Development processes – Requirement Analysis

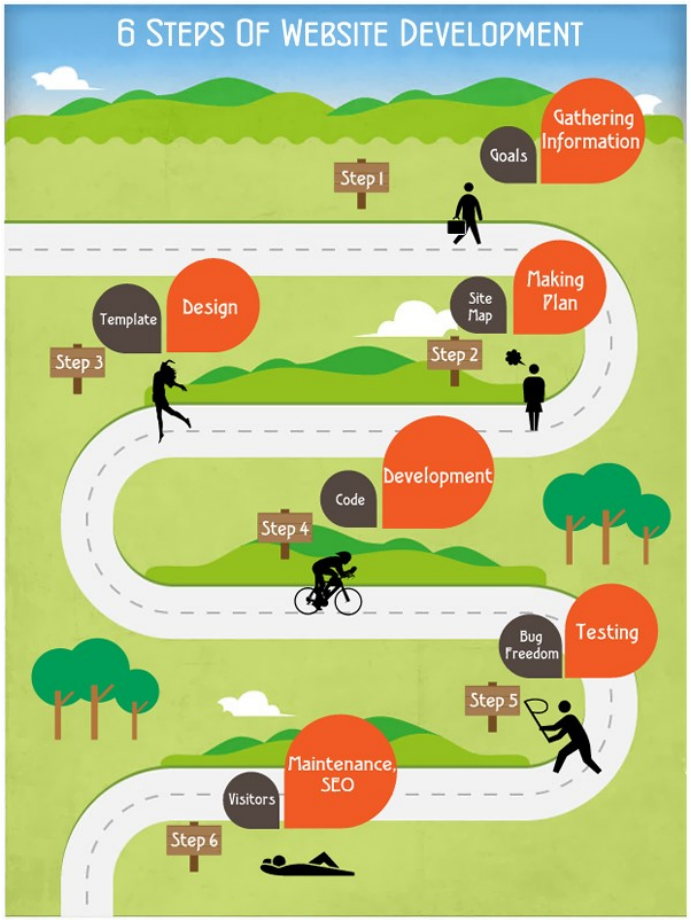
School of Computer Science | Software Analysis &Design
Marta Kristín Lárusdóttir, Assistant professor
Hannes Pétursson, Lecturer

Content

- Introducing development processes
- The reasons for why we analyze and design software
- A typical requirement list

- Reading:
  - Chapter 1, cp 2 (p. 33 – 42)

