PXL – IT
42TIN1280 Software Analysis
Documentation of Requirements using NL

Nathalie Fuchs

Luc Doumen

DE HOGESCHOOL
MET HET NETWERK

Hogeschool PXL – Elfde-Liniestraat 24 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook

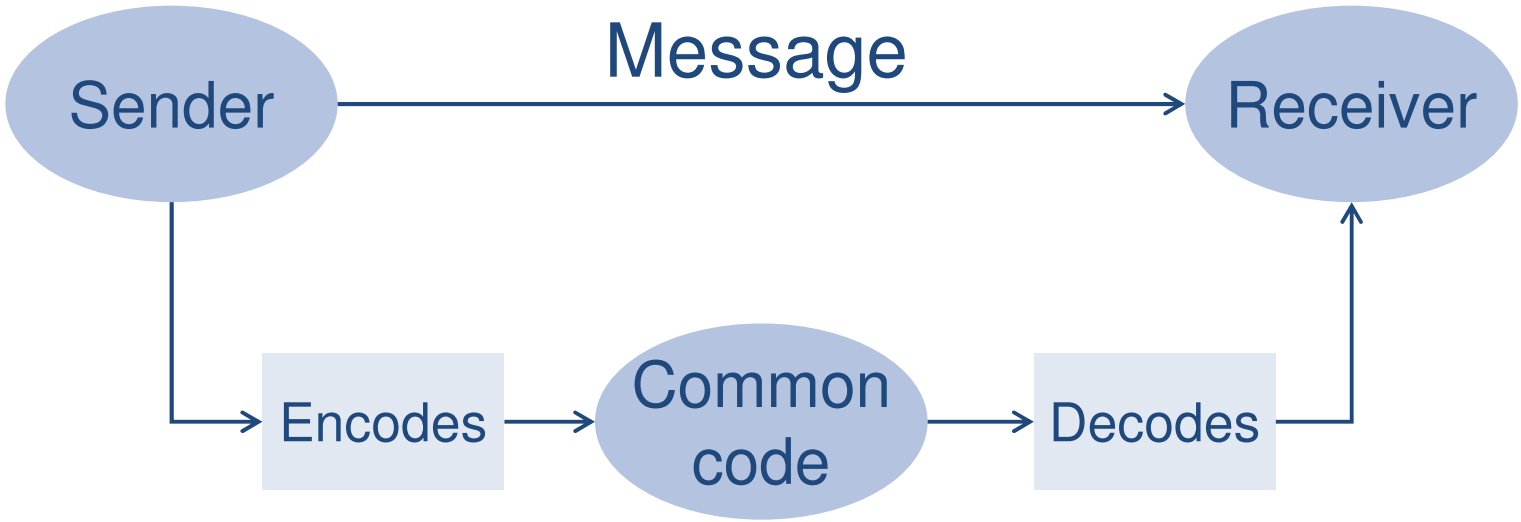
Content

- Natural language: what, language effects, etc.
- Five most relevant “transformations”
- Requirements writing according to the template
  - Dealing with transformations
- Recap IEEE 830 – SRS, standardized document structures & requirements classification
- Fit criteria
- Requirement card & attributes
- Key learning points
- Questions & answers


Natural language

Natural Language

- Every stakeholder can “easy” understand it
- Is universal, can be used to describe the desired circumstances
- … but is often ambiguous and interpretable



Language Effects

- It is about perceptions and transformations of the documentation of the perceived ‘things’
- ‘What is said’ versus ‘What is meant’
- ***It is the responsibility of the requirement engineer to determine what the originator of the requirement really meant!***
- Five most relevant transformations
  - Nominalization - Nouns without reference index
  - Universal quantifiers - Incompletely specified conditions
  - Incompletely specified process words


Five most relevant “transformations”

Five most relevant Transformations (1)

1. Nominalization
   - Using nouns for complex processes
   - E.g. concepts like ‘registration’, ‘administration’, ‘acquisition’, etc.
   - Only to be used when the process is completely clear / defined ➔ Glossary!

Five most relevant Transformations (1)

1. Nominalization - example
   - After a **system breakdown** a **restart** shall automatically be initiated
     - What will happen during a restart?
     - When will this restart be executed?
     - Who initiates the restart?
     - What failure and exceptions might happen during the restart?
     - What failures and exceptions might lead to a system breakdown?

- **Is this clear for every stakeholder?**
- **What is a system breakdown?**
- **Apparently, I can restart?**

Five most relevant Transformations (2)

2. Nouns without reference index
   - E.g. “The data will be shown on the terminal to the user”
     - What data, on which terminal, to which user?
     - Better: “The system shall show the billing data on the terminal where the registered user is signed on”

3. Universal quantifiers
   - E.g. ‘never’, ‘always’, ‘none’, ‘all’, ‘everybody’, etc.

4. Incompletely specified conditions

   E.g. ‘in case’, ‘when’, ‘dependent on’, ...

Five most relevant Transformations (3)

5. Incompletely specified process words
   - E.g. ‘to transfer’: from where, where to?
   - Passive phrasing ➔ active phrasing

Fuzzy terms list

Mostly	As needed
Might	Make sense
Appropriate	Might make sense
Graceful	At minimum
Major	Slowly
May be of use	Including but not limited to
And/or	Suitable
Various	Clean and stable interface
Several	

Exercise – new clock radio (part 01)

- As a team determine what are the defects in the (natural language) requirements for the new clock radio? (cf. next slide)

- It is a defect when it is a violation of a rule!

Exercise – transformations (continued)

Topic – New clock radio

The development of the innovative alarm clock and music system shall raise the profit of the company by 10% within 3 months after delivery to the wholesale dealer.

The clock shall produce no tic sound. Because people don't like to be woken up abruptly, the new clock radio shall wake up the sleeping people more smoothly than any other alarm clock.

The luminosity of the lighted digits shall be changeable continuously.

The clock radio shall be small and lightweight, because the users also want to use it on travel. Therefore the clock radio shall be usable with different power voltages. To achieve a minimum weight the alarm clock must not contain any iron parts.

The alarm clock shall be able to wake its environment with music or a wake-up call.

The clock shall be very easy to use. The snooze key shall be easily accessible. The clock shall withstand a heavy blow from a sleeper and a fall from 50 cm.

The clock shall be very low in energy consumption. The system may only be operated with batteries of type 3434.

Requirements writing according to the template

Dealing with transformations

- Generic, syntactical requirements templates
  - The blueprint that determines the syntactical structure of a single requirement
  - For complete and unambiguous sentences
  - To create precisely phrased requirements

- ***Five*** steps to write requirements according to the template

Dealing with transformations

1. Determine legal obligation/relevance
   - Fixing of liability
   - Strongly recommended (should)
   - Used in the future (will)
   - E.g. ‘shall’, ‘should’& ‘will’
   - If liabilities change, then requirements change too
   - Use of attributes is another way to document liabilities

Dealing with transformations

1. Determine legal obligation/relevance
   ➔ verb convention

Shall	The word “shall” in the text expresses a recognised requirement on the System.
Should	The word “should” in the text expresses a recommendation or advice on implementing such a recommendation. It is expected that the Requirement Authority will require such recommendations or advice to be followed unless good reasons are stated for not doing so.
Must	The word “must” in the text is used for legislative or regulatory requirements (e.g. Health and Safety) with which both the Requirement Authority and the Design Authority shall comply.
Will	The word “will” in the text indicates a requirement deemed to be outside the individual scope of the System. The existence of a “will” generally implies a requirement for provision of a service by a unit external to the System. The System Design Authority is implicitly authorized to rely on the provision of the service.

