Part I: The Fundamentals

Part II: Requirements Engineering Practices

Part III: Enablers and Stumbling Blocks

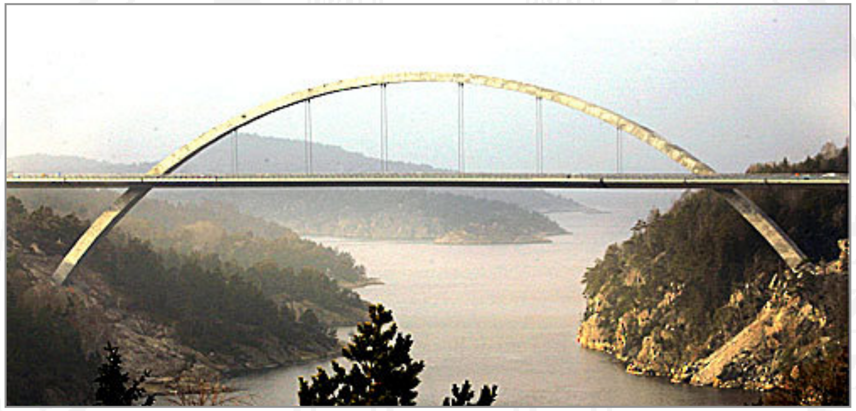
Conclusions

References

5 Documenting requirements

Bridging the gap:

Stakeholders



System builders

Photo © Lise Aserud / DPA

The need:

- Communicating requirements
- Having a basis for contracts and acceptance decisions

The means: Documented requirements

5.1 Requirements Engineering work products

DEFINITION. Work product – A recorded, intermediate or final result of information generated in a work process.
Synonym: artifact

Work products are characterized by their

- Purpose
- Representation (free text, structured text, lists, graphics, drawings,...)
- Size (single requirements, sets of requirements, documents (or document-like structures)
- Lifespan (temporary, evolving, durable)

Note that a work product may contain other work products

Work products and their purposes

Single requirements

- Sentence in natural language – expressing an individual requirement
- User story – specifying a function or behavior from a stakeholder’s perspective

Work products and their purposes – 2

Sets of requirements

- Use case – specifying a system function from a stakeholder's perspective
- Graphic model – specifying various aspects, e.g., context, activity, behavior
- Task description – specifying a task to perform
- External interface – specifying the information exchanged between a system and an actor in the system context
- Epic – providing a high-level view of a stakeholder need
- Feature – A distinguishing characteristic of a system that provides value for stakeholders

Work products and their purposes – 3

Documents and document-like structures

- System requirements specification, business requirements specification, stakeholder or user requirements specification – providing a baselined or released requirements document
- Product and sprint backlog – managing a list of work items, including requirements
- Story map – visual arrangement of user stories
- Vision – providing a conceptual overview of the goals and capabilities of a system

Work products and their purposes – 4

Other RE-related work products

- Glossary – providing an unambiguous and agreed common terminology
- Textual note or graphic sketch – serving for communication and understanding
- Prototype – understanding or validating requirements

5.2 Classic requirements specifications

Full-fledged requirements specifications are typically needed

- When customers want contractually fixed requirements, costs and deadlines
- When systems are built by an external contractor based on a set of given requirements (tendering, outsourcing)
- In regulated environments where regulators check compliance of developed systems to their requirements

Document types

[ISO/IEC/IEEE 2018]

- Stakeholder requirements specification (also called customer requirements specification)
  What the stakeholders want (independent of any system providing it)
- System requirements specification
  The system or product to be developed and its context
- Software requirements specification
  If the system is a pure software system
- Business requirements specification
  High-level specification of business needs or goals

Stakeholder requirements specification

- Written when stakeholder needs shall be documented before any system development considerations are made
- Typically written by domain experts on the customer side (maybe with help of RE consultants)
- If a stakeholder requirements specification is written, it precedes and informs system or software requirements specifications

System/software requirements specification

- The classic form of a requirements specification
- No methodological difference between system requirements specification and software requirements specification
- Typically written by requirements engineers on the supplier side

5.3 Requirements in agile development

No classic requirements specification document (unless mandated by regulators)

Various work products that ...