Introducing Development Processes

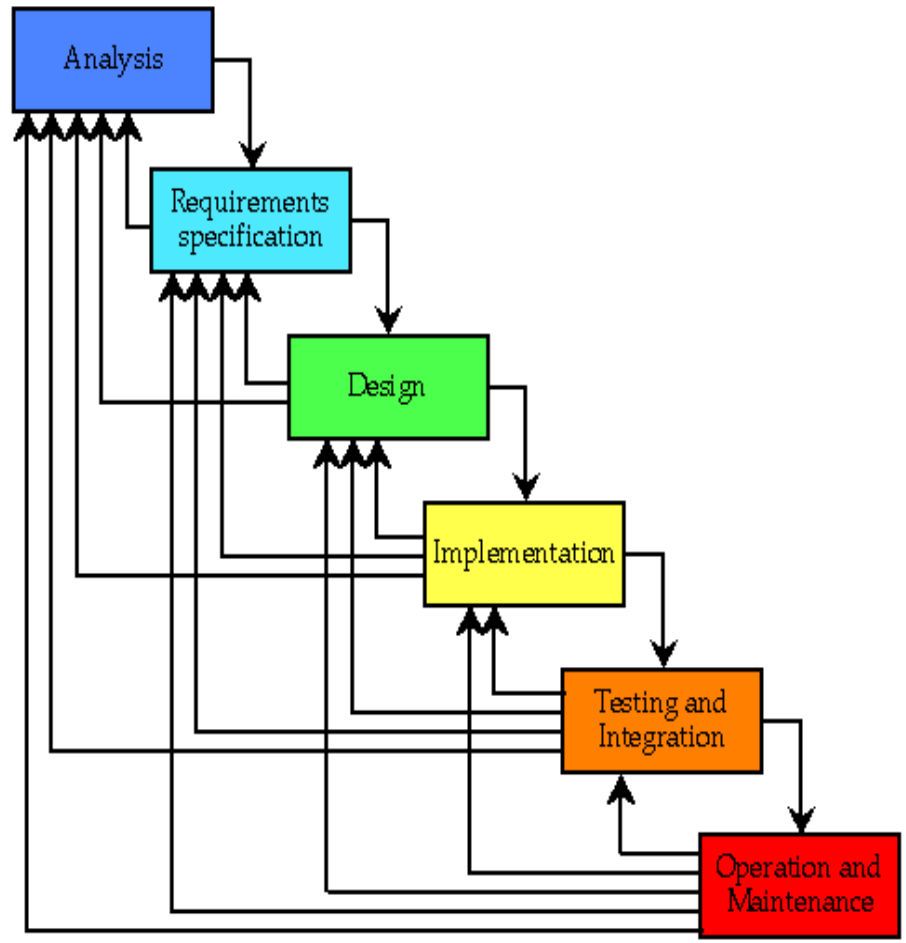


What are the steps of the development process?

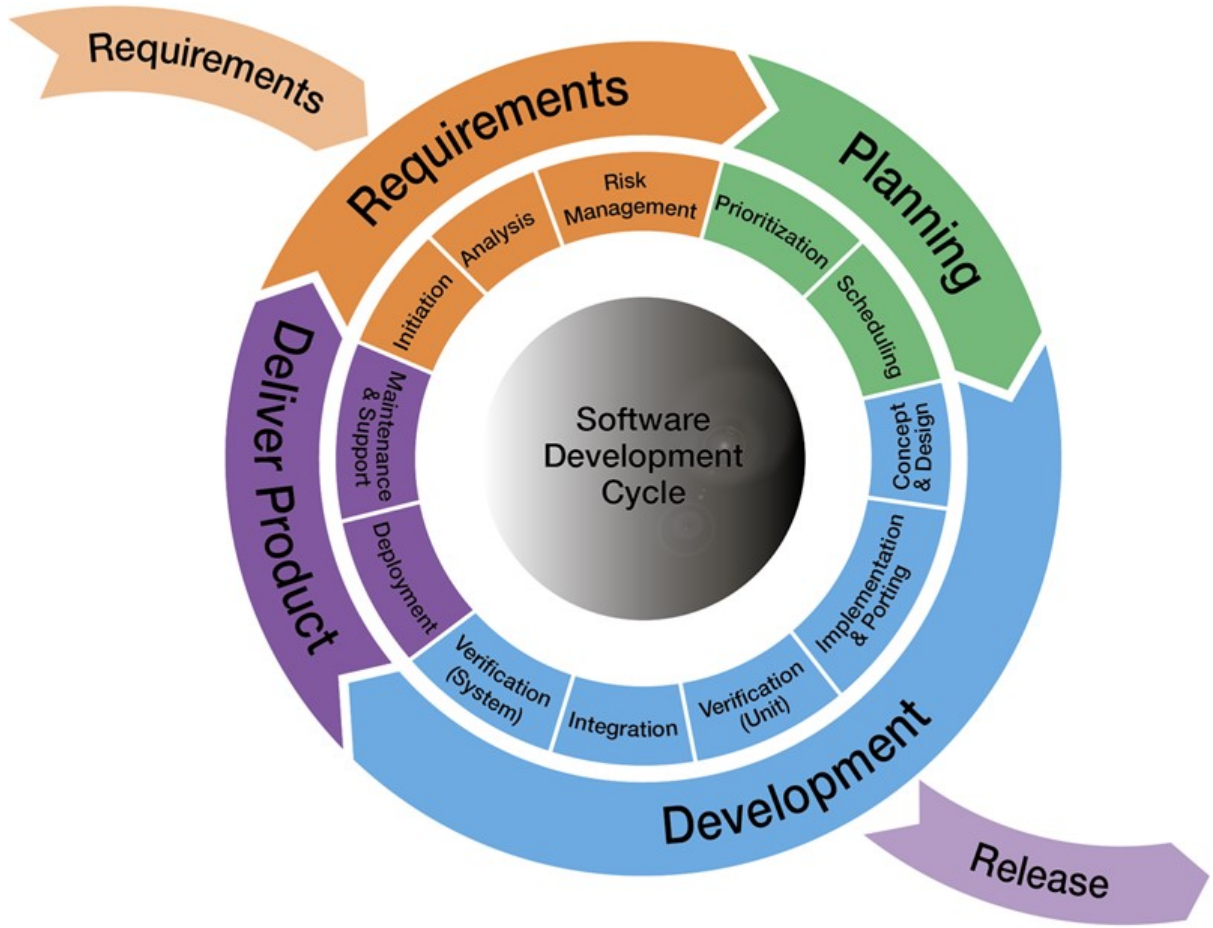
- The development process is at least 4 steps:
  - Analysis – Design – Implementation – Testing
    - Sometimes, "Initiation" is added as a first step, and "Support, maintenance and improvement" added as a last step
    - Documentation is sometimes a separate step, but usually is just a part of the whole process (or a part of the implementation)
- The precise mix of these steps depends on the methodology used (see later in the course about the waterfall model, XP and Agile)

