026).

Summary

Prioritization ensures value delivery

Techniques guide decision-making

Negotiation resolves conflicts

Feasibility ensures practicality

Continuous improvement needed

References

1. Beatty, K. W. (2013). Software Requirements (3rd ed.). Washington: Microsoft Press.
2. Microsoft. (2024, August 27). Understanding the RICE Model and its framework. Retrieved May 3, 2026, from Microsoft website: https://www.microsoft.com/en-us/microsoft-365-life-hacks/organization/understanding-the-rice-model-and-its-framework

Thank You!