- ... specify requirements: vision, stories, epics, use cases,...
- ... have requirements-related content: Prototypes, mock-ups, storyboards, roadmap, early product versions (e.g., MVP – minimum viable product)

Value-driven creation of artifacts

Agile requirements work products

- Requirements primarily captured as a collection of user stories, organized in a product backlog
- A system vision provides an abstract overview of the system to be developed
- On an intermediate level of abstraction, epics and features can serve to group user stories
- Stories may be sub-divided into tasks
- Use cases/scenarios and other models may be used to provide structure and context

5.4 Glossary

RE typically is a multi-person endeavor

→ Danger of missing shared understanding in terminology

DEFINITION. Glossary – A collection of definitions of terms that are relevant in some domain.

A glossary defines

- Context-specific terms
- Everyday terms that have a special meaning in the given context
- Abbreviations and acronyms
- Synonyms (different terms denoting the same thing)
- Homonyms (using the same term for different things)

Rules for creating and maintaining a glossary

❍ Consistently structured

❍ Centrally managed

❍ Defined responsibilities for creation and maintenance

❍ Maintained over the entire course of a project

❍ Usage of terms as defined in the glossary is mandatory

❍ Stakeholders must agree upon the glossary

5.5 Prototypes

DEFINITION. Prototype – A preliminary realization of some part of a system serving for exploring, communicating or validating concepts and requirements.

The realization may be in any physical form, from paper and post-its over clickable pages to executable source code.

In RE, a prototype is a means for

- specifying requirements by example
- validating requirements
- supporting stakeholder communication and shared understanding

Forms of Prototypes in RE

[Lichter et al. 1994]

- *Exploratory prototype*:
  - Creating shared understanding
  - Clarifying requirements
  - Validating requirements on different levels of fidelity
  - Thrown away after use

- *Evolutionary prototype*:
  - Pilot system forming the nucleus of a system to be developed
  - Final system evolves by incrementally extending and improving the prototype

Exploratory prototypes

- *Wireframe*
  - Low-fidelity prototype
  - Built with paper or other simple materials
  - Primarily serves for discussing and validating design ideas and user interface concepts
- *Mock-up*
  - Medium-fidelity prototype
  - Real screens and click flows, but without real functionality
  - Primarily serves for specifying and validating user interfaces

Exploratory prototypes – 2

- *Native prototype*
  - High-fidelity prototype
  - Implements critical parts of a system to an extent that stakeholders can work with the prototype
  - Primarily serves for validating that the prototyped part of the system will work and behave as expected

Exploratory prototypes can be expensive work products

- Choose proper level of fidelity
- Trade-off between cost and value gained

5.6 Aspects to be documented

Independently of any language, method, and documentation style, four aspects need to be documented:

- Context
  - Objects, actors and assumptions in the context of a system
  - Embedding of a system in its context
  - Interaction between a system and the actors in the context
- Functionality
  - Data: Usage and structure
  - Functions: Results, preconditions, processing
  - Behavior: Dynamic system behavior as observable by users
  - Both normal and abnormal cases must be specified

Aspects to be documented – 2

❍ Quality

Performance

- Data volume
- Reaction time
- Processing speed
- Specify measurable values if possible
- Specify more than just average values

Specific Qualities

- “-ilities” such as Usability, Reliability, Availability, etc.

Aspects to be documented – 3

- Constraints

  Restrictions that must be obeyed / satisfied

  - Technical: given interfaces or protocols, etc.
  - Legal: laws, standards, regulations
  - Cultural
  - Environmental
  - Physical
  - Solutions / restrictions demanded by important stakeholders

5.7 How to document

Sample standards for classic requirements documents

IEEE Std 830-1998 (outdated, but still in use)

- Three parts
- System requirements only
- Representation of specific requirements tailorable

VOLERE

- 27 chapters
- System and project requirements

Enterprise-specific standards

- Imposed by customer or given by supplier

IEEE Std 830-1998

[IEEE 1988]

