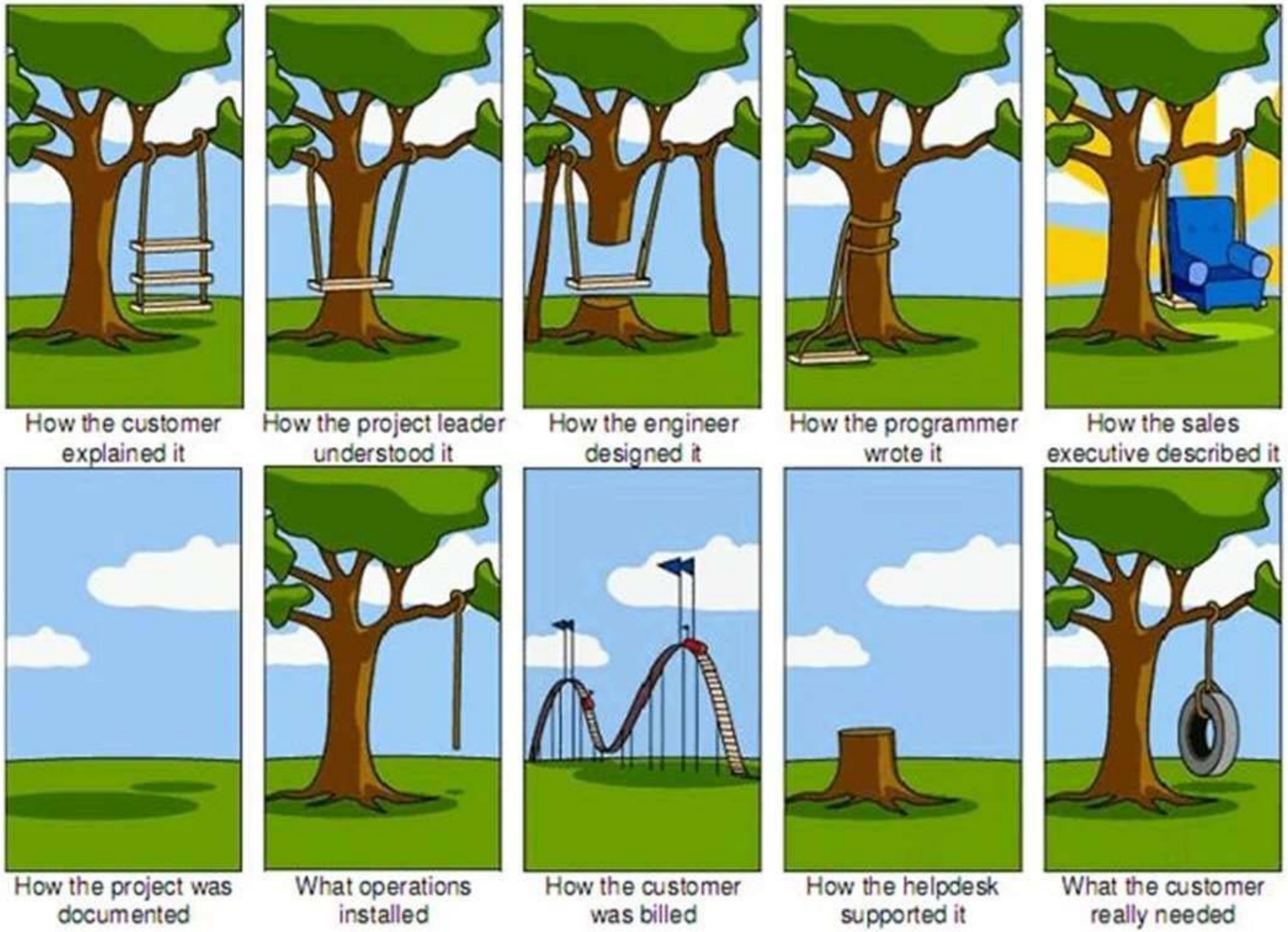
1. Requirements Engineering - Overview


I'LL GO UP AND FIND OUT WHAT THEY NEED AND THE REST OF YOU START CODING !
SHELTON


How the customer explained it
How the project leader understood it
How the engineer designed it
How the programmer wrote it
How the sales executive described it
How the project was documented
What operations installed
How the customer was billed
How the helpdesk supported it
What the customer really needed

Requirements Engineering

Quality means fitness-for-purpose. Cannot say anything about quality unless you understand the purpose.

Communication is as important as the analysis.

