CHAPTER 1
The essential of software requirement

Objectives

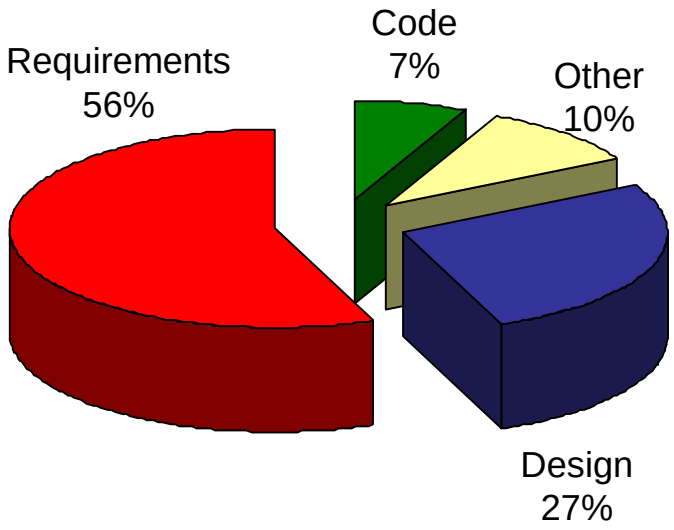
- Understand some key terms used in the software requirements domain.
- Distinguish product requirements from project requirements.
- Distinguish requirements development from requirements management.
- Be alert to several requirements-related problems that can arise

Contents

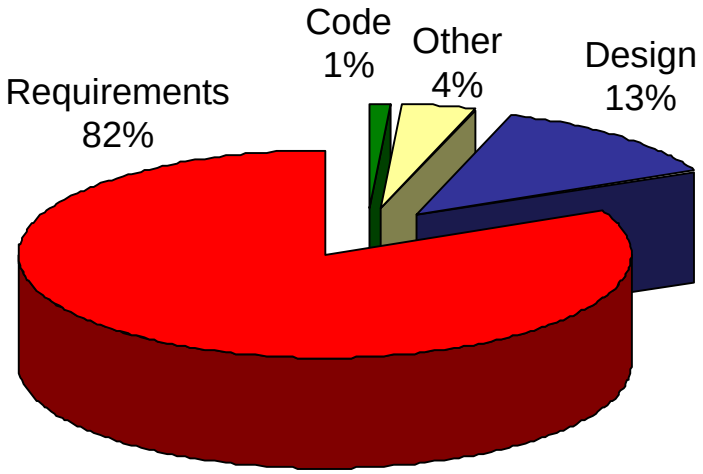
1. Why is requirement engineering?
2. Software requirements defined
3. Requirements development and management
4. When bad requirements happen to good people
5. Benefits from a high-quality requirements process

Why is requirement engineering?

- **Distribution of Defects**



- **Distribution of Effort to Fix Defects**



Source: Martin & Leffinwell

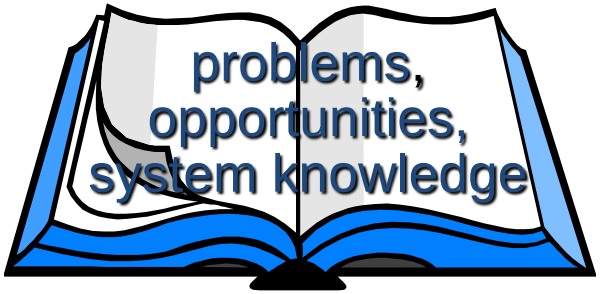
Software requirements defined

- Brian Lawrence: “anything that drives design choices” (*Lawrence 1997*)

- “**Requirements are a specification of what should be implemented. They are descriptions of how the system should behave, or of a system property or attribute. They may be a constraint on the development process of the system**” *Ian Sommerville and Pete Sawyer (1997)*

The scope of RE: the *WHY*, *WHAT*, *WHO* dimensions

System-as-is



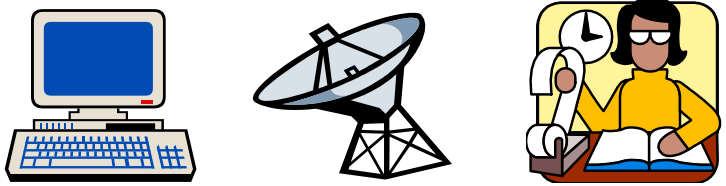
System-to-be

Objectives

satisfy

requirements,
constraints,
assumptions

assignment



WHY
a new system?

