CHAPTER 2

Requirements from the customer's perspective

Read by yourself

- Chapter 2
  - Who is the customer?
  - Differentiate stakeholders and customers
  - The gap between what the customer needs and what the developer builds
  - Parties involve the reaching agreements
  - What is requirements baseline?
  - The customer's rights and customer's responsibilities
- Chapter 3
  - Requirements development process framework
  - The distribution of requirements development related in different type of SDLC
  - The goal of every phase in Requirements development process framework

Objectives

- After finish this chapter, student should understand that the customer-development relationship is so critical to software project success.
- This chapter also discusses the critical issue of reaching agreement on a set of requirements planned for a specific release or development iteration
- Requirements Bill of Rights for Software Customers and a corresponding Requirements Bill of Responsibilities for Software Customers could bring to student the importance of customer—and specifically end user—involvement in requirements development.

Contents

1. The expectation gap
2. Who is the customer
3. The customer-development partnership
4. Requirements Bill of Rights for Software Customers
5. Requirements Bill of Responsibilities for Software Customers
6. Creating a culture that respects requirements
7. Identifying decision makers
8. Reaching agreement on requirements

The expectation gap

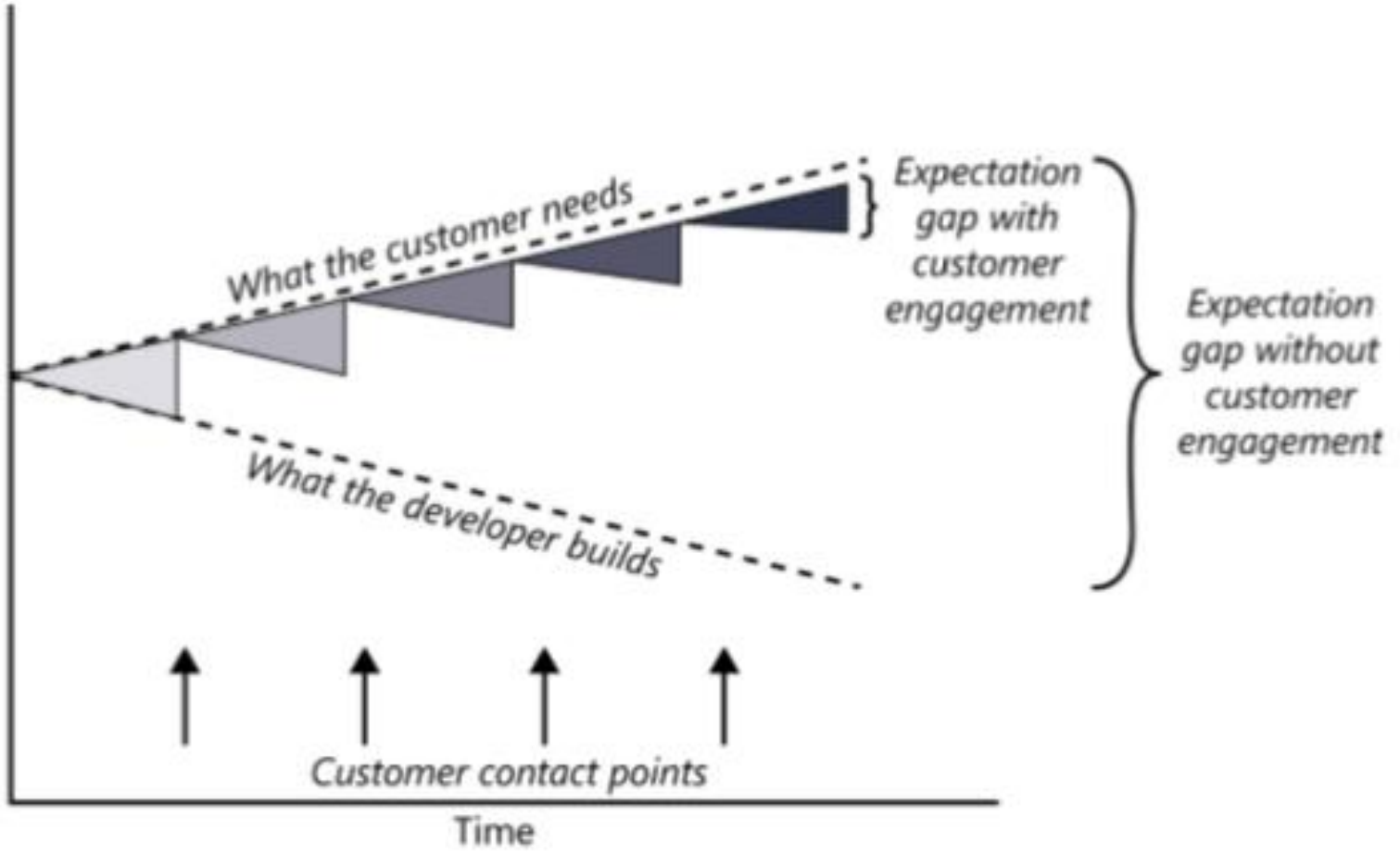


FIGURE 2-1 Frequent customer engagement reduces the expectation gap.

Who is the customer?