Dealing with transformations

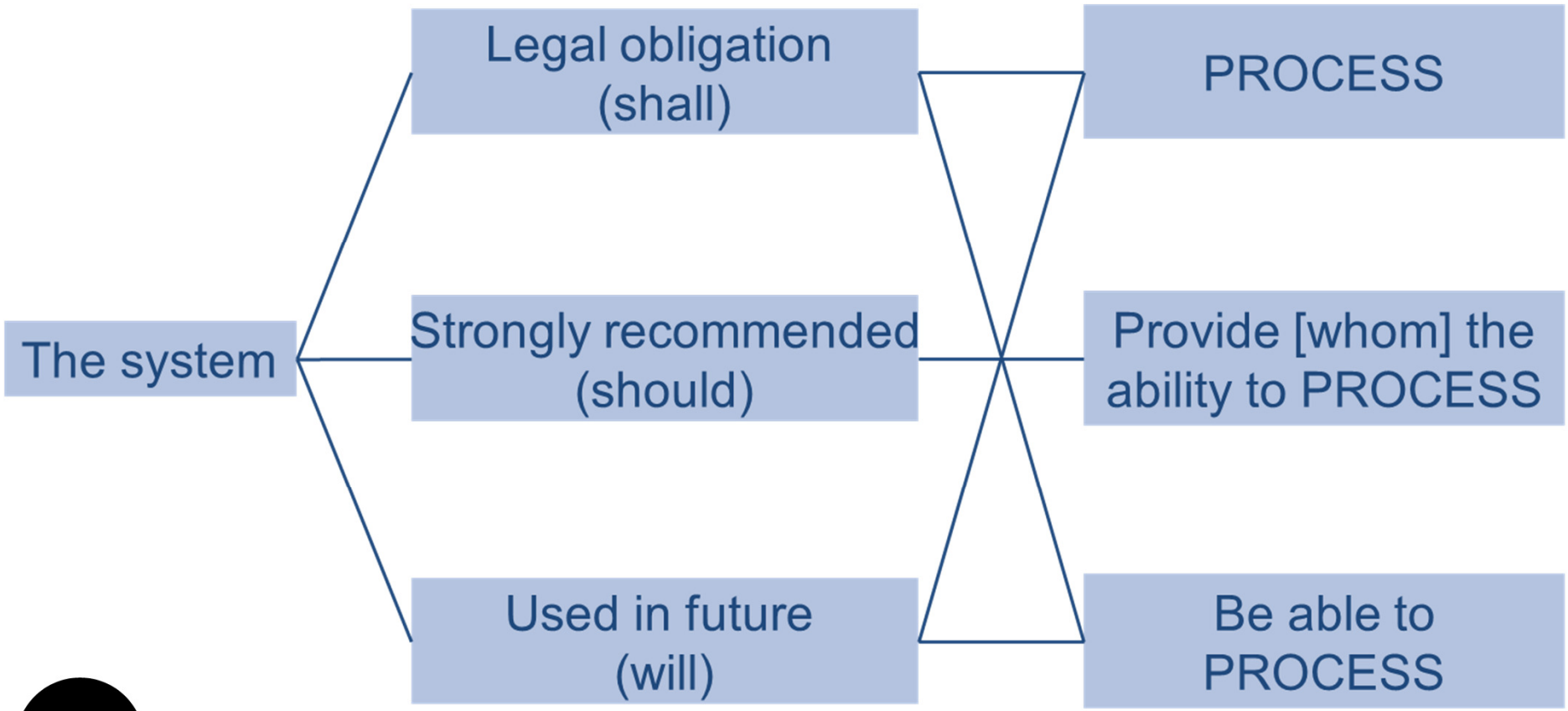
2. Determine the core of the requirement
   - The core of the requirement is the required functionality (e.g. print, calculate, transfer)
   - Processes are activities or events: they shall be described using verbs
   - It might be necessary to add the (process) verbs to the glossary

Dealing with transformations

3. Characterize the activity
   - The way the system works is closely linked to the process words
   - Three types of system activities
     - Autonomous system activity: The system carries out the process by itself
     - User interaction; The system provides the user with process functionality
     - Interface requirement: The system carries out the process dependent on a third factor (for instance a different system), remains passive and waits for an external event

Dealing with transformations

3. Characterize the activity - basic structure



Dealing with transformations

- The first three steps …
  - Our planned system should offer a print option
    - Option is indispensable for us
    - “Print” is the process word
    - The system gives the administrator the option to
  - Requirement no. 1, version 1:
    - The system **shall** provide the **administrator** the ability to **print**

Dealing with transformations

4. Insert objects
   - Further elements are required for the completion
   - A closer or supplementing characterization of the process word
   - ‘What’, ‘where’

   - Requirement no. 1, version 2:
     - The system **shall** provide the **administrator** the ability to **print** the error message to the network printer

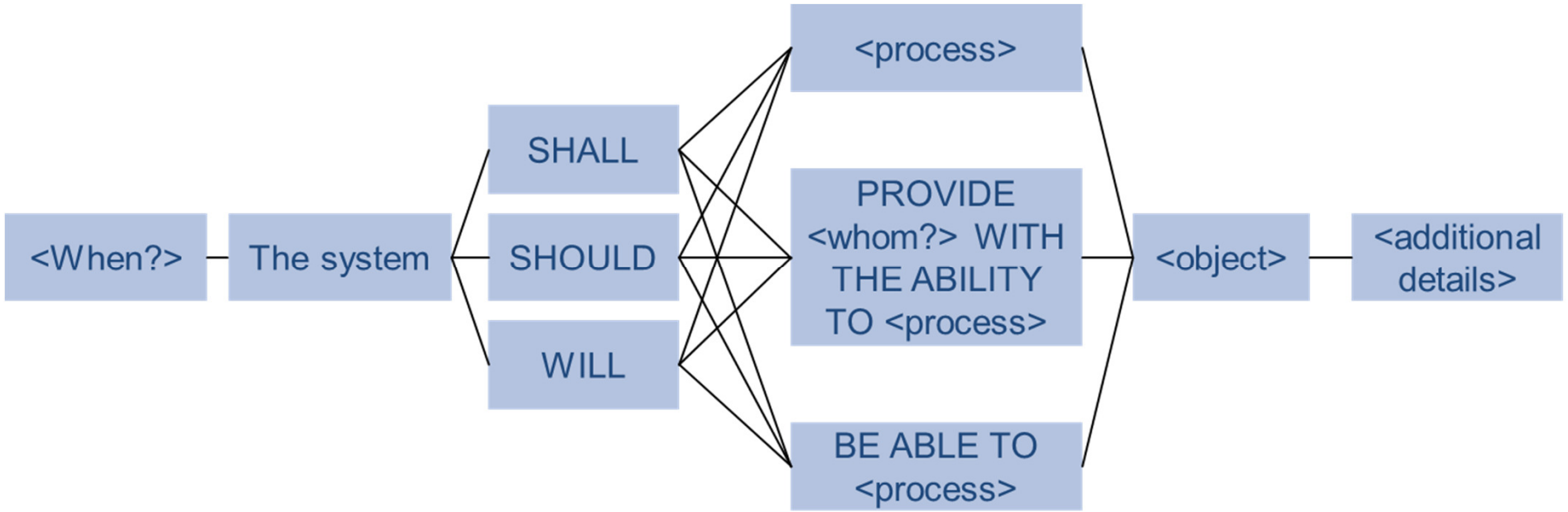
Dealing with transformations

5. Determine conditions
   - The functionality is only given or provided under certain logical or temporal conditions

   - Requirement no. 1, version 3:
     - If an error message has been generated, the system **shall** provide the **administrator** the ability to **print** the error message to the network printer