1. Introduction
   - 1.1 Purpose
   - 1.2 Scope
   - 1.3 Definitions, acronyms, and abbreviations
   - 1.4 References
   - 1.5 Overview
2. Overall description
   - 2.1 Product perspective
   - 2.2 Product functions
   - 2.3 User characteristics
   - 2.4 Constraints
   - 2.5 Assumptions and dependencies
3. Specific requirements

Appendixes

Index

Variants:
Organize by
- Mode
- User class
- Object
- Feature
- Stimulus
- Function

VOLERE

[Robertson and Robertson 2012]
[https://www.volere.org/templates/]

Project Drivers

1. The Purpose of the Project
2. The Stakeholders

Project Constraints

3. Mandated Constraints
4. Naming Conventions and Terminology
5. Relevant Facts and Assumptions

Context and Functionality

6. The Scope of the Work
7. Business Data Model & Data Dictionary
8. The Scope of the Product
9. Functional Requirements

Non-Functional Requirements

10. Look and Feel Requirements
11. Usability and Humanity Requirements
12. Performance Requirements
13. Operational & Environmental Requirements
14. Maintainability and Support Requirements
15. Security Requirements
16. Cultural Requirements
17. Complience Requirements

Project & Product Issues

18. Open Issues
19. Off-the-Shelf Solutions
20. New Problems
21. Tasks
22. Migration to the New Product
23. Risks
24. Costs
25. User Documentation and Training
26. Waiting Room
27. Ideas for Solutions

Subtitles added by MG, inspired by an earlier version of the template

Guidelines for agile requirements

- Standard template for writing user stories (cf. Chapter 8)
- Organizing stories in a product backlog
- Artifact / work product structures provided by textbooks

[Leffingwell 2011]

General guideline: do things only if they add value

How to document – language options

Informally

- Plain natural language (narrative text)

Semi-formally

- Structured natural language (using templates or forms)
- Graphic models

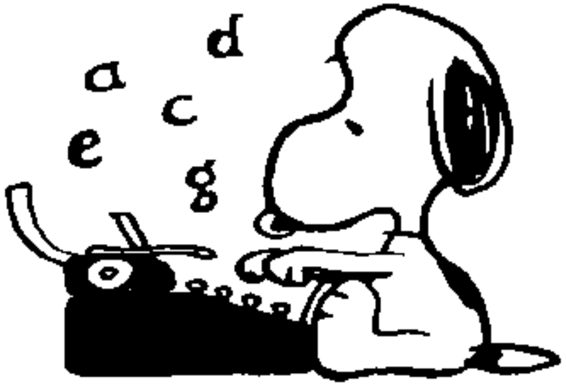
  Typically as diagrams which are enriched with natural language text

