UNIVERSITÄT DUISBURG ESSEN

Offen im Denken

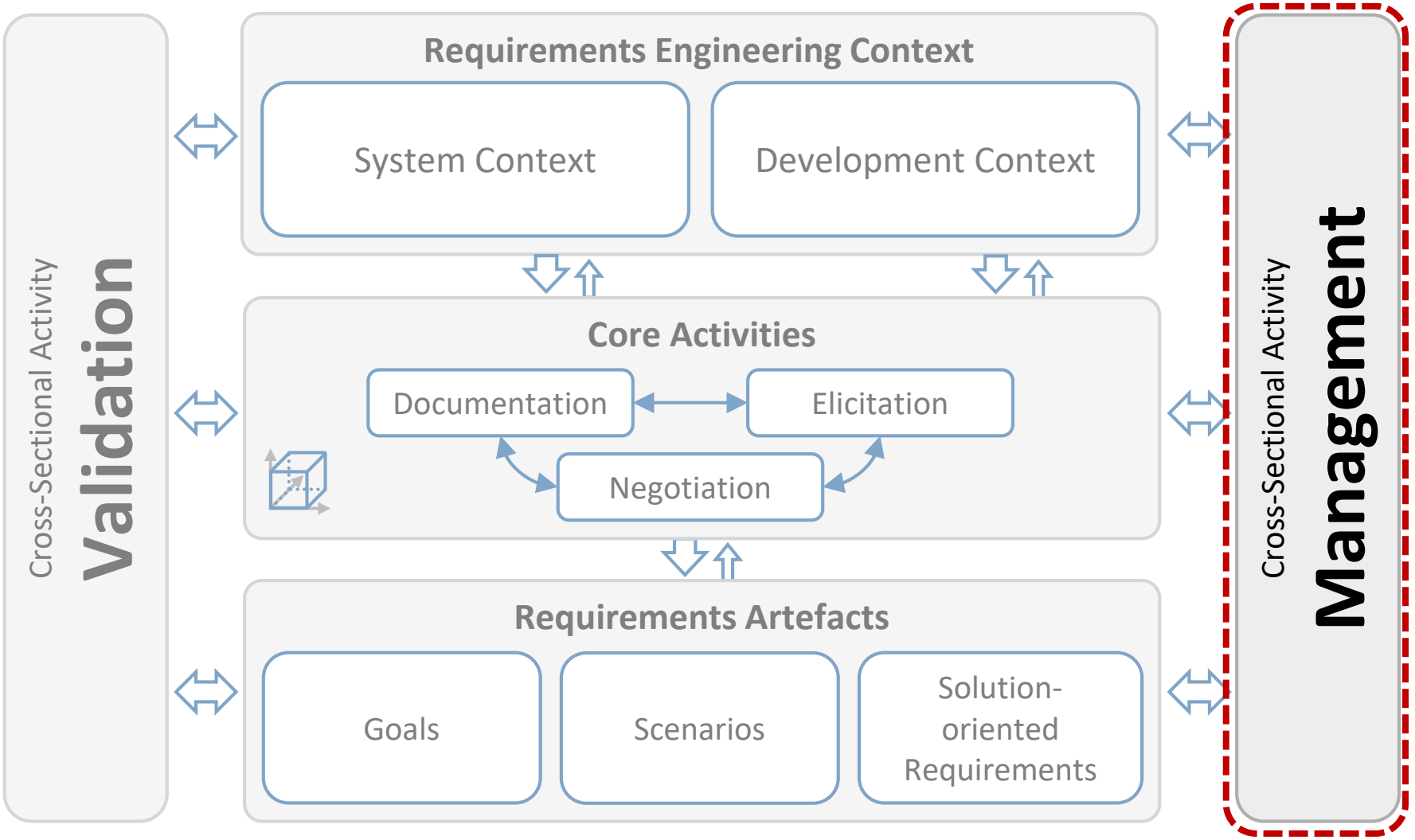
Requirements Engineering & Management

Cross-Sectional Activities – Management

Prof. Dr. Klaus Pohl

PALUNO
The Ruhr Institute for Software Technology

Framework for Requirements Engineering



Agenda

1. Introduction to Requirements Management
2. Requirements Prioritization
3. Version and Configuration Management of Requirements Artefacts
4. Requirements Traceability
5. Change Management of Requirements

1. Introduction to Requirements Management

Goals of the Management Activity

D Management is one of the two cross-sectional requirements engineering activities. The goal of management in requirements engineering is to:

1. **Observe** the requirements engineering **context** to **detect context changes**.
2. **Manage** the execution of requirements engineering **activities**.
3. **Manage** the requirements **artefacts**.

Observing the Requirements Engineering Context

Goal: Identify the changes in the context and estimate the impact of these changes.

- Context changes can occur in each part of the requirements engineering context.
- A major challenge for context observation is to detect and anticipate changes in the context as early as possible.
- An important aspect of context observation is the selection of specific observation techniques.
  - Context objects critical to the success of the system (e.g., laws and regulations) should be observed using continuous context monitoring.
  - For less critical aspects, periodical scanning of the context can be sufficient.

Managing the Requirements Artefacts

Goal: Continuously **keep track** of all requirements **artefacts**, their relevant **attributes** and **relationships** as well as their **evolution**.

- Five main sub-tasks:
  - Definition of a requirements **attribute scheme**
  - Requirements **traceability**
  - Requirements **change management**
  - Requirements **configuration management**
  - Requirements **prioritization**

Managing the Requirements Engineering Activities

Goal: **Monitor**, **control**, and **adjust** the **planned workflow** of elicitation, documentation, negotiation, validation and management activities.

Two basic approaches:

- **Phase-oriented approach**:
  Same sequence of activities is applied to all requirements artefacts.
- **Situational approach**:
  Determines the activities to be executed next based on an assessment of the current status of the existing requirements artefacts.

2. Requirements Prioritization

Goals of Requirements Prioritization

Problem:

- Due to **resource constraints** and **limitations** (time, budget, capacity of people, …) not all requirements can be considered with the same intensity and be realized to the same degree.

Goals:

- Use of limited resources in a **systematic way** to ensure requirements are considered **in order of** their **importance**
- **Classification** of requirements **into priority classes**
- **Ordering of priority classes** according to a certain goal

Priority of Requirements

D The **priority of a requirement** documents the **importance** of the requirement with regard to one or several **prioritization criteria**. The priority may be determined either **for each** requirement **in isolation** or by **pairwise comparison** of requirements.

- A **prioritization goal** defines the purpose of prioritizing requirements and indicates the prioritization criteria to be used.
- **Prioritization criteria** are used to determine the priority of requirements.

Prioritization Goals in the RE Activities (1)

- **Elicitation**
  - Which requirements should be elaborated next?
  - Which requirement sources (e.g., stakeholders) should be considered first?
- **Documentation**
  - Determine an ordering in which the requirements shall be documented to satisfy the predefined documentation rules and formats.
- **Negotiation**
  - Requirements can be prioritized with respect to their influence on the project success.
  - The prioritization can then be used to resolve the conflicts for the most important requirements first.

Prioritization Goals in the RE Activities (2)

- **Validation**
  - Determine the order in which the requirements should be validated.
  - Determine different intensities of validation for the requirements.
  - Define the order in which detected defects should be resolved.
- **Management**
  - Define which change requests should be processed first - most urgent changes are integrated first.

- In addition, **requirements prioritization** is essential for many **other activities** during the development process.
  - Design, implementation, project management, product management, release planning, etc.

Prioritization Process

- Requirements prioritization should always be **directed by a clear goal** and should **involve all relevant stakeholders.**

- Prioritization preparation **activities** :
  1. Determine **artefacts** to be prioritized.
  2. Select **prioritization criteria**.
  3. Identify relevant **stakeholders**.
  4. Select one or more **prioritization techniques**.