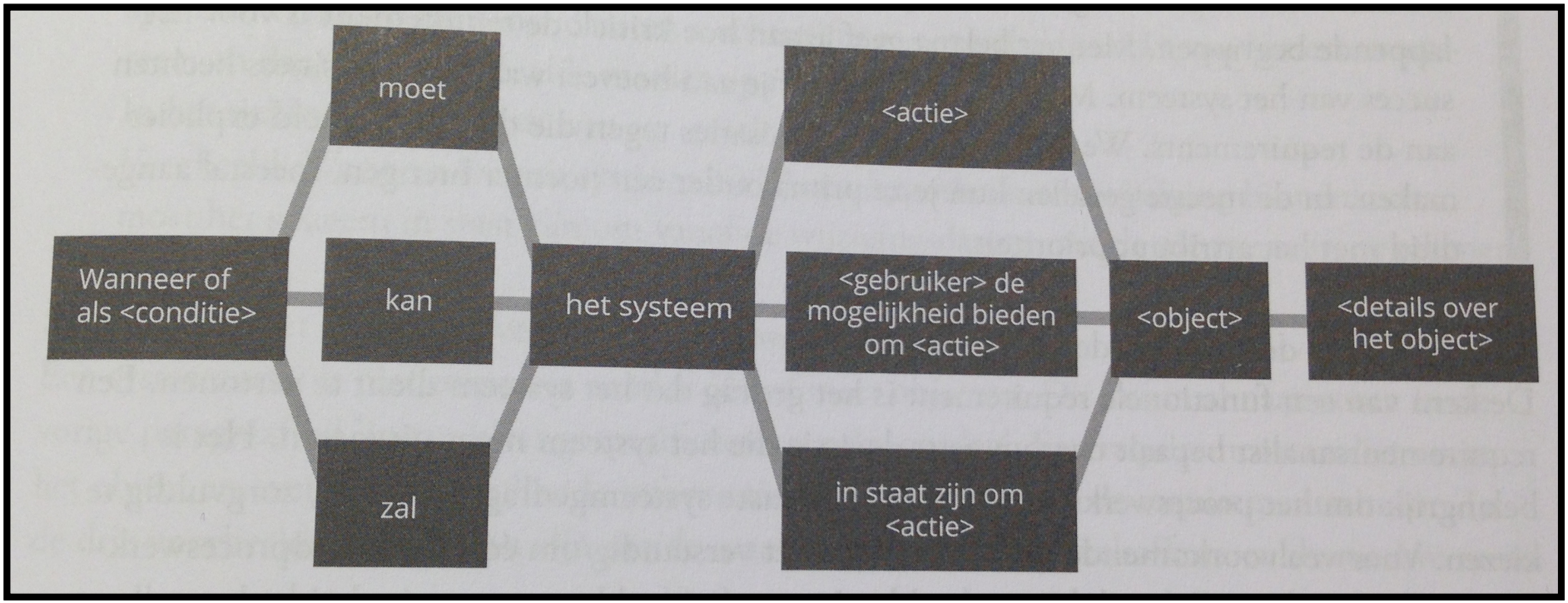
Dealing with transformations

Complete requirement template



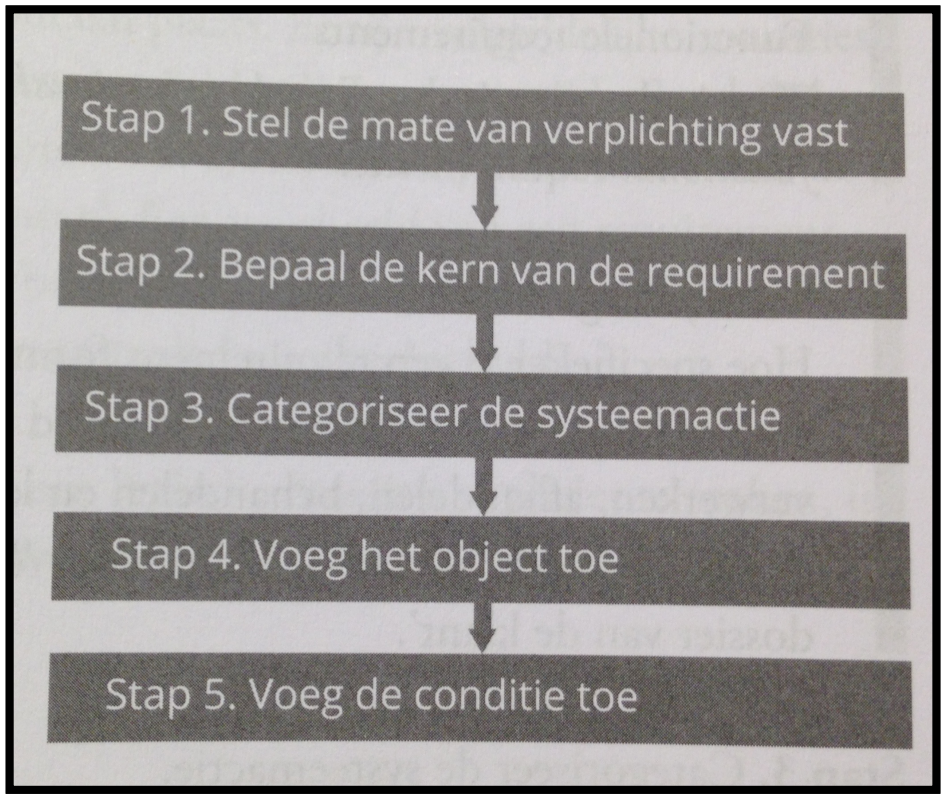
Dealing with transformations

Complete requirement template



Dealing with transformations

Complete requirement template



Mate van verplichting	Hulpwerkwoord in het Nederlands	Hulpwerkwoord in het Engels
Wettelijk verplicht	Moeten	Shall
Noodzakelijk	Kunnen	Should
Wenselijk	Zullen	Will

Dealing with transformations

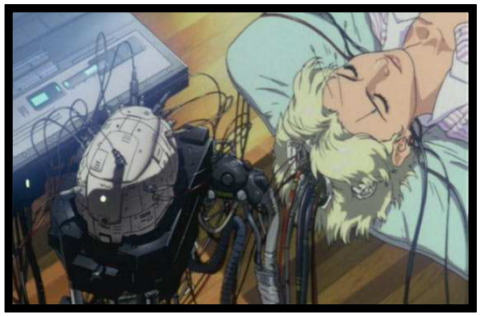
Use templates

- The <stakeholder> shall be able to <capability> **URS**
  - The order clerk shall be able to raise an invoice
- The <product> shall be able to <action> <entity>
  - The launcher shall be able to launch missiles
- The <product> shall provide <user> <function> <object> every <performance> <unit> **SRS**
  - The coffee machine shall provide the user the ability to get a hot drink every 10 seconds

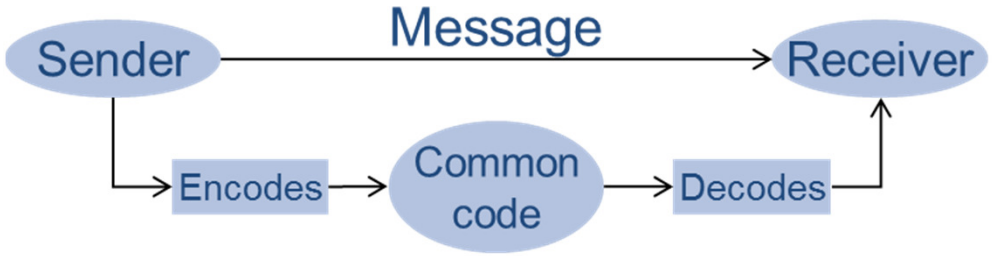
www.requirementsengineering.info

Recap IEEE 830 – SRS, standardized document structures & requirements classification

Why do we need documentation?



- If we could look into each others brains, we wouldn't need documentation …
- Words can help us to communicate



- Be careful, documentation is not enough!
  - Formal / informal language
  - Different interpretations ("I didn't say he killed his wife")

Source: I always get my sin

Language Effects Illustration (FYI)

What is said …	What is meant …	What is understood …
With the greatest respect.	I think you are wrong (or a fool).	He is interested in what I have to say.
(That's) not bad.	That's good or very good.	That's not good enough.
Quite good.	A bit disappointing.	Rather good.
Could we consider some other options?	I don't like your idea.	They have not decided yet.
I will think about it.	It's a bad idea, so I will most definitely not do it.	They think it's a good idea: let's keep developing it
I'm sure it's my fault.	It is your fault!	It was their fault.
That is an original point of view.	You must be crazy.	They like my ideas.
You'll get there eventually.	You don't stand a chance in hell.	They agree I'm heading in the right direction.
We will look into it, we will study the subject.	We will not do anything about it.	They are interested! They will study the subject.

