UNIVERSITÄT DUISBURG ESSEN

Offen im Denken

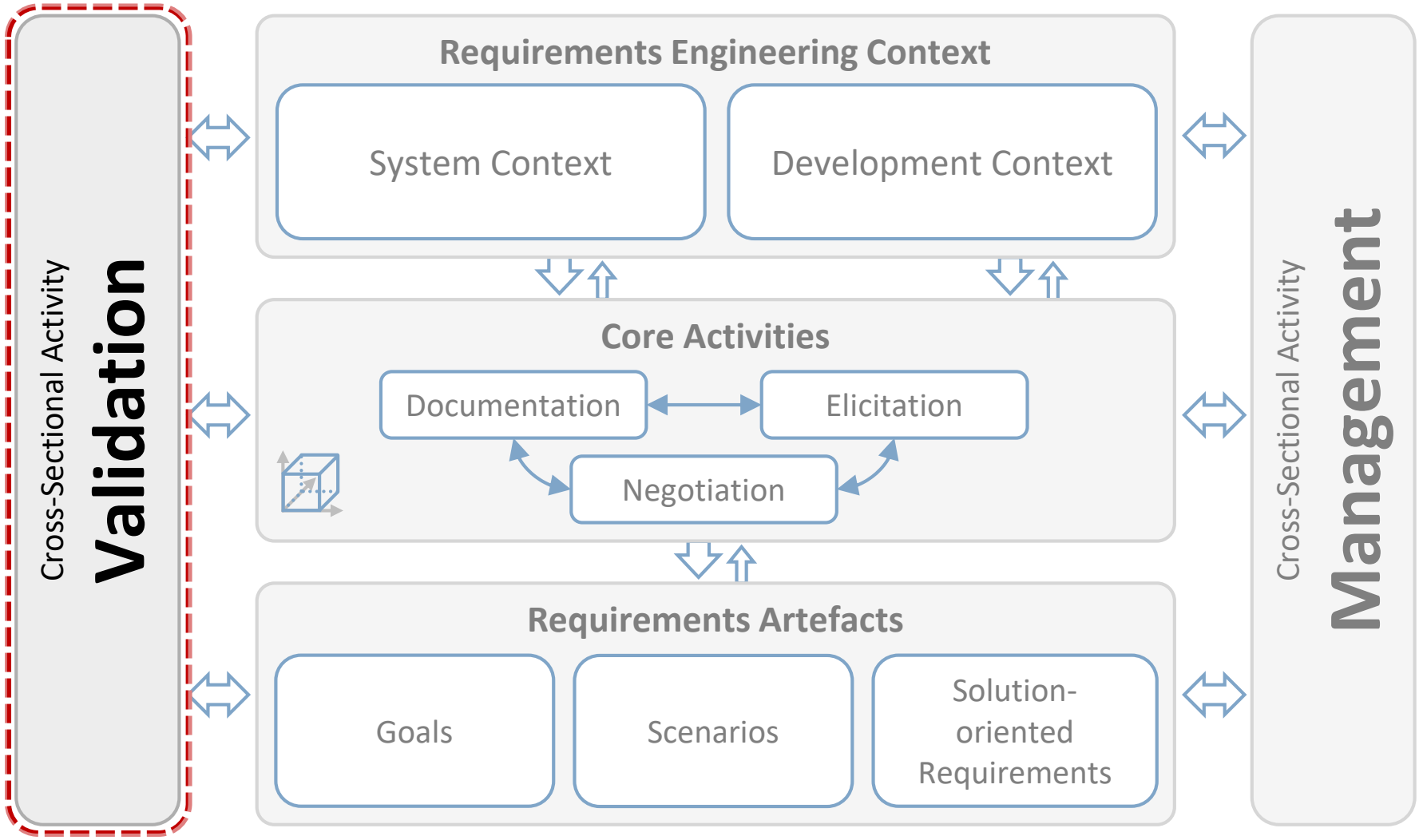
Requirements Engineering & Management

Cross-Sectional Activities – Validation

Prof. Dr. Klaus Pohl

PALUNO
The Ruhr Institute for Software Technology

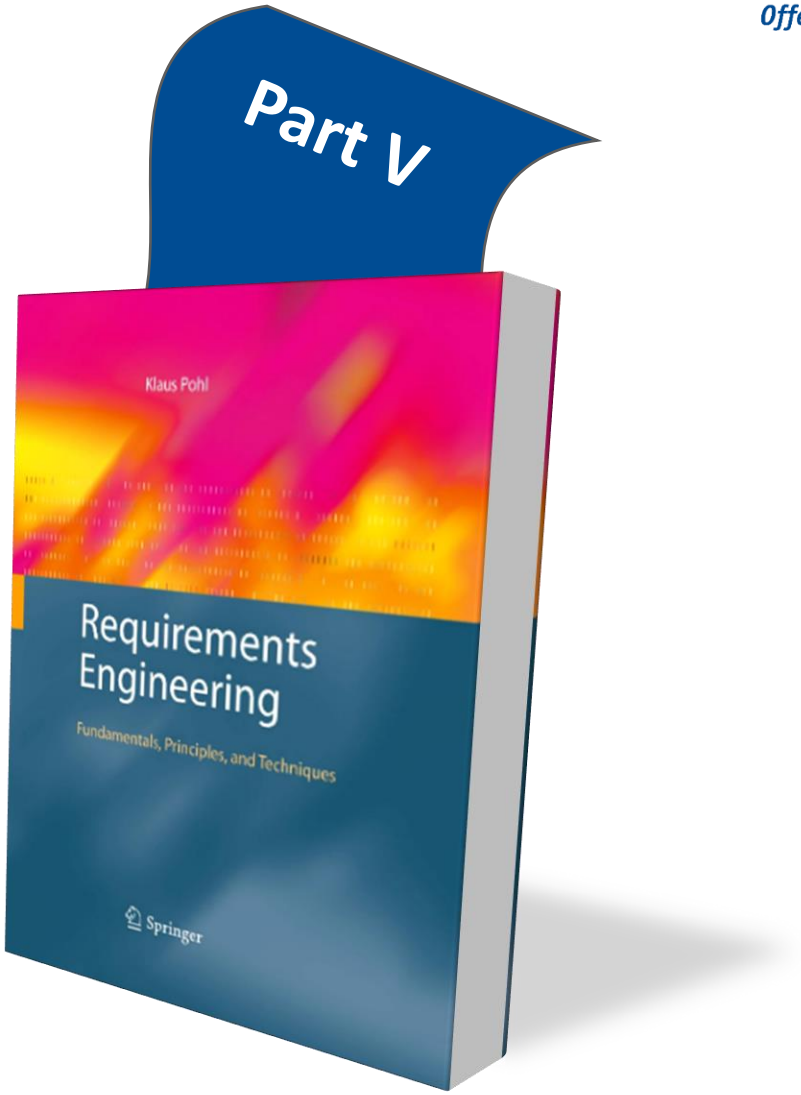
Requirements Engineering Framework



Agenda

1. Introduction to Requirements Validation
2. Quality Gates
3. Validation Principles
4. Validation Techniques – Overview
5. Inspections

Corresponding Parts and Chapters in Textbook



Klaus Pohl: Requirements Engineering – Fundamentals, Principles, and Techniques.
Springer, Berlin Heidelberg, 2010.