1. Determining the Artefacts

- To prevent prioritization errors, only artefacts of the **same type**, and on the **same level of abstraction should be jointly prioritized**.
- Hint: Prioritize **top-down** (high-level requirements first, e.g., stakeholder goals)

E

Prioritization goals for different artefact types:

1. Goals: Determining relevance for market success
2. Use cases: Determining implementation in releases
3. Scenarios: Determining criticality of exception scenarios
4. …

[1]

2. Selecting Prioritization Criteria (1)

Typical prioritization criteria include:

- **Importance**: Urgency of implementing, importance for acceptance of the system, importance for architectural design, strategic importance to market position
- **Cost**: Financial resources needed for implementation
- **Damage**: Disadvantage from neglecting the requirement
- **Duration**: Time needed to realize the requirement
- **Risk**: Realization risk of a requirement (e.g., due to technological problems)
- **Volatility**: Likelihood of a requirement changes
- **Business value**: Value for the business (e.g. Scrum uses business value/cost benefit to prioritize use cases to be implemented first)

Prioritization can be performed according to **a single or multiple** prioritization criteria.

[1]

2. Selecting Prioritization Criteria (2)

The **prioritization goal** indicates the **criteria** to be used **for prioritizing** the requirements.

Prioritization criteria relevant for different goals:

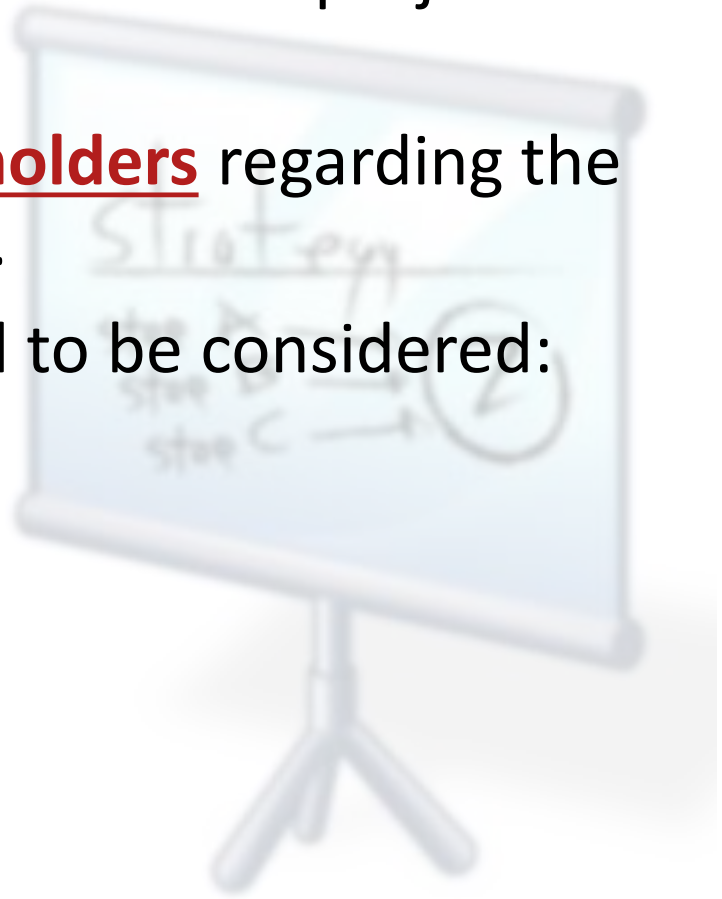
- Resolving conflicts:
  - Damage – Resolve conflicts for high-damage requirements first.
  - Volatility – Volatile requirements are likely to cause more conflicts.
- Intensity and order of validation activities:
  - Cost – Early validation may avoid costly modifications later.
  - Duration – Early validation may avoid additional development time.
- Order for documentation:

  Importance – Acceptance criteria should be documented in detail.

[1]

3. Determining the Stakeholders

- Each stakeholder has **specific knowledge** about the project context and the requirements to be prioritized.
- To ensure **correct results**, **relevant stakeholders** regarding the prioritization criteria need to be **involved**.
- Typically, the following **stakeholders** need to be considered:
  - Development team
  - Project management
  - Customer/user
  - Quality assurance team



[1]

4. Selecting a Prioritization Technique (1)

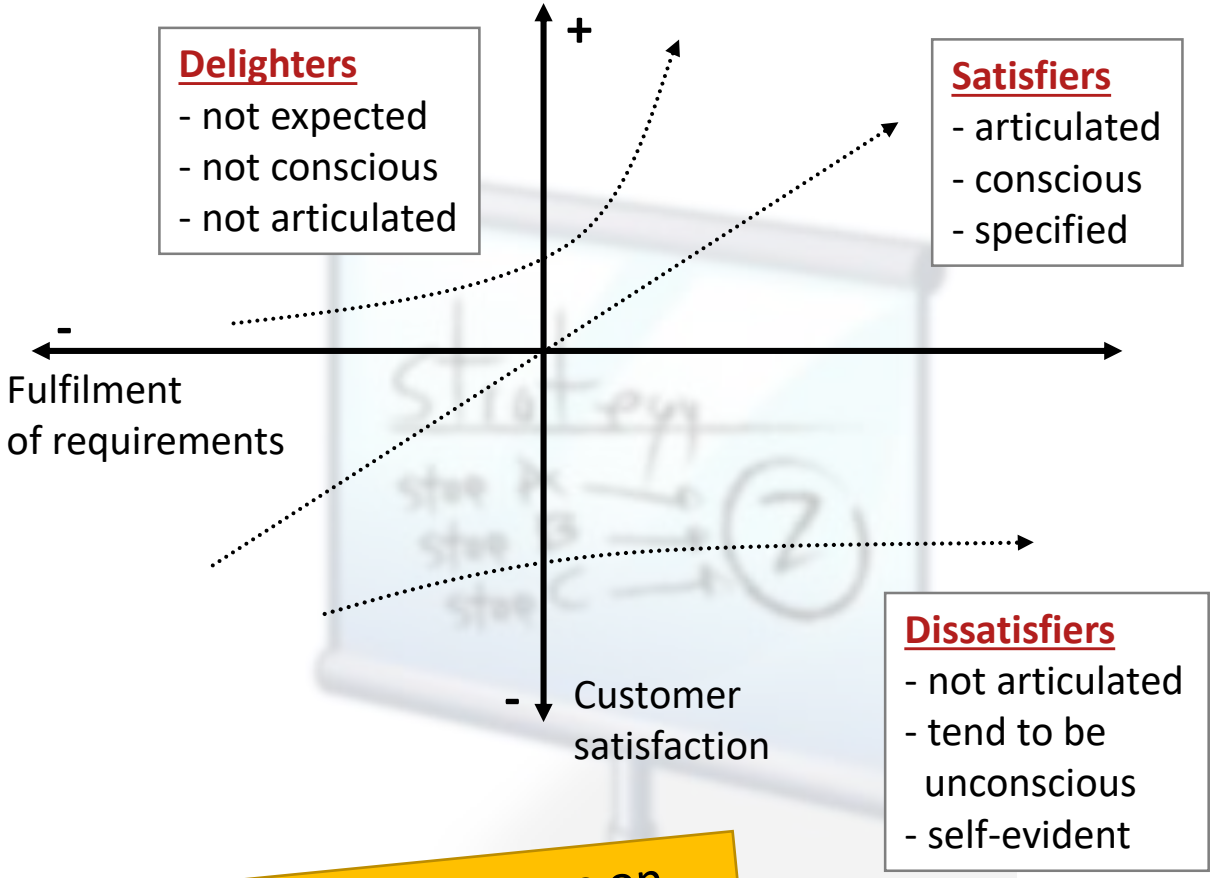
Frequently used **prioritization techniques**:

- Ad-hoc Ranking:
  - Typically done for one prioritization criterion (e.g., risk)
  - **Avoid** three point scales (e.g., high, medium, low) with meaningless, "neutral" values
  - Use a **four point** scale (e.g., high, likely high, likely low, low) for ranking requirements instead!
- Top-Ten:
  - Select a fixed number of **top-priority requirements** (typically ten)
  - Rank these requirements in descending order according to the defined prioritization criterion
- One-Criterion Classification
  - **Define meaningful classes** for assigning priority values to requirements
    - For example, "relatively stable": the likelihood for a change is less than 10%
  - Classify the requirements accordingly
- ...