Deliverables

- Initiation
  - Project proposal document
- Analysis
  - Requirement analysis report
- Design
  - System Design Documents
  - Test specifications
- Implementation
  - Code
  - Documentation
- Testing
  - Test results, based on test spec



Iterative and Incremental Software Development



These Steps Need to be Taken

- Either all requirements analysed at the same time
  - Then you would do all the design, and so on !!
  - Waterfall process
- Or
  - Requirement analysis, design, implementation and testing in a 2 week period
- Either way
  - All these need to be done
  - It is just a matter of in which order

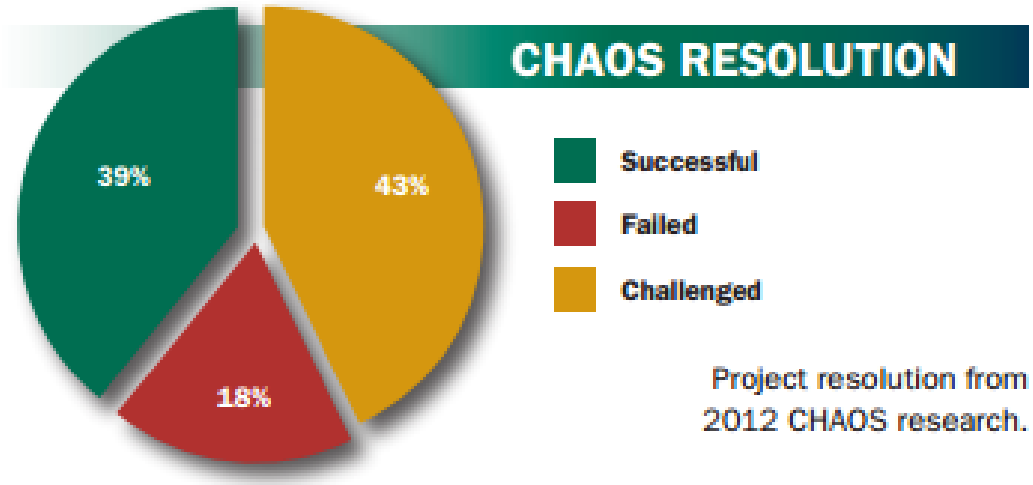
Focus of the course

- This course focuses on the analysis and design and some parts of testing
  - What it involves and methods which can be used
- We leave out
  - Project management
  - Big part of initiation
  - Code development by programming languages (implementation)
  - The above will be learnt in other courses
- We start with requirement capture
  - Usually takes place after problem analysis phase but sometimes blend into each other

Software for information systems

- All software exists within a system
- A lot is involved when developing software, e.g.:
  - Business drivers and business processes
  - Technology drivers
  - Development
  - Stakeholders
  - Users
  - People, people, people
- So a lot to think about to produce software of good quality