WHAT
services?

WHO
will be
responsible
for what ?

List of assignments

1. The landlord management system
2. Temporary employees management system
3. Vegetables and Fruits ordering system
4. Cinema ticket management system
5. Rental vehicles system
6. Parking location management system
7. Food ordering system
8. Traveling booking management

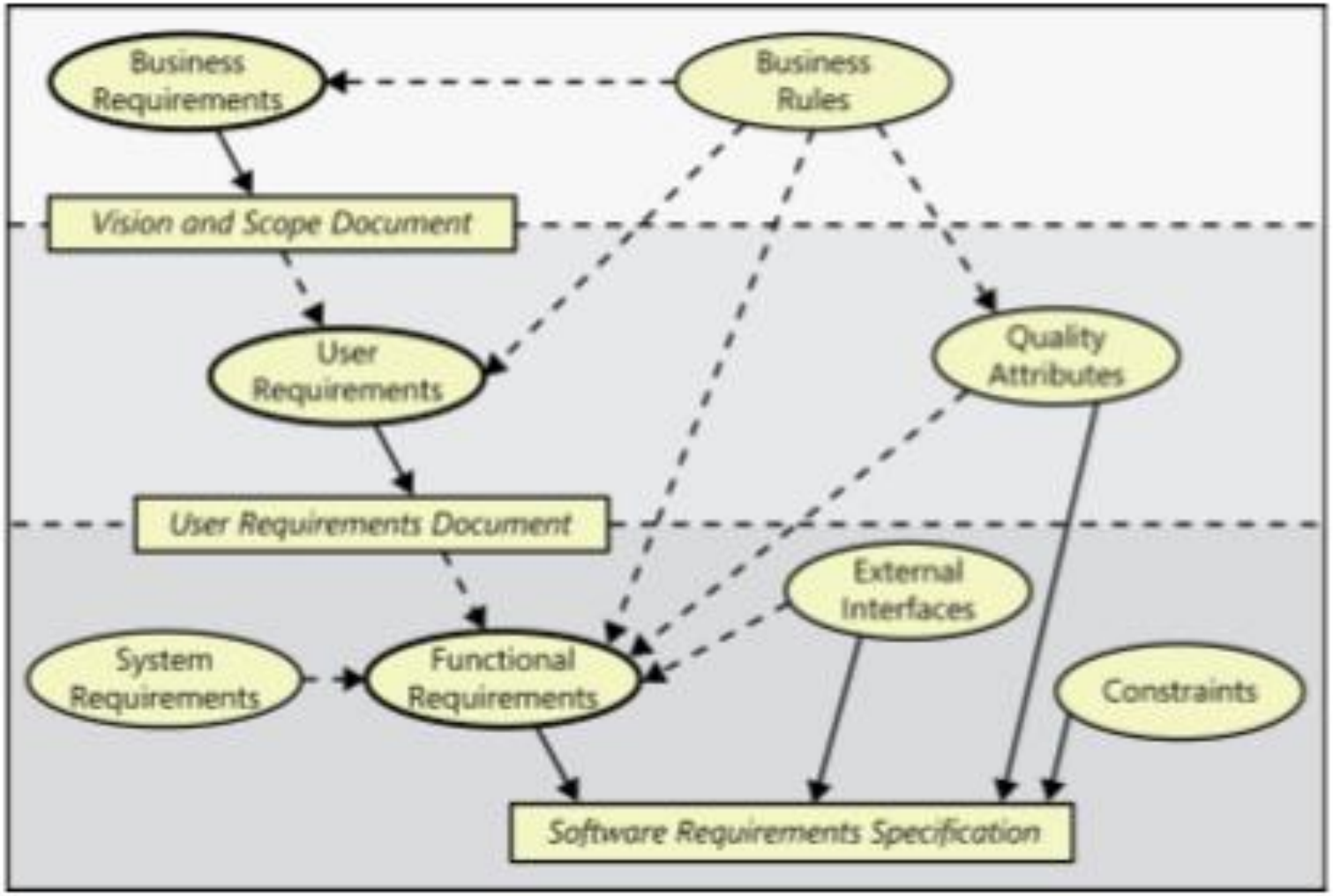
Group	AssignmentNumber
1	6
2	5
3	8
4	7
5	2
6	3
7	4
8	1