Other Reasons for Documentation

- Requirements ‘may’ have a long lifespan
- Accessibility for many persons
- Legal relevance / compliance
- We need a good overview
- Requirements may become very complex
- Stakeholder needs
  - Users (are my whishes / needs satisfied?)
  - Project Manager (consequences for budget/schedule)
  - System Engineers (translate to solution/architecture)
  - Testers (testability, traceability to test cases)
  - Software Engineers (translation to design and code)
  - Subcontractors (define work/contract – outsourcing!)
  - Change Management, Operation & Maintenance…

Two aspects

- Needs to be readable
  - The structure of the document, how it is **organized** and how the flow **supports the reader** to place individual requirements in context

- Needs to be manageable
  - Qualities of individual requirements, clarity and preciseness and how they are divided into **single traceable items**
  - MS-Word doesn't provide attributes, identifiers etc.

Perspectives on system to be developed

- Data perspective
  - Structure of data
  - Usage and dependency relationships in context
- Functional perspective
  - What information (data) from the system context is manipulated by the system and what data flows from the system to the system context
- Behavioral perspective
  - Documenting the system, embedded in the system context, in a state oriented way

Writing the specification

- Three main sections
  - Introduction
  - Overall description
    - Constraints
  - Specific requirements
    - Functional requirements (grouped)
    - Quality requirements
- Therefore ➔ 

Putting together what we have gathered so far

Document Structure

- Need for structured contents (reference structures)
- Lots of standards and templates available
  - **IEEE 830-1998 (Recommended Practice for Software Requirements Specifications, www.ieee.org)**
  - Other examples
    - IEEE 1233–1998 (Reference structure “System Requirements Specification”)
    - Volere Framework of Atlantic System Guild (www.volere.co.uk)
    - Rational Unified Process (RUP)
    - V-Model XT (Germany)
    - Etc.

Using Standards & Templates

- Can and should be tailored to your project!
- Should at least contain:
  - The goals
  - The functional requirements
  - The non-functional requirements
    - Quality requirements
    - Constraints
  - A glossary
  - A list of abbreviations used

Requirements Document Template

- **Section 1: Introduction**
  - 1.1 The purpose of the product
  - 1.2 Scope of the product (context diagram)
  - 1.3 Definitions, acronyms and abbreviations
  - 1.4 References
  - 1.5 Overview

•Describe purpose of this SRS
•Describe intended audience

•Identify the software product
•Enumerate what the system will and will not do
•Describe user classes and benefits for each

•Define the vocabulary of the SRS (may reference appendix)

•List all referenced documents including sources (e.g., Use Case Model and Problem Statement; Experts in the field)

•Describe the content of the rest of the SRS
•Describe how the SRS is organized

Requirements Document Template

- **Section 2: Overall description**
  - 2.1 Product perspective
    - General overview, use cases, interfaces
  - 2.2 Product functions
    - A list of functionalities (incl. quality related) provided
    - Grouping should support a communication point
  - 2.3 User characteristics
  - 2.4 Constraints
  - 2.5 Assumptions and dependencies
  - 2.6 Apportioning of requirements
    - Requirements that may be delayed

•Present the business case and operational concept of the system
•Describe how the proposed system fits into the business context
•Describe external interfaces: system, user, hardware, software, communication
•Describe constraints: memory, operational, site adaptation

Requirements Document Template

- **Section 2: Overall description**
  - 2.1 Product perspective
    - General overview, use cases, interfaces
  - 2.2 Product functions
    - A list of functionalities (incl. quality related) provided
    - Grouping should support a communication point
  - 2.3 User characteristics
  - 2.4 Constraints
  - 2.5 Assumptions and dependencies
  - 2.6 Apportioning of requirements
    - Requirements that may be delayed

•Summarize the major functional capabilities
•Include the Use Case Diagram and supporting narrative (identify actors and use cases)
•Include Data Flow Diagram if appropriate

•Describe and justify technical skills and capabilities of each user class

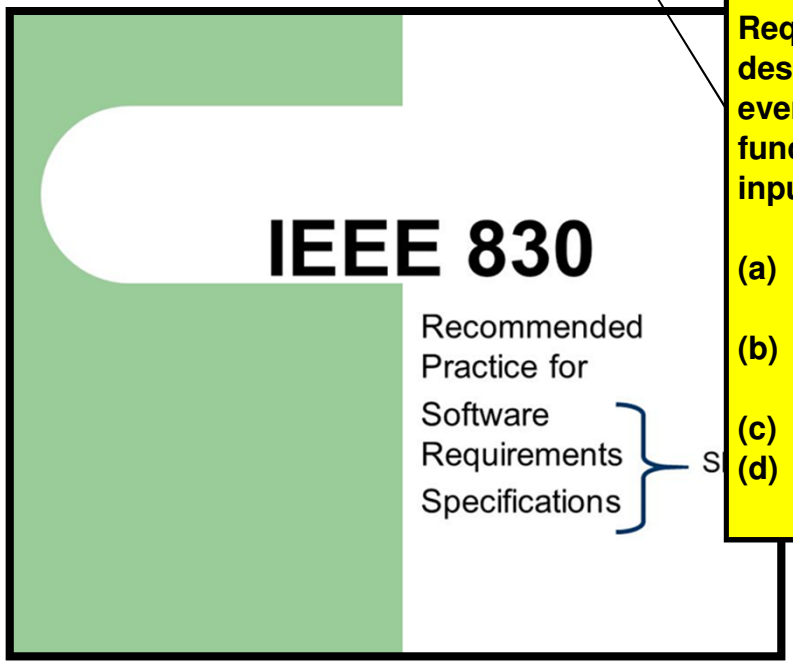
Requirements Document Template

- **Section 2: Overall description**
  - 2.1 Product perspective
    - General overview, use cases, interfaces
  - 2.2 Product functions
    - A list of functionalities (incl. quality related) provided
    - Grouping should support a communication point
  - 2.3 User characteristics
  - 2.4 Constraints
  - 2.5 Assumptions and dependencies
  - 2.6 Apportioning of requirements
    - Requirements that may be delayed

•Describe other constraints that will limit developer's options; e.g., regulatory policies; target platform, database, network software and protocols, development standards requirements

Requirements Document Template

- **Section 3: Specific Require**
  - 3.1 Functional requirements
  - 3.2 Quality requirements



Specify software requirements in sufficient detail to enable designers to design a system to satisfy those requirements and testers to verify requirements

State requirements that are externally perceivable by users, operators, or externally connected systems

Requirements should include, at a minimum, a description of every input (stimulus) into the system, every output (response) from the system, and all functions performed by the system in response to an input or in support of an output

(a) Requirements should have characteristics of high quality requirements
(b) Requirements should be cross-referenced to their source.
(c) Requirements should be uniquely identifiable
(d) Requirements should be organized to maximize readability

Requirements Document Template

- **Section 3: Specific Requirements**
  - 3.1 External Interfaces
  - 3.2 Functions
  - 3.3 Performance Requirements
  - 3.4 Logical Database Requirements
  - 3.5 Design Constraints
  - 3.6 Software System Quality Attributes
  - 3.7 Object Oriented Models

•Detail all inputs and outputs (complement, not duplicate, information presented in section 2)
•Examples: GUI screens, file formats

•Include detailed specifications of each use case, including collaboration and other diagrams useful for this purpose

•Include:
a) Types of information used
b) Data entities and their relationships

•Should include:
a) Standards compliance
b) Accounting & Auditing procedures

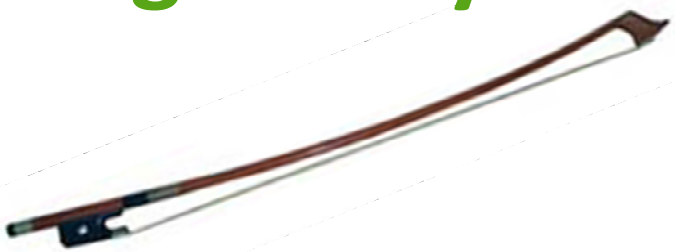
•The main body of requirements organized in a variety of possible ways:
a) Architecture Specification
b) Class Diagram
c) State and Collaboration Diagrams
d) Activity Diagram (concurrent/distributed)

Why do we need a glossary?