[1]

4. Selecting a Prioritization Technique (2)

- **Kano Classification:**
  - Classification according to the **influence** of a requirement's realization on **customer satisfaction**
  - Three **classes** of requirements:
    - **Dissatisfier**: Must-be requirement
    - **Satisfier**: Requirement consciously demanded by the customer
    - **Delighter**: Attractive requirement the customer is not (yet) aware of



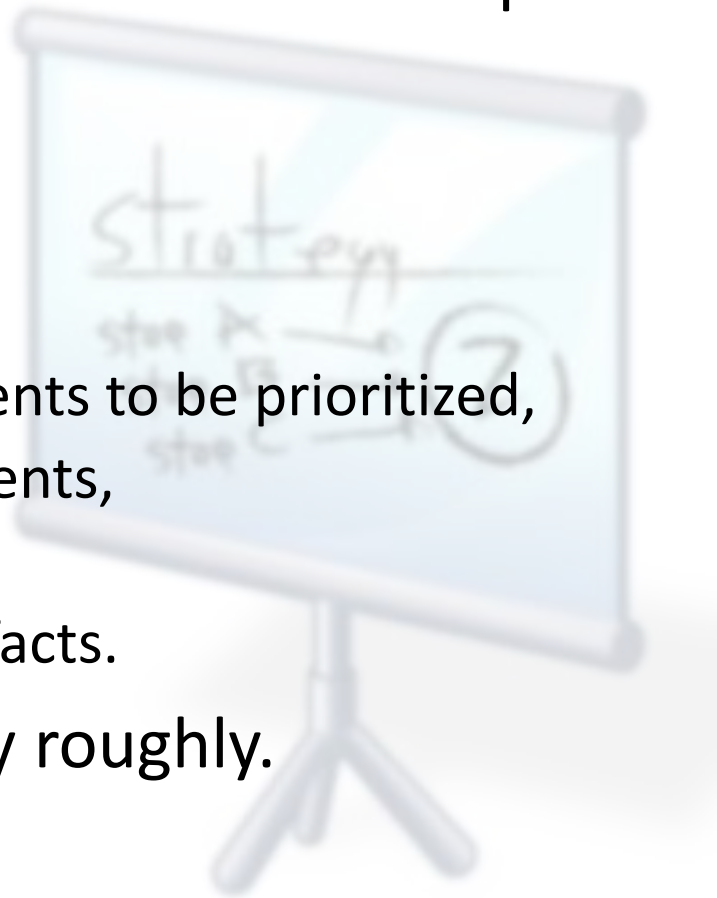
based on [Kano et al. 1984]

More details in Lectures on "Requirements Elicitation"

[1]

4. Selecting a Prioritization Technique (3)

- **Factors** influencing the **selection** of a prioritization technique:
  - Number of requirements to be prioritized
  - Effort of applying the technique
- **Prioritization effort** is influenced by:
  - the number of stakeholders involved,
  - the degree of understanding of the requirements to be prioritized,
  - the degree of agreement about the requirements,
  - the prioritization criteria used, and
  - the content defined in the requirements artefacts.
- **Prioritization** effort can be estimated only roughly.



[1]

3. Version and Configuration Management of Requirements Artefacts

Configuration and Release Management

D Configuration management (CM) is the discipline of **controlling changes** in large and complex systems. Its goal is to prevent the chaos caused by the numerous **corrections**, **extensions**, and **adaptations** which are applied to any large system over its lifetime. Based on [Tichy 2003]

- Requirements artefacts are (just like other artefacts such as source code) **configuration items** that change over time and need to be managed.

D Software release management is "the process through which software is **made available** to and obtained by its users". [Hoek et al. 1997]

- Configuration and release management for the software are **based on the requirements artefacts** (i.e. configurations of requirements).
- A **system release** reflects a certain configuration of requirements artefacts which are implemented and delivered to the customer.

Configuration Items in Requirements Engineering (1)

- A **configuration item** is a **single entity** in the **configuration management process** which can change over time.
- The **requirements engineer** has to **define the configuration items**.
- **Potential configuration items** could include:
  - A single requirement artefact
  - A set of requirements artefacts (of the same or different types)
  - A requirement document
  - A functional model
  - A behavioural model
  - A data model
  - …

Configuration Items in Requirements Engineering (2)

We differentiate between **three key levels of configuration items**:

- **Document level**:
  At this level, documents are the smallest unit (configuration item) to be considered during configuration management.
- **Requirements artefact level**:
  At this level, requirements artefacts (e.g., goals, scenarios, solution-oriented requirements) are the smallest unit (configuration item) to be considered.
- **Requirements attribute level**:
  At this level, individual attributes of requirements artefacts are the smallest unit (configuration item) to be considered.

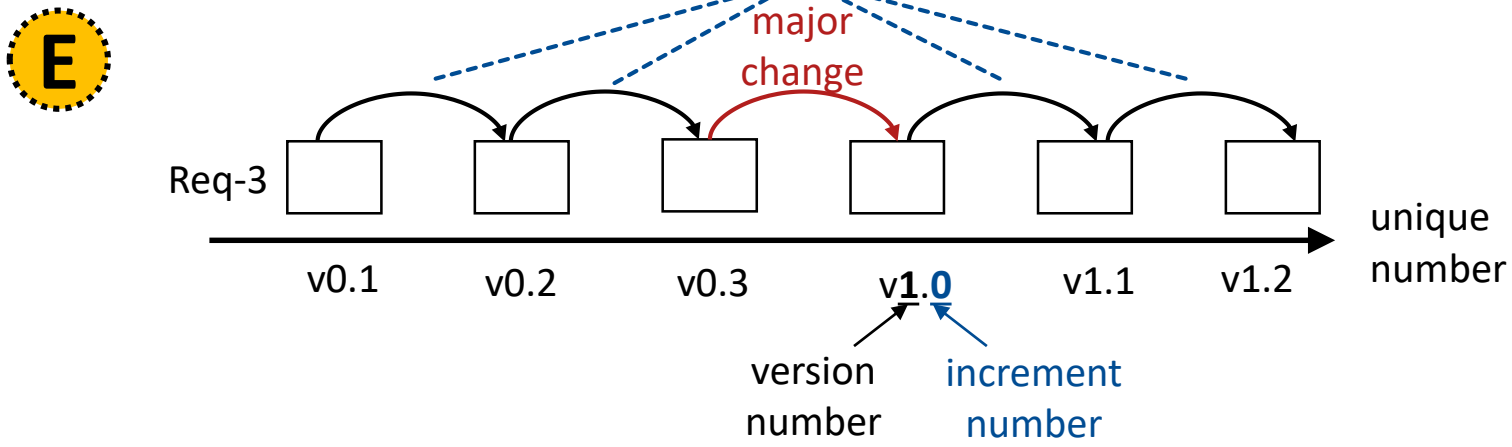
Versions of Requirements Artefacts (1)

Requirements artefacts change over time for various reasons, including:

- A requirements artefact can be **specified at different levels of details** over time. (→ content dimension)
- At a given time, stakeholders may agree with the defined artefact. They might well disagree with the artefact definition later on, i.e. **the agreement achieved** about an artefact **can change over time**. (→ agreement dimension)
- The artefact might be represented using **different representation formats over time**, e.g., informal at the beginning, more formal ones towards the end of the requirements engineering process. (→ documentation dimension)
- A requirements artefact might be **withdrawn**.
- **Quality attributes** for the artefact might change.
- **Contextual information** for the artefact might change.
- …

Version of Requirements Artefacts (2)

- A **version of a requirements artefact** can be regarded as a **defined state** of the artefact **at a given point in time**.
- Differentiation between:
  - **Version**: Major changes / releases
  - **Increment**: Minor changes
- A version of an artefact must be clearly **identifiable**!
- **Unique identifier**: [v]version.increment
  - Increase in version sets increment to 0.