Formally

- Formal models, typically based on mathematical logic and set theory

General rules for requirements documentation

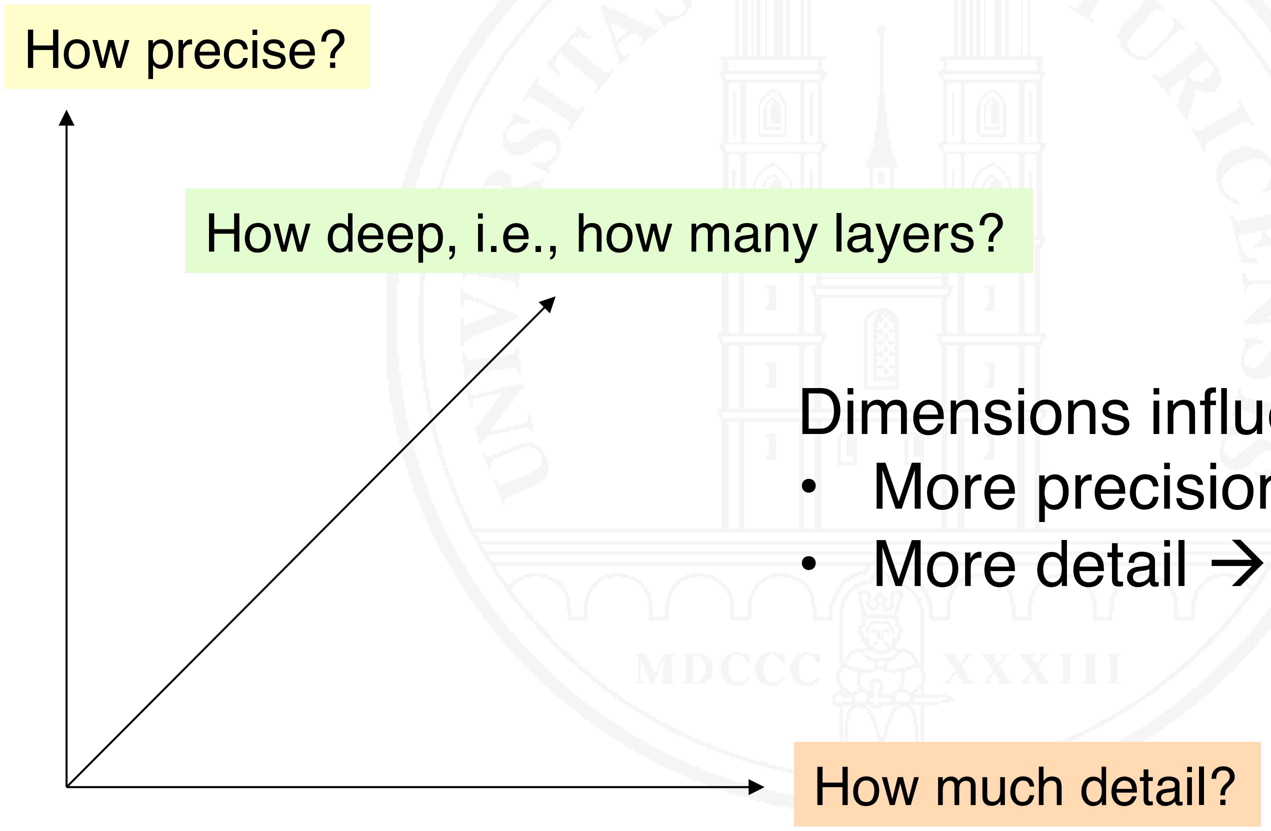
- Specify requirements as small, identifiable units whenever possible
- Record metadata such as source, author, date, status
- Use structure templates
- Adapt the degree of detail to the risk associated with a requirement
- Specify normal and exceptional cases
- Don’t forget quality requirements and constraints



© UFS, Inc.

Precision – Detail – Depth

Three dimensions:



How precise?

How deep, i.e., how many layers?

How much detail?

Dimensions influence each other:

- More precision → more detail
- More detail → more depth

Precision: reduce ambiguity

Restrict your language

Use a glossary

Define acceptance test cases

Quantify where appropriate

Formalize

Meine Liebe,
Du fehlst mir morgens, mittags und abends.
5-8
DAS IST ZU UNGENAU...
WENN DU EINEM MÄDCHEN SCHREIBST, MUSST DU DICH PRÄZISE AUSDRÜCKEN.
Du fehlst mir um 8 Uhr 15, 11 Uhr 45 und 21 Uhr 36...
SCHULZ
© 1985 United Feature Syndicate, Inc.

Snoopy quantifies ... unfortunately, I have it only in German

Detail

What's better?

"The participant entry form has fields for name, first name, sex, ..."

"The participant entry form has the following fields (in this order): Name (40 characters, required), First Name (40 characters, required), Sex (two radio buttons labeled male and female, selections exclude each other, no default, required),..."

It depends.

- Degree of implicit shared understanding of problem
- Degree of freedom left to designers and programmers
- Cost vs. value of detailed specification
- The risk you are willing to take

Depth

The more precise, the more information is needed

➜ Preserve readability with a hierarchical structure

"...

4.3 Administration of participants

4.3.1 Entering a new participant

4.3.1.1 New participant entry form

4.3.1.2 New participant confirmation

4.3.2 Updating a participant record

..."