Software Development is not Easy

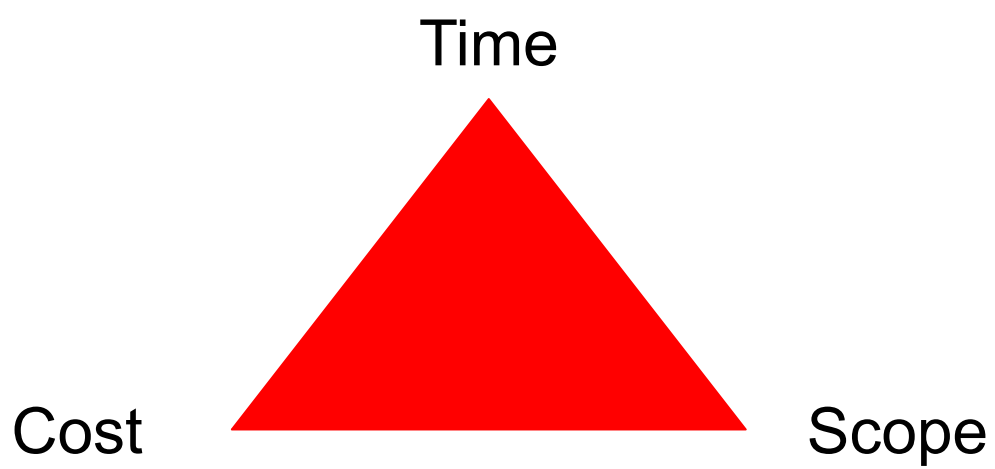


- Chaos Manifesto, Standish Group, 2013
- More success rate by looking at processes, methods, skills, costs, tools, decisions, optimization, internal and external influences, and team chemistry

- http://versionone.com/assets/img/files/ChaosManifesto2013.pdf

Project overruns (from the Chaos Manifesto)

- Time overruns 74%
- Cost overruns 59%
- Specified features & functions delivered 69%



The project management triple constraints

Success Factors (as Standish group sees it today)

- **1. Executive Support**
- **2. User Involvement**
- **3. Clear Business Objectives**
- 4. **Emotional Maturity**
- **5. Optimizing Scope**
- **6. Agile Process**
- 7. Project Management Expertise
- **8. Skilled Resources**
- **9. Execution**
- 10. Tools & Infrastructure

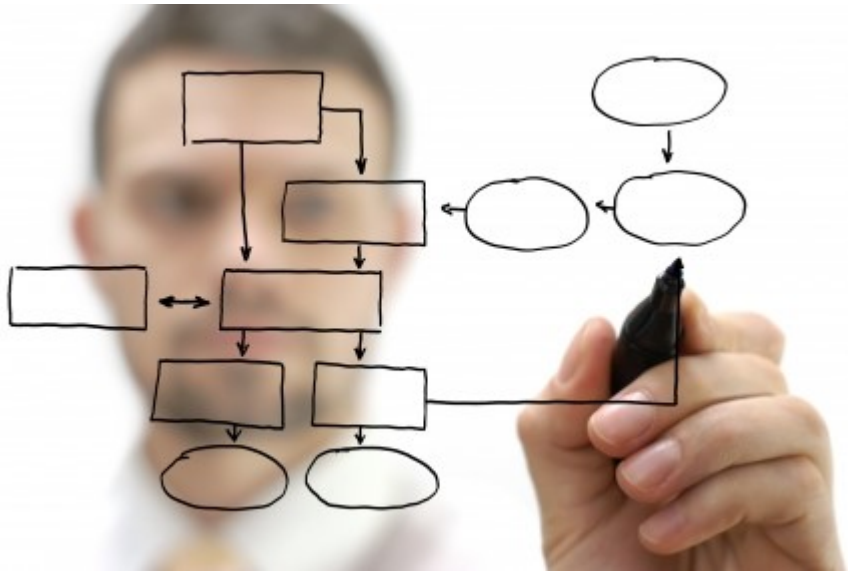
- Bold = related to this course
- http://www.standishgroup.com/chaos_news/newsletter.php?id=54

This course you will learn about (in red)

1. Executive Support
2. **User Involvement**
3. **Clear Business Objectives**
4. Emotional Maturity
5. **Optimizing Scope**
6. **Agile Process**
7. Project Management Expertise
8. Skilled Resources
9. Execution
10. Tools & Infrastructure