Not a phase or stage

- **Requirements Engineering (RE)** is a set of activities concerned with identifying and communicating the purpose of a software-intensive system, and the contexts in which it will be used. Hence, RE acts as the bridge between the real world needs of users, customers, and other constituencies affected by a software system, and the capabilities and opportunities afforded by software-intensive technologies

and partly about what is possible

Designers need to know how and where the system will be used.

Requirements are partly about what is needed.

Need to identify all the stakeholders - not just the customer and user

RE (Requirements Engineering)

- **Requirements engineering** is the **process** of establishing
  - **System services** that the customer requires from a system and
  - **Constraints** under which it operates and is developed.

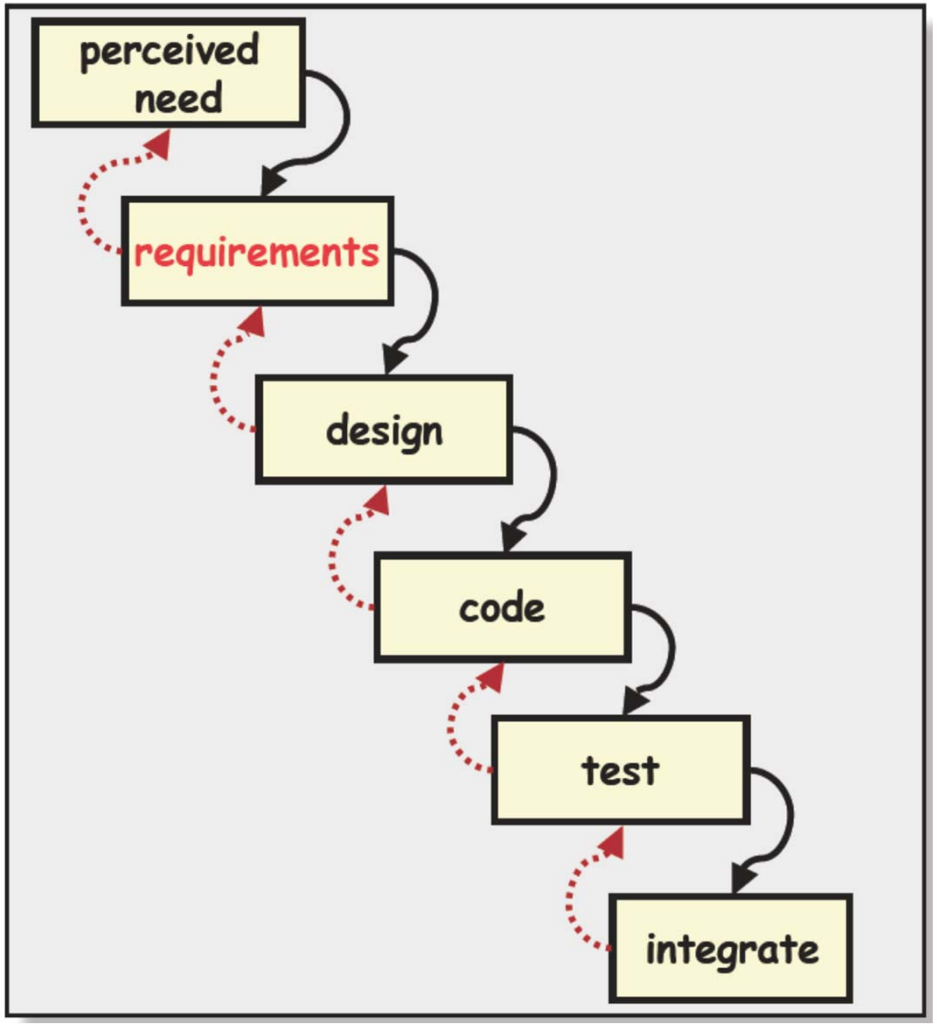
- **Requirements** are
  - **Descriptions** of the **system services** and **constraints**, generated from the RE processes.
    - User-level facility descriptions
    - Detailed specifications of expected system behavior
    - A general system properties
    - Specific constraints on the system
    - Information on how to carry out some computation
    - Constraints on the development of the system
  - **System services** → Functional requirements
  - **Constraints** → Non-functional requirements