- Avoiding (false) interpretations
- Making requirements more understandable
- Contains terms that
  - Could (potentially) be ambiguous in the context considered
  - Are central for the project and the application domain

- **Not a list of all terms used!**

Why do we need a glossary?



- Homonyms
- Synonyms
- Common terms with a specific meaning in the context of the project
- Specific terms of the domain
- Technical terms
- Acronyms and abbreviations
- …
- Company specific and supplementary project specific

Writing requirements

- Functional requirements:
  - Describe a capability that the stakeholder needs or an action the product must take
    (what does the product have to do to complete this use case step?)
  - Use the form: “The product shall be able to do a specific thing (for a specific actor)”
- A single sentence with a single indicative verb - use simple direct language
- Don’t write a solution. Examine your requirements for unwanted technology

Guidelines

- Short and concise sentences and paragraphs
- One requirement per sentence (no compound requirements), no nesting

The 2 most important style rules!

- Consistent terminology
- Avoid generalizations, clear references
- Use ‘must’, ‘can’ and similar words carefully (‘shall’ is better)

Quality gateway

- Each individual requirement must pass through the quality gateway to be added to the specification – it's the only way in …
- Requirements must be …
  - Agreed
  - Ranked
  - Unambiguous
  - Valid and up-to-date
  - Correct
  - Consistent
  - Verifiable
  - Realizable
  - Traceable
  - Complete
  - Understandable
  - …

The rule set!

Requirements rules

- Specify the contents and format of a requirement and requirements document
- Also referred to as standard or guideline
- Rules are agreed upon at a higher level than the author like at organizational (or project) level
- Rules belong to the requirements process (not the review process)
- During reviews the req.'s are 'tested' against the rules (= verification) - objective!
- Increase defect finding capability!

Rules and checklists

- See any defects with this requirement?

“The objective is to get higher adaptability using product Y”

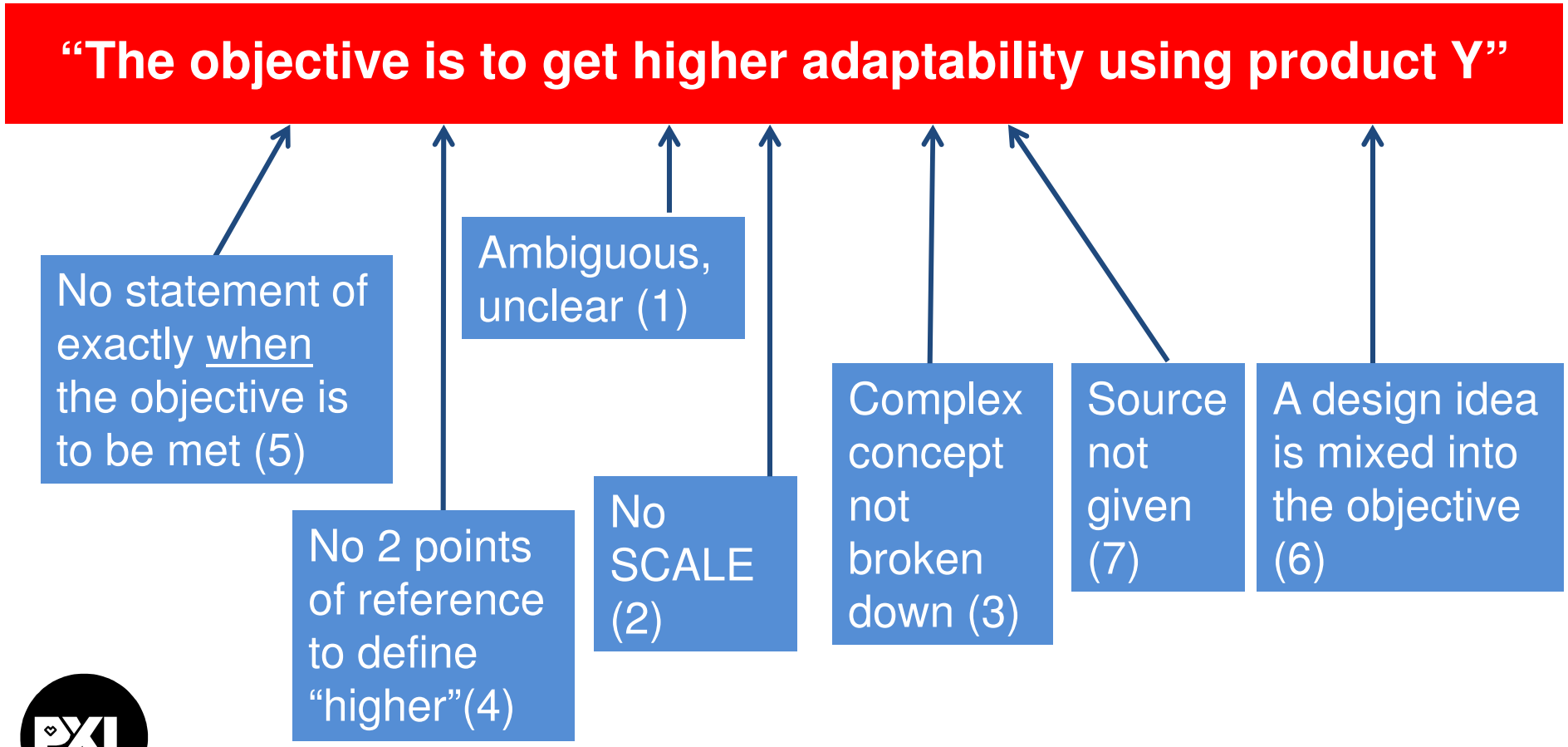
Rules for quality requirements

Example

1. They should be unambiguously clear to the intended reader.
2. They shall specify a scale of measure to define the concept.
3. They shall break down complex concepts into a set of measurable concepts.
4. To define "relative" terms like "higher" they shall specify at least two points of reference on the defined scale.
5. They shall specify exactly when a quality level is to be available.
6. They shall not mix design ideas in the specification of objectives.
7. The process input or "source" (like contract, standard, marketing plan) of the requirement shall be given.