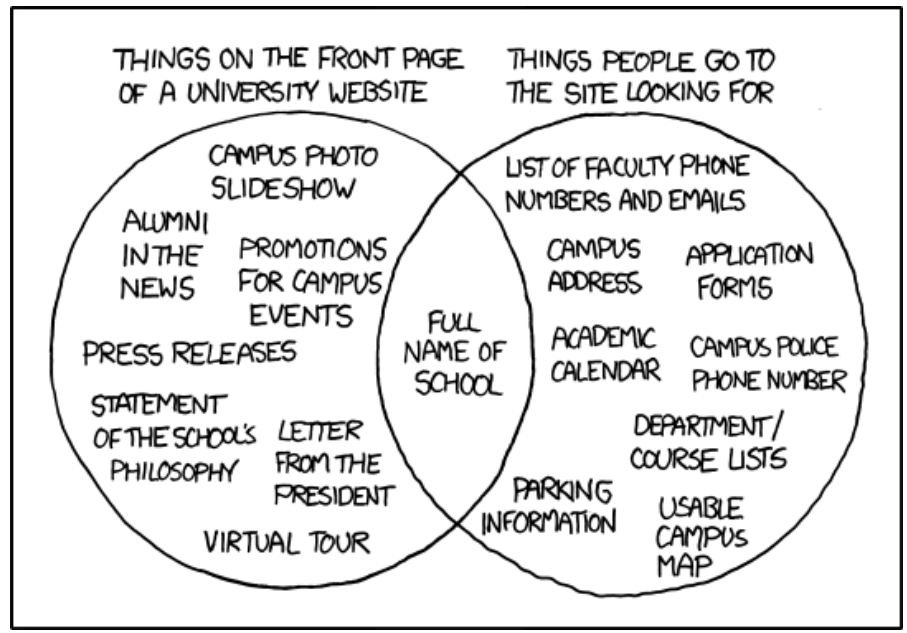
- http://www.standishgroup.com/chaos_news/newsletter.php?id=54

Requirement Analysis



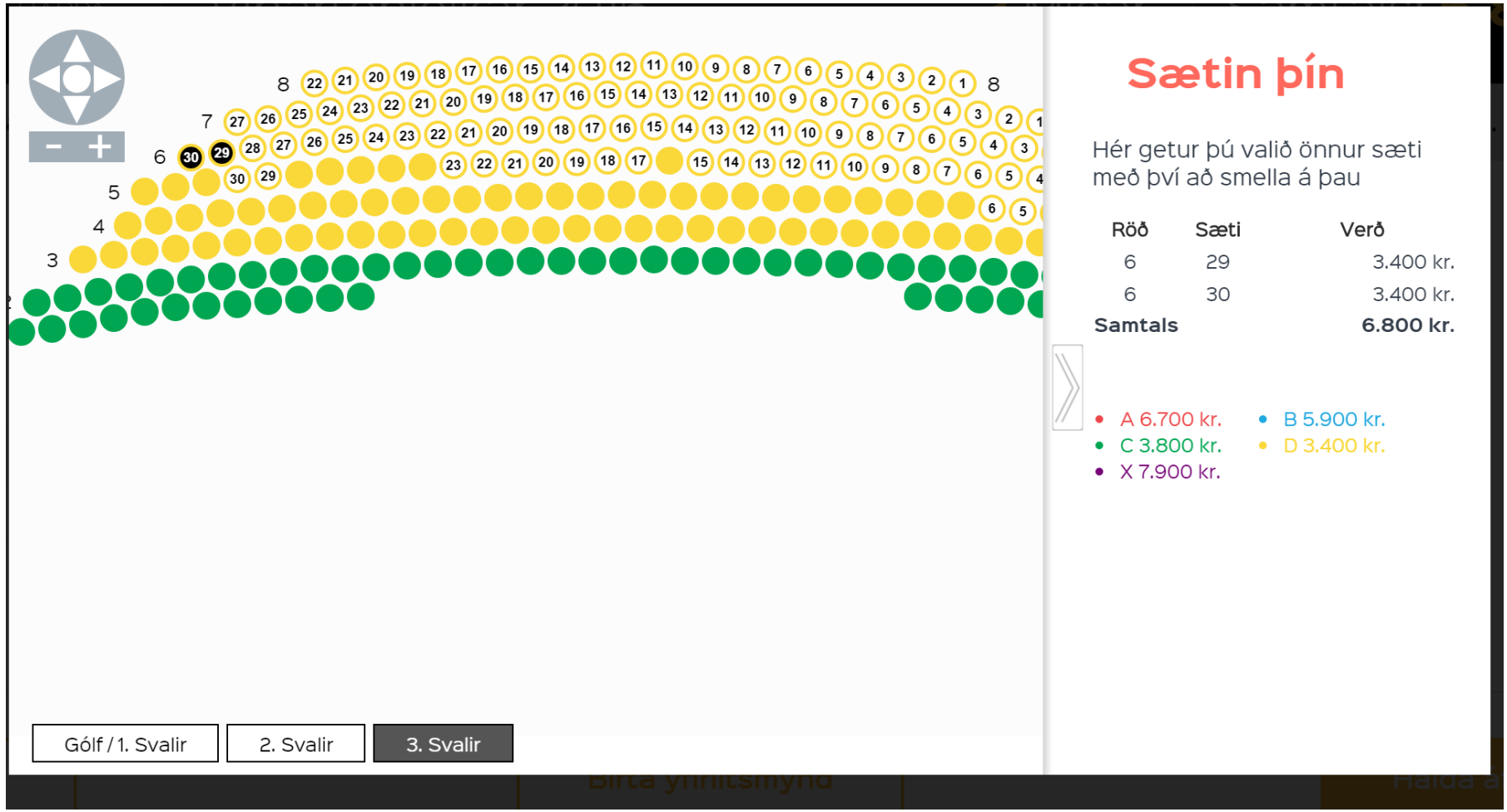
Why analyze and design?