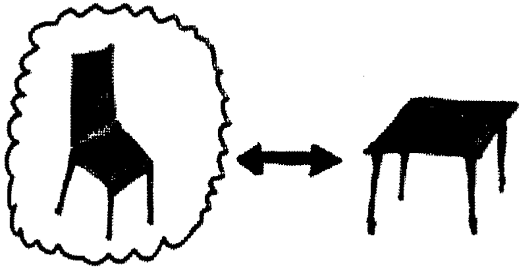
5.8 Quality of documented requirements

Two aspects of requirements quality

- Quality of individual requirements
- Quality of requirements specification documents

Hint: Don’t confuse quality of requirements with quality requirements

Quality of individual requirements



For individual requirements, strive for requirements that are...

● Adequate	True and agreed stakeholder needs
● Understandable	Prerequisite for shared understanding
● Verifiable	Conformance of implementation can be checked
● Unambiguous	True shared understanding
● Complete	No missing parts
● Feasible	Non-feasible requirements are a waste of effort
● Necessary	Part of the relevant system scope
● Traceable	Linked to other requirements-related items

Quality of requirements artifacts

When creating a requirements specification, strive for a document that is

● Consistent	No contradictions
● Complete	Contains all relevant requirements
● Conformant	Conforms to prescribed artifact structure, format or style
● Modifiable	Because change will happen
● Non-redundant	Requirements do not overlap
● Structured	Improves readability of artifact
● Traceable	Linked to related artifacts

Quality criteria are in the eye of the beholder

- No general consensus
- Different, overlapping sets of quality criteria used in
  - this course
  - RE textbooks
  - RE standards (e.g., ISO/IEC/IEEE 29148:2018)
  - Quasi-standards such as the IREB Certified Professional for Requirements Engineering (see http://www.ireb.org)

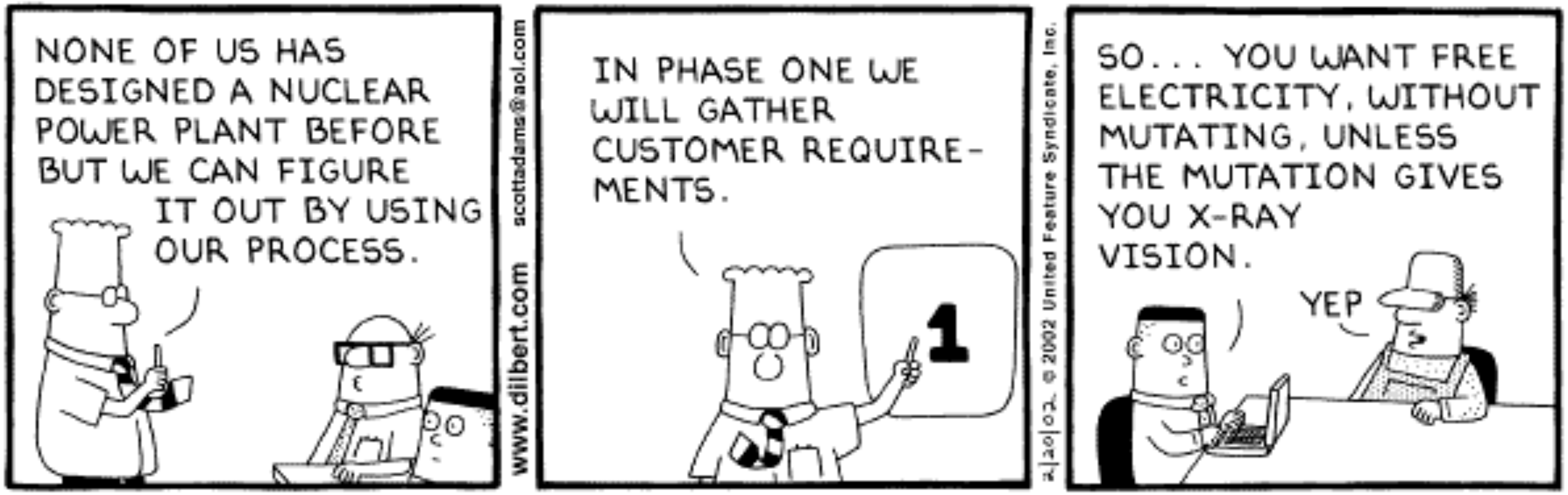
Not all qualities are equally important

- Adequacy and understandability are key
- Verifiability and Consistency are very important
- Achieving total completeness and unambiguity is neither possible nor economically feasible in most cases
- The importance of feasibility, traceability, conformance, etc. of requirements depends on the concrete project/situation