Requirements Configurations (1)

- A **configuration** of requirement artefacts **consists of** a set of requirements **artefacts**, or more precisely a stet of versions of the artefacts.
- Configurations have **two dimensions**:
  - **A product** dimension: different artefact types (e.g., goals, scenarios, etc.)
  - **A version** dimension: different versions of the artefacts (valid at a certain point in time or for a given time frame).
- Typically, a configuration is **build for a specific purpose**, e.g., defining a configuration of requirement artefacts reviewed in an inspection or to be realised in the next system release.

Requirements Configurations (2)

A configuration of requirements artefacts should fulfil the following properties:

- **Consistency**: The versions of the artefacts grouped together are free of conflicts.
- **Unique identification (UID)**: A configuration has an identifier to identify the configuration unambiguously.
- **Not changeable**: A configuration freezes a particular state. Any change in artefacts leads to a new artefact version and a new configuration.
- **Basis for roll-back**: Configurations and their grouped artefacts provide the basis for roll-back to previous and consistent states in the process.

Requirements Baselines (1)

- A requirements baseline is a specific configuration of requirements artefacts.
- Baselines constitute **reference points** in the development process, i.e. subsequent changes are documented relative to them. [Tichy 2003]
- Baselines have the properties of a requirements configuration plus the following **additional properties**:
  - **Basis for the definition of system releases**
    - A requirements baseline defines the requirement artefacts realized in a certain system release (system version delivered to the costumer).
  - **Visible to the customer**
    - A requirements baseline is a requirements configuration visible to the customer.
  - **Subject to change management**
    - The requirements artefact contained in a requirements baseline can only be changed through a defined change management process.

Requirements Baselines (2)

Requirements baselines support different activities in the development process:

- **Basis for planning system releases**

  Requirements baselines serve as basis for planning system releases. They include a set of “stable” requirement artefacts and are visible to the customer.

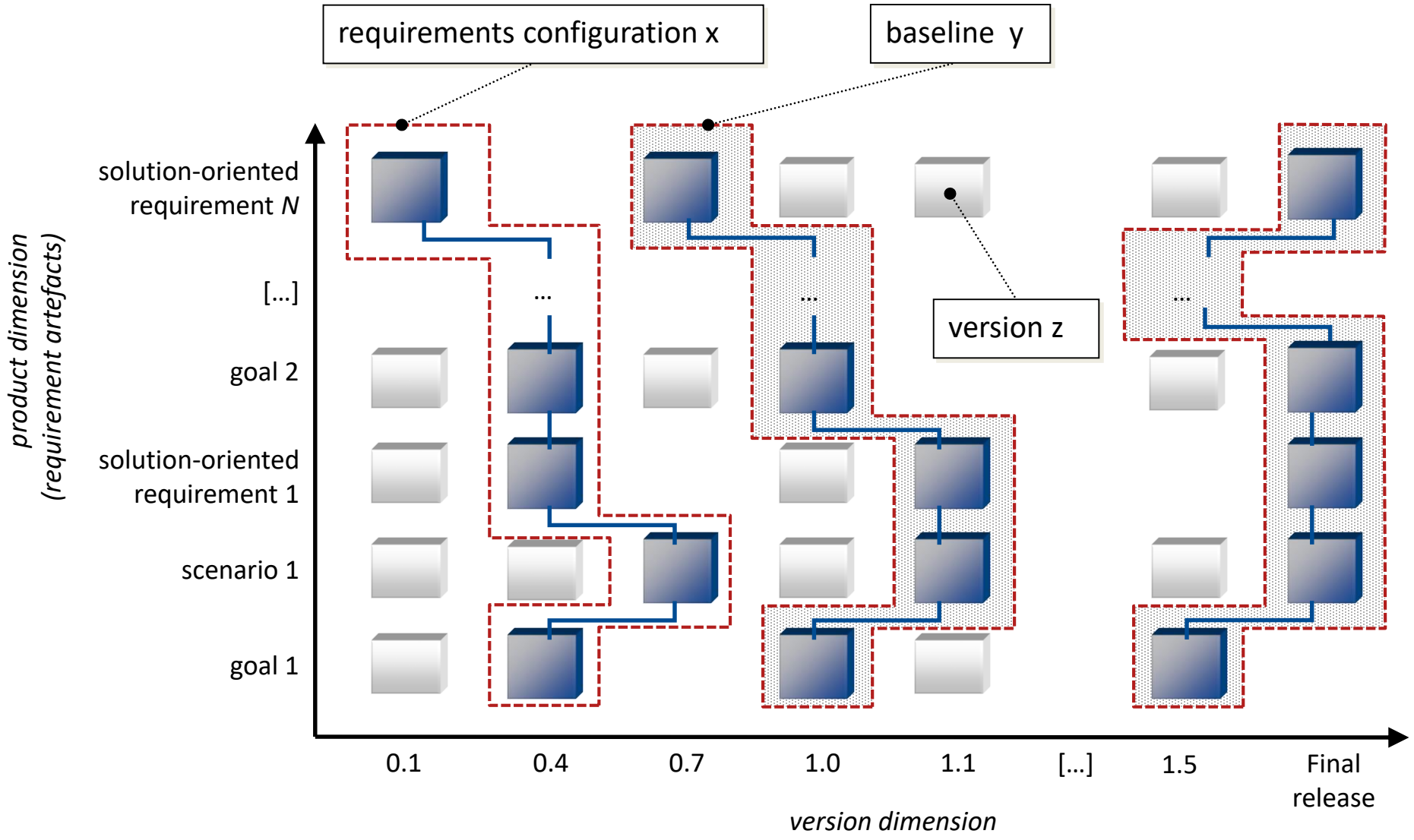
- **Estimation of realization effort**

  A requirements baseline forms the basis for estimating the development effort required for the realization of a particular system release.

- **Comparison with competitors’ products**

  Requirements baselines are used for comparing the planned system release with competing systems on the market.

Requirements Configurations and Baselines



4. Requirements Traceability

Definition of Requirements Traceability

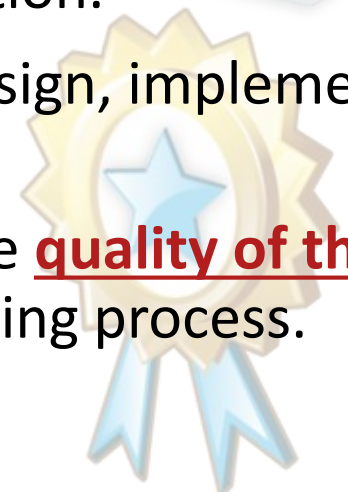
D "Requirements **traceability** refers to the **ability to describe and follow the life of a requirement**, in both a **forwards and backwards direction**
(i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases)."

[Gotel and Finkelstein 1994]

Motivation for Traceability

Requirements **traceability**:

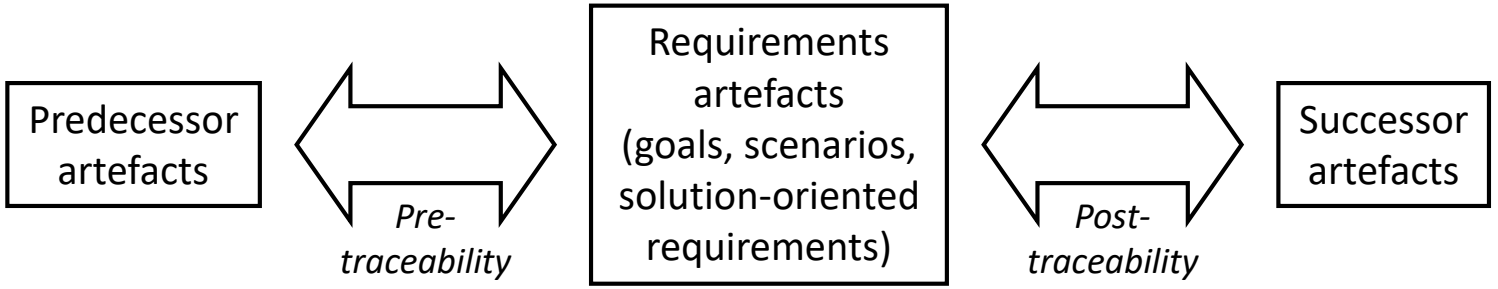
- Facilitates the **identification of the sources** which lead to the definition of a requirement.
- Documents the **influence a requirement** had on the system development process and the **rationale** for defining requirements.
- Is **key for a systematic impact analysis** of change, in case a requirement changes.
- Supports the **integrity and completeness** of documentation.
- Has **impact** on **all system development activities** like design, implementation, test, system operation and maintenance.
- The quality of the **traceability** information **influences** the **quality of the system developed** and the manageability of the whole engineering process.