1. Introduction to Requirements Validation

Validation vs. Verification

Validation	Verification
Am I building **the right requirements (system)**?	Am I building **the requirements (system) right**?
Checking if the artefacts are **appropriate** Result: appropriate or inappropriate!	**Proving the correctness** of software or specification Result: correct or incorrect!

[Boehm 1984], [Dzida and Freitag 1998]

Two Basic Types of Quality Assurance

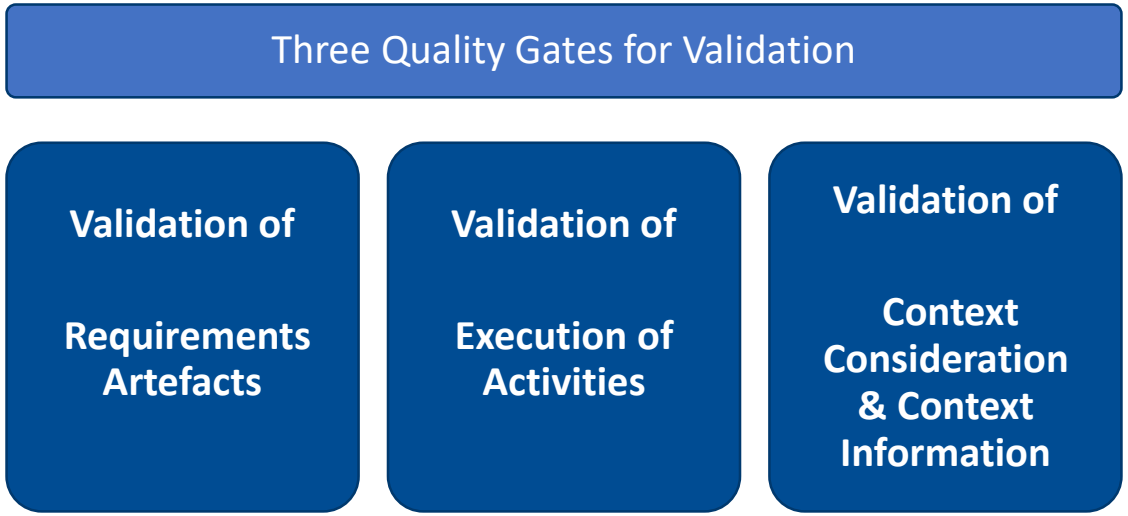
- **Constructive quality assurance**
  - Techniques aim at the prevention of defects when developing artefacts, e.g. guidance to adhere to well-known modelling rules.

