Software Requirements Engineering

SE 311

Has a SW shortage ever frustrated you?

Introduction

- Software defines our lives

Self-driving Vehicles

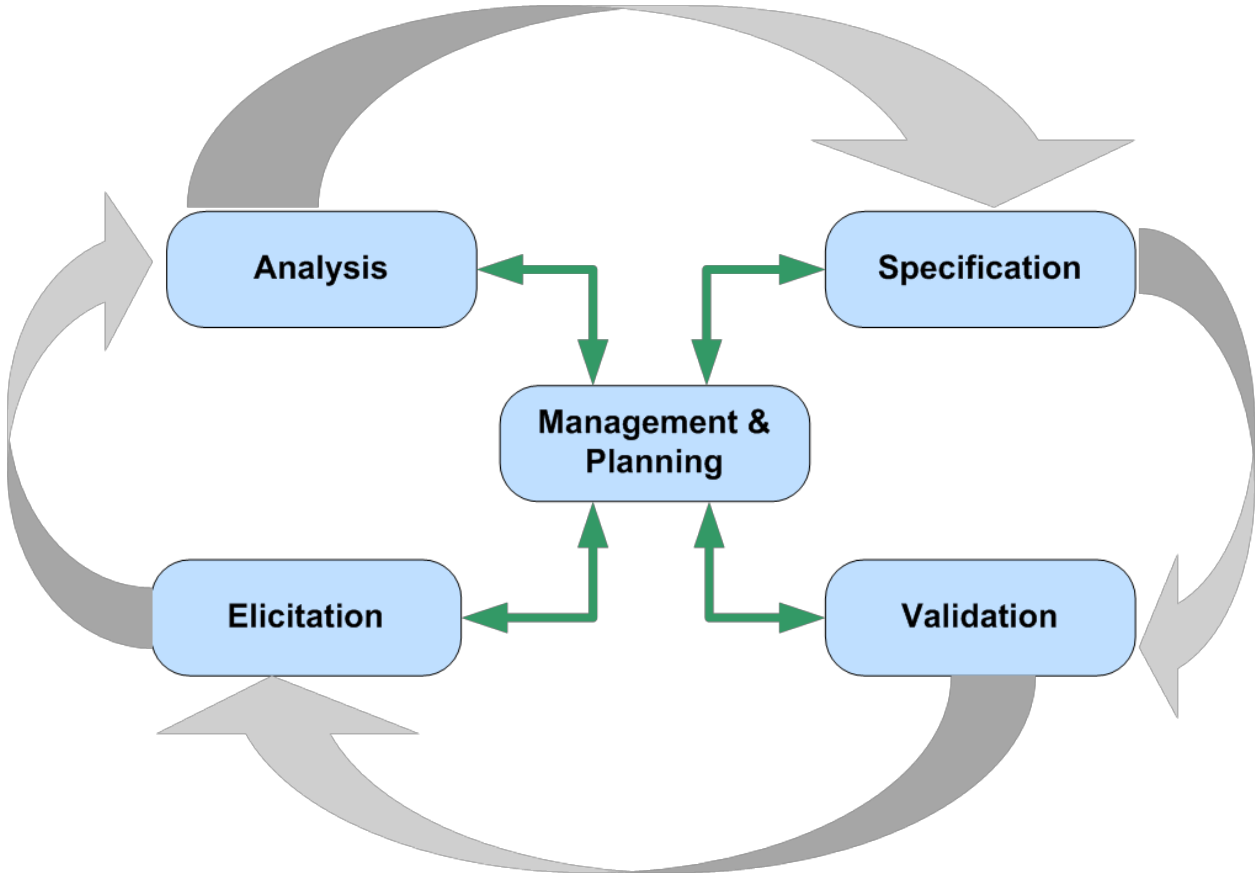
Unmanned Aerial Vehicles

Medical Sensors & Actuators

Robots

Requirements Engineering (RE) Process

- A software system is assessed in terms of the extent to which it satisfies the purpose for which it was intended.
- RE is the process of discovering that purpose by defining a set of software requirements to delineate the constraints and demands of the system



Given that

- The **size** of the worldwide software industry in is measured in billions in an annual bases
- **Errors** introduced during requirements activities account for 40 to 50 pecrcent of all defects found in SW products
- The hardest single part of conceptually defining a software system is the task of **defining the software requirements**.
- “no other part of the work so cripples the resulting system if done wrong” and “no other part is more difficult to rectify later” (Brooks, 1987)

Course Objectives

- CLO1: Recognize the different types of requirements and importance of software requirement engineering.
- CLO2: Identify multiple techniques to elicit requirements from stakeholders, choosing from among alternative methods as appropriate for different situations.
- CLO3: Analyze documenting requirements for high-level software requirements engineering.
- CLO4: Design and develop models for requirement documentation using goals, use cases, UML class and activity diagrams and state charts.
- CLO5: Evaluate the software requirements using validation and negotiation techniques regarding priorities and scope with the stakeholders.
- CLO6: Measure the documented requirements using quality criteria techniques for accuracy, unambiguity, consistent and completeness of the requirements.
- CLO7: Demonstrate the management of requirements as they change over time. (e.g., over multiple releases and over a product line)

Class Project

- **Identify** a *real* problem in any domain of your choice where you think a SW solution is needed to solve the problem
- **Identify** *real* problem stakeholders
- **Conduct** what you've learned about RE in order to capture stakeholders needs and produce a **Software Requirements Specification** (SRS) for your proposed solution

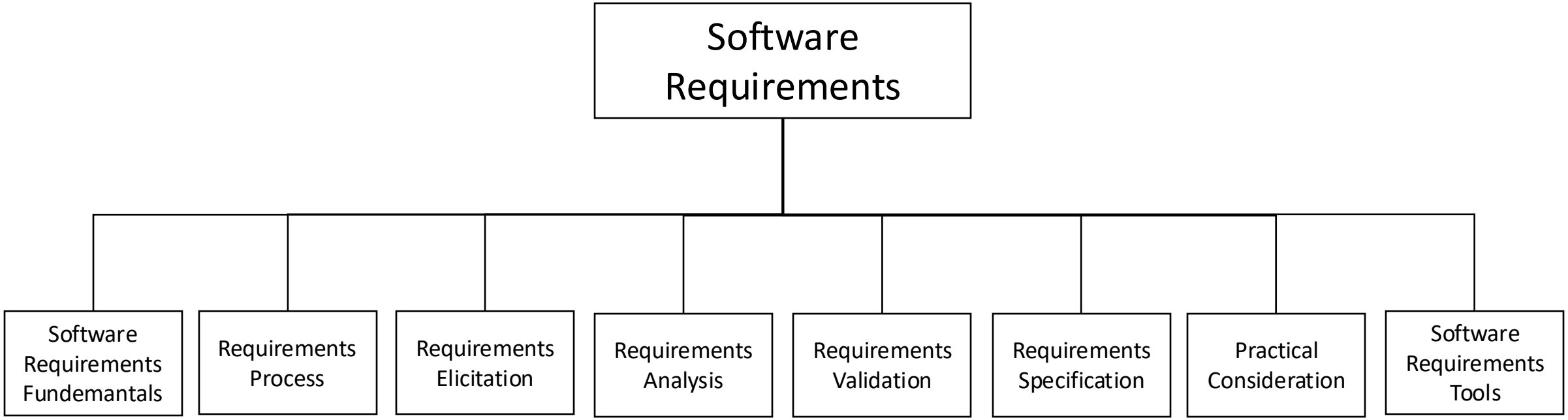
Class Project

- Teams of 3-4 students
- Select a team leader to coordinate communication and deliver submissions
- Four deliverables:

