Week 2 – Requirement engineering Schedule

- What / Who / Why … requirement engineering
- Serious game: about requirement elicitation
- Serious game: try to do it
- More about Requirement specification

Serious Game: CyberVideo

- Teams of 4-5 students
  - 1 Client
  - 3 or 4 providers
- Starting document
  - Providers: Call for tender (few lines)
  - Client: informal description (3 pages)
- Objectives Providers:
  - Ask client question in order to find all the requirements described in the informal description
  - Propose a specification document

Serious Game: CyberVideo

Providers:

- What elicitation method did you use?
- Did you chose it on purpose?
- Could you have done differently? How?

- What was difficult?
- How you are going to write the specification?

Schedule

- What / Who / Why … requirement engineering
- Serious game
- More about Requirement specification
- Exercises / Homework
  - Express requirements for CyberVideo
  - Read (and learn) about requirement elicitation

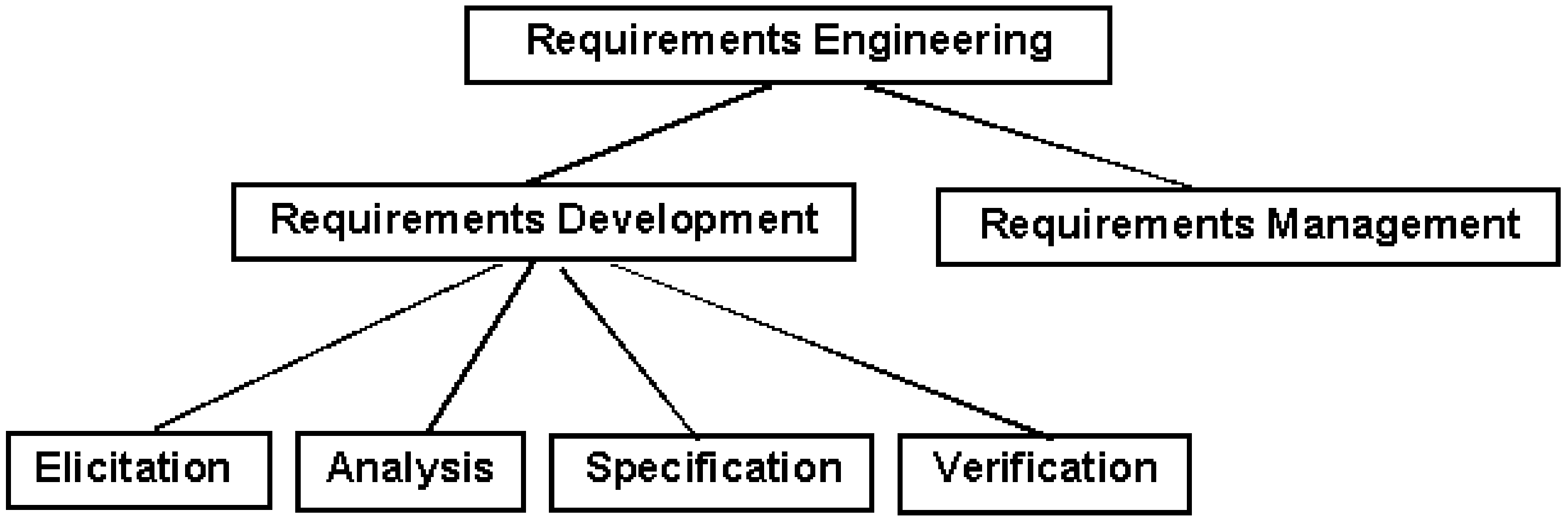


Figure 2. Subdisciplines of requirements engineering.

How is organized Requirement Engineering?

> Requirements development

REQUIREMENT SPECIFICATION

What is a requirement?