- **Analytical quality assurance**
  - Techniques aim at checking the quality of a developed artefact, e.g. software testing, formal verification, or requirements validation.

based on [Liggesmeyer 2009]

Validation Goals

1. Checking whether the **outputs of activities** fulfil defined quality criteria (e.g., checking requirements artefacts and additional information)
2. Checking whether the **inputs of activities** fulfil defined quality criteria (e.g., checking context consideration)
3. Checking whether the **execution of activities** adheres to process definitions and activity guidelines



Validation

D Validation denotes **checking** whether **inputs** (e.g., context information), **execution**, and **outputs** of the requirements engineering activities **fulfil defined quality criteria**.

Validation is performed by involving **relevant stakeholders**, other **requirement sources** (standards, laws, etc.) as well as **external reviewers**, if necessary.

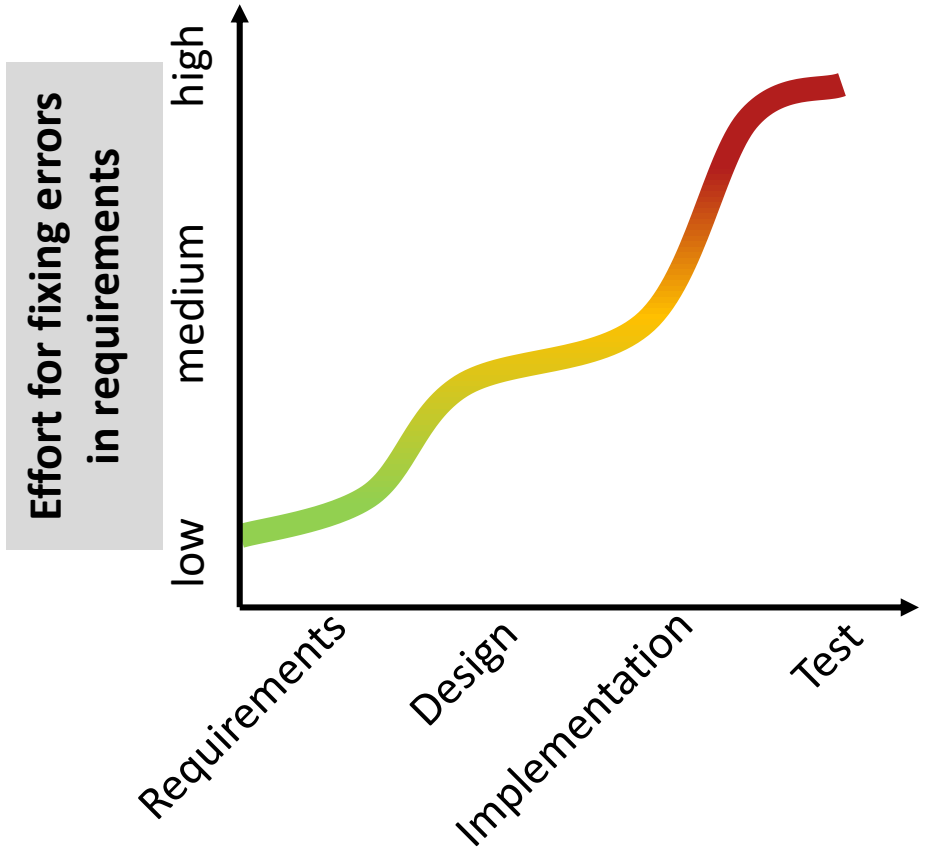
Risks of Insufficient Validation

- Error propagation
  - If an **error in a requirements artefact is not detected** before the release of the system, **all artefacts** (requirements artefacts, design artefacts, source code test artefacts, and manuals) **need to be revised**.
    → Corrective activities cause very **high costs**!