First deliverable	Second deliverable	Third deliverable	Fourth deliverable
• Vision & scope document	• Elicitation notes and evidence of elicitation process • A list of prioritized user requirements	• SRS that is approved by stakeholders • A prototype of the SW	• Presentation

Software Requirements Fundamentals

Software Requirements Knowledge Area

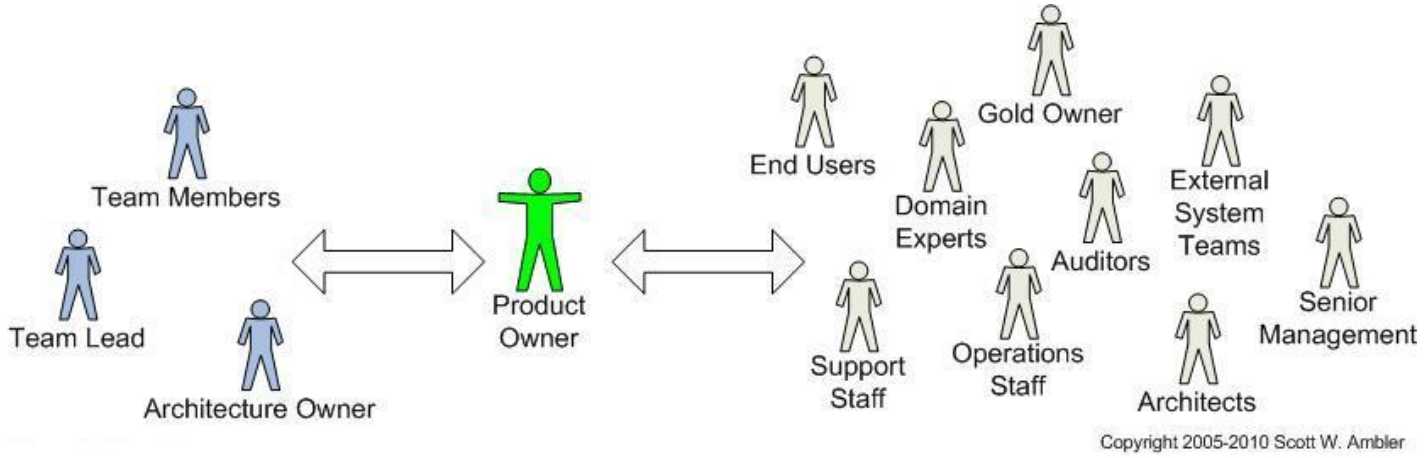


What is a *software requirement*?

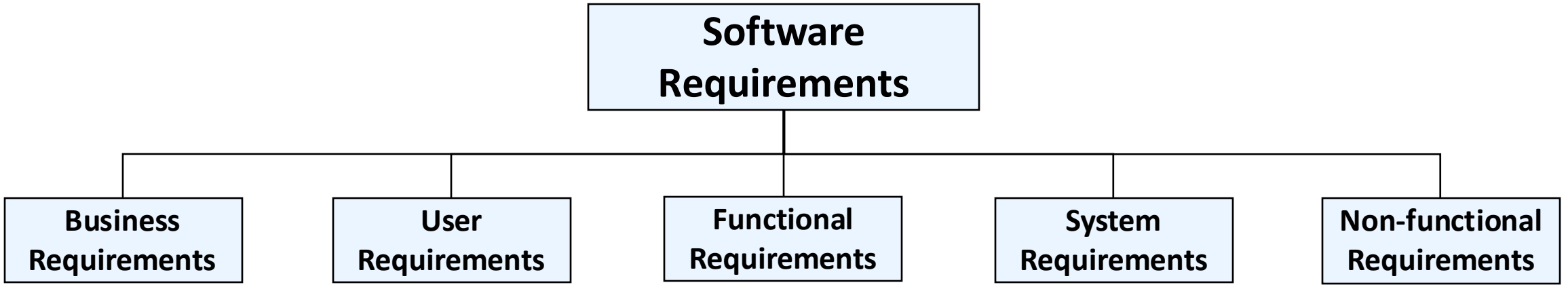
- A statement which translates or expresses a need and its associated constraints and conditions (IEEE/ISO/IEC 29148-2018)
- At its most basic, a software requirement is a property that must be exhibited by something in order to solve some problem in the real world (SWEBOK V3)

What are *software requirements*?

- The set of requirements form a complex matrix of the needs of many stakeholders who are spread across different levels of the organization and who are somehow linked to the problem



Types of Requirements



Definitions

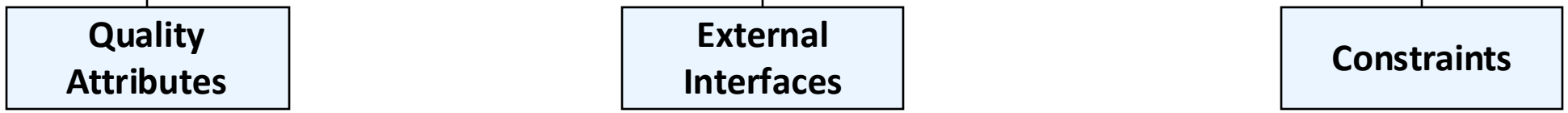
Business Requirements	User Requirements	Functional Requirements	System Requirements	Non-functional Requirements
Describe why the organization is implementing the system.	Describe goals or tasks the users must be able to perform with the product that will provide value to someone.	Specify the behaviors the product will exhibit under specific conditions	Describe the requirements for a product that is composed of multiple components or subsystems	Describe how *well* the products carries out its tasks. • Quality attributes • External interfaces • Constraints

Examples

Business Requirements	User Requirements	Functional Requirements	System Requirements	Non-functional Requirements
”Airline wants to reduce airport counter staff costs by 25 percent.“	”User needs to be able to check-in for flight using airline’s website.”	”If the Passenger’s profile does not indicate a seating preference, the reservation system shall assign a seat.”	”The airport check-in kiosks need a barcode reader to scan a passport.”	”The airline’s website needs to be available 98% of the time OR the app needs to be iOS compatible.”