- Express what customers want or need
  - A « **need** » is something **mandatory** that we must have
  - A « **want** » is **nice** to have but not always mandatory
- A requirement can be
  - A goal
  - A provided function
  - A quality
  - A property (domain, organization)
  - A constraint

Functional requirements

- **Services provided** by the system
  - Description of the expected function or behavior
  - A general property
  - Expected UI
- Example (**Software library**)
  - The software has to manage books borrowing
  - The checkout function starts by reading the subscriber card
  - A subscriber has to pay 20 euros per year
  - A subscriber is defined by his name, age, etc.
  - All needed information have to be displayed in a single window

Functional requirements – Davis, 93

- An **object**
  - A **client** is identified by his name, age and address
- A **functionality**
  - A client can **borrow** up to 5 books
- A **state**
  - A book is **available, borrowed or lost**

- Several of them together

Objects – Davis, 93

- **Entity** clearly identified
  - Concept related to the software
- Requirements specify the objects:
  - Name and meaning
  - Structure
  - Scope
- Library Example
  - A **client** is identified by his **name**, **age** and **address**
  - A **book** is defined by its **title** and **author(s)**

Functions – Davis, 93

- **Activity** clearly defined in the domain
  - Tasks, services, processes
  - Related to the software
- Requirements specify the functions:
  - Name
  - Interface, data
  - Behavior
  - Demanded ressources
- Library Example
  - **Book checkout**: to borrow a book, the client has first to show his library card and then the book

State – Davis, 93

- Characterize the situation of an entity
  - Can be expressed as a predicate
  - Can change over the time
  - Influence the behavior of the entity
- Requirements specify the states
  - All possible states
  - Transitions
  - Possible properties
- Library Example
  - A book is **available, borrowed or lost**

Objects, functions, states

- Requirements may established **relations** among **objects**, **functions** and **states**
  - A **client** can **borrow** a book when he has paid his bill and has less than 5 **borrowed** **books**
- Analysis methods focuses on single aspect
  - Object
  - Function
  - States

Capturing functional requirements Example of questions – Pfleeger

- Functions
  - What does the system should do?
  - When?
  - Several functional modes?
  - Appropriate responses to stimuli ?
- Data
  - Format?
  - How long should be?

Non-functional requirements

Constraints

A constraint under which a software operates or is developed

Process

- Tools
- Standards

Domain

- Usage
- Regulation / Law

Development

- COTS (OS, middleware, …)
- Methods

Context

- Existing applications
- People

Non-functional requirements

Constraints

- Example (Software library)
  - UML **must** be used for the modeling phase (at design)
  - An architectural design document **must** be provided

Process

  - The system **must** use Oracle for persistency functions
  - Java annotations **may** be used for development

Development

  - No historic is maintained for subscribers

Domain

  - The system **must** interface with legacy systems

Context

Non-functional requirements
Qualities

- **External** or **internal** qualities of
  - the provided functions
  - the global system
- **Includes**
  - Security, Logging
  - Storage, Configuration
  - Performance, Cost
  - Interoperability, Flexibility
  - Accessibility, Disaster recovery
- Must be **quantified** (to be evaluated)

Non-functional requirements

Qualities

- What about… ?
  - “The system should be easy to use”
  - “The system should be robust and quick”

Non-functional requirements
Qualities

- What about… ?
  - “The system should be easy to use”
  - “The system should be robust and quick”

- Different interpretation
- Source of conflict

=> Need to quantify the NF-requirements

Non-functional requirements
Qualities

- Example (**Software library**)
  - Book checkout must be made **in less than 1 minute**
  - Any function must be **done in less than 2 minutes**
  - Backtracking must **always** be possible when borrowing a book
  - The software system must be **available 6 days a week**

Example

- First formulation
  - The system should be easy to use for a experimented user and should be organized to limit the number of errors.
- Second formulation
  - A user with 5 years of experience should be able to use the system after 2 hour long formation.
  - After the formation, the average number of errors made by a user should not be more than 2 by day.

Performance

- Quality perceived by **users** (external)
- Requirements to be specified
  - Number of transactions per second
  - Arrival rate of inputs
  - Refreshing time
  - Response time for a given pattern of events
  - What to do when expected quantities are exceeded
    - Failure, ignorance of additional inputs, degraded services

Usability

- Quality perceived by **users** (external)
- Requirements to be specified
  - Provided UI
  - Error messages
  - keyboard shortcut
  - Backtrack possibilities
  - Techniques to help users and to improve confidence
  - Amount of expected training
  - Facilities to avoid misuses

Availability & Reliability

- Quality perceived by **users** (external)
- Requirements to be specified
  - Max. number of bug per Kline during integration
  - Min. duration without a problem
  - Is there a time to perform maintenance?
  - Maximum time allowed for restarting the system
  - Must backup copies be stored at a different location?
  - Must the system detect and isolate fault
- Difficult to assess !