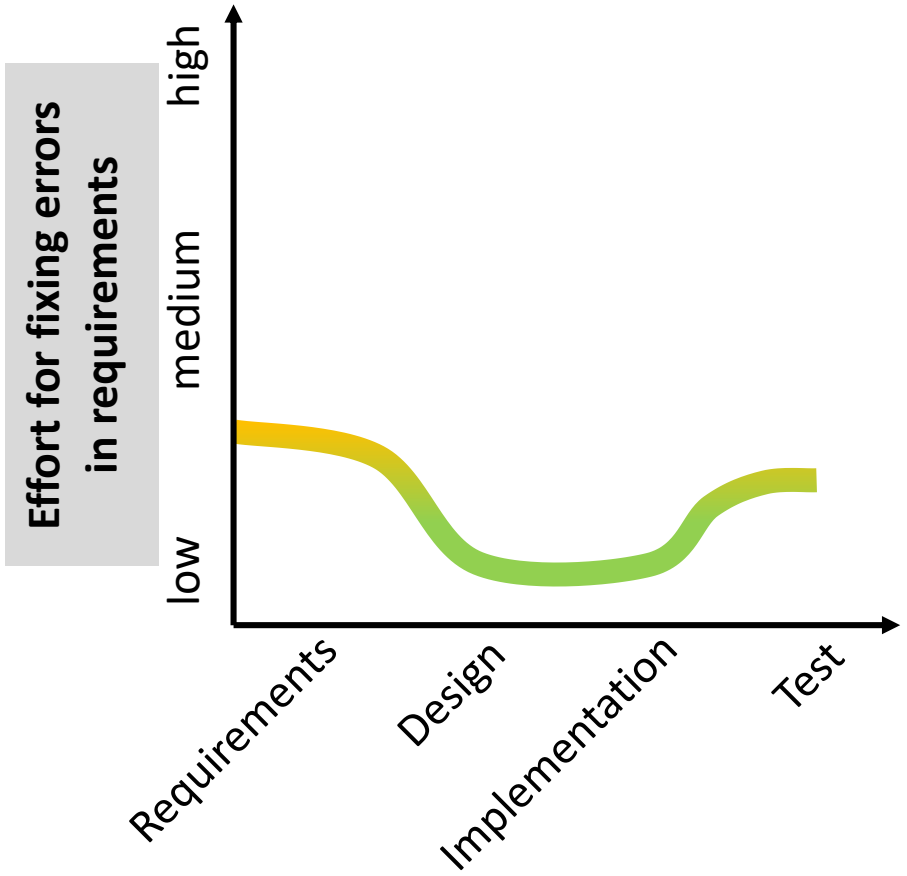
- Legal issues
  - Requirements are often **part of the contract between client and contractor**.
  - **Overlooked defects** in requirements may **lead to disputes and conflicts** concerning the actual contractual liabilities.

Reduction of Costs and Risks in later Phases

Without validation during requirements engineering



With validation during requirements engineering



2. Quality Gates

Overview

Three Quality Gates for Validation

Validation of the Requirements Artefacts

Validation of the Execution of Activities

Validation of Context Consideration & Context Information

Quality Aspects for Requirements Artefacts

All three dimensions of requirements engineering are relevant for the acceptance of a requirement. **Check all three dimensions**!

Content Dimension	Documentation Dimension	Agreement Dimension
• Check whether **all relevant requirements** are known and understood to the **required level of detail**.	• Check whether the **requirements are documented** according to the defined **documentation and specification guidelines**.	• Check whether **sufficient agreement** is reached about the documented requirements. • Check for each known conflict whether it **has been resolved**. • Check for **unidentified conflicts**.

Content Dimension

- **Aim of validation**:
  Check for **deficiencies regarding the content** of the requirements.
- Risks of content insufficiencies:
  - **Defects** in the content of the requirements can lead to **significant delays in subsequent development** activities.
  - Significant effort has to be spent for **eliciting missing information** as well **mitigating the problems** caused by incorrect or missing information.
  - **Defects creep unnoticed** into architectural design, implementation, and test artefacts.

Documentation Dimension

- **Aim of validation:**
  Check whether the documented requirements **adhere to the documentation guidelines** defined for the document.
- Risks of documentation insufficiencies:
  - Documented requirements are **unusable** or **incomprehensible** for the stakeholders.
  - Gaps and insufficiencies of requirements remain undetected because insufficient documentation disables stakeholders to fully comprehend the documented requirements.
  - **Delay of development activities** for which requirements must be available in a specific format.
  - Other **criteria** (e.g. correctness or completeness) **cannot be checked**.

Agreement Dimension

- **Aim of validation:**
  Check whether there is a sufficient agreement between the relevant stakeholders about the final draft of the requirements and if the detected conflicts have been resolved.
- Risks of agreement insufficiencies:
  - Disagreement between stakeholders remains undetected and surfaces the latest in change request during system operation.
  - Insufficient agreement can lead to significant delays in subsequent development activities.
  - Late agreement can cause changes of artefacts and thus cause higher efforts for integrating those changes, e.g. during design or testing.
  - Validation often provides the last opportunity for stakeholders to suggest changes without affecting design and implementation artefacts.

Validating the Execution of Activities

- Check whether each required requirements activity has been performed properly (according to the activity definition).
  - Have **all relevant stakeholders** been involved in the execution of the activity?
  - Was the **context considered** properly?
  - Have **all required inputs** been **considered**?
  - Have **all required outputs** been **created**?
  - Have **all activities** defined in the process definition been **performed**?
  - Have the guidelines for **executing** the activity be considered?
- Detect **unwanted** and **possibly critical deviations**. Have they been handled properly?