SDLC and RE Process

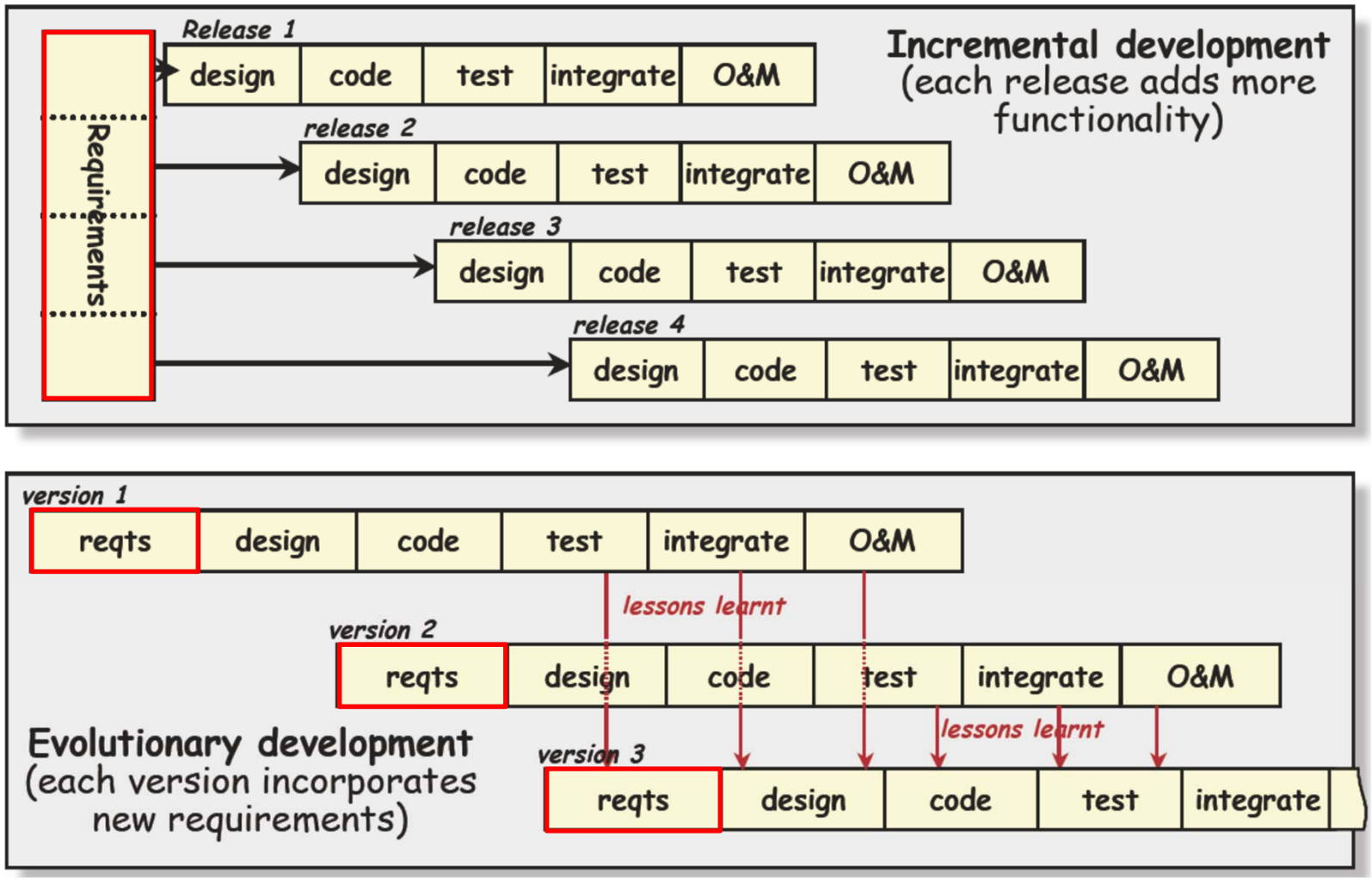
- Requirements engineering process should be adapted to a specific SDLC.
  - **RE process + Development process**

- **Software development lifecycle (SDLC)** models
  - Waterfall
  - Incremental, Evolutionary
  - Spiral, Iterative, Agile, RUP