☞ Strive for value, not for blind satisfaction of requirements quality criteria!

6 Requirements Engineering processes

DEFINITION. Process – A set of interrelated activities performed in a given order to process information or materials.


NONE OF US HAS DESIGNED A NUCLEAR POWER PLANT BEFORE BUT WE CAN FIGURE IT OUT BY USING OUR PROCESS.
www.dilbert.com scottadams@aol.com
IN PHASE ONE WE WILL GATHER CUSTOMER REQUIRE-MENTS.
1
2/20/02 © 2002 United Feature Syndicate, Inc.
SO... YOU WANT FREE ELECTRICITY, WITHOUT MUTATING, UNLESS THE MUTATION GIVES YOU X-RAY VISION.
YEP

[Armour 2004, Reinertsen 1997, 2009]

The principal tasks

Requirements Specification

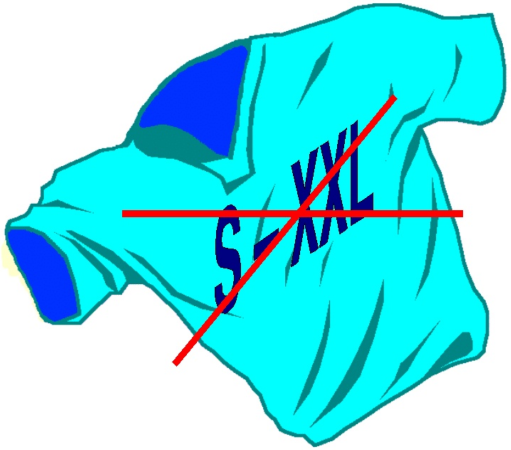
- Elicitation & Analysis
- Documentation
- Validation

Requirements Management

- Identification and metadata
- Requirements prioritization
- Change and release management
- Traceability

An RE process organizes how to carry out RE tasks, using appropriate practices and producing needed work products

No ‘one size fits all’ process



Some influencing factors

- The embedding development process
- Size and criticality of system
- Degree of shared understanding
- Project type (Customer order or development for a market; new system or evolving an existing one; developing or using COTS
- Availability of stakeholders
- Constraints