All Images from [1]

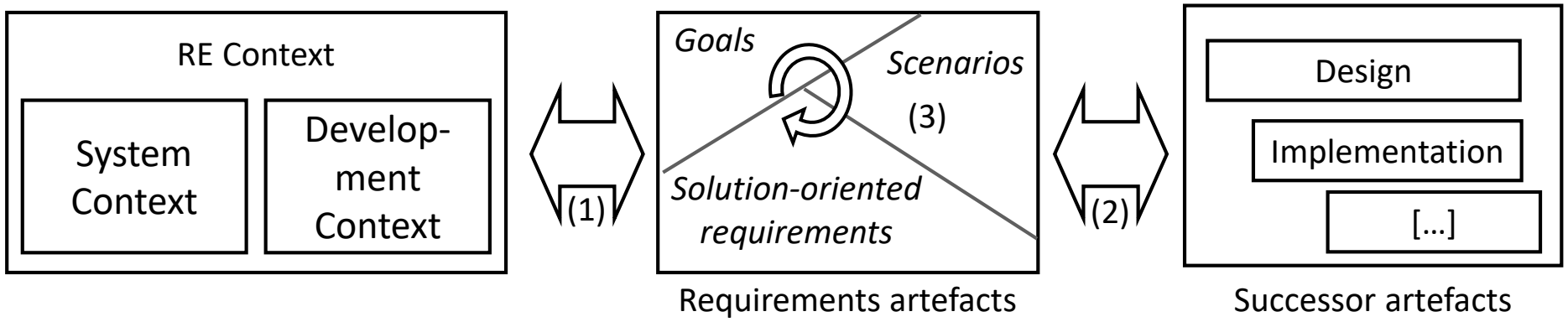
Pre- and Post-traceability of Requirements

- **Pre-traceability**: Traceability of requirements artefacts to its predecessor artefacts, i.e. to its source or origin
- **Post-traceability**: Traceability from a requirements artefacts to its successor artefacts, i.e. architectural components, implementation in the source code, test cases

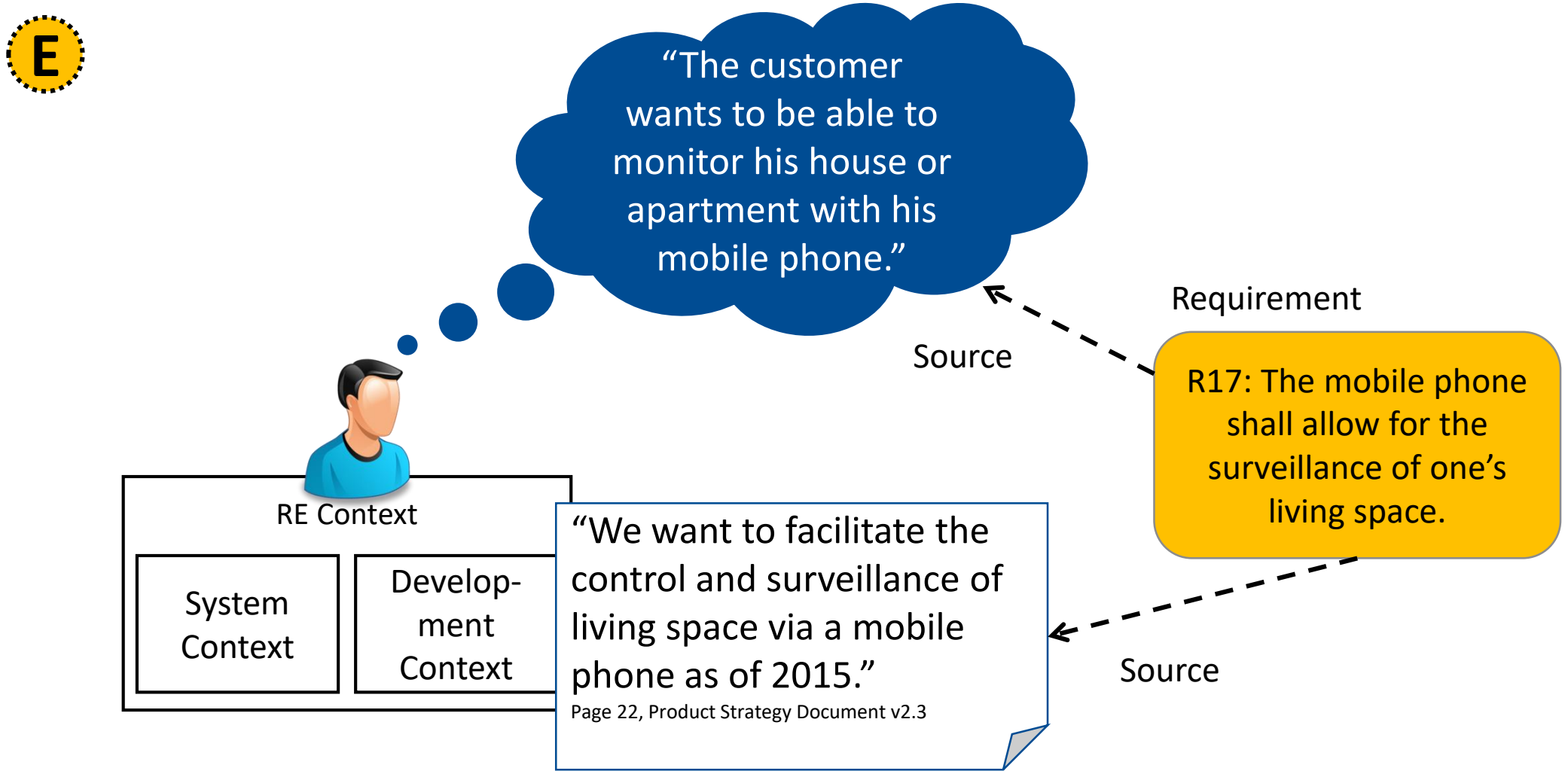


Pre- and Post-traceability of Requirements
Extended Pre- and Post-traceability

- **Pre-traceability of requirements** (1): Traceability to the context
- **Post-traceability of requirements** (2): Traceability to the realization
- **Traceability between requirements artefacts** (3): of the same artefact type / of different artefact type

