Requirements and qualities

Martin Osovský

Motivation

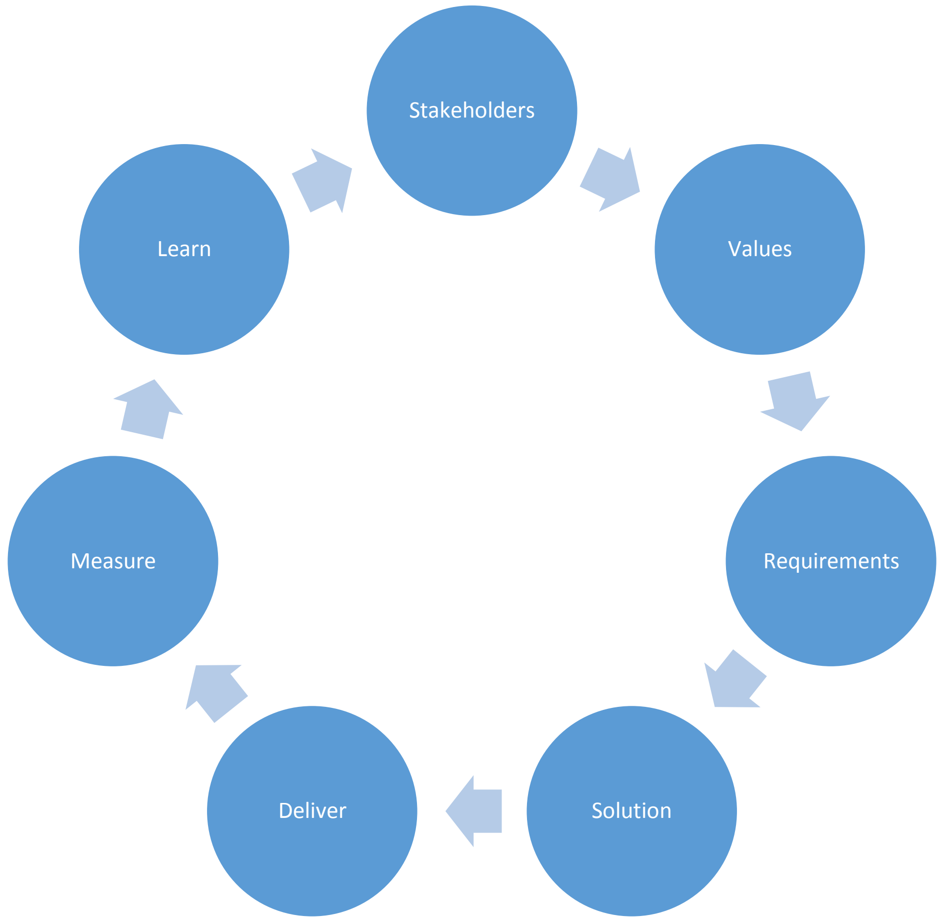
- When you know exactly what to do, it is usually easy to do it.
- As an architect I often need quality requirements
  - To be able to decide about the most important things
  - To be able to VALIDATE architectural decisions
- Architecture is driven by qualitative requirements
- Many seemingly not tensible aspects of a product can be well defined as quality requirements
  - Like maintainability
  - Or extensibility, time to market etc.

What is an requirement?

- A Stakeholder Valued System State under Certain Conditions

- Key parts are
  - There is a concrete Stakeholder
  - His value in the product (e.g. what makes him happy with the product)
  - System state defined exactly and in terms of the system
  - Conditions, constraints and assumptions are explicitly stated

Evolving a process/product



Biggest problems with requirements

- Focus on “customer needs”
- No value in the requirements
- No clarity, nice words
- Focus on function (see non-functional requirements)
- Too much focus on testability
- No background (who and why?)
- No dependencies and relations (isolated requirements)
- No quality control over requirements
- No distinguishing between types of requirements

The worst problems

- Bad quality of top level key requirements
  - Missed stakeholder, no values, no key requirements identified
- Mixing means and ends (“I need a brick” problem)
  - Design in requirements
  - No background
  - No evolution with stakeholders