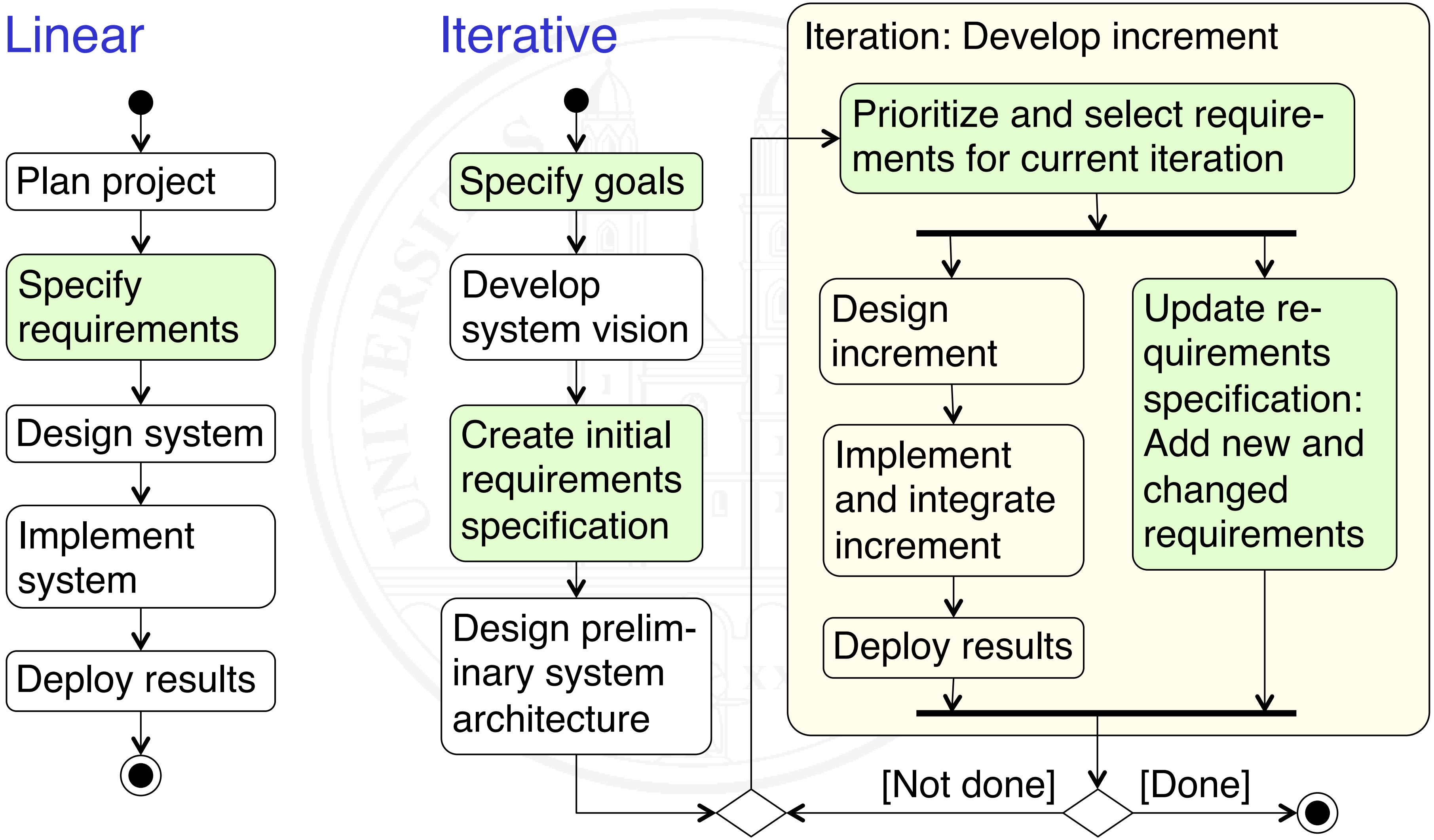
➪ Tailor the process from some principal configuration options and a rich set of RE practices

Process facets

There are three process facets, from which an RE process can be configured

❍ Time facet:	Linear vs. Iterative
❍ Purpose facet:	Prescriptive vs. Explorative vs. COTS-Driven
❍ Target facet:	Customer-Specific vs. Market-Oriented

Time facet: Process structure


Linear
Plan project
Specify requirements
Design system
Implement system
Deploy results
Iterative
Specify goals
Develop system vision
Create initial requirements specification
Design preliminary system architecture
Iteration: Develop increment
Prioritize and select requirements for current iteration
Design increment
Implement and integrate increment
Deploy results
Update requirements specification: Add new and changed requirements
[Not done]
[Done]

Time facet – Selection criteria

- Linear
  - System development process is plan-driven and mostly linear
  - Clear, stable, a priori known requirements
  - Comprehensive requirements specification required as a contractual basis for outsourcing design and implementation
  - Regulatory authorities require a requirements freeze
- Iterative
  - Evolving requirements – not known up-front
  - Short feedback loops established for mitigating risk
  - Long duration of project
  - Ability to change requirements easily is important

Purpose facet: Prescriptive – Explorative

Prescriptive

Requirements specification is a contract: All requirements are binding and must be implemented

Selection criteria:

- Functionality determines cost and deadlines
- Customer requires fixed-price contract
- Design and implementation tendered or outsourced