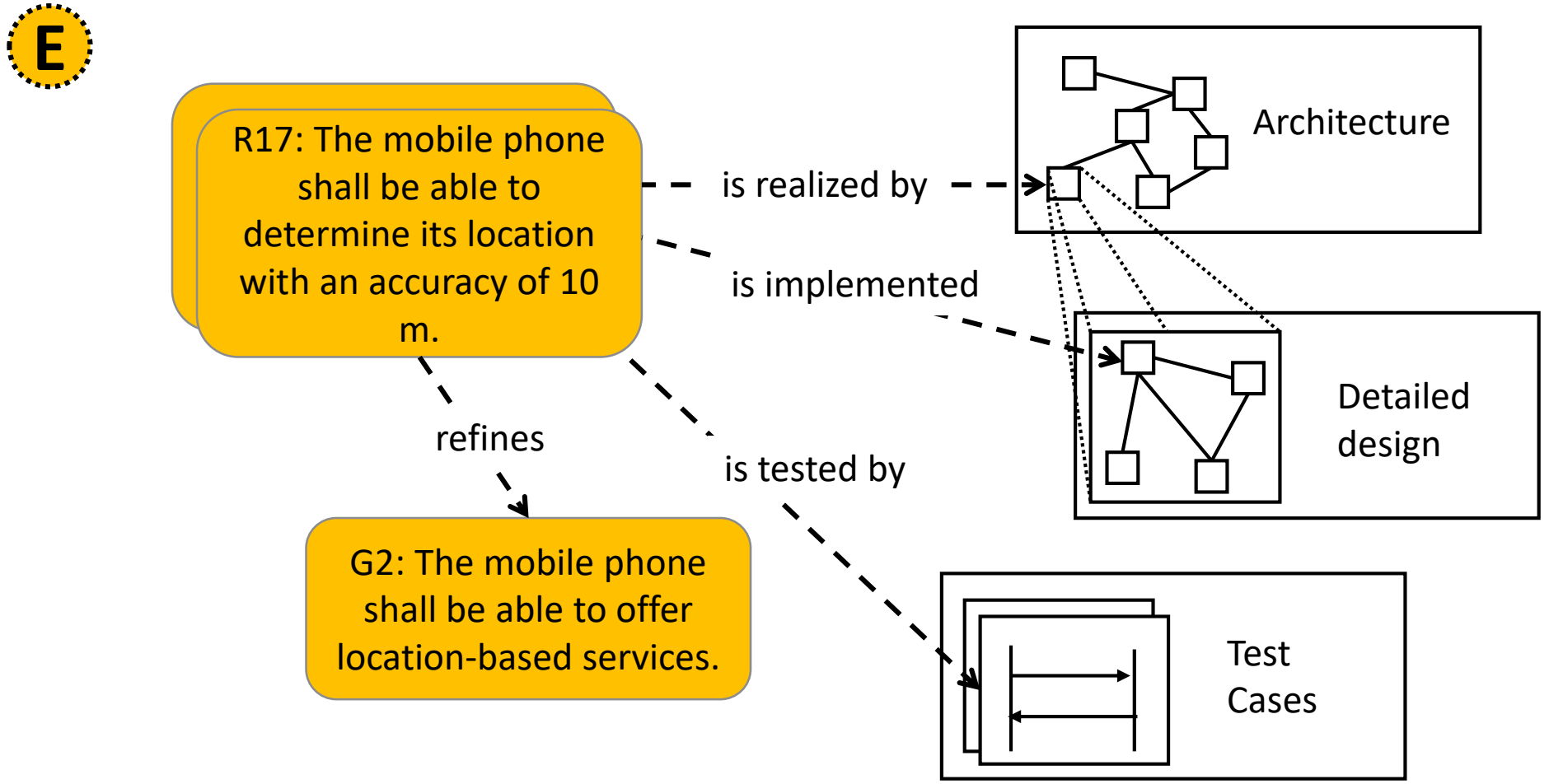


Pre-traceability of a Requirement



All Images from [1]

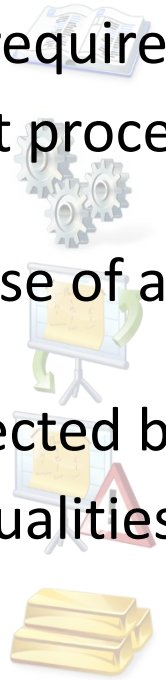
Post-traceability of a Requirement



Benefits of Requirements Traceability

Traceability has **positive effects** on various aspects of the development process, e.g.:

- **Accountability**: Allows assigning development effort to individual requirements
- **Process improvement**: Allows tracing problems in the development process back to their causes
- **Change management**: Allows analysing the affected artefacts in case of a change
- **Risk management**: Allows identification of artefacts potentially affected by a risk
- **Avoidance of "gold plating"**: Allows identification of functions or qualities not specified in the requirements
- **Additional benefits**: Verifiability and acceptance, quality assurance, maintenance, re-engineering, reuse, etc.



All Images from [1]

Documenting Traceability Relationships

Traceability relationships can be documented as:

1. Textual annotation
2. Attribute scheme
3. Traceability Matrices
4. Traceability Graphs

(1) Textual Annotation and (2) Attribute Scheme

- **(1) Textual Annotation**:

  Documented via an annotated comment.

  R17: For selecting the trip destination, the navigation system shall display the last ten trip destinations.

  Is based on R9

- **(2) Attribute scheme**:

  Documented via attributes in a scheme.

  ID: R17
  Description: For selecting the trip destination, the navigation system shall display the last ten trip destinations.
  Based on: R9

(3) Traceability Matrices (1)

Traceability Matrix for a **single** relationship type:

- Top left corner indicates the relationship type.
- Source artefacts: Left column
- Target artefacts: Top row.
- Traceability links are indicated in the matrix body.

satisfies	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
Scenario 1	X				
Scenario 2				X	
Scenario 3		X			
Scenario 4			X		X
Scenario 5		X			

(3) Traceability Matrices (2)

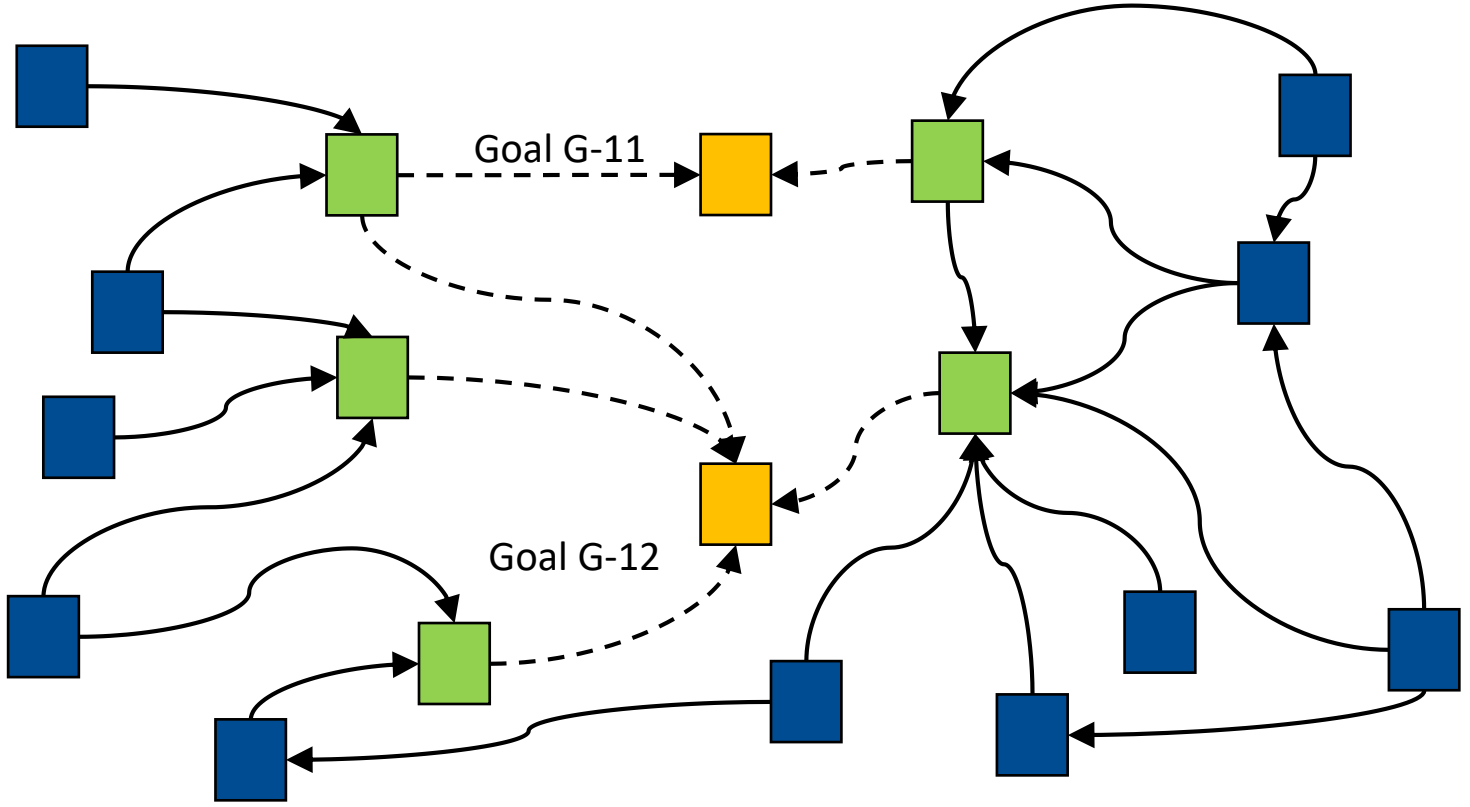
Traceability Matrix for **multiple** relationship types:

- Source artefacts: Left column
- Target artefacts: Top row
- Traceability links are indicated in the matrix body together with the name of the relationship type.