Validating Context Consideration (1)

- **Errors in context consideration cause requirements errors such as**:
  - missing requirements.
  - inappropriate documentation formats.
  - undetected disagreement of stakeholders about requirements!
- Errors in requirements can, among others, be caused by
  - **missing** context information,
  - **incorrect** context information,
  - **insufficiently considered** context information,
  - **incomplete consideration of requirements sources**.

Validating Context Consideration (2)

- **Validate each perspective on the requirements engineering context**:
  - All perspectives on the system context
  - All perspectives on the development context
- Use **dedicated checklists** for each context perspective for validation.

Validation of Context Information

Aim of validation:
Check whether documented context information (including context models, context assumptions, context description) are sufficiently documented.

- **Similar to the validation of requirements artefacts**, check the documented context information by considering the **documentation, content,** and **agreement dimension**.
- **Check especially for:**
  - Missing context information
  - Incorrect context information
  - Insufficiently considered context information
  - Incomplete requirements sources

3. Validation Principles

Six Principles

Assure and improve the quality of the validation outcome!

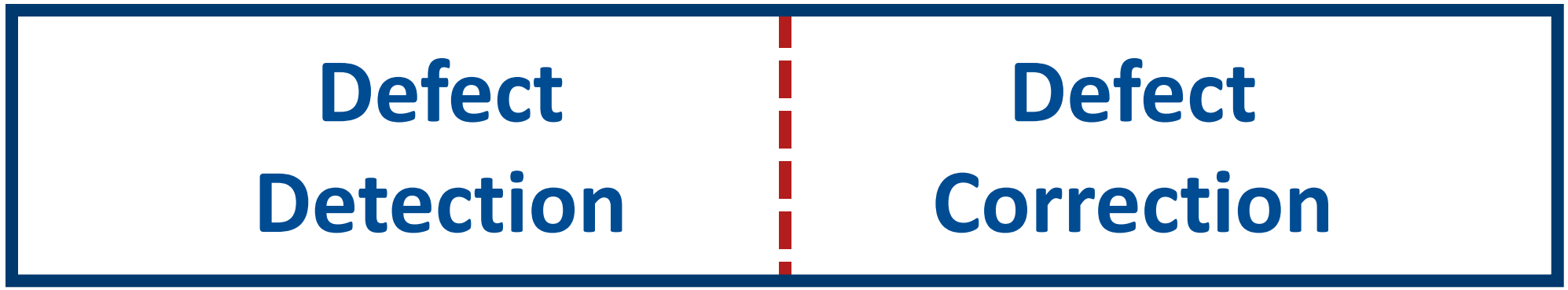
1. Involving the right stakeholder
2. Separating defect detection from defect correction
3. Leveraging multiple independent views
4. Use appropriate documentation formats
5. Exemplary creation of (development) artefacts
6. Repeated validation

Principle 1 - Involving the Right Stakeholder

- Independence of reviewers based on [IEEE Std 1012-2004]
  - **Managerial** independence
  - **Financial** independence
- Involve the right stakeholders **depending on the validation goal** and the **subject**.
  - Use the structure of the requirements engineering context to identify potential relevant stakeholders.
- For critical validation tasks or if required expertise is not available within the project (organization) consider the involvement of external experts (validators).

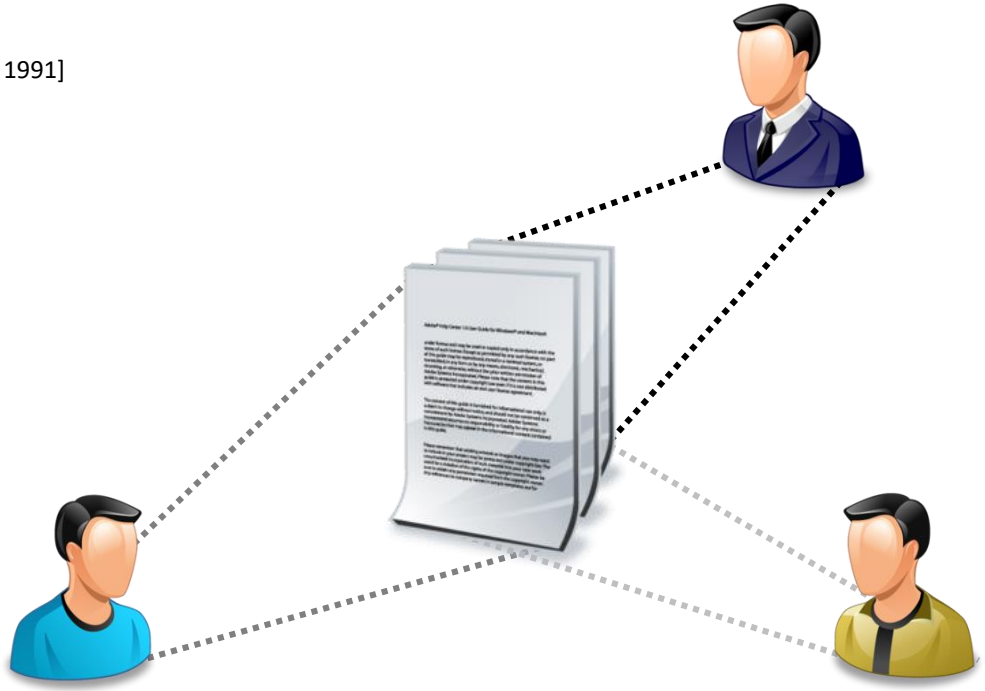
Principle 2 - Separating Defect Detection and Correction