Security

- Quality perceived by **users** (external)
- Requirements to be specified
  - Must access to the system or information be controlled?
  - Should each user’s data isolated from the data of each other?
  - Should user programs be isolated from other programs and from the operating system?
  - Should precautions be taken against theft or vandalism

Maintenability

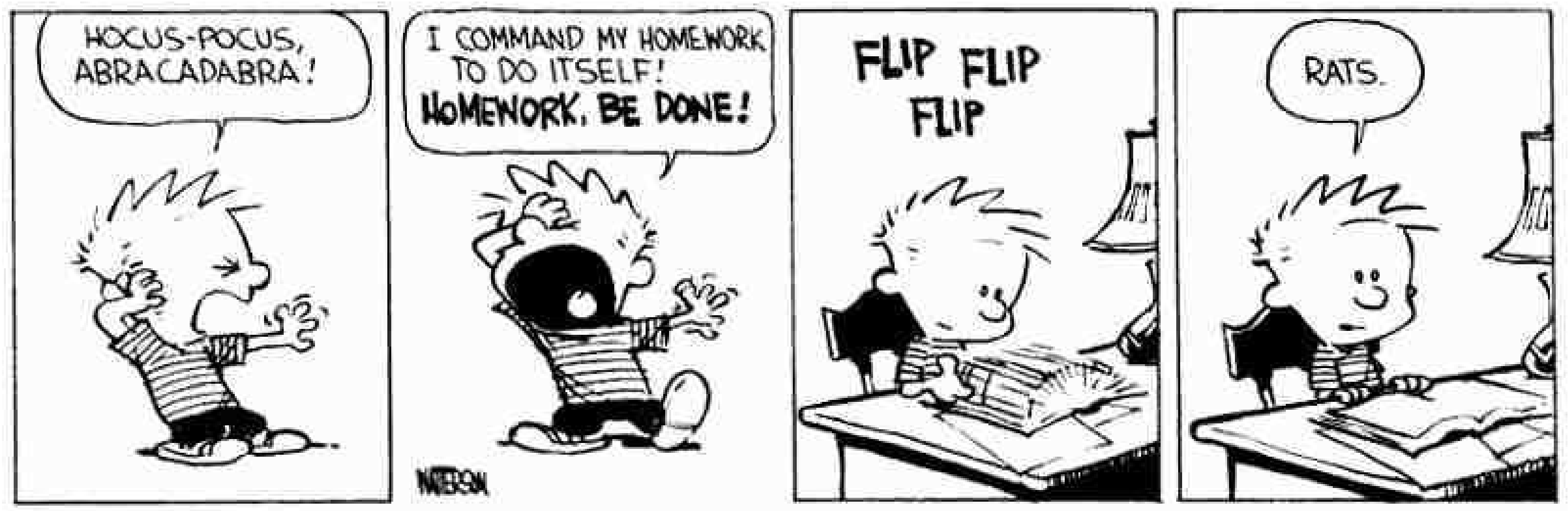
- Quality perceived by **engineers** (internal)
- Requirements to be specified
  - When and in what ways might the system be changed in the future?
  - How easy should it be to add features to the system?
  - How easy should it be to port the system from one platform (computer, OS) to another

Schedule

- What / Who / Why … requirement engineering
- Serious game
- More about Requirement specification
- Exercises
  - Express requirements for CyberVideo
  - Read (and learn) about requirement elicitation

Exercises/HomeWork

- Express some requirements about CyberVideo
  - Functional
  - Non-Functional


HOCUS-POCUS, ABRACADABRA!
I COMMAND MY HOMEWORK TO DO ITSELF! HOMEWORK, BE DONE!
WATERSON
FLIP FLIP FLIP
RATS.

Exercise

- Read and learn about requirement elicitation
- At the end of the work, you should be able to
  - cite several elicitation methods
  - explain the principle of a given elicitation method
  - give advantages/limits of a given elicitation method

For the final evaluation

- You should know
  - Challenges and issues of requirement engineering
  - Advantages and limits of the different requirement elicitation methods
- You should be able to
  - Identify incorrect formulation of NF requirement
  - Propose an alternative formulation