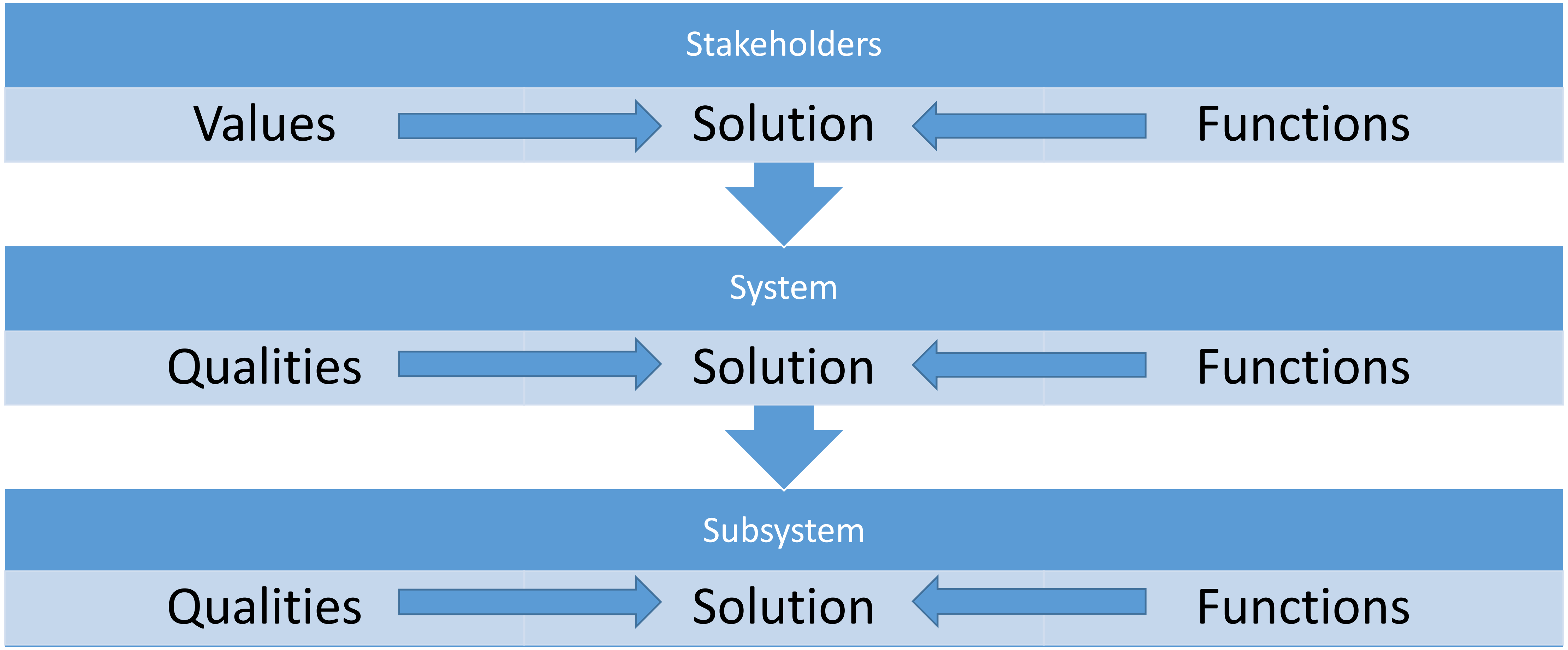
The value vs design

Why do you require a 'password' for Security!	• That's what I asked for!
What kind of security do you want?	• **Against stolen information**
How strong security against stolen info are you willing to pay for?.	• **At least 99% chance they cannot break in within 1 hour**
So that is your real requirement ?	• **Yep.**
Can we make that official?	• **Of course!**

Types of requirements

- Functions are for free
  - There is usually no clear value in the functions itself
  - **What** the system does
- Quality requirements are the key
  - **They are not nonfunctional**
  - **How well** does the system does it
  - Performance requirements
  - Resource requirements (e.g. budget)
- Constraints
  - Design constraints – explicitly stated by someone

Levels of requirements



Anatomy of an requirement

- Clearly there must be
  - A name (or tag)
  - Stakeholders
  - Type (level, complexity, …)
  - Dependencies (consists of, influences (positively, negatively)
- We use
  - Ambition
  - Scale
  - Meter
  - Targets and Benchmarks
  - Constraints

Quantification and measurements

- Quality requirements must be quantified in order to be measured
  - You cannot improve what you cannot measure
  - You also don’t know that it got worse
- Not every requirement is quantifiable by itself
- Every requirement can be decomposed to quantifiable ones
- Quantification is done using Scales
- Measurement using Meters