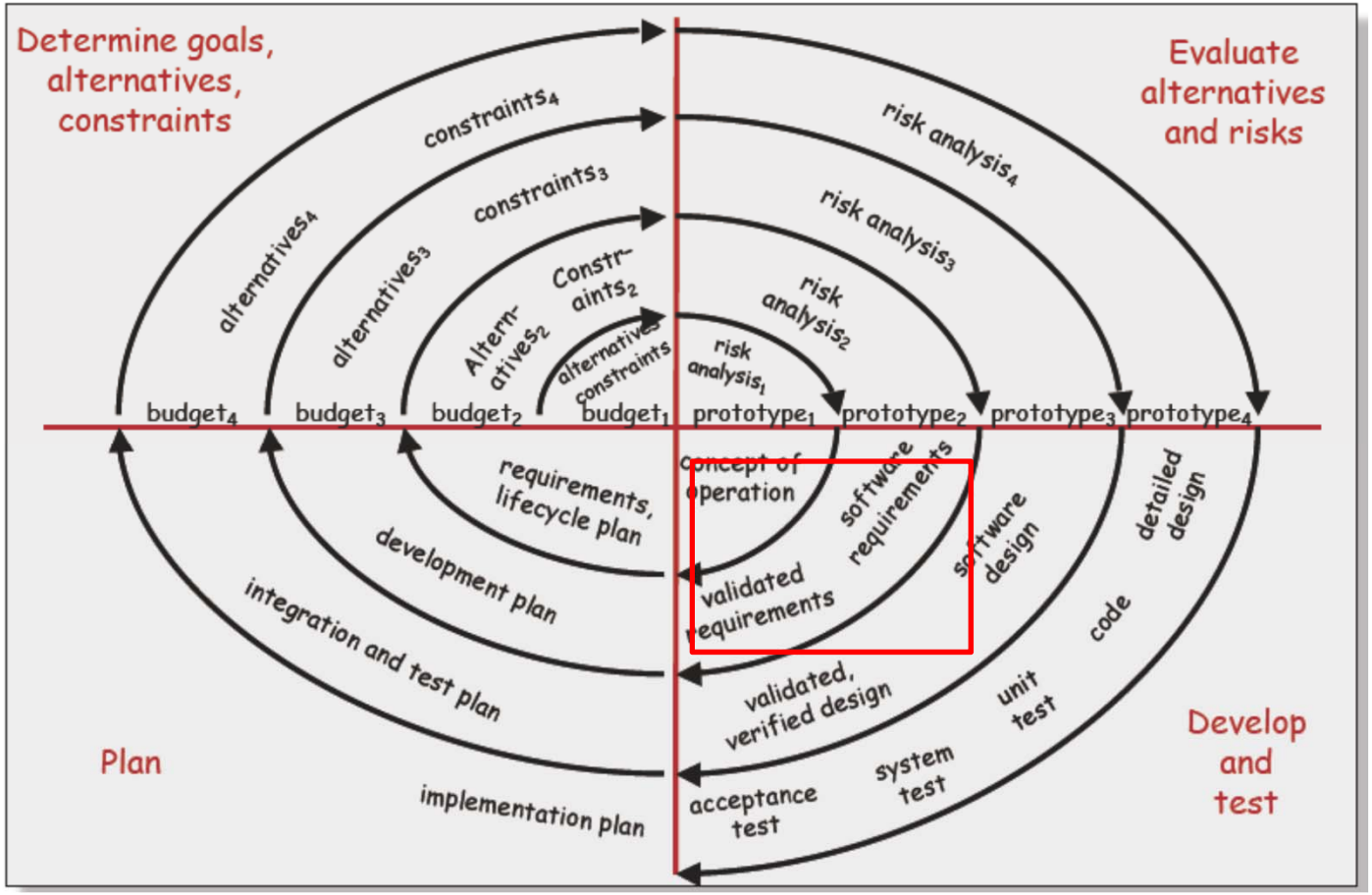
Waterfall Model



Phased Lifecycle Models

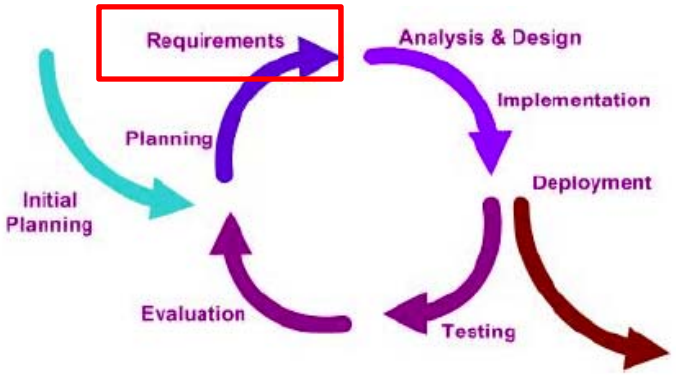


The Spiral Model

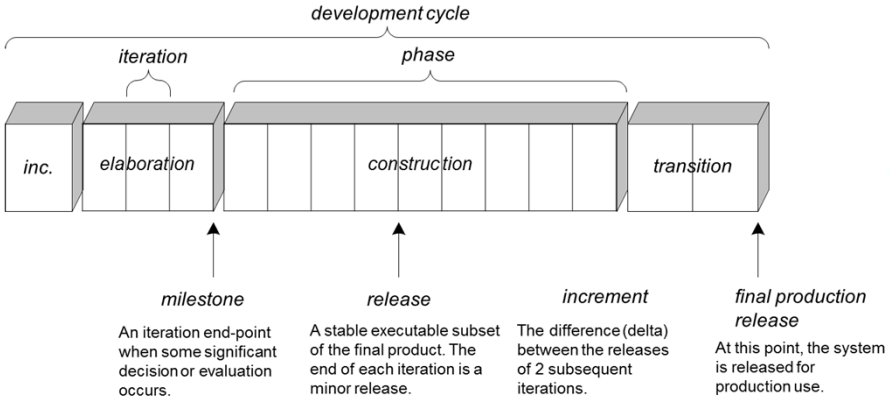