- People need to be able to use our software
- We need to **think hard** in order for giving the end user the opportunity **“not to think”**



- Need input from all aspects of the system to be built, especially the users
- Fun article on specs by Joel Spolsky in 2000: [Painless Functional Specifications - Part 1: Why Bother?]

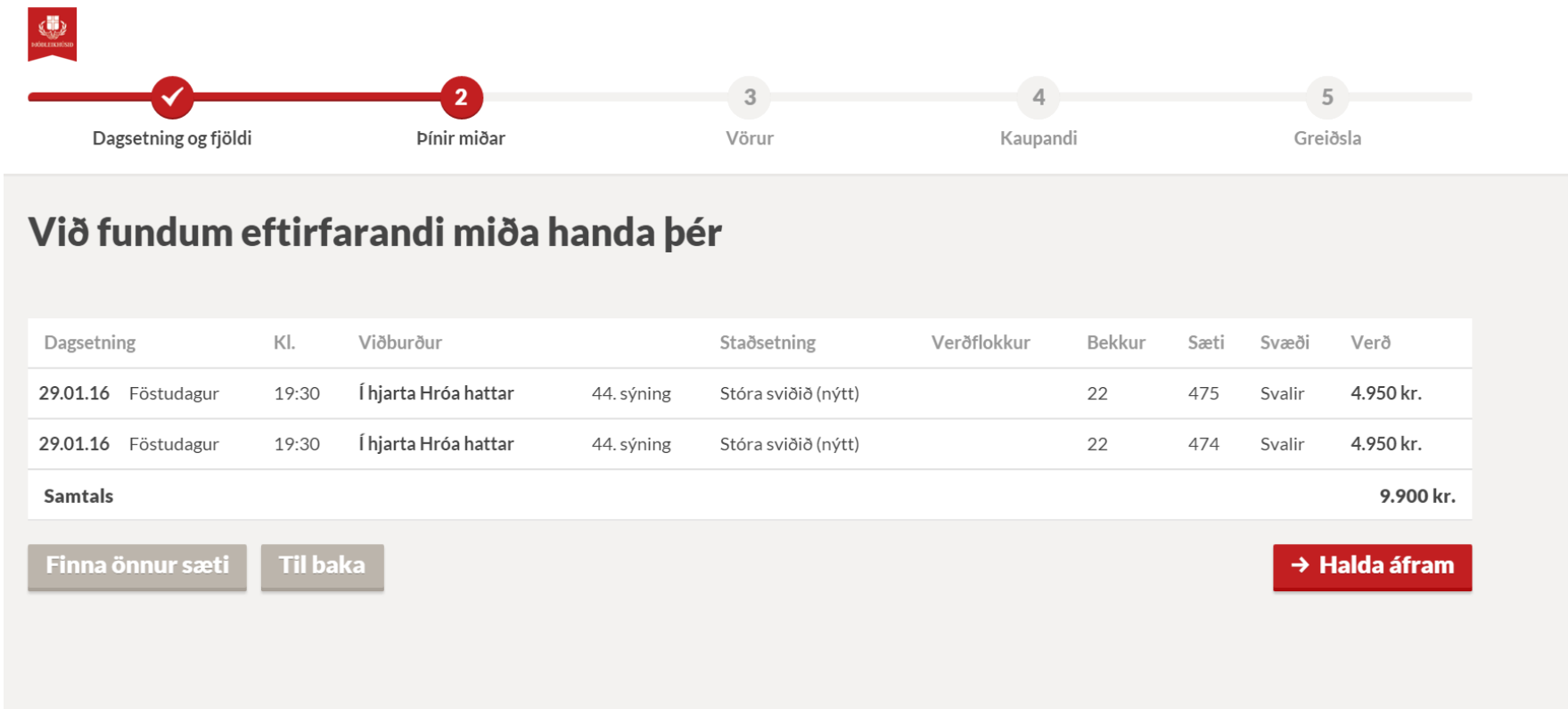
Does the user interface matter?



