RE - Alles hat zwei Seiten,
Ein Streitgespräch unter Kollegen

Sven Krause

Product Developer

Business Analyst

Projekt- & Q-Mgt.

Beratung & Coaching

sven.krause@zuehlke.com
Senior Business Consultant
Zuehlke Engineering AG

Bernd Waldmann

Systems Engineer

Requirements Analyst

Medical Device Expert

Hearing Scientist

bernd.waldmann@phonak.com
Director Systems Engineering
Phonak AG

Learning objectives

- Know about “popular” discussion topics in RE
- Understand that there is no silver bullet: there are good arguments pro & con
- Learn how to choose the right approach for your setting

What and How ...

What and How must be clearly separated

- Requirements should not define solutions (the HOW), but only the problem (the WHAT)
- If you state solutions in requirements, you restrict the choices the implementers can make
- Product Managers don't know about technical solutions, but they know about customer needs.
- Developers know about technical solutions and feasibility, let them write technical specifications.
- This is the well established separation of requirements (Lastenheft) and design specifications (Pflichtenheft)
- Matches organizational setup: business vs. technical

What and How cannot be clearly separated

- Design happens in successive refinements: from Needs to Concept to Requirements to Design Spec …
- WHAT/HOW depends on a reader's viewpoint: one person's WHAT is another person's HOW
- Business people like to think in terms of solutions, not problems – so let's talk about solutions in their language
- Both are interface documents for communication between different domains, e.g. Marketing – R&D, or Architect – Developer
- Lastenheft / Pflichtenheft are from the contracting business and have legal meaning, we are more interested in cooperative processes

What and How ...

Synthesis

What and How must be clearly separated

What and How cannot be clearly separated

- *"It's not whether a statement is a 'requirement' or a 'design' that matters, but whether the statement places appropriate constraints on the people that will read it"* (Erik Simmons, Intel)

- At least, trace back to "requirements" (user needs) from "design specifications" (system behavior)

What and How ...
Environments

What and How must be clearly separated

… may be appropriate for:

- purchaser ./. supplier setting
- green field projects

What and How cannot be clearly separated

… may be appropriate for:

- in-house implementers
- brown-field and maintenance projects
- products based on given architecture
- agile projects

Use Case

Use cases are helpful tools for eliciting and documenting requirements

- Good starter to define requirements for processes & platforms
- Simple and understandable representation of complex subject
- use cases are the only model which is understandable for the business
- Help to focus on the most significant specification area – the interaction between system and user/customer
- use cases help to understand the “big picture”, see individual requirements in perspective
- use cases typically involve multiple products or components, and help to describe system behavior

Use cases are useless, developers can only work from line-item requirements

- Use cases are hard to convert to developer tasks, test instructions etc.
- How do you calculate % done and test coverage from use cases?
- Typically, half the use case is about what the user does – so it is outside the realm of influence of the developer
- Creating use cases takes too much time – first you need to write the use case, and then you need to write the “derived” requirements anyway

Use Case
Synthesis

Use cases are helpful tools for eliciting and documenting requirements

Use cases are useless, developers can only work from line-item requirements

- Use cases don't replace requirements, they complement requirements
- Use cases are a tool for requirements elicitation
- Use cases are useful, but it is a question of the right moment
- You have to differentiate "business use cases" against "technical use cases"
- Use case has to be refined by additional models (UML activity diagram; BPMN, …)

Use Case
Environments

Use cases are helpful tools for eliciting and documenting requirements

… may be appropriate for:

- business solutions with a software part
- requirements for systems
- products with complex user interaction

Use cases are useless, developers can only work from line-item requirements

… may be appropriate for:

- component requirements
- very technical products
- developers with excellent domain knowledge

RM Tool

RM tools can improve the quality of requirements

- A trace matrix is necessary to prove coverage (completeness) and detect gold-plating – and that is hard to maintain without an RM tool
- Context is important: rationales, traceability
  … and this is hard to maintain without an RM tool
- Different stakeholders need different content
  … and a good RM tool can create multiple views (custom reports)

A fool with a tool is still a fool

- The integration into development process is too complex (there is no tool which is accepted by business)
- RM tools don't help with content, just with the formalities
- Process engineering best practice: choose a tool that supports your process, don't adjust your process to the tool
- Total cost-of-ownership of a RM tool is much higher than the license fee: maintenance, user training, customizations, tool operators

RM Tool
Synthesis

RM tools can improve the quality of requirements

A fool with a tool is still a fool

- *„Investing in putting bad requirements into manage-ment tools is a lot like rearranging the deck chairs on the Titanic.“* (Ivy Hooks)
- There are no RE tools, only RM tools: the tool will not teach you requirements engineering
- Be clear about your expectations for a tool: traceability? reporting? teach you RE?
- Match the tool to your process and maturity

RM Tool
Environments

RM tools can improve the quality of requirements

… may be appropriate for:

- highly regulated environments (traceability)
- mature requirements process
- resources for tool maintenance available
- requirements engineering know-how is available

A fool with a tool is still a fool

… may be appropriate for:

- highly agile development, little reuse
- organization without mature RE knowledge
- no resources for coaching & tool maintenance available

DANKE

Alan M. Davis
Requirements Engineering, Both Sides of the Issues. 16th IEEE RE Conference, 2008

Patrick Strasser
(Swisscom)

Frank H. Ritz
(Ritz Engineering)

Marcel Hebeisen
(Swisscom)

Matthias Pohle
(Swisscom)

Thomas Haas
(Infogem AG)

Ralf Fahney
(Fahney Anforderungsingenieurwesen)

Samuel Fricker
(Uni Zürich, Fuchs Informatik)

Reinhard Stoiber
(Uni Zürich)

Nicolas Fauser
(Swisscom)

Bei Interesse: Die SAQ RE Arbeitsgruppe

http://www.saq.ch/de/fachgruppen/informatik/arbeitsteams/requirements-engineering/

Sven.krause@zuehlke.com

Survey at RE Forum 2010 – 44 participants

1. **What and How**	a. What and How must be clearly separated	55%	
	b. What and How cannot be clearly separated	45%	
2. Use Cases	c. Use cases are helpful tools for eliciting and documenting requirements	93%	
	d. Use cases are useless, developers can only work from line-item requirements	7%	
3. RM Tool Enviroment	e. RM tools can improve the quality of requirements	50%	
	f. A fool with a tool is still a fool	50%	

Combination		
acf	29%	
ace	24%	
bce	24%