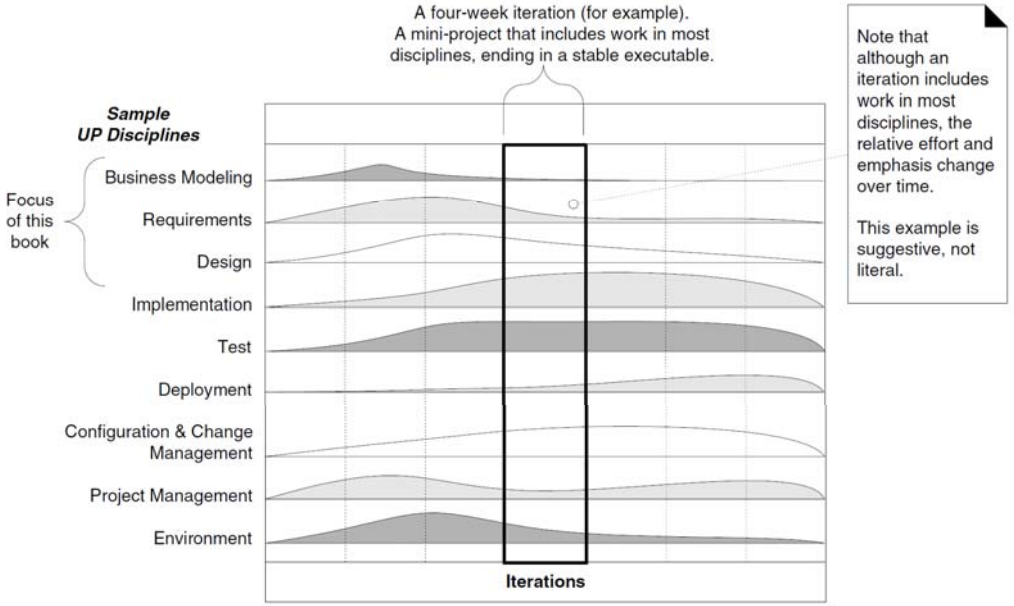
Agile Models and RUP

- **Basic Philosophy of Agile**
  - Individual over processes and tools
  - Working software over documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan



- Evolved into **RUP** **(Rational Unified Process)**

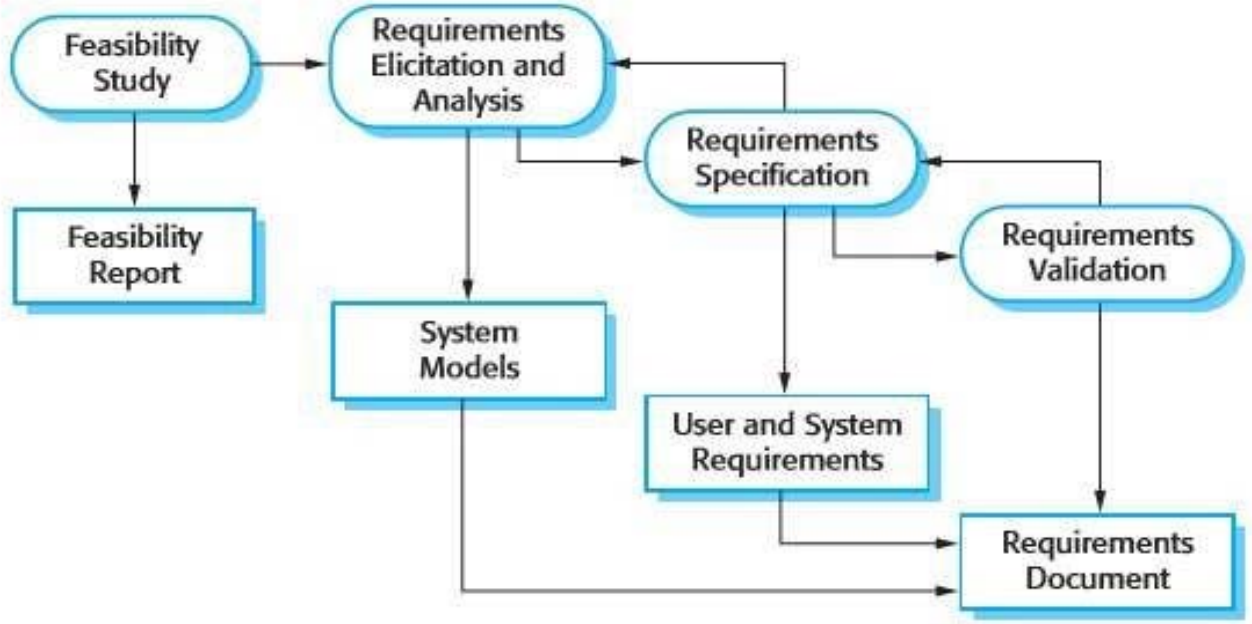




Requirements Engineering Processes

- **Requirement engineering processes** vary depending on
  - Application(target) domain
  - People involved
  - Organization developing the requirements
  - Software development processes used

- **Generic activities** common to all RE processes :



1. Feasibility Study

- Decides whether or not the proposed system is worth to develop

- **A short focused study** to check
  - ***"If the system contributes to organizational objectives"***
  - ***"If the system can be engineered using current technology and within budget"***
  - ***"If the system can be integrated with other systems that are used"***

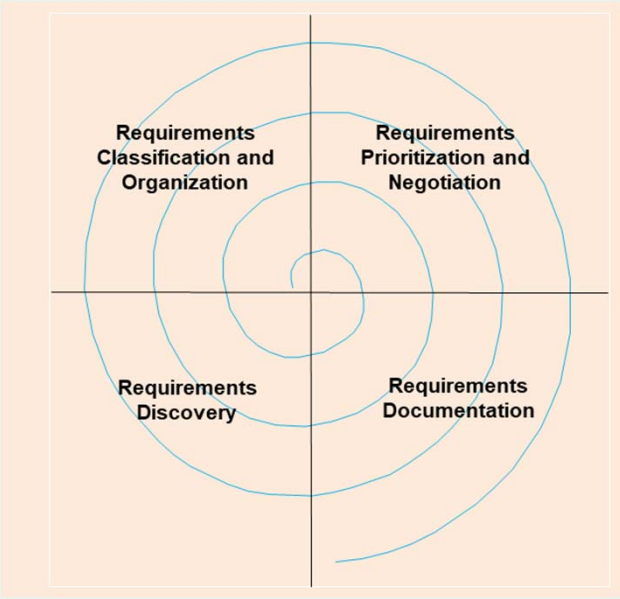
- **Questions**
  - What if the system was not implemented?
  - What are the problems in the current process?
  - How will the proposed system help to satisfy customer's requirements?
  - What will be the integration problems?
  - Is new technology needed? What skills?
  - What facilities must be supported by the proposed system?

2. Requirements Elicitation and Analysis

- Called also **Requirements Discovery** to find out
  - Application domain, services that the system should provide
  - System's operational constraints