This is the user interface for buying tickets in Harpa (tix.is)

Does the user interface matter?

The same task buying tickets? Which do you think is better?



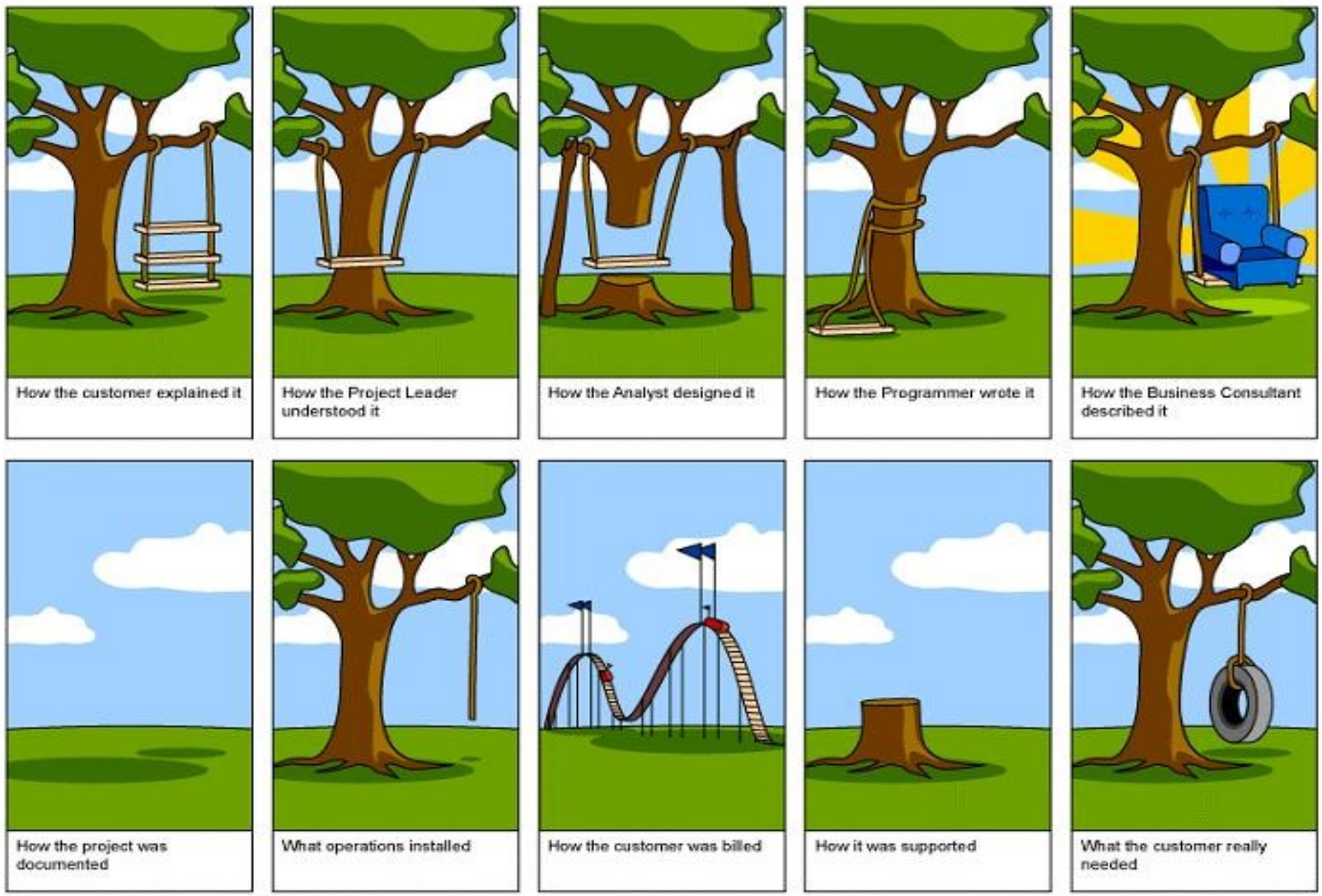
This is the user interface for buying tickets in Þjóðleikhúsið (midi.is)

If the interface does not matter

- Why do people complain when Facebook changes their user interface??