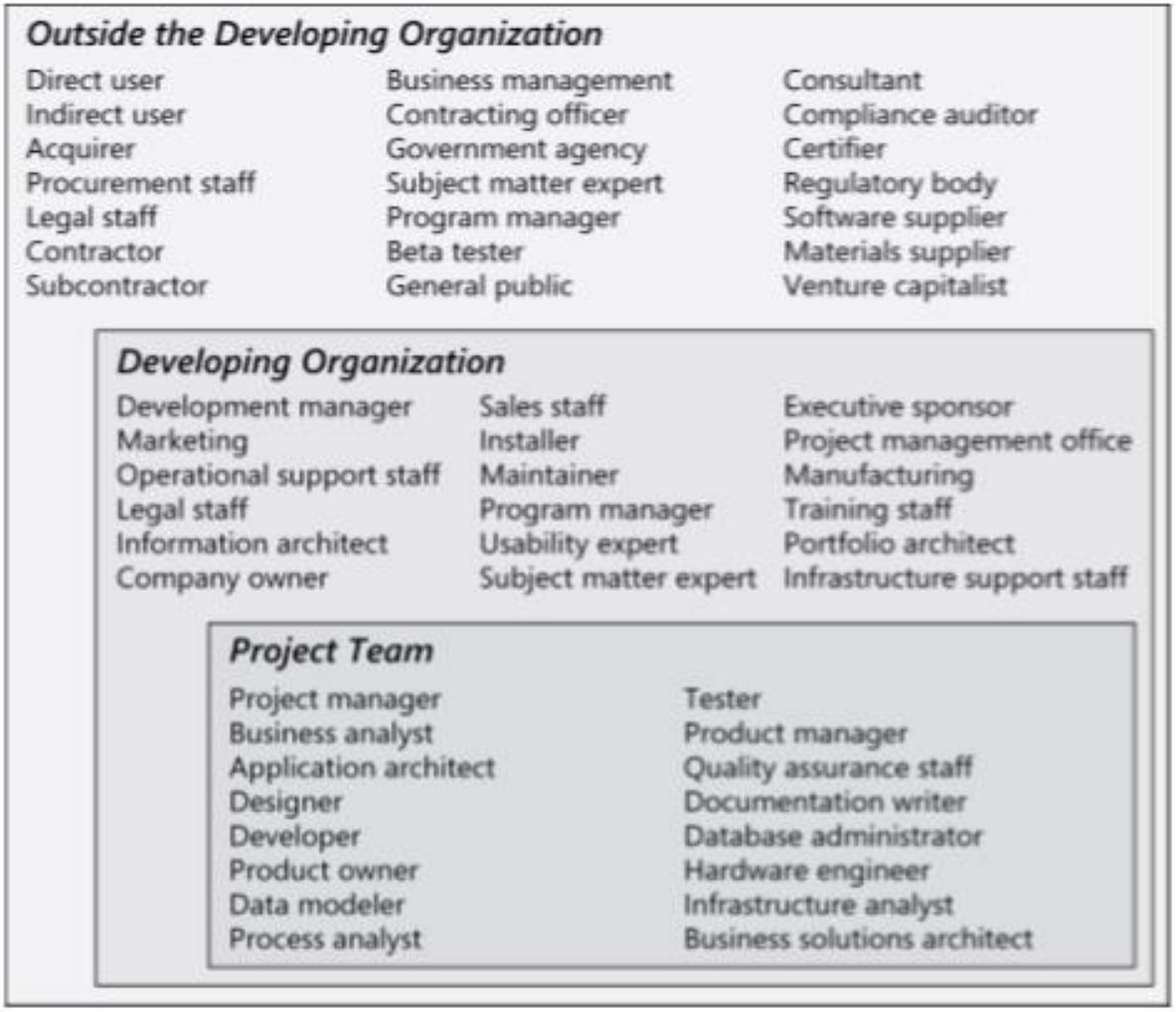


FIGURE 2-2 Potential stakeholders within the project team, within the developing organization, and outside the organization.

The customer-development partnership

- Customers have the right to:

1. Expect BAs to speak your language.
2. Expect BAs to learn about your business and your objectives.
3. Expect BAs to record requirements in an appropriate form.
4. Receive explanations of requirements practices and deliverables.
5. Change your requirements.
6. Expect an environment of mutual respect.
7. Hear ideas and alternatives for your requirements and for their solution.
8. Describe characteristics that will make the product easy to use.
9. Hear about ways to adjust requirements to accelerate development through reuse.
10. Receive a system that meets your functional needs and quality expectations.

The customer-development partnership

- **Customers have the responsibility to**:

1. Educate BAs and developers about your business.
2. Dedicate the time that it takes to provide and clarify requirements.
3. Be specific and precise when providing input about requirements.
4. Make timely decisions about requirements when asked.
5. Respect a developer's assessment of the cost and feasibility of requirements.
6. Set realistic requirement priorities in collaboration with developers.
7. Review requirements and evaluate prototypes.
8. Establish acceptance criteria.
9. Promptly communicate changes to the requirements.
10. Respect the requirements development process

Identifying decision makers

- decision leader
- decision rule

Reaching agreement on requirements

- Customers agree that the requirements address their needs.
- Developers agree that they understand the requirements and that they are feasible.
- Testers agree that the requirements are verifiable.
- Management agrees that the requirements will achieve their business objectives.

❖**The requirements baseline**

❖**What if you don’t reach agreement?**

❖**Agreeing on requirements on agile projects**

The requirements baseline (p39)

- A requirements baseline is a set of requirements that has been reviewed and agreed upon and serves as the basis for further development
- A meaningful baselining process gives all the major stakeholders confidence in the following ways:
  - Customer management or marketing is confident that the project scope won't explode out of control, because customers manage the scope change decisions.
  - User representatives have confidence that the development team will work with them to deliver the right solution, even if they didn't think of every requirement before construction began.
  - Development management has confidence because the development team has a business partner who will keep the project focused on achieving its objectives and will work with development to balance schedule, cost, functionality, and quality.
  - Business analysts and project managers are confident that they can manage changes to the project in a way that will keep chaos to a minimum.
  - Quality assurance and test teams can confidently develop their test scripts and be fully prepared for their project activities.

What if you don’t reach agreement?

Agreeing on requirements on agile projects

CHAPTER 3
Requirements engineering good practices

Objectives

- Student should enhance every stage in the requirement development process framework and how to implement these stages.
- Student should obtain a wide range of good practice in software requirements to prepare for the next chapters
- Need to help student understand these practices aren't suitable for every situation, so use good judgment, common sense, and experience. Even the best practices need to be selected, applied, and adapted thoughtfully to appropriate situations by skilled business analysts.
- Student must enhance that different practices might be most appropriate for understanding the requirements for different portions of a given project.
- Use cases and user interface prototypes might help for the client side, whereas interface analysis is more valuable on the server side…etc.