	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
Scenario 1	satisfies				
Scenario 2	based on	conflicts		satisfies	
Scenario 3		satisfies			
Scenario 4	conflicts		satisfies		satisfies
Scenario 5		satisfies		based on	

Hint: **Traceability matrices** are **not really suited** to visualize traceability links for a **large number of requirements**.

(4) Traceability Graph (Illustration) (1)



Traceability information:

- Nodes represent artefacts
- Edges represent traceability links

goal
scenario
functional requirement

based-on
satisfies

(4) Traceability Graph (2)

- **Partial graphs** can be derived to focus on particular traceability information, e.g.:
  - All “satisfies” relationships and their corresponding artefacts. (satisfies graph)
  - All “scenario” artefacts with their respective relationships (scenario graph)
- Foundation for **impact analysis** in requirements change management: **Traceability paths** indicate impact chains of artefacts.

5. Change Management of Requirements

Fundamentals (1)

- The requirements engineering **context changes** over time, e.g.,
  - evolution of stakeholder needs due to new insights,
  - changes to (inter-)national laws,
  - new technologies are invented,
  - new products of competitors are introduced.
- During **system operation problems** can occur, e.g.,
  - data inconsistencies,
  - system errors,
  - insufficient system quality.
- Requirements artefacts must be **adapted** to cope with context changes and problems in system operation originated in requirements artefacts.
- **Change management process** is a explicitly defined process to systematically manage changes in requirements artefacts.

Fundamentals (2)

R98: The navigation system shall calculate the estimated duration of a trip. To calculate the estimated duration of the trip, for motorways, an average speed of 68 mph shall be assumed.

- The feedback of many customers using the system clearly indicates that the estimated driving times are always too optimistic (→ **context change**).

- **Consequently, R98 needs to be changed:**

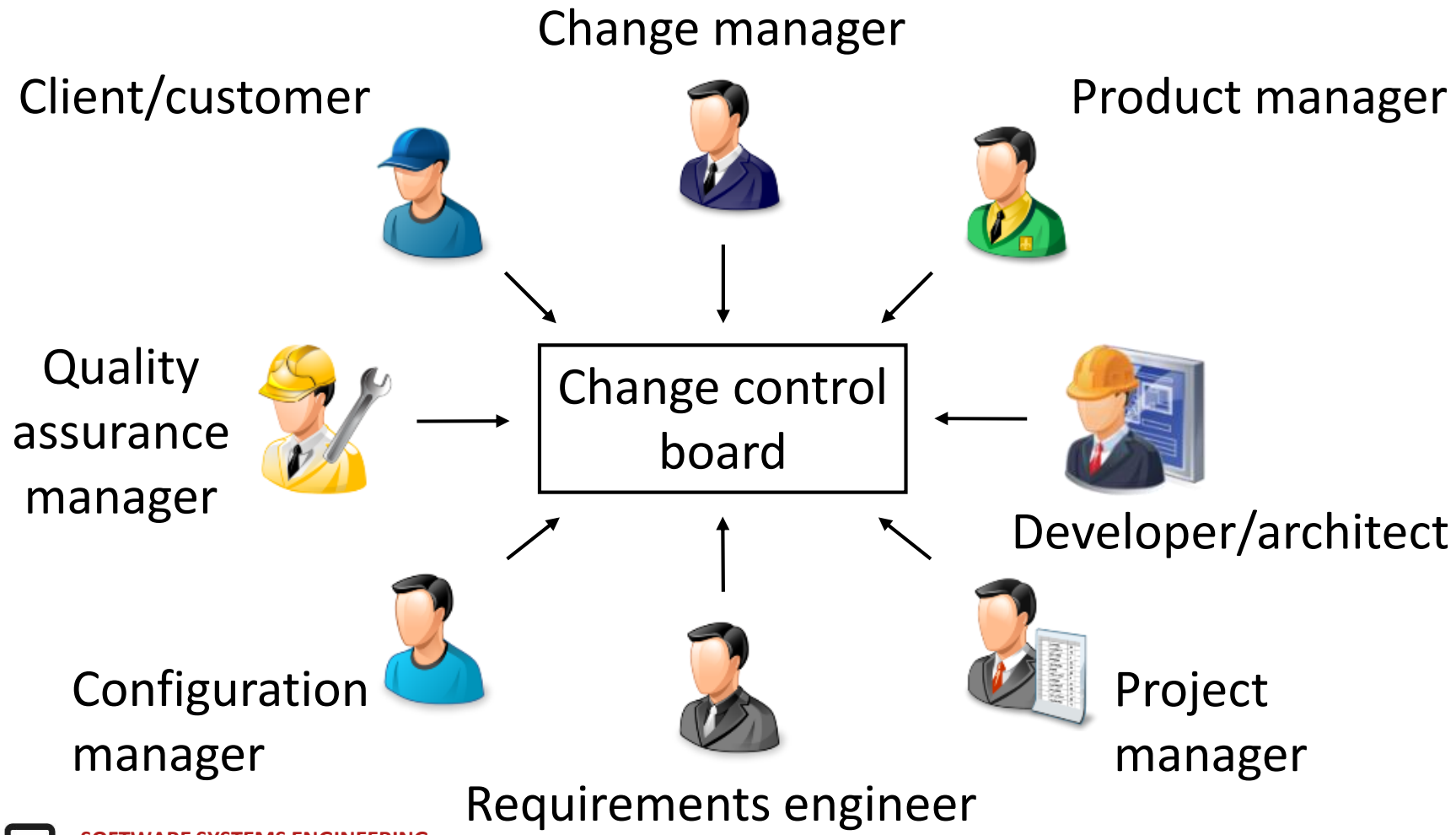
  R98.r1: The navigation system shall calculate the estimated duration of a trip. To calculate the estimated duration of the trip, for motorways, an average speed of 55 mph shall be assumed.

Change Control Board (1)

- The change control board (CCB) is an establishment to receive **change requests** and to **decide about the processing of the requests**.
- The change control board is responsible for:
  - **Classification of incoming change requests**: The change control board analyses each incoming change request and assigns the changes to different categories.
  - **Effort estimation for change integration**: For each change request, the change control board estimates the effort required for integrating the change.
  - **Evaluation of change requests and decision-making**: The change control board evaluates the change request with regard to the relation between effort and benefit. Bases on the result the request is accepted or rejected.
  - **Prioritization of accepted change requests**: The change control board prioritizes the accepted change request and assigns each request to a system release or to a dedicated project for integrating the change.