- **Focus on the identification** of potential requirement defects.
- Purposeful use of available resources for validation.
- **Don‘t deal with defect correction**!
- **Avoid focusing** on single defects.



Principle 3 - Leveraging Multiple Independent Views

- Obtain multiple **independent views**, e.g., by using perspective based-reading.
- **Consolidate** independent validation results in a **systematic** way. based on [Leite and Freeman 1991]



all icons from [1]

Principle 4 - Use of Appropriate Documentation Formats

- Chose documentation formats based on stakeholders **preferences and knowledge**.
- **Use the strengths** of documentation formats to **compensate weaknesses** of other formats.

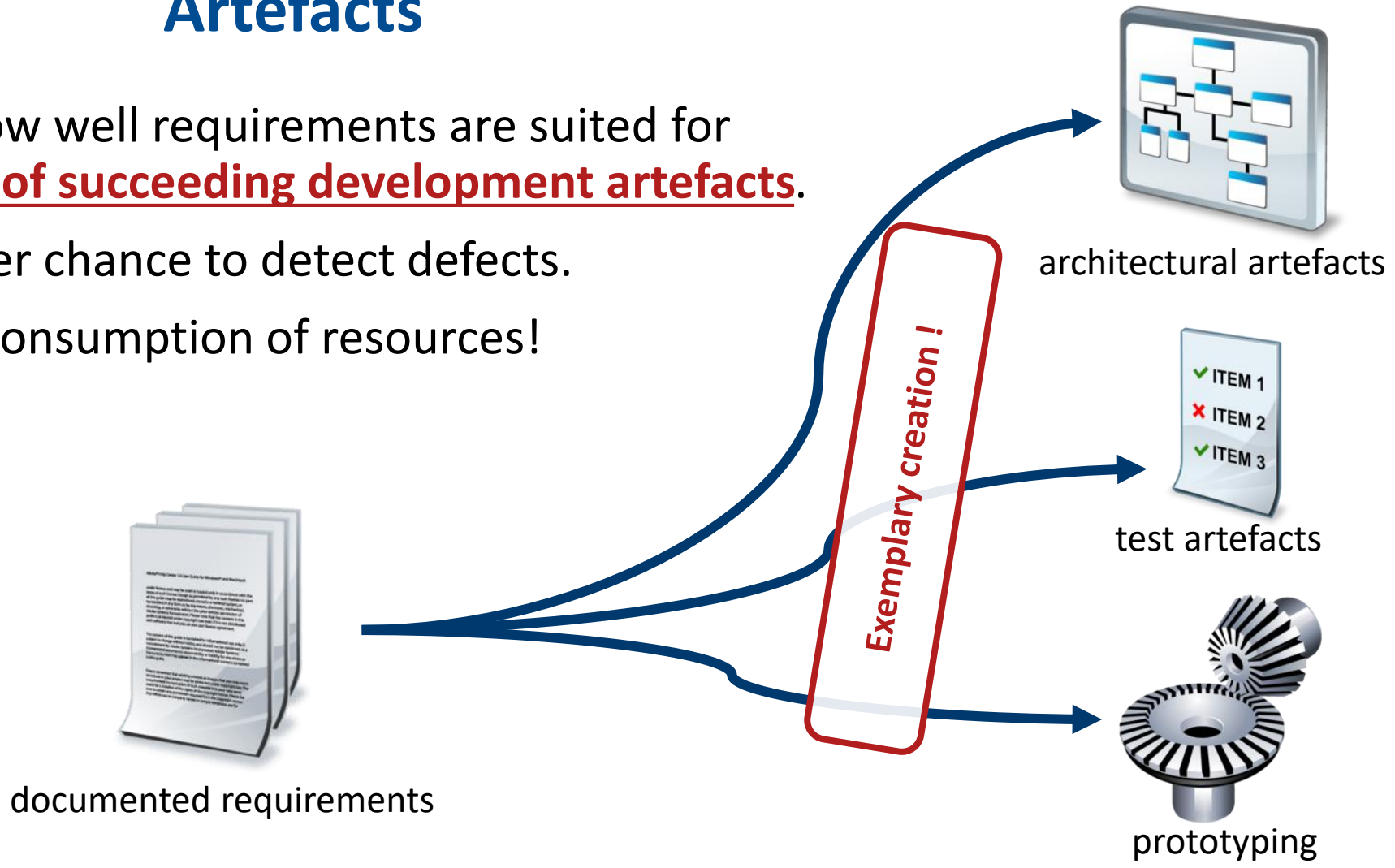
Models	Natural Language
• Structured representation of objects and relationships • Limited expressiveness	• General comprehensibility • High expressiveness • Ambiguity • Weak support for presenting complex, interrelated facts in a structured manner