Non-functional Requirements

(Also called quality of service requirements, quality factors, and the "-ilities")



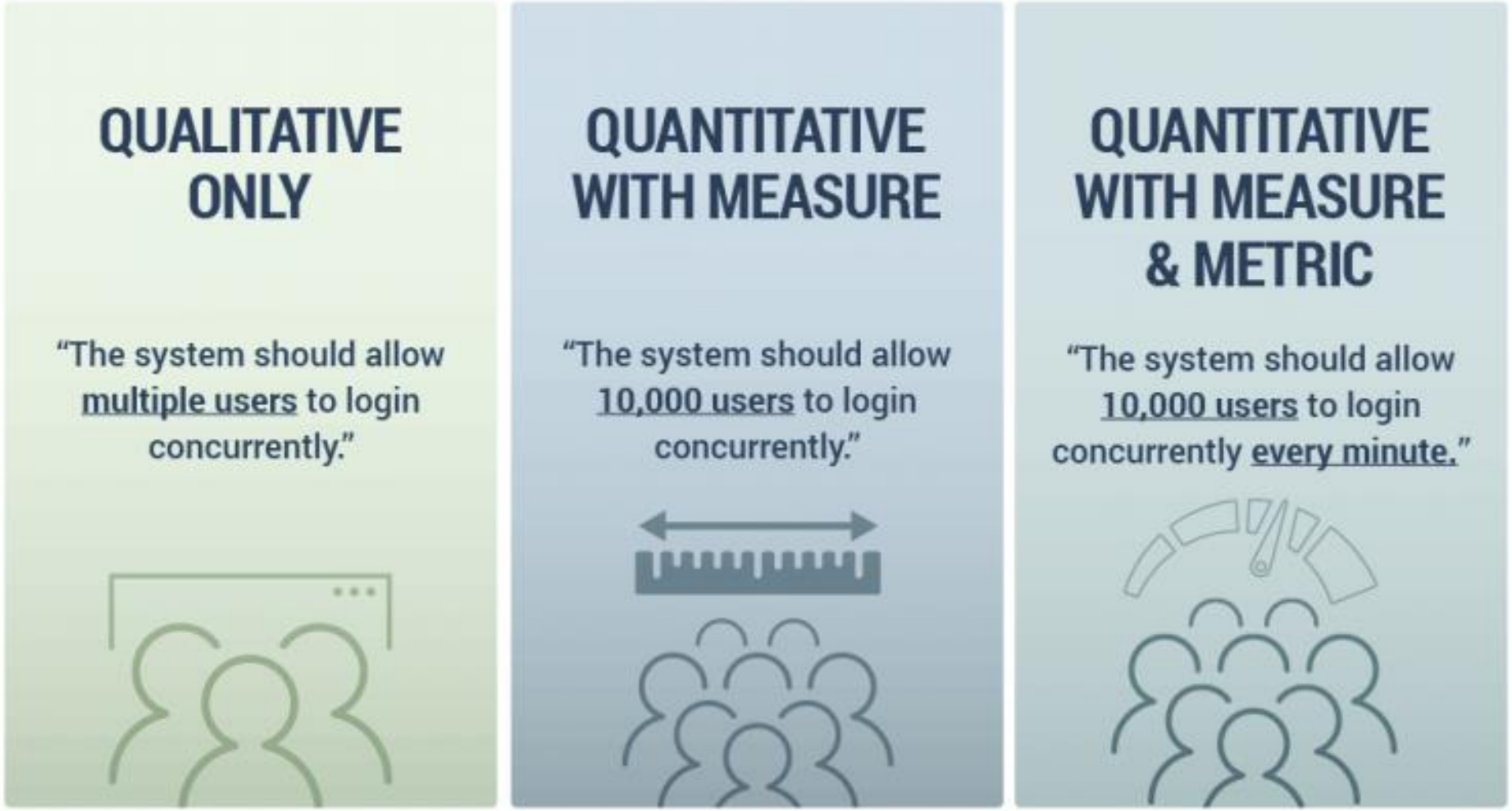
Definitions

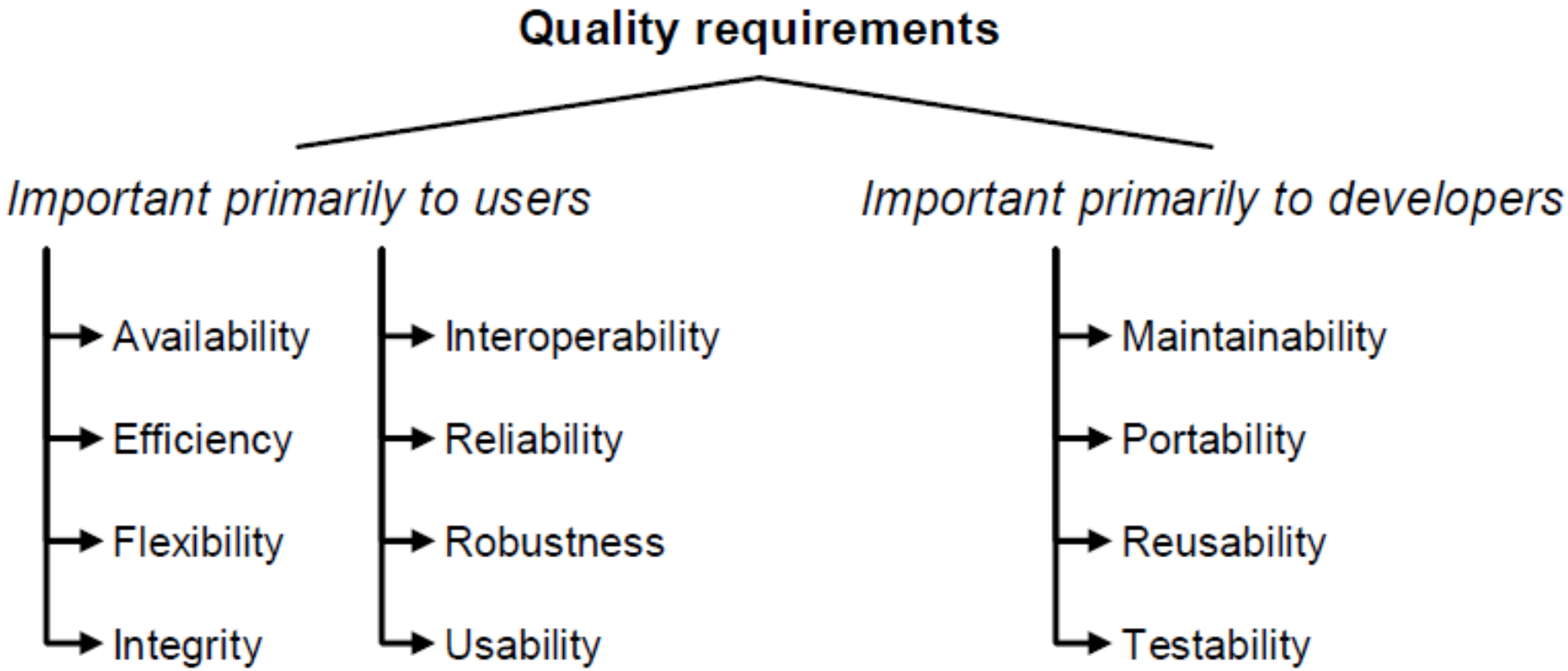
Quality Attributes	External Interfaces	Constraints
Describe the product's characteristics in various dimensions that are important either to users or to developers and maintainers (performance, safety, availability, compatibility, …etc.)	Describe the connections between the system and the outside world (other SW, HW, users, communication interfaces).	Impose restrictions on the options available to the developer during construction of the product (development process, documentation, programming language, cost and delivery …etc.)