Decomposition

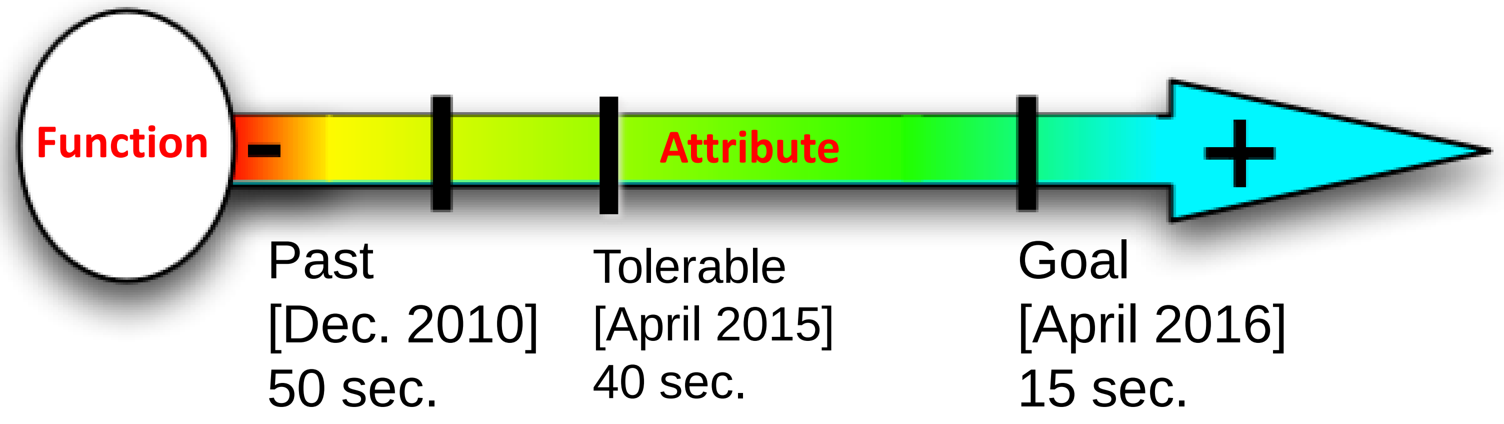
- The high level requirements are usually Complex (not quantifiable)
- They can be decomposed to Scalar ones in an iterative process (remember – learning)
- That gives a nice hierarchy
- Scalar requirements have a scale
- Examples:
  - Security
  - Usability
  - Friendship, Love, Music, Poetry

Scales

- Every quality has at most one scale
- A quality with no scale is Complex and needs to be decomposed
- Scale consist of Qualifiers and Units (usually in a form of x per y)
  - Examples
    - Kilometers per hour (current speed)
    - Usability.Intuitevness : % chance that defined [User] can successfully complete defined [Tasks] Immediately, with no External help.
    - Beauty : Average % of evaluation points using [Survey] given by defined [Experts] per screen.
    - It is suprising how many of these things can be googled
- Designing scales is an agile (i.e. learning) process and it’s fun
  - Use <> brakets to delay a definition (fuzzy brackets)

The levels

- It is essential to have targets, bechmarks and constraints



The levels in more details

- Targets
  - Goal : where we want to be
  - Stretch : where we wish to be
- Benchmarks
  - Past : where we are (were)
- Constraints
  - Fail : the value is not there at all
  - Survival : the value is at its lowest
- These numbers must be qualified like this
  - Goal[May 2015, Desert, Expert User, Humidity:90%] = 99%

There must not be design in the requirements

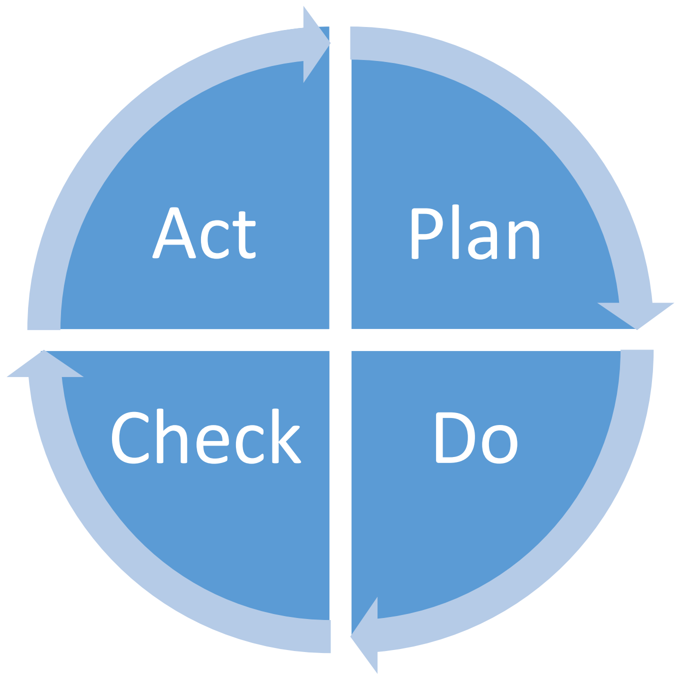
- Only in the form of a
  - Hypothesis
  - Constraint
- All design given by the customer must be
  - Treated as a Solution Idea (not required, suggested)
  - And thoroughly investigated to know the background
- Examples
  - The application is protected by a password
  - The system shall use a loadbalancer
  - The administrator will be notified by an SMS or email

Key abilities of an requirement analyst

- Know the key stakeholders (internal and external)
- Ask for the background (the real value needed)
- Filter out design (very difficult)
- Identify stakeholder values
- Map them to proper quality requirements
- Decompose these to scalar qualities
- Define scales for these qualities
- Gather facts, i.e. numbers about past (measuring) and desired values (learning)

The whole process goes in circles

- Values, markers and stakeholders evolve
- We learn by measuring the product continuously
- Always have the wheel in mind:


Act
Plan
Check
Do