Defect detection by **transformation**!

Principle 5 - Exemplary Creation of (Development) Artefacts

Check how well requirements are suited for **creation of succeeding development artefacts**.

- Greater chance to detect defects.
- High consumption of resources!



all icons from [1]

Principle 6 - Repeated Validation

- Stakeholders continuously **gain new knowledge** during requirements engineering. This influences validation results!
- **A single validation** and release of a requirement is **no guarantee for stable validity**!
- Repeated validation of already validated requirements, among others especially
  - **in long-term** projects.
  - Changes of **stakeholder**, **context** or **requirements**.
  - **when reusing** requirements.
  - for systems with a high degree of **innovation**.
  - **in poorly understood domains**.
  - **high gain of additional or new knowledge**.

4. Validation Techniques - Overview

Reviews

D A **requirements review** is a process or meeting during which the **requirements** for a system, hardware item, or software item **are presented** to project personnel, managers, users, customers, or other interested parties **for comment or approval**.

[ISO/IEC/IEEE 24765:2010]

- There are several **types of reviews**.
- In the following we focus on the most common ones.

Overview

Validation Techniques

- **Review**
  - Commenting Artefacts
  - Walkthrough
  - Inspection
- **Validation using Prototypes**

In this lecture!

Details on **all validation techniques** can be found in:

For each of those techniques **during preparation**:

- Define a clear **validation goal**.
- Identify **required participants**.
- **Communicate goals and expected results** with participants.
- **Organize suitable location** and **equipment**.

Review Techniques – Overview

	Commenting Artefacts	Walkthrough	Inspection
Goal	• Very flexible for feedback on early sketches to detailed examination of artefacts	• Feedback on early sketches of artefacts • Elicitation of new ideas • Agreement on requirements	• Extensive search for defects in a manageable extent of requirements
Group session	• Optional	• Yes	• Yes

Review Techniques – Rough Comparison

	Commenting Artefacts	Walkthrough	Inspection
Effort	Low to medium	Low to medium	High
Suited for artefacts with high quality	Medium	Very low	High
Suited for artefacts with low quality	High	High	Low

Commenting Artefacts

- **Simple technique** for requirements validation executed by one or multiple stakeholders (inspectors).
- Author sends the material to be validated to the inspector(s)
  - If multiple inspectors are involved: Sequential inspections or parallel inspections
  - Inspector reviews the material, comments on it and passes the material and the review
- Individual **defect detection** and **documentation**.
- **Consolidation** of results performed by the author.
- Options:
  - Instructions for reviewing the material and documenting the results
  - Execution in a group meeting

all icons from [1]

5. Inspections

Preparation

Planning

- Assign the following **roles** to the stakeholders involved:
  Organizer, moderator, author, reader/presenter, inspectors, minute-taker.
- Select a **neutral moderator** and a **neutral reader**
  - The **author can NOT** be the moderator nor the reader!

Overview

- In a first meeting, let **the author present** the requirements **artefacts** which shall be **inspected** to the other participants.
- Give the **participants** the opportunity to **ask questions** about the requirements artefacts.
  The meeting shall support the inspection team to understand the requirements and their relationships with each other.

Execution (1)

Defect detection

- The **inspectors detect potential defects in the requirements artefacts** under inspection.
- Experiment with defect detection by **individual inspectors vs. defect detection** in groups in order to find out which way of detecting defects works best for your projects or your organization.
- Guide the defect detection by means of appropriate instructions, e.g. provided as checklists.
- Provide **templates** or forms **for documenting defects**, decisions, and rationales.
- Ensure that each inspector documents all detected defects.

Execution (2)

Defect collection

- **Collect** the **detected defects**, **consolidate** them and **document them** including a classification of each defect.
- Let the **reader present** the inspected **artefacts piece by piece**, e.g., paragraph by paragraph.
- Make sure **all detected defects are collected** and classified. including questions posed to the author as well as improvement suggestions.
- Create a specific defect class for **requirements conflicts**.
- Pay attention to **complete documentation** of the results.