Examples

Quality Attributes	External Interfaces	Constraints
"The flight search results should load within 5 seconds after the user hits the "Search" button".	"A link to the Tawakkalna system database needs to be established to verify the COVID-19 status of a passenger using his national ID number".	"The system development process and deliverable documents shall conform to the process and deliverables defined in MIL-STD-498".

Expression of Non-functional Requirements




Quality requirements
Important primarily to users
Important primarily to developers
Availability
Efficiency
Flexibility
Integrity
Interoperability
Reliability
Robustness
Usability
Maintainability
Portability
Reusability
Testability

- Product properties
- How should the system do?
- Helps verify performance.
- More difficult to capture.

- Product features
- What should the system do?
- Helps verify functionality.
- Easier to capture.

Non-Functional Requirements VS. Functional Requirements

"When sending welcome emails, the server must send them within 10 minutes of registration".

"When packing slips are printed, they must be on both sides of 5" x 8" sheets of white paper".

"When a site visitor creates an account, the server shall send a welcome email".

"When order status changes to fulfillment, the local printer shall print a packing slip".

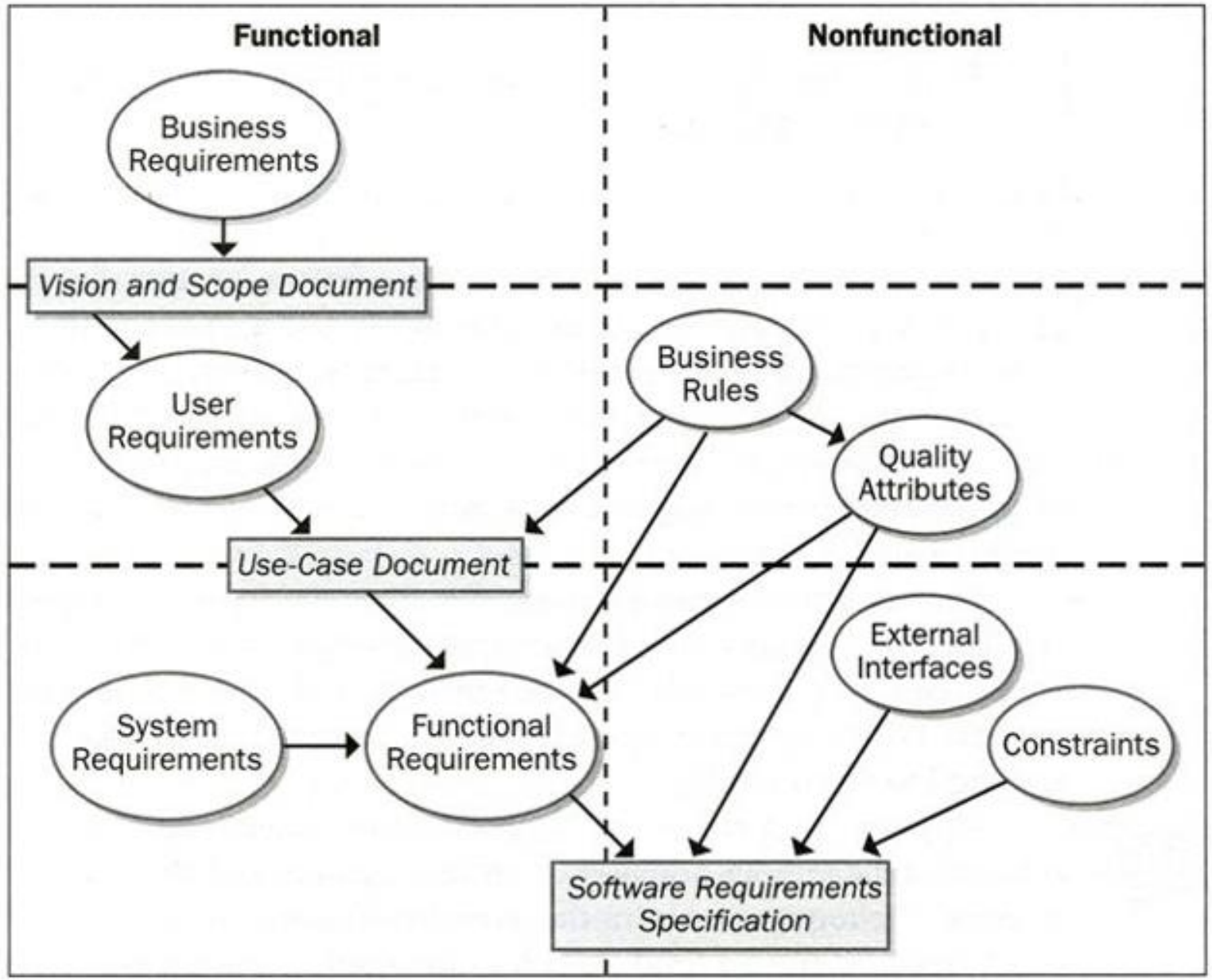
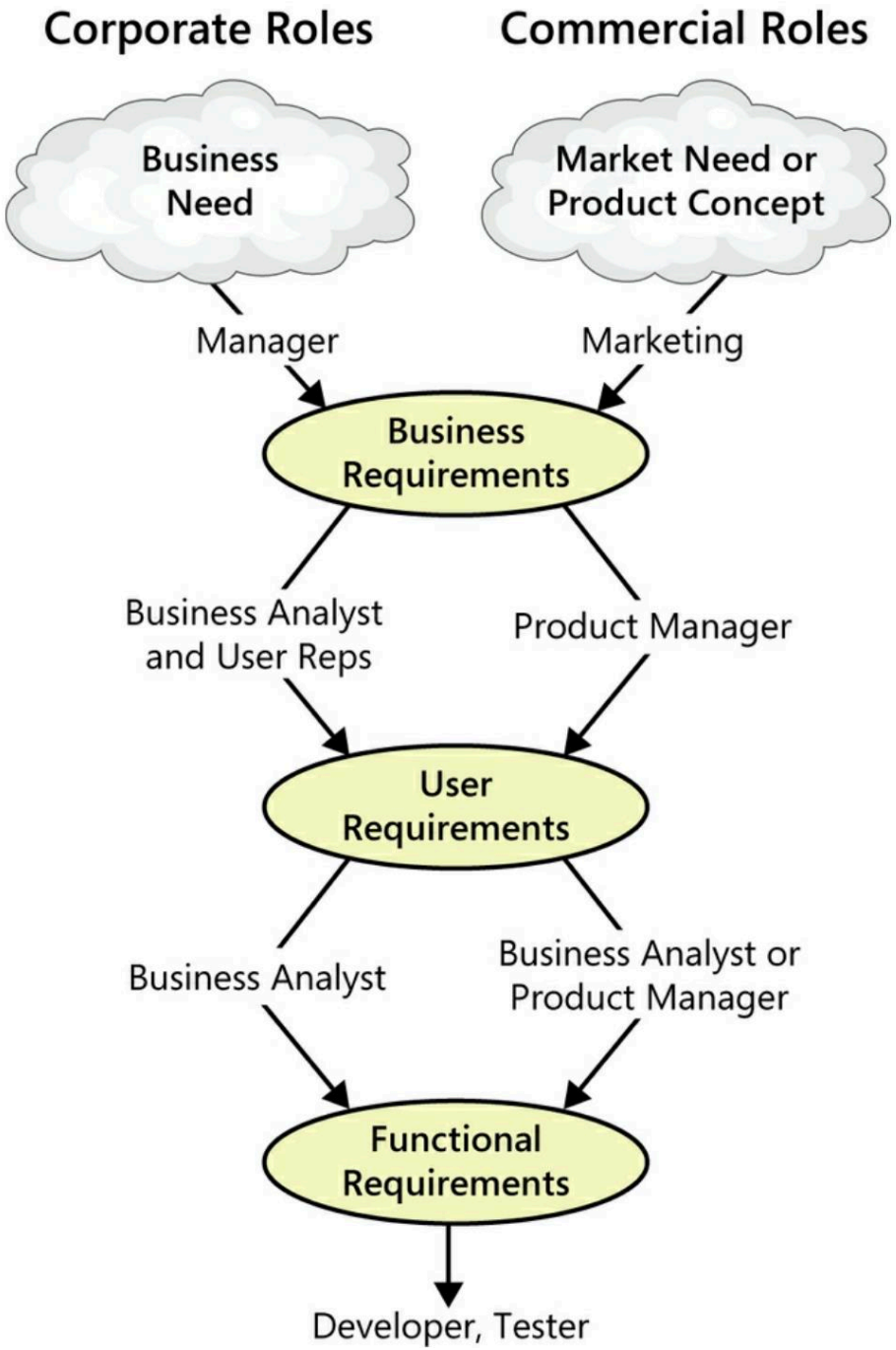