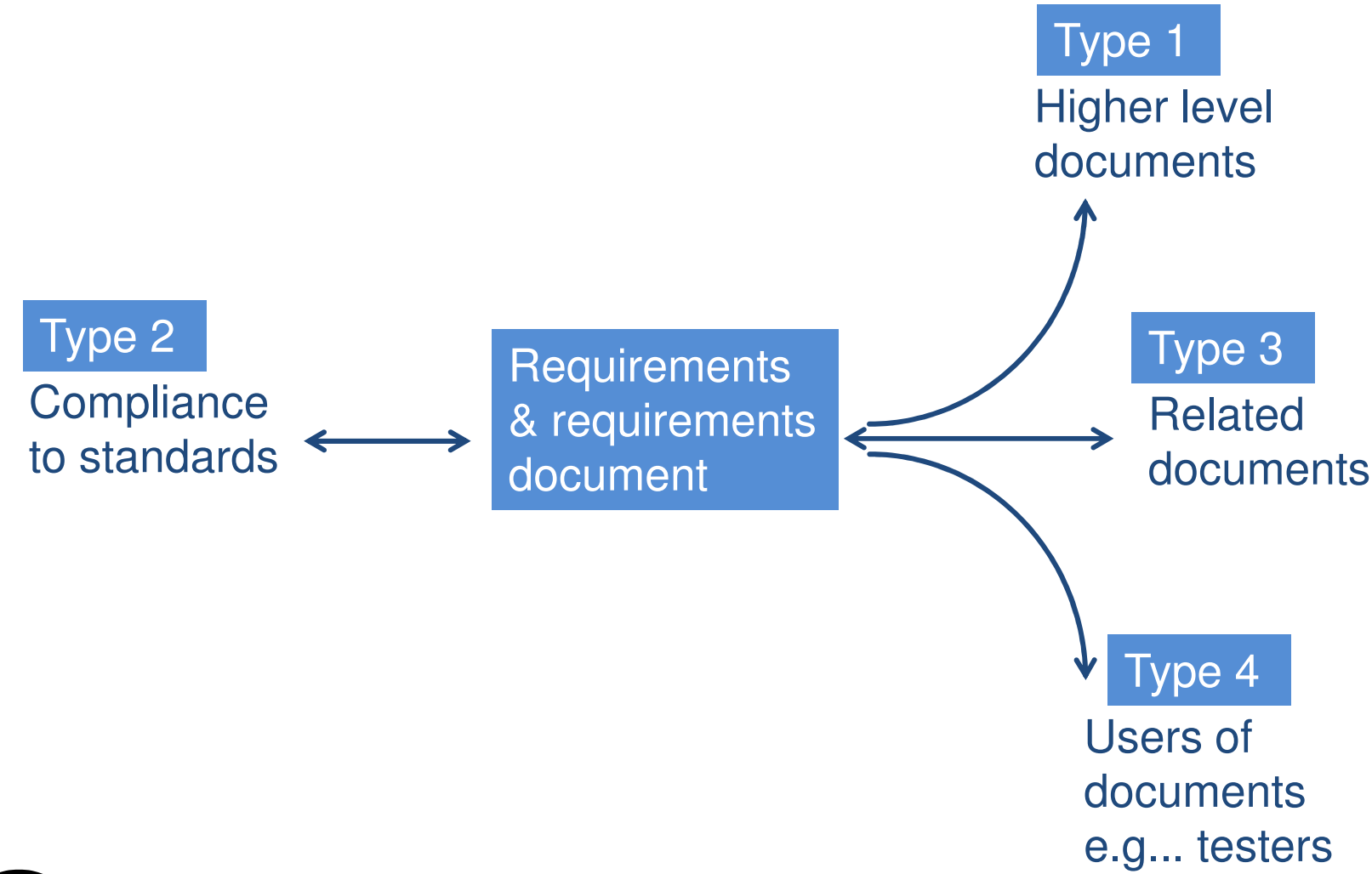
Rules and checklists

- See any defects with this requirement?



Rules

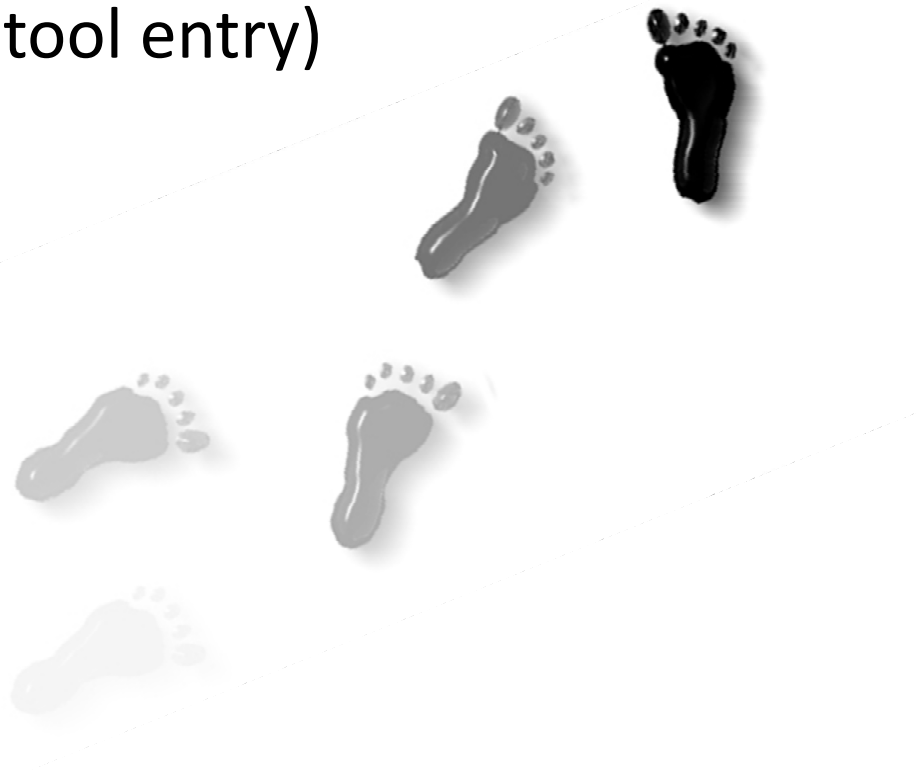
Shall be company specific!



Type 1 & 3 Rules (Tracing)

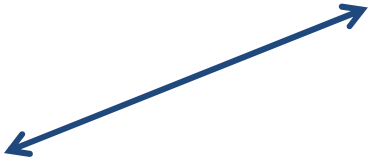
- Necessary
- Each requirement shall be necessary (supported by a higher level entity, e.g. a document, a requirement or a defect management tool entry)

- External consistency
- Complete
- References
- Traceability
- Knowledge responsible



Type 2 Rules (Format)

- Standards
- Identification
- Purpose
- Annotation
- Changes
- Grouping
- Unique
- Internal consistency
- Language

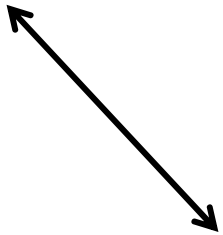


All forms of annotation, comments, notes, suggestions, examples, or other items not part of the official document shall be clearly indicated as such. This will be documented by using the attribute 'additional information' starting with a term that states the type of additional information.

Type 4 Rules (Usage/Content)