Follow-up

- **Reworking the information gathered** during the inspection meeting for further processing and **communicate** the information to **responsible persons** (e.g. the author).
- **Provide** the **identified defects** to the author.
- Obey the **follow-up checks** required by the inspection team.
- **Express your gratitude** to the reviewers for their participation.

Critical Success Factors

- **Process and organization**
  - Strict adherence to a defined inspection process
  - Acceptance of the role-specific tasks
  - Management commitment
  - Ability to deal with criticism in an adequate way
- **Artefacts**
  - Size of the artefacts
  - Complexity of the artefacts
  - Quality of the artefacts regarding content, documentation, and agreement
- **Selection of inspection team members**
  - Number of inspectors
  - Coverage of the four context facets
  - Experience of the inspectors

Benefits

- Inspections are **well suited** for **detail validation** of **artefacts (of high quality)**.
- Check for **achieved understanding** (content dimension)
- Check for **comprehensibility and unambiguity** (documentation dimension)
  - Inspections provide a good check if the chosen documentation format is suitable for the stakeholders and if the artefacts are defined comprehensible and unambiguous.
- Check for **achieved agreement** (agreement dimension)
  - Inspections also check the achieved agreement about the documented requirements (if the relevant stakeholders are involved in the inspection process).

Effort

- Medium to high
- **Several inspectors** check each artefact in detail – therefor high effort.
- No addition effort for the creation of additional development artefacts.

Literature (1)

[Boehm 1984] B. Boehm: Verifying and Validating Software Requirements and Design Specifications. IEEE Software, Vol. 1, No. 1, IEEE Press, Los Alamitos, 1984, pp. 75-88.

[Dzida and Freitag 1998] W. Dzida, R. Freitag: Making Use of Scenarios for Validating Analysis and Design. IEEE Transactions on Software Engineering, Vol. 24, No. 12, 1998, pp. 1182-1196.

[IEEE Std 1012-2004] Institute of Electrical and Electronics Engineers: IEEE Standard for Software Verification and Validation (IEEE Std 1012-2004). IEEE Computer Society, New York, 2005.

[ISO/IEC/IEEE 24765:2010] ISO/IEC/IEEE 24765:2010: Systems and software engineering – Vocabulary. 2010.

[Leite and Freeman 1991] J. C. S. P. Leite, P. A. Freeman: Requirements Validation through Viewpoint Resolution. IEEE Transactions on Software Engineering, Vo. 17, No. 12, 1991, pp. 1253-1269.

Literature (2)

[Liggesmeyer 2009] P. Liggesmeyer: Software-Qualität – Testen, Analysieren und Verifizieren von Software. 2nd edition, Spektrum, Akademischer Verlag, Heidelberg, 2009.

[Robertson and Robertson 2006] S. Robertson, J. Robertson: Mastering the Requirements Process. 2nd edition, Addison-Wesley, Amsterdam, 2006.

[Shull et al. 2000] F. Shull, I. Rus, V. Basili: How Perspective-Based Reading Can Improve Requirements Inspections. IEEEComputer, Vol. 33, No. 7, 2000, pp.73-79.

[Sommerville 2007] I. Sommerville: Software Engineering. 8th edition, Addison-Wesley, Bosten, 2007.

[Sommerville and Sawyer 2000] I. Sommerville, P. Sawyer: Requirements Engineering – A good Practice Guide. Wiley, Chichester, 2000.

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

[Wiegers 2002] K. E. Wiegers: Peer Reviews in Software – A Practical Guide. Addison-Wesley, Bosten, 2002.

Literature for Further Reading

[Bäumer et al. 1996] D. Bäumer, W. Bischofberger, H. Lichter, H. Züllighoven: User Interfae Prototyping – Concepts, Tools, and Experience. In: Proceedings of the 18th International Conference on Software Engineering (ICSE'96), IEEE Computer Society Press, Los Alamitos, 1996, pp. 532-541.

[Brykczynski 1999] B. Brykczynski: A Survey of Software Inspection Checklists. ACM SIGSOFT Software Engineering Notes, Vol. 24, No. 1, 1999, pp. 82-89.

[Fagan 1976] M. E. Fagan: Design and Code Inspections to Reduce Errors in Program Development. IBM Systems Journal, Vol. 15, No. 3, 1976, pp. 258-287.

[Fagan 1986] M. E. Fagan: Advances in Software Inspections. IEEE Transations on Software Engineering, Vol. SE-12, No. 7, 1986, pp. 258-287.

[Gilb and Graham 1993] T. Gilb, D. Graham: Software Inspection. Addison-Wesley, 1993.

[Laitenberger and DeBaud 2000] O. Laitenberger, J.-M. DeBaud: An Encompassing Life Cycle Centric Survey of Software Inspection. Journal of Systems and Software, Vol. 50, No. 1, 2000, pp. 5-31.

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