Figure 1-1 Relationship of several types of requirements information.

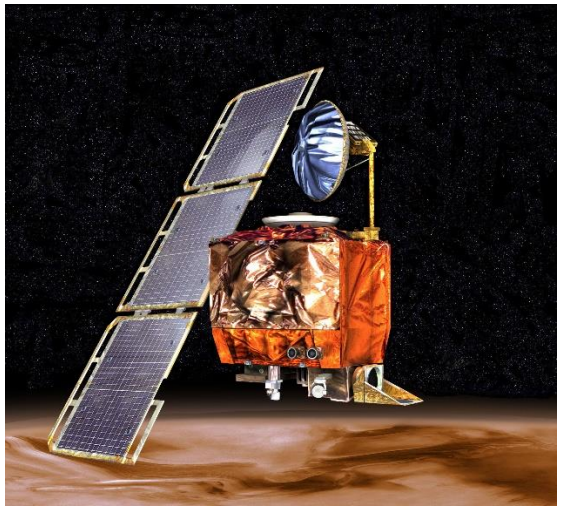

Corporate Roles
Commercial Roles
Business Need
Market Need or Product Concept
Manager
Marketing
Business Requirements
Business Analyst and User Reps
Product Manager
User Requirements
Business Analyst
Business Analyst or Product Manager
Functional Requirements
Developer, Tester

Domain Requirements

- Derived from application domain
- Describe system characteristics and features that reflect the domain
- May be new functional requirements or constraints on existing requirements
- Example: “A patient’s designated physician is the only entity allowed to retrieve a patient’s medical reports”.

Why RE?

Mars Climate Orbiter

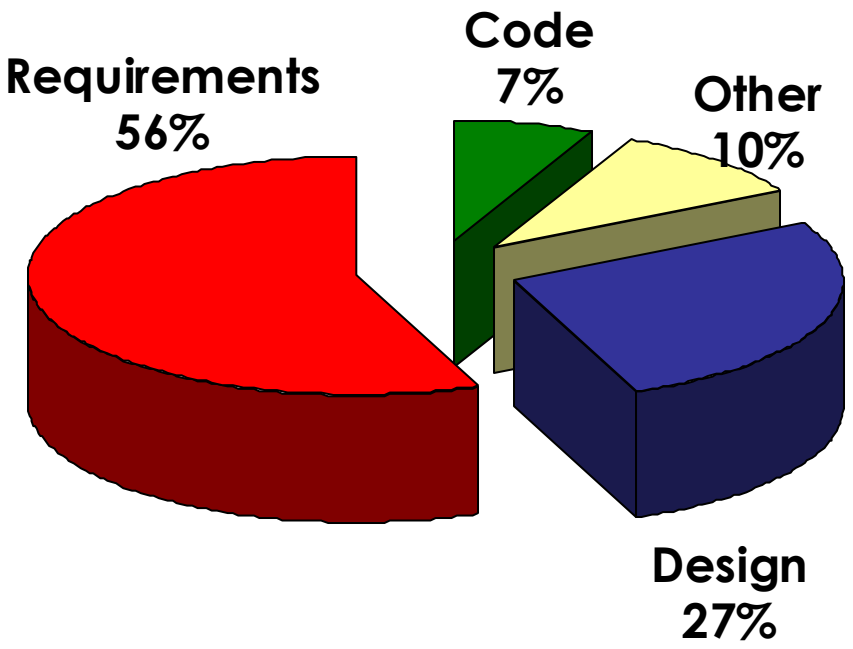


- Cost about $235.9M
- Launched by NASA in 1998 to study the Martian climate.
- It disappeared around Mars in 1999
- Failure caused by a measurement mismatch between two SW systems: the NASA team used the metric system, while the spacecraft builder Lockheed Martin used the English system for a key function