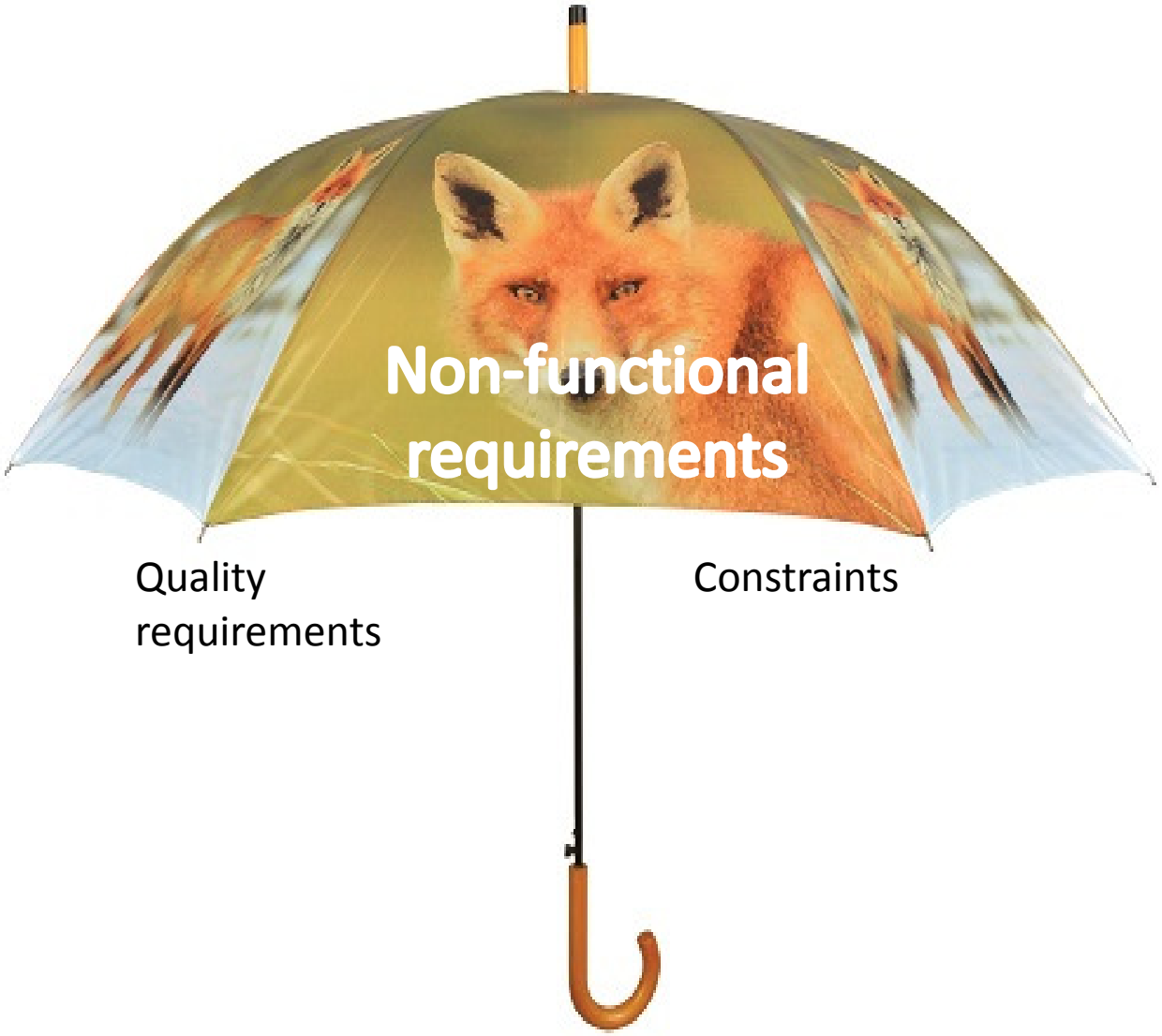
- Detail and specific
- Brief
- Unambiguous
- Level
- Priority
- Rationale
- Quantify
- Compound

- Technically achievable
- Testable
- Uncertain
- Resource feasibility



Req's shall be unambiguous to the intended readership. Req's shall have only one interpretation. For example the word shall is used and not the word should. Ambiguous words like adequate, fault tolerant, and user friendly shall be avoided. Words like can shall only be used when more than one option is available. Preferred directive language (active voice) shall be used, e.g. specifies and not can specify.