Types of requirements

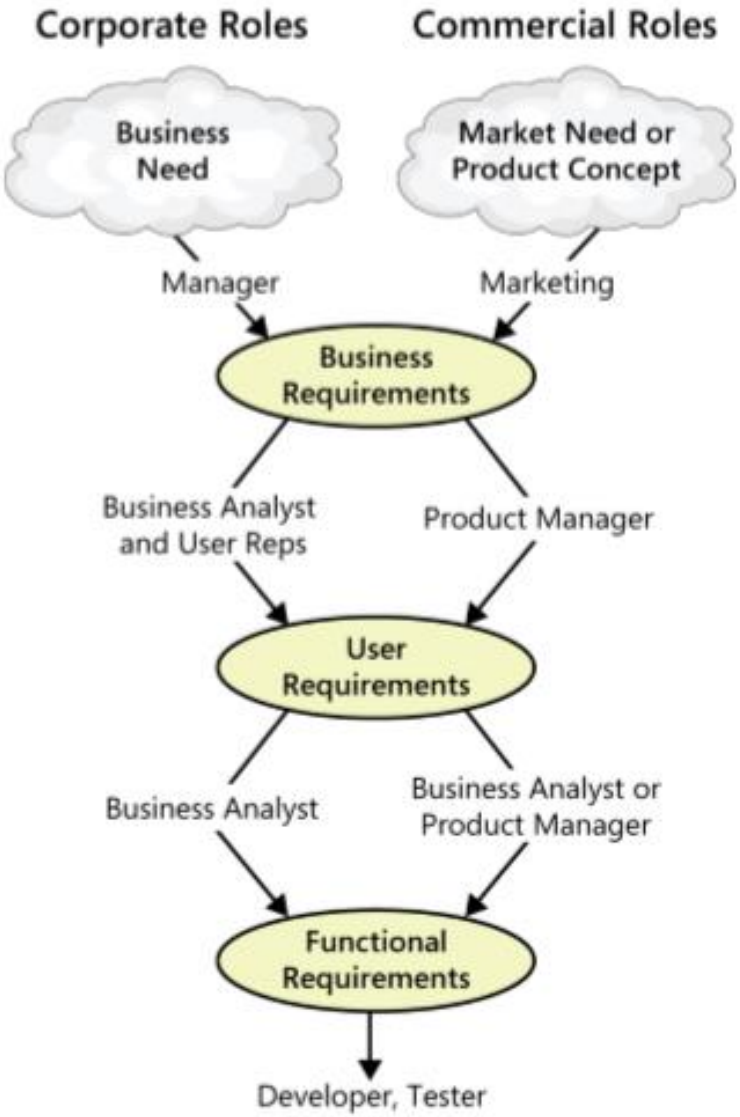
TABLE 1-1 Some types of requirements information

Term	Definition
Business requirement	A high-level business objective of the organization that builds a product or of a customer who procures it.
Business rule	A policy, guideline, standard, or regulation that defines or constrains some aspect of the business. Not a software requirement in itself, but the origin of several types of software requirements.
Constraint	A restriction that is imposed on the choices available to the developer for the design and construction of a product.
External interface requirement	A description of a connection between a software system and a user, another software system, or a hardware device.
Feature	One or more logically related system capabilities that provide value to a user and are described by a set of functional requirements.
Functional requirement	A description of a behavior that a system will exhibit under specific conditions.
Nonfunctional requirement	A description of a property or characteristic that a system must exhibit or a constraint that it must respect.
Quality attribute	A kind of nonfunctional requirement that describes a service or performance characteristic of a product.
System requirement	A top-level requirement for a product that contains multiple subsystems, which could be all software or software and hardware.
User requirement	A goal or task that specific classes of users must be able to perform with a system, or a desired product attribute.

Levels and types of requirements



The participation of stakeholders in requirements development



Product vs. project requirements

- Product requirements:
  - properties of a software system to be built
- Projects requirements:
  - do have other expectations and deliverables that are not a part of the software the team implements, but that are necessary to the successful completion of the project as a whole