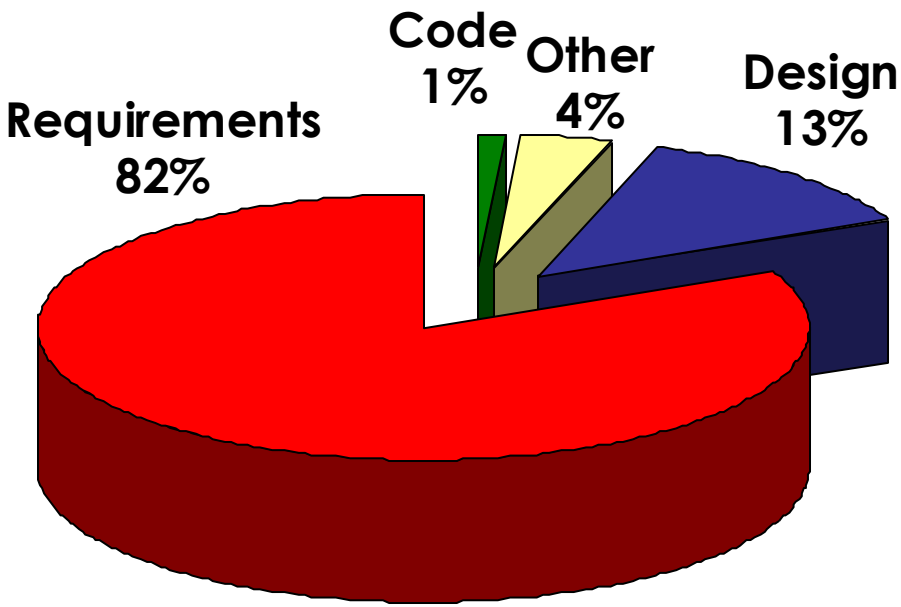
Integrated Resource Management Project (GIRES)

- GIRES (Gestion intégrée des ressources)
  - To replace >1000 existing systems in 140 organisations / departments
  - Affecting 68,000 employees
- The project started in 1998 with an estimated timeframe of 8 years and a budget of $80 million
- Could not cope with changes to the requirements and was cancelled in 2003 after investing
- Total cost: $400 million

Why Focus on RE?

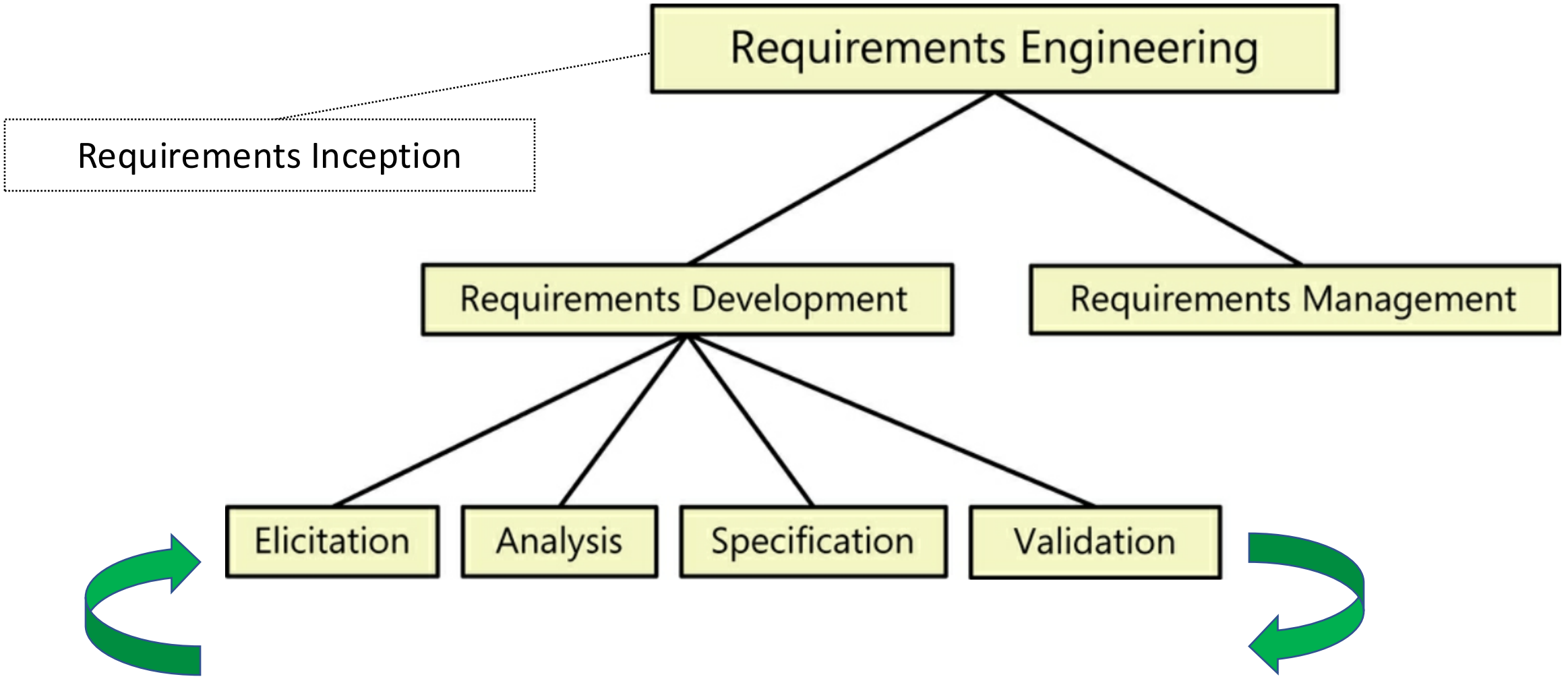


Distribution of Defects



Distribution of Effort to Fix Defects

RE Activities



RE Activity

Inception: Start the process (business need, market opportunity, great idea), business case, feasibility study, system scope, risks, etc.

Elicitation: All the tasks involved with discovering and gathering the requirements (interviews, document analysis)

Analysis: Develop a deep understating of the requirements on an individual level and as a whole (decomposition, conflicts, gaps)

Specification: Documenting these requirements in a well-organized way suitable for comprehension by their intended audiences.

Validation: Reviewing the documented requirements by internal teams and the customer to confirm that it's complete and properly represented and that it meets the customer needs.