Profile from 2005

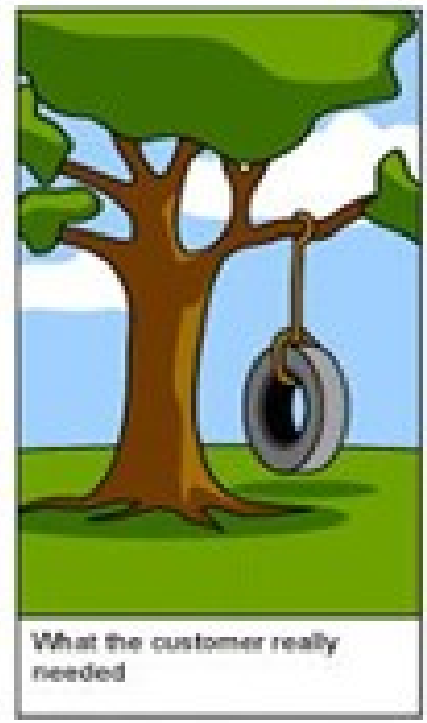
Does requirement analysis matter?



... maybe we need to learn to communicate?

The picture should be like this

This is what was analysed and designed



This is what the customer got

Requirement analysis report

- In Icelandic: þarfagreiningarskýrsla

- A document which explains ***what*** the system should do
  - Note: it will say **nothing about how** said functionality will be implemented!
- Can we give a finite list of what should be in this document?
  - No, anything that might be useful to explain what the system should do could be in there

What is the purpose of this document?

1. First and foremost: state *what* the system should do

2. Communicating with stakeholders of the project
   - requires the document to be written in a non-technical language, regular users/ managers/ developers must all be able to understand the report

3. Used as input to later stages of the development process
   - development
   - testing
   - documentation

Contents of a requirement analysis report

- Although this won't be an exhaustive list, the following is often found in a requirement analysis report:
  - general description (free text)
  - requirement list
  - prioritization of requirements
  - use case diagram
  - list of use cases
  - role list
  - etc.

General description

Typically, a general description of the system/project would mention:

- The main purpose of the system
- What assumptions are made about the system
- Implementation suggestions
  - is it a web application? for a smart phone? etc.
- System boundaries
  - where do we stop?
  - will this system interface with other systems?
  - what is out of scope?
- Environment

What is a requirement?

- A well defined, testable statement that can be verified
- Example of a good requirement:
  - “All users shall be able to rent a book" (short, single responsibility)
- A bad requirement:
  - "The system should be really fast" (vague, how fast is really fast?)

– Better:
  - "The average response time should be less than 500 milliseconds when executing a query" (measurable)

Functional and non-functional requirements

- Functional requirements
  - implemented features
    - “it should be possible to borrow a book"
  - rules
    - "a user may not borrow more than 3 books simultaneously"
  - etc.
- Non-functional requirements
  - requirements such as:
    - Extensibility/scalability
    - Portability, reusability
    - Performance, hardware concerns
    - Cost savings
    - Usability/user experience
    - Training needs, etc…

Requirement list

- A requirement list will often look something like this:

Number	Name (and possibly a short description	Use case number(s)	Priority (A/B/C)	Status (approved)
...	...	...	...	...

  - Number:
    - an incrementing number which identifies each requirement
  - Name/description:
    - short description of the requirement
  - Use case number(s):
    - a list of use cases which have to do with this requirement
  - Priority:
    - how important this requirement is:
    - A: absolutely essential
    - B: useful, but not mission critical
    - C: nice-to-have
  - Status:
    - approved/not approved (not always used)

How do we gather information?

- Requirements can be gathered in several ways:
  - Interviews
  - Questionnaires
  - Prototypes
  - Observing users
  - Examine other systems (older versions)

- We often use more than one in the same project

  - Will be covered better later...

Summary

- Looked at the development process
- Discussed the reasons for why we analyze and design software
- Examined what deliverables are created during the process
- Looked at a typical requirement list