Contents

1. Requirements engineering good practices
2. A requirements development process framework
   - Elicitation
   - Analysis
   - Specification
   - Validation
3. A representative requirements development process
4. Requirements management

Requirements engineering good practices

Elicitation	Analysis	Specification	Validation
■ Define vision and scope ■ Identify user classes ■ Select product champions ■ Conduct focus groups ■ Identify user requirements ■ Identify system events and responses ■ Hold elicitation interviews ■ Hold facilitated elicitation workshops ■ Observe users performing their jobs ■ Distribute questionnaires ■ Perform document analysis ■ Examine problem reports ■ Reuse existing requirements	■ Model the application environment ■ Create prototypes ■ Analyze feasibility ■ Prioritize requirements ■ Create a data dictionary ■ Model the requirements ■ Analyze interfaces ■ Allocate requirements to subsystems	■ Adopt requirement document templates ■ Identify requirement origins ■ Uniquely label each requirement ■ Record business rules ■ Specify nonfunctional requirements	■ Review the requirements ■ Test the requirements ■ Define acceptance criteria ■ Simulate the requirements

Requirements management	Knowledge	Project management
■ Establish a change control process ■ Perform change impact analysis ■ Establish baselines and control versions of requirements sets ■ Maintain change history ■ Track requirements status ■ Track requirements issues ■ Maintain a requirements traceability matrix ■ Use a requirements management tool	■ Train business analysts ■ Educate stakeholders about requirements ■ Educate developers about application domain ■ Define a requirements engineering process ■ Create a glossary	■ Select an appropriate life cycle ■ Plan requirements approach ■ Estimate requirements effort ■ Base plans on requirements ■ Identify requirements decision makers ■ Renegotiate commitments ■ Manage requirements risks ■ Track requirements effort ■ Review past lessons learned

A requirements development process framework

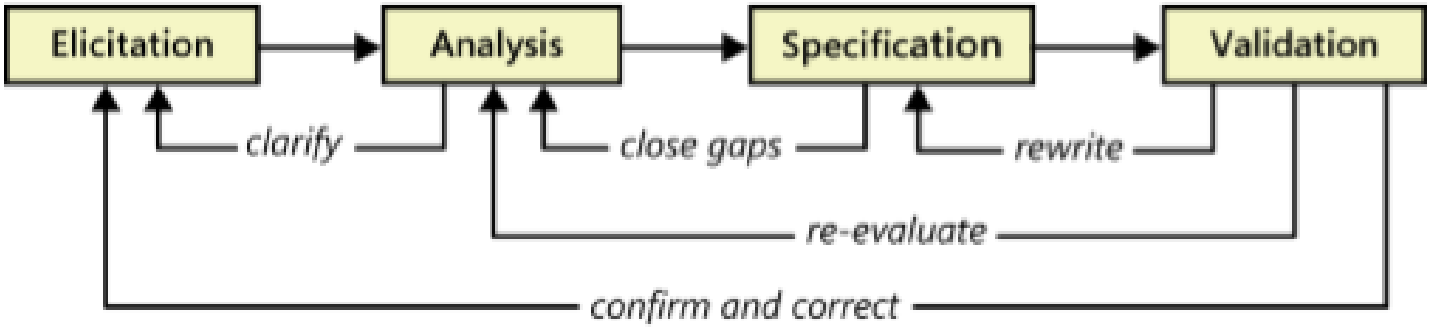
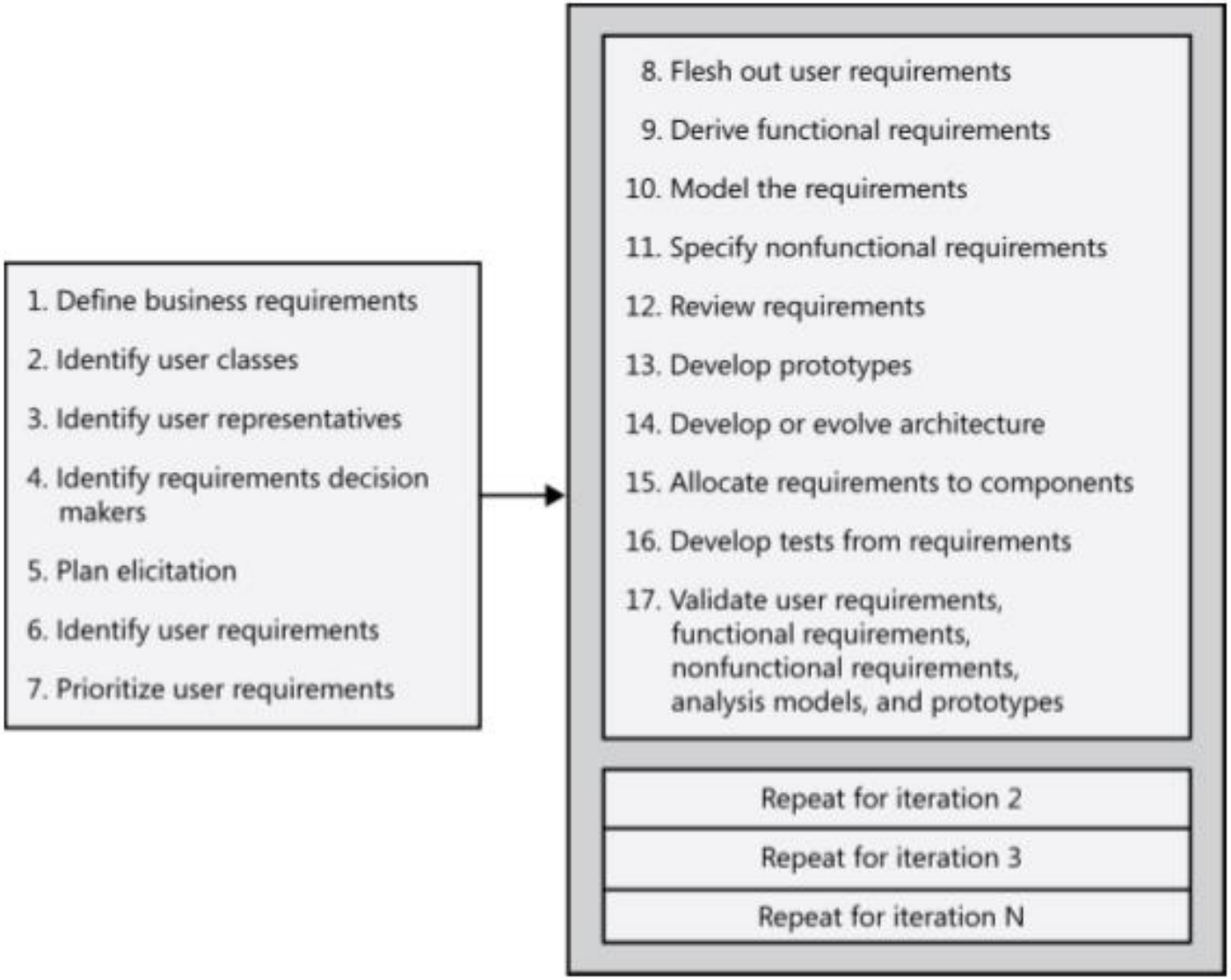