Management: Managing the requirements beyond being agreed upon and through implementation up until the project is completed (tracing reqs, tracking status, changes – scope, assessing impact, accepting or refusing)

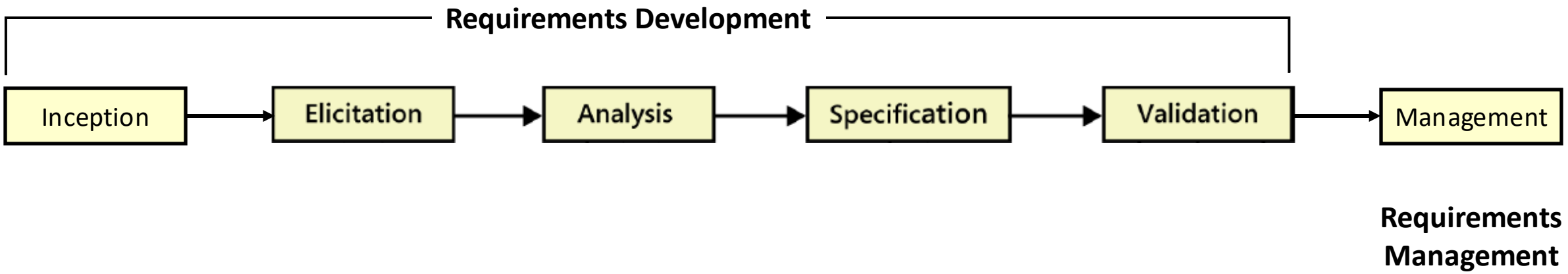
Regardless of what development life cycle is being used (waterfall, iterative, agile, …etc.) these are the thing you need to do regarding requirements.
Iteration is key to requirements development success.

Requirements Engineering (RE) Process

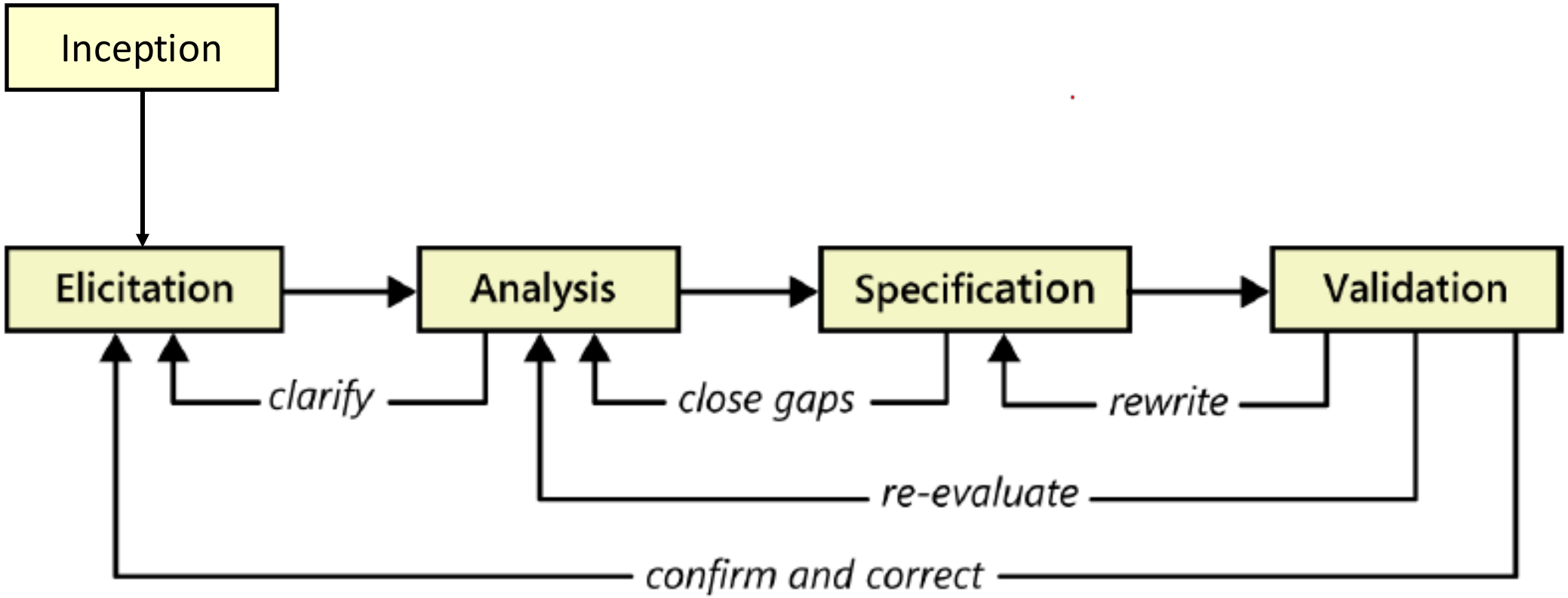
- A *process* is a sequence of steps performed for a given purpose (e.g. SW development process)