Requirements development and management

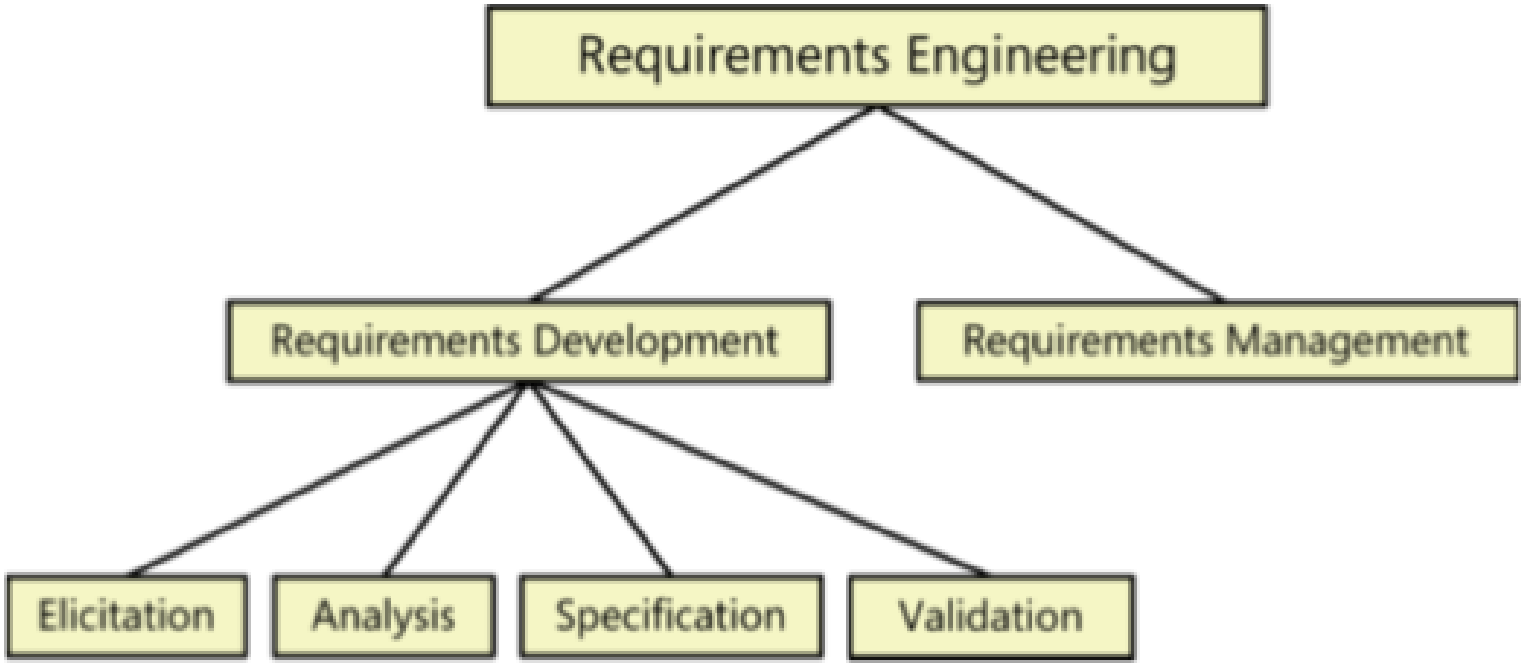


FIGURE 1-4 Subdisciplines of software requirements engineering.

Elicitation

- Identifying the product's expected user classes and other stakeholders.
- Understanding user tasks and goals and the business objectives with which those tasks align.
- Learning about the environment in which the new product will be used.
- Working with individuals who represent each user class to understand their functionality needs and their quality expectations.

Analysis

- Analyzing the information received from users to distinguish their task goals from functional requirements, quality expectations, business rules, suggested solutions, and other information
- Decomposing high-level requirements into an appropriate level of detail
- Deriving functional requirements from other requirements information
- Understanding the relative importance of quality attributes
- Allocating requirements to software components defined in the system architecture
- Negotiating implementation priorities
- Identifying gaps in requirements or unnecessary requirements as they relate to the defined scope

Specification

- Translating the collected user needs into written requirements and diagrams suitable for comprehension, review, and use by their intended audiences.

Validation

- Reviewing the documented requirements to correct any problems before the development group accepts them.
- Developing acceptance tests and criteria to confirm that a product based on the requirements would meet customer needs and achieve the business objectives.