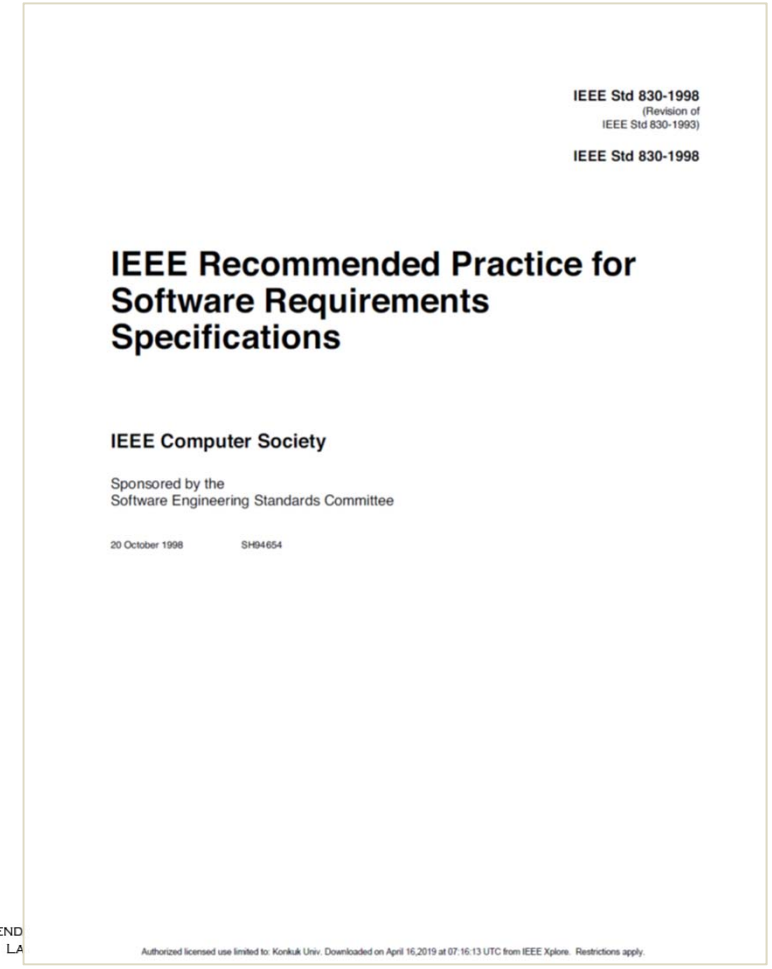
- Should involve various stakeholders
  - End-users, managers, engineers, domain experts, trade unions, etc.

- 4 activities performed iteratively
  - **Requirements discovery**
  - **Requirements classification and organization**
  - **Prioritization and negotiation**
  - **Requirements documentation**



3. Requirements Specification

- Write elicited, analyzed, negotiated, prioritized and selected requirements into **documents** according to the **IEEE 830-1998 Standard**



IEEE Std 830-1998
(Revision of
IEEE Std 830-1993)

IEEE Std 830-1998

IEEE Recommended Practice for Software Requirements Specifications

IEEE Computer Society

Sponsored by the
Software Engineering Standards Committee

20 October 1998 SH94654

Table of Contents

Figure 1—Prototype SRS outline

4. Requirements Validation

- Demonstrate whether the requirements we defined are **what the customer really wants**

- **Requirements validation checks:**
  - Validity : *Does the system provide the functions which support the customer's needs well?*
  - Consistency : *Are there any requirements conflicts?*
  - Completeness : *Are all functions required by the customer included?*
  - Realism : *Can the requirements be implemented with available budget and technology?*
  - Verifiability : *Can the requirements be checked?*