- RE is a process, not a discrete activity

RE Process

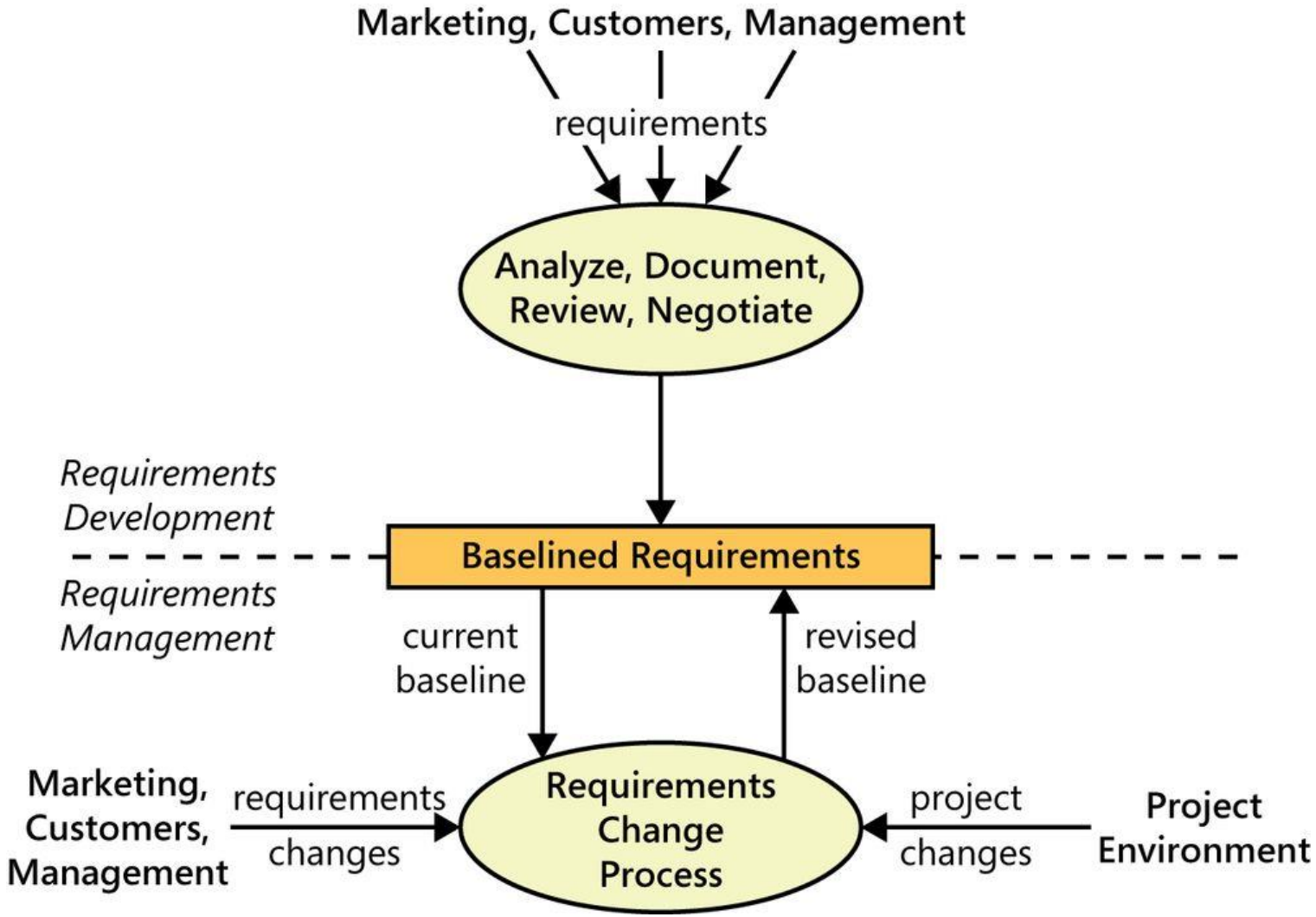

Requirements Development
Inception
Elicitation
Analysis
Specification
Validation
Management
Requirements Management

Requirements Development Process



"Progressive refinement of detail"

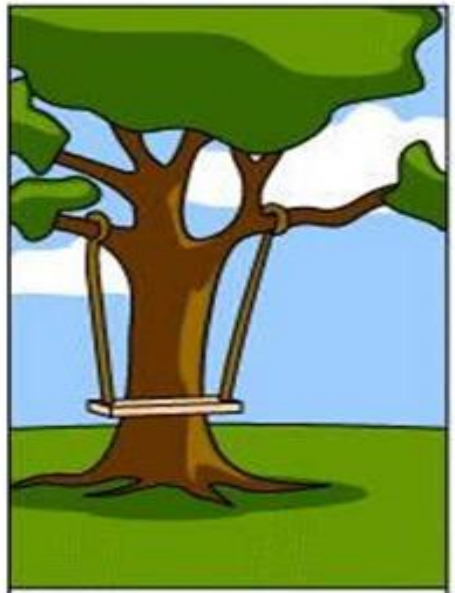
Boundary Between RD and RM



RE Challenges

- Lack of the right expertise
- Insufficient user involvement
- Overlooked stakeholders
- Unrealistic or unclear goals
- Ambiguous requirements
- Conflicting requirements
- Creeping user requirements
- Changing requirements

How the customer explained it

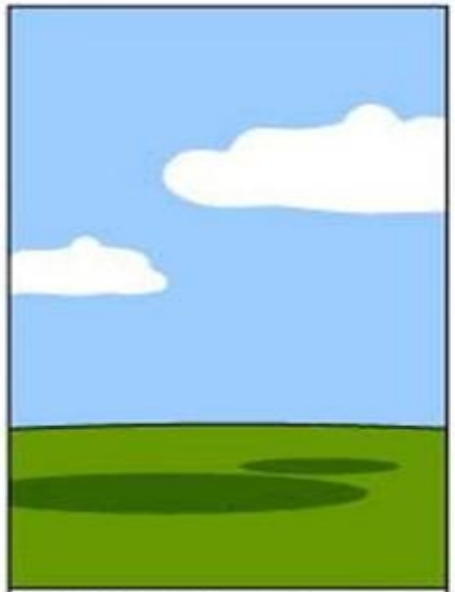


How the Project Leader understood it

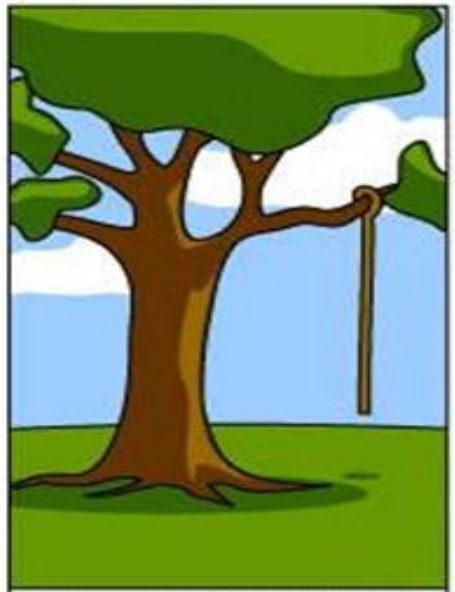
How the Analyst designed it

How the Programmer wrote it

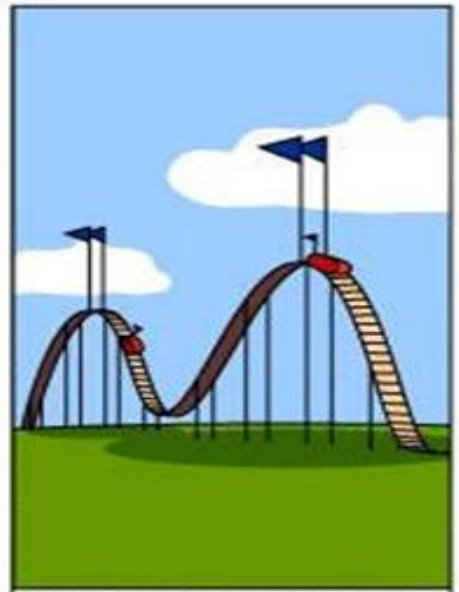
How the Business Consultant described it



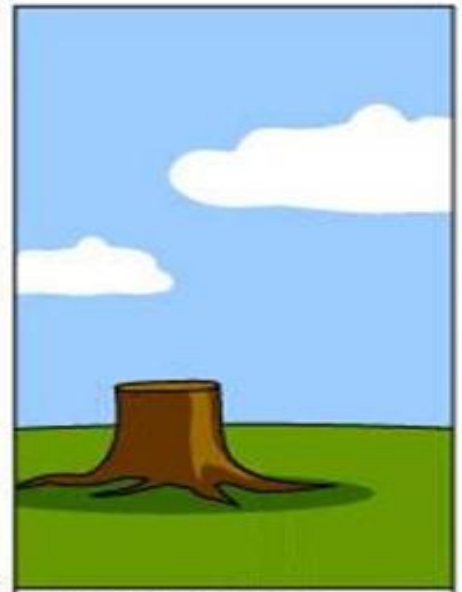
How the project was documented



What operations installed



How the customer was billed



How it was supported

What the customer really needed