Requirements management

- Defining the requirements baseline, a snapshot in time that represents an agreed-upon, reviewed, and approved set of functional and nonfunctional requirements, often for a specific product release or development iteration
- Evaluating the impact of proposed requirements changes and incorporating approved changes into the project in a controlled way
- Keeping project plans current with the requirements as they evolve
- Negotiating new commitments based on the estimated impact of requirements changes
- Defining the relationships and dependencies that exist between requirements
- Tracing individual requirements to their corresponding designs, source code, and tests
- Tracking requirements status and change activity throughout the project

Requirements management & requirements development

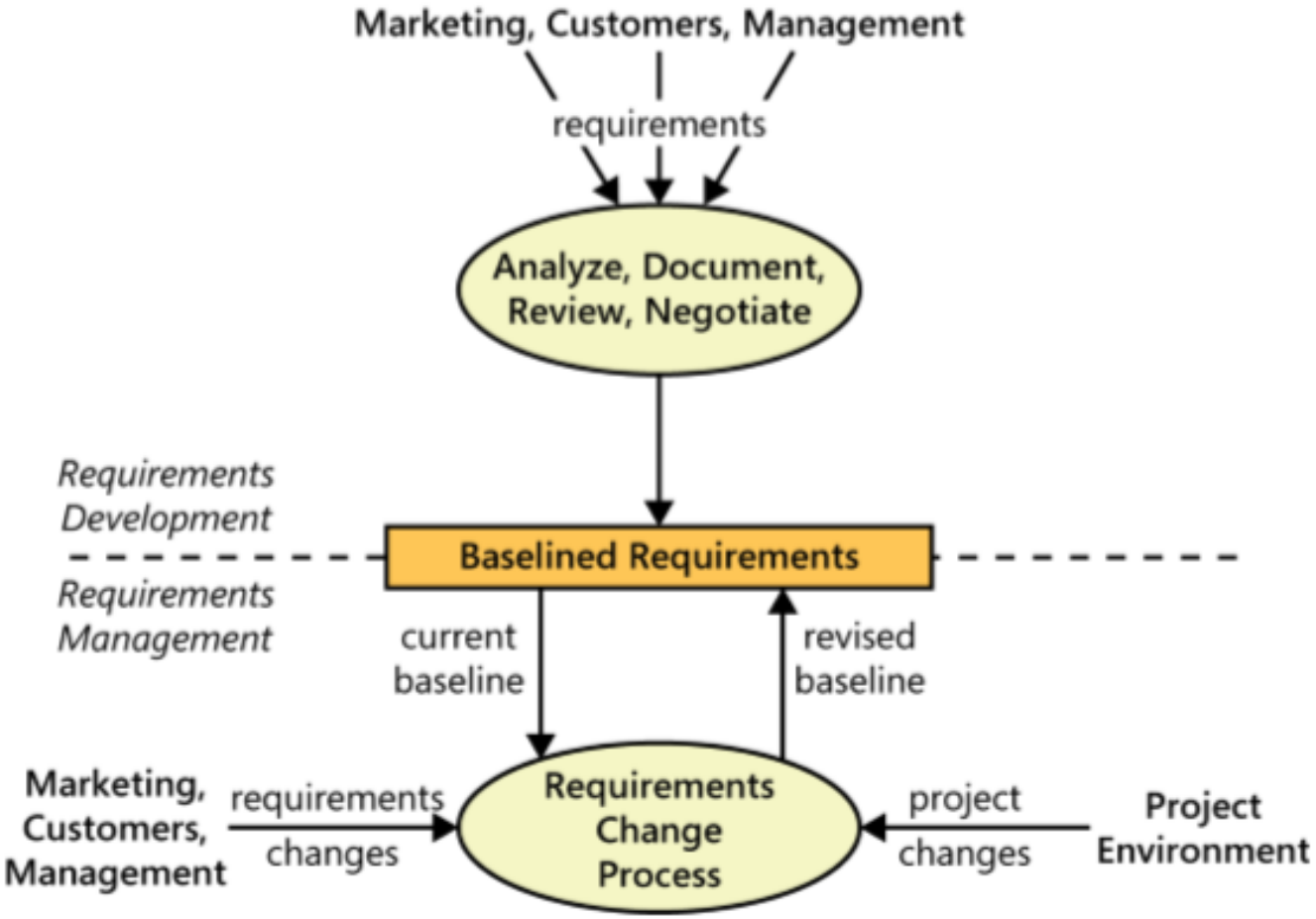


FIGURE 1-5 The boundary between requirements development and requirements management.

When bad requirements happen to good people

- Insufficient user involvement
- Inaccurate planning
- Creeping user requirements
- Ambiguous requirements
- Gold plating
- Overlooked stakeholders

Benefits from a high-quality requirements process

- Fewer defects in requirements and in the delivered product.
- Reduced development rework.
- Faster development and delivery.
- Fewer unnecessary and unused features
- Lower enhancement costs
- Fewer miscommunications.
- Reduced scope creep.
- Reduced project chaos.
- Higher customer and team member satisfaction.
- Products that do what they’re supposed to do.