Change Control Board (2)

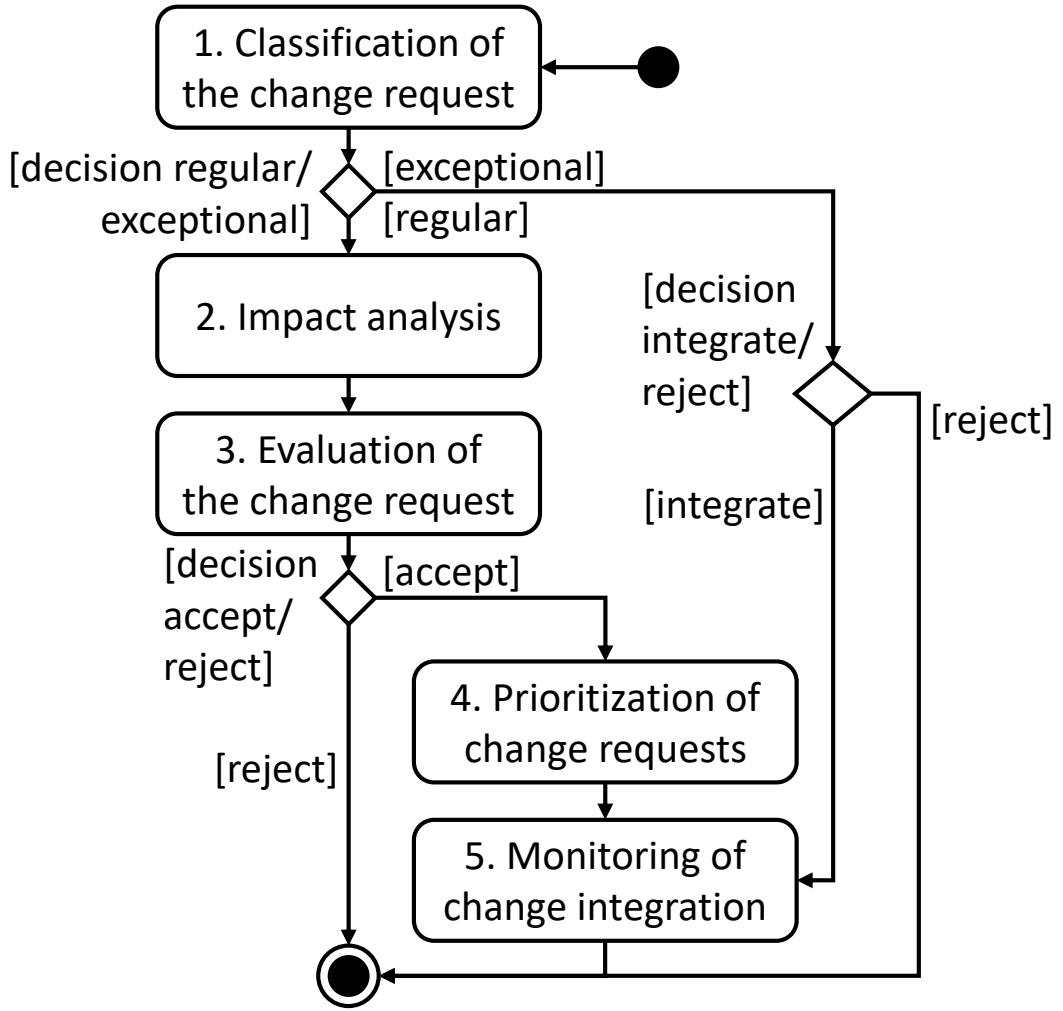
Typical **members** of a **change control board** are:



All Images from [1]

Change Management Process

To cope with **requirements changes throughout the entire lifecycle** of a system a systematic **change management process** for managing the changes requests is important!



Step 1. Classification of the Change Request

Three typical change categories:

Corrective Change

Originated in a system errors (e.g., erroneous behaviour) encountered during operation, which is caused by one or more incorrect specified requirements.

Adaptive Change

Originated in changes in the requirements engineering context (e.g., changed stakeholder intentions, new regulations and laws, …).

Exceptional Change

A change is crucial and must be integrated immediately (in many cases corrective changes).

All Images from [1]

Summary (1)

- The goal of management in requirements engineering is to observe the requirements engineering context, manage the execution of requirements engineering activities, and manage the requirements artefacts.
- Requirements prioritization aims at handling constraints and limitations (time, effort, people) for the use of requirements.
- Requirements prioritization is a structured process which ensures all relevant requirements are adequately considered.
- Prioritization techniques are selected and combined with regard to the number of requirements and the effort required to execute the prioritization technique.
- Requirements artefacts are identified by unique numbers (version- and increment number).
- A configuration of requirement artefacts comprises a set of related versions of requirements artefacts and is characterized by the properties “consistency”, “unique identification”, “not changeable”, and “basis for roll-back”.
- Configurations are described in a product- and version dimension making up a requirements baseline.

Summary (2)

- There are three types of traceability: pre-, post-requirements traceability and traceability between requirements.
- Traceability relationships can be documented by using textual annotations, attribute schemes, traceability matrices and traceability graphs.
- A change of the system context and a problem encountered during system operation can lead to requirements changes.
- The change control board decides about requirement changes and receives change requests.
- The change control board is responsible for classification of incoming change requests, effort estimation for change integration, evaluation of change requests and decision-making, prioritization of accepted change requests.
- Three types of requirements change requests: corrective change, adaptive change and exceptional change.

Literature

[Dahlstedt and Persson 2005] A. G. Dahlstedt, A. Persson: Requirements Interdependencies – State of the Art and Future Challenges. In: A. Aurum, C. Wohlin (Eds.): Engineering and managing Software Requirements, Springer, berlin, Heidelberg, 2005, pp. 95-116.

[Leffingwell and Widrig 2000] D. Leffingwell, D. Widrig: Managing Software Requirements – A Unified Approach. Addison-Wesley, Reading, 2000.

[Kano et al. 1984] N. Kano, S. Tsuji, N. Seraku, F. Takahashi: Attractive Quality and Must-Be Quality (in Japanese). Journal of the Japanese Society for Quality Control, Vol. 14, No. 2, 1984, pp. 147-156.

[Pohl 1996a] K. Pohl: Process-Centered Requirements Engineering. Wiley, Research Studies, Advanced Software Development Series, Taunton, Somerset, 1996.

[Wiegers 1999] K. E. Wiegers: Software Requirements. Microsoft Press, Redmond, 1999.

[Wiegers 2003] K. E. Wiegers: Software Requirements. 2nd edition, Microsoft Press, Redmond, 2003.

Literature for Further Reading (1)

[Karlsson et al. 1997] J. Karlsson, S. Olssen, K. Ryan: Improved Practical Support for Large-Scale Requirements Prioritising. Requirements Engineering, Vol. 2, No. 1, Springer, London, 1997, pp. 51-60.

[Lehtola et al. 2004] L. Lehtola, M. Kauppinen, S. Kujala: Requirements Prioritization Challenges in Practice. In: Proceedings of the 5th International Conference on Product Focused Software Process Improvement (PROFES'04), Springer, Berlin, Heidelberg, New York, 2004, pp. 497-508.

[Davis 1993] A. M. Davis: Software Requirements – Objects, Functions, and States. 2nd edition, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[Davis 2003] A. Davis: The Art of Requirements Triage. IEEE Computer, Vol. 36, No. 3, 2003, pp. 42-49.

Literature for Further Reading (2)

[IEEE Std 830-1998] Institute of Electrical and Electronics Engineers: IEEE Recommended Practice for Software Requirements Specification (IEEE Std 830-1998). IEEE Computer Society, New York, 1998.

[Hoek et al. 1997] A. v.d. Hoek, R.S. Hall, D. Heimbigner, A.L. Wolf: Software Release Management. Proceedings of the 6th European SOFTWARE ENGINEERING conference held jointly with the 5th ACM SIGSOFT international symposium on Foundations of software engineering, ACM, 2003, pp. 159-175.

[Tichy 2004] W.F. Tichy: Software Configuration Management. Encyclopedia of Computer Science. 4th Edition, Wiley 2003, pp. 1601-1604.

Image References

[1] Licensed by http://www.iconshock.com/

[2] Provided by Microsoft Office

Legend

Definition

Example

UNIVERSITÄT DUISBURG ESSEN

Offen im Denken

Requirements Engineering & Management

Vielen Dank für Ihre Aufmerksamkeit

PALUNO
The Ruhr Institute for Software Technology