Explorative

Only goals known, concrete requirements have to be explored

Selection criteria:

- Stakeholders strongly involved, continuous feedback
- Deadlines and cost constrain functionality
- Prioritizing and negotiating requirements to be implemented

Purpose facet: COTS-Driven

COTS-Driven

Requirements must reflect functionality of chosen COTS solution

Selection Criteria:

- System will be implemented with COTS software
- Only requirements not covered by the COTS solution shall be specified

COTS (Commercial Off The Shelf) – A system or component that is not developed, but bought as a standard product from an external supplier

Target facet

Customer-Specific

System is ordered by a customer and developed by a supplier for this customer

Selection criteria:

- Individual persons identifiable for all stakeholder roles
- Stakeholders on customer side are main source for requirements
- Contractual customer-supplier relationship affects process

Market-Oriented

System is developed as a product or service for a market

Selection criteria:

- Prospective users not individually identifiable
- Requirements specified by supplier
- Supplier has to guess/estimate/ elicit the needs of the envisaged customers/users
- Marketing people, digital designers and system architects are primary stakeholders

Caveats and hints

- Linear RE processes only work if a sophisticated process for changing requirements is in place
- Linear RE processes imply long feedback loops: intensive validation of requirements must be performed
- Market-oriented RE processes crucially depend on fast feedback from pilot users for validating whether the product will actually satisfy needs of the targeted user segment
- *Market-Oriented* does not combine well with *Linear* and *Prescriptive*
- Frequent combinations: *Linear* and *Prescriptive*
  Explorative and *Iterative*

Typical RE process configurations

Participatory: Iterative & Explorative & Customer-Specific

- Main application case
  Supplier and customer closely collaborate; customer stakeholders strongly involved both in specification and development processes
- Typical work products
  Product backlog with user stories and/or task descriptions, prototypes
- Typical information flow
  Continuous interaction between stakeholders, product owners, requirements engineers, and developers

Typical RE process configurations – 2

Contractual: Typically Linear (sometimes Iterative) & Prescriptive & Customer-Specific

- Main application case
  Specification constitutes contractual basis for development of a system by people not involved in the specification and with little stakeholder interaction after the requirements phase
- Typical work products
  Classic system requirements specification, consisting of textual requirements and models.
- Typical information flow
  Primarily from stakeholders to requirements engineers

Typical RE process configurations – 3

Product-oriented: Iterative & Explorative & Market-Oriented

- Main application case
  An organization specifies and develops software in order to sell/ distribute it as a product or service
- Typical work products
  Product backlog, prototypes
- Typical information flow
  Interaction between product owner, marketing, requirements engineers, digital designers, developers and (maybe) fast feedback by (pilot) customers/users

Typical RE process configurations – 4

COTS-Aware: [Iterative I Linear] & COTS-Driven & Customer-Specific

- Main application case:
  The requirements specification is part of a project where the solution is mainly implemented by buying and configuring COTS
- Typical work products:
  Process models describing the alignment of business processes and the COTS solution, partial requirements specification, covering what is not provided by the COTS solution
- Typical information flow:
  Primarily from stakeholders and COTS solution experts to requirements engineers

Agile requirements process

Pushes incrementality and exploration to the extreme

- Fixed-length iterations of 1-6 weeks
- Product owner or customer representative always available and has power to make immediate decisions
- Only goals and vision established upfront
- Requirements loosely specified as stories (with details captured in acceptance criteria)
- Use cases or other means used for providing structure & context
- At the beginning of each iteration
  - Customer/product owner prioritizes requirements
  - Developers select what to implement in that iteration
- Short feedback cycle from requirements to deployed system

Characteristics of an "ideal" RE process

- Strongly interactive
- Close and intensive collaboration between
  - Stakeholders (know the domain and the problem)
  - Requirements engineers (know how to specify)
- Very short feedback cycles
- Risk-aware and feasibility-aware
  - Technical risks/feasibility
  - Deadline risks/feasibility
- Careful negotiation / resolution of conflicting requirements
- Focus on establishing shared understanding
- Strives for innovation