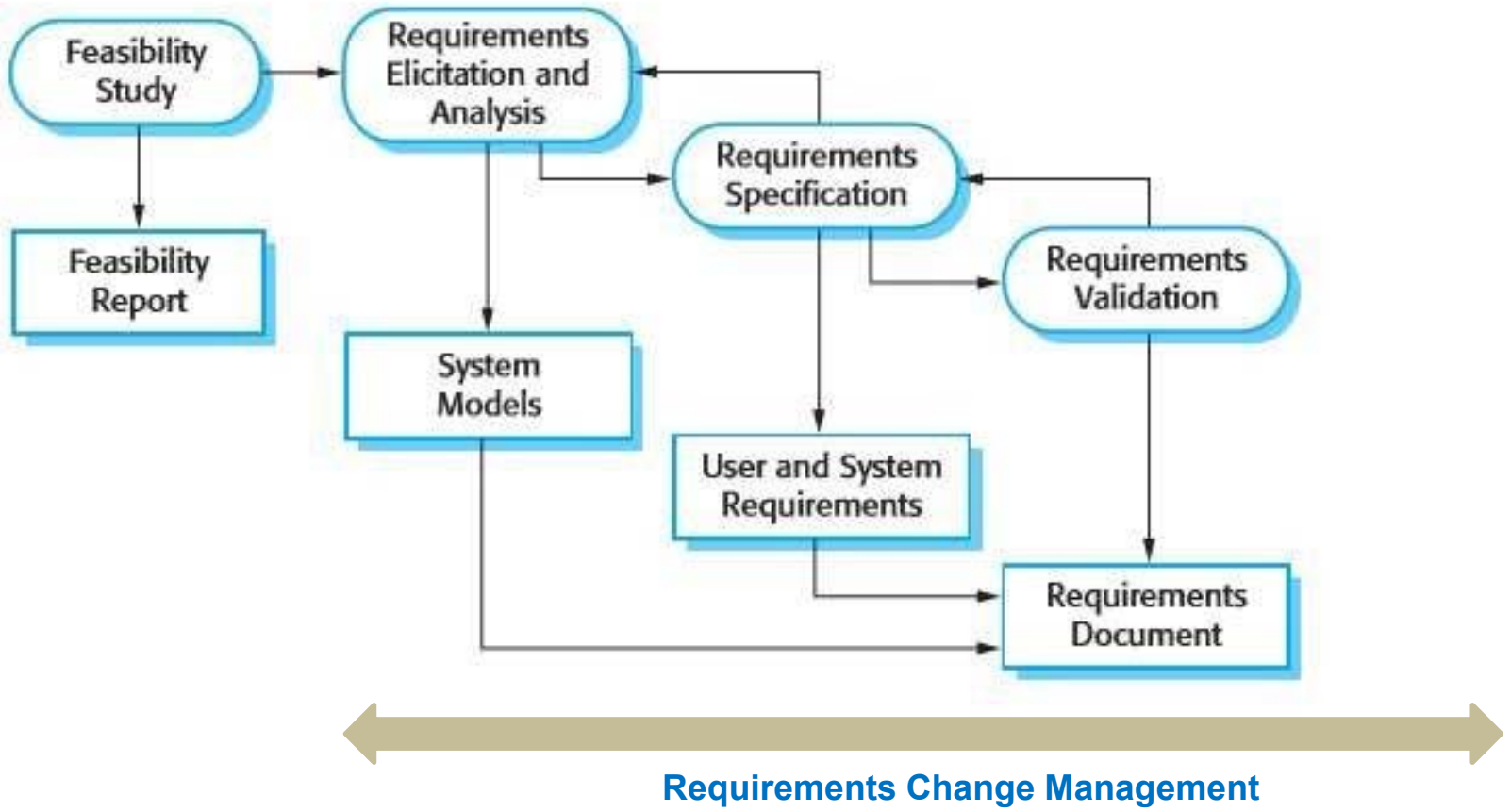
- **Requirements validation tools:**
  - Requirements reviews
  - Prototyping
  - Test case generation

5. Requirements Change Management

- The process of **managing requirements change** during the RE process and the overall system development
  - Requirements are inevitably incomplete and inconsistent.
  - New requirements emerge during the process, as business needs change and a better understanding of the system is developed.

- **Traceability** is the heart of requirements management.
  - Source ↔ Requirements ↔ Design ↔ Code

Requirements Engineering Process



Exercise 1: Requirements Practices at Workplace

- **Let's read the paper below and discuss on the requirements practice at your workplace.**

Requirements Eng (2006) 11: 1–3
DOI 10.1007/s00766-004-0206-4

VIEWPOINTS

Alan M. Davis · Didar Zowghi

Good requirements practices are neither necessary nor sufficient

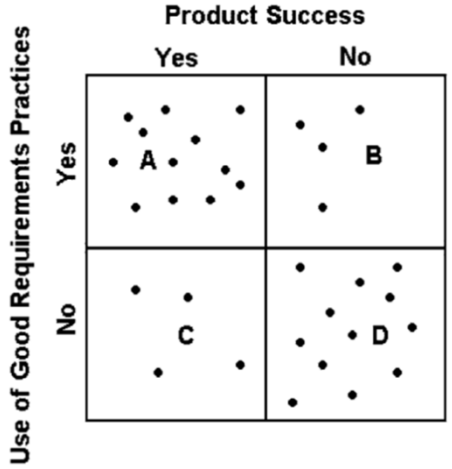


Fig. 1 Relationship of good requirements practices to success

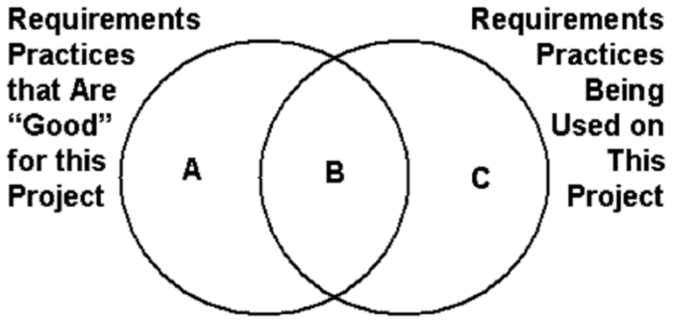


Fig. 2 Relationship of good requirements practices to success