Requirements classification

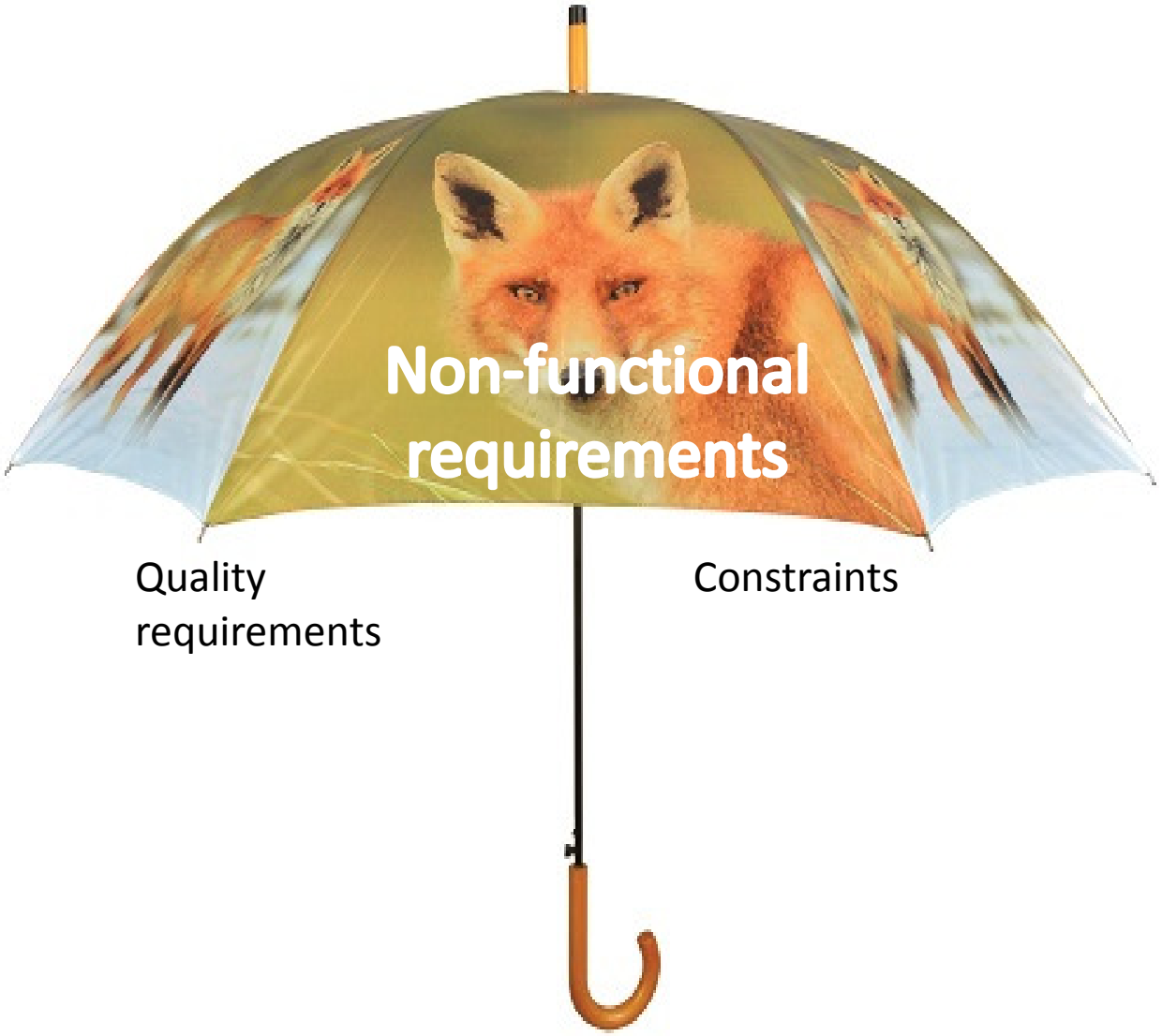


Functional requirements

Quality requirements

Constraints

Requirements classification



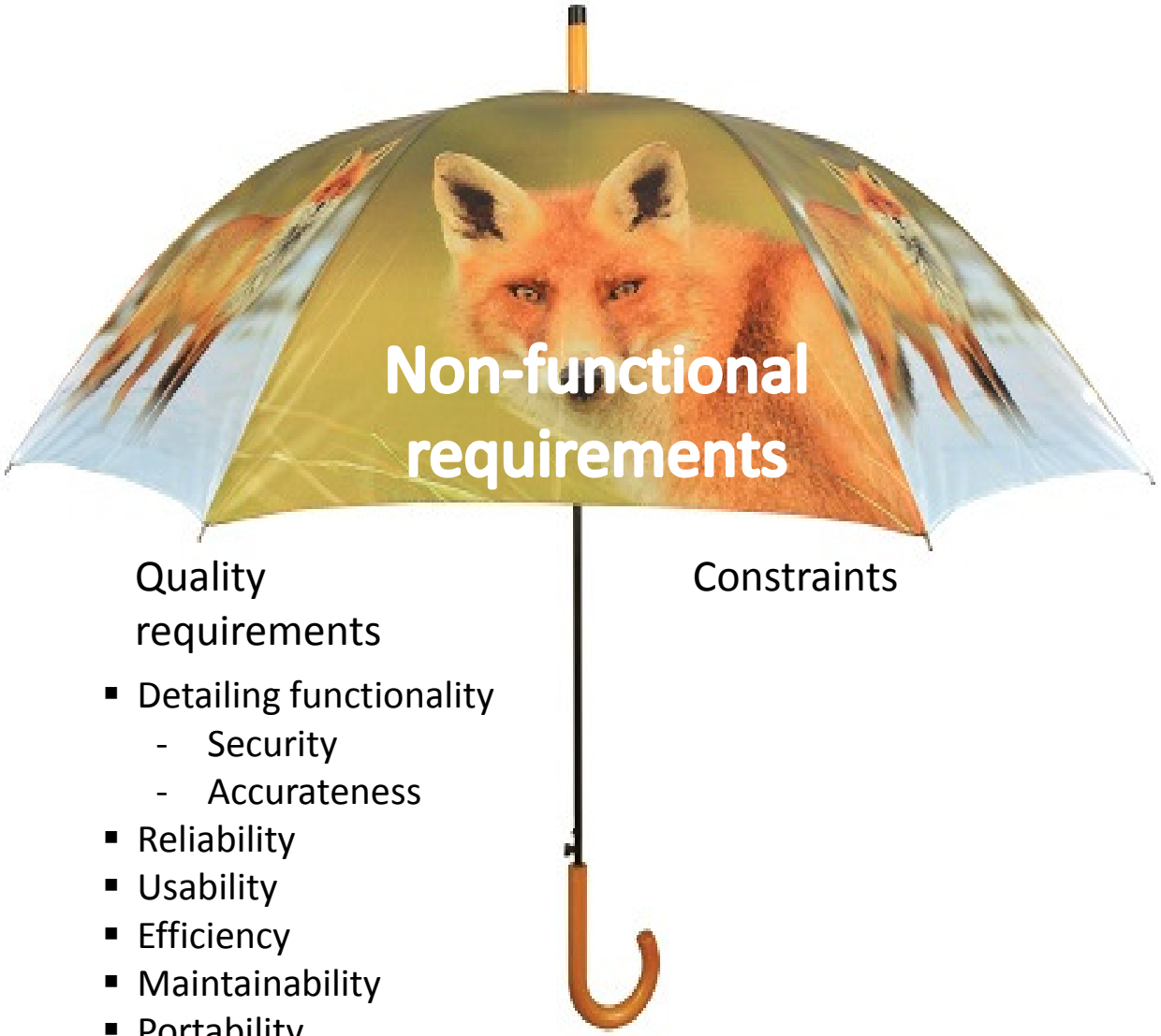
Functional requirements

- Functionality
- Behavior
- Data

Quality requirements

Constraints

Requirements classification



Functional requirements

- Functionality
- Behavior
- Data

Quality requirements

- Detailing functionality
  - Security
  - Accurateness
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

Constraints

Requirements classification

Functional requirements

- Functionality
- Behavior
- Data

Quality requirements

- Detailing functionality
  - Security
  - Accurateness
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

Constraints

Fit criteria

Fit criteria (1)

Involve tester here!

- "Fit" means that a solution completely satisfies the defined requirement
- Need to attach a quantification to the requirement
  - Make it measurable
  - For use cases this is the post condition (outcome)
  - The quantification of the requirement is its fit criterion
- The fit criterion may quantify the behavior, the performance, or some other quality of the requirement
  - So more important for quality requirements
  - Functional requirements should already be testable

Fit criteria (2)

- Fit criteria apply to both functional and non-functional requirements
- !!! Analyze requirement description and determine requirement rationale to find the appropriate scale of measurement for fit criteria
- Make it possible to determine whether a requirement has been satisfied or not
- It is usually necessary to negotiate fit criteria with the stakeholder
  - E.g. 90% of the customers must be able to get the correct ticket from the product in no more than 25 seconds

Fit criteria (3) - Examples from practice

Requirement	Fit criteria (detailed requirements)
The tool shall create a Project based on a Template.	If templates exist, the tool shall provide an existing template for the project. If no templates exist, the project shall be based on an empty template.
The tool shall accept customizations to the list of predefined quality attributes.	The list of predefined quality attributes can be expanded, shrunk and modified.
The tool shall add a Factor to a Project.	A factor shall be unique within the scope of a project.

Fit criteria (4) – Another example

- Functional Requirement
  - Description:
    The system shall record the weather station readings
  - Rationale:
    So readings are not lost
  - Fit criterion:
    The recorded weather station readings shall match the readings sent by the weather station

Fit criteria (5) - Another example

- Non-functional Requirement
  - Description:
    The system shall be user friendly
  - Rationale:
    So new users can learn system fast
  - Fit criterion:
    New users shall be able to add, change and delete roads within 30 minutes of their first attempt at using the product

Fit criteria (6) - Examples from practice

	URS	SRS	DRS
Complete	Not Applicable, since not all BRQs will be translated into URQs.	All URQs are incorporated in one or more SRQs.	All SRQs are incorporated in one or more DRQs. Note that usually there is one SRS and more than one DRS document.
Unambiguous	The requirements shall be at the level of unambiguousness to allow product team level decisions to be taken.	The requirements shall be at the level of unambiguousness to allow for project planning in terms of effort and time.	The requirements shall provide enough information to allow for the execution of individual deliverables and tasks (e.g. detailed design, test design).
Priority	Generic rule applies.	Generic rule applies.	Not applicable. DRQ's do not have a separate priority. The DRQ priority is derived from the SRQ to which it can be traced.

Fit criteria (7)

- Objective measurement to determine if one meets the weight requirement

- Subjective measurements to determine if the wine meets the taste requirements (with some statistical relevance)

Fit criteria (8) – Another example

- Usability requirement example
  - Description:
    The product shall be able to be used by a member of the panel without training
  - Fit criterion:
    90% of a panel of representatives of the general public can successfully complete the task within 5 minutes

Fit criteria (9) – lessons learned

- Quality Requirements
- Many different stakeholders
- Large impact
- Often lead to new functional requirements
- Immense influence on architecture
- Difficult to specify the fit criteria (acceptance criteria)
- Candidates for re-use
- Linked to functional requirements (constraints!)

Requirement card & attributes

Requirements card

Requirement #	:	*Priority*	:
Requirement Type	:	*Use Case*	:
Description	:		
Rationale	:		
Source	:		
Fit Criterion	:		
Supporting material	:	*Annotation:*	

Requirements attributes (1)

- ID
  - To allow traceability
- Type
  - Allows req.'s to be sorted, grouping allows the requirements to be checked on completeness and for conflicts, e.g. by non-functional, by business process
- Use case number
  - For traceability and change control purposes
  - Again for grouping etc.
- Description
  - The intent of the requirement (may initially be ambiguous) - the stakeholders' whishes & needs

Requirements attributes (2)

- Rationale
  - Reason behind the requirement's existence. Helps to clarify and understand the requirement. (to find 'gold plating' req.'s)
- Source
  - Name of the person who raised the requirement.
- Priority
  - Measure of the business importance. For negotiation, but also for risk-based testing.
- Others …
  - Dependencies, supporting materials, annotation, history

Exercise – new clock radio (part 02)

- You have already found defects in the requirements for the new clock radio
- Assignment
  - As a team re-write both functional and quality requirements for the new alarm clock
  - Use the template (requirements cards) and fill out (at least) the following attributes:
    - Unique id, description, rationale and when appropriate the fit criterion. Source = your name!
    - System level requirements, English language


Key Learning Points

Key Learning Points (1)

- Reasons for requirements documentation
- Requirements perspectives
- Creating Requirements
  - Using natural language
    - Prose, very flexible, for all kinds of requirements, no need for stakeholders to learn new modelling techniques
  - Using conceptual models
    - Compact, formal, standardization, Use case diagrams, class diagrams, activity diagrams, state charts
  - A combination of both ➔ For all perspectives

Key Learning Points (2)

- Advantages & disadvantages of natural language
- Advantages of standardized document structures
- One widespread document structure
- Quality criteria for requirements & requirements documents
- Two most important style rules
- Glossaries
- 5 transformational processes
- 5 steps of writing requirements using a template

Questions & answers