FIGURE 3-1 Requirements development is an iterative process.

A representative requirements development process



Requirements elicitation

- Define product vision and project scope
- Identify user classes and their characteristics
- Select a product champion for each user class
- Conduct focus groups with typical users
- Work with user representatives to identify user requirements
- Identify system events and responses
- Hold elicitation interviews
- Hold facilitated elicitation workshops
- Observe users performing their jobs
- Distribute questionnaires
- Perform document analysis
- Examine problem reports of current systems for requirement ideas
- Reuse existing requirements

Requirements analysis

- Model the application environment
- Create user interface and technical prototypes
- Analyze requirement feasibility
- Prioritize the requirements
- Create a data dictionary
- Model the requirements
- Analyze interfaces between your system and the outside world
- Allocate requirements to subsystems

Requirements specification

- Adopt requirement document templates
- Identify requirement origins
- Uniquely label each requirement
- Record business rules
- Specify nonfunctional requirements

Requirements validation

- Review the requirements
- Test the requirements
- Define acceptance criteria
- Simulate the requirements

Requirements management

- Establish a requirements change control process
- Perform impact analysis on requirements changes
- Establish baselines and control versions of requirements sets
- Maintain a history of requirements changes
- Track the status of each requirement
- Track requirements issues
- Maintain a requirements traceability matrix
- Use a requirements management tool

Knowledge

- Train business analysts
- Educate stakeholders about requirements
- Educate developers about the application domain
- Define a requirements engineering process
- Create a glossary

Project management

- Select an appropriate software development life cycle
- Plan requirements approach
- Estimate requirements effort
- Base project plans on requirements
- Identify requirements decision makers
- Renegotiate project commitments when requirements change
- Analyze, document, and manage requirements-related risks
- Track the effort spent on requirements
- Review lessons learned regarding requirements on other projects