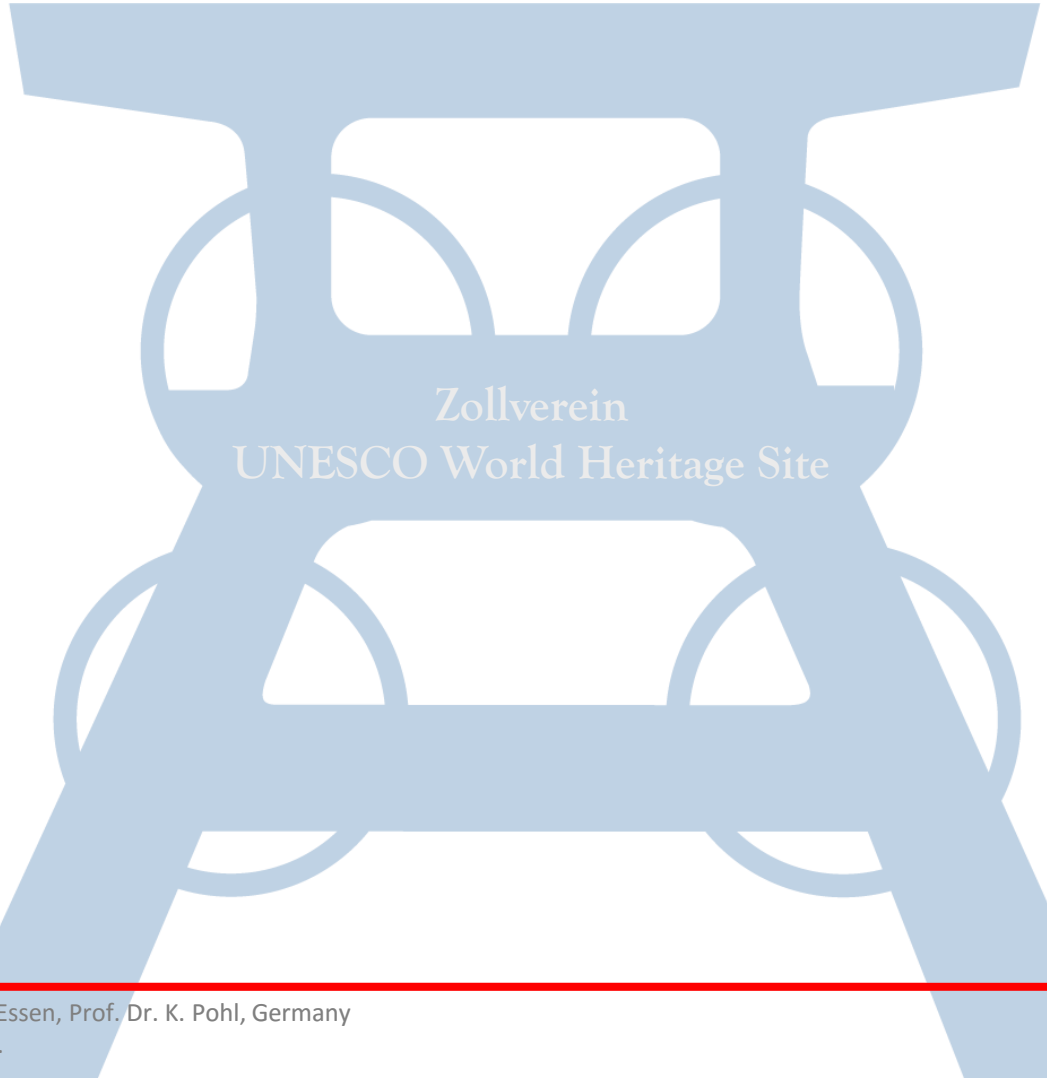
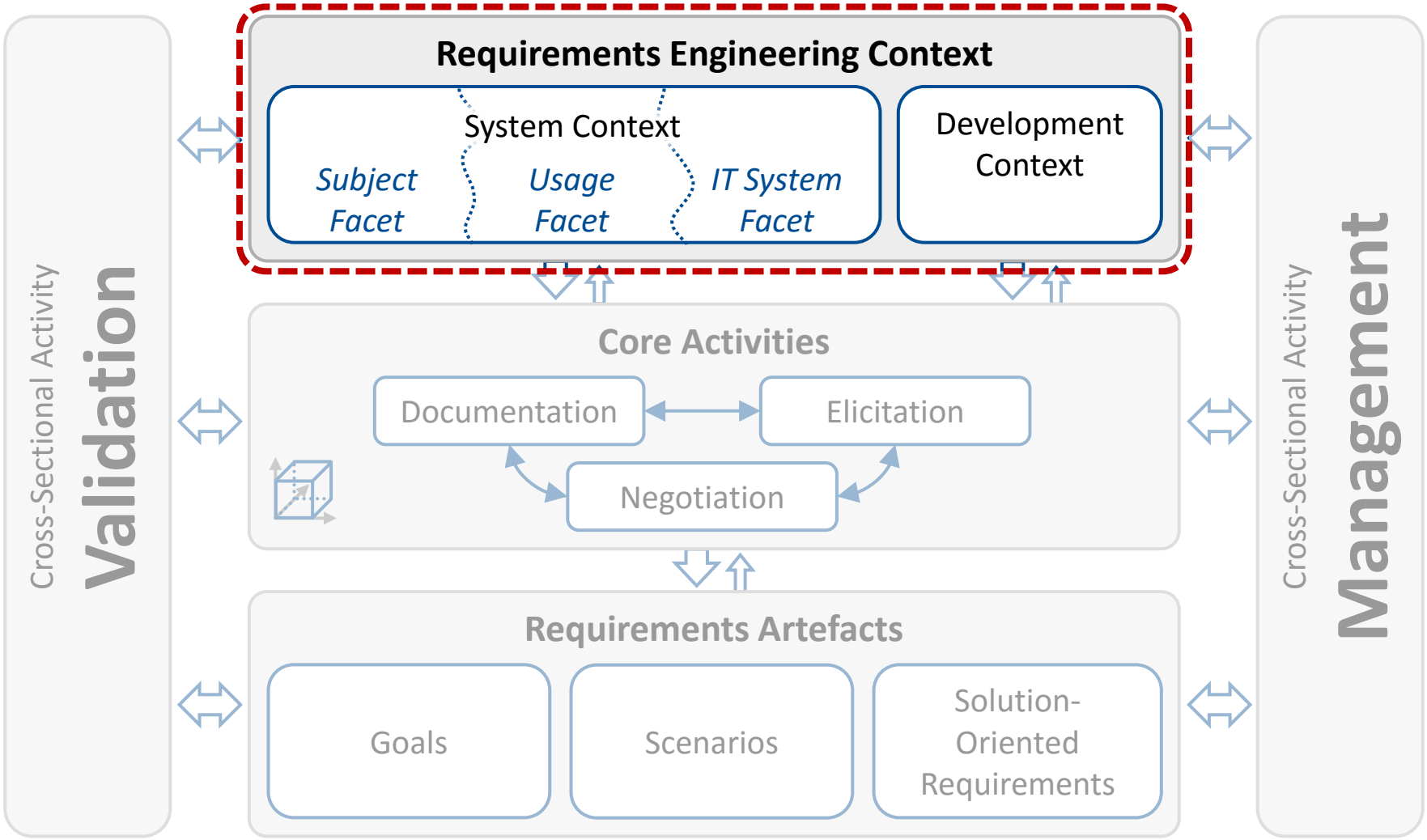
Requirements Engineering & Management

Context II



Requirements Engineering Framework



Teaching Goals

The students

- learn about the importance of the development context.
- learn the difference between system context and development context.
- understand the overall structure of the requirements engineering context.
- learn that a context object can have different roles in different context parts.
- learn to differentiate between the system and the system context.
- learn what can be changed during system development and what cannot be changed.

[2]

Structure of this Lecture

- Development Context
- Requirements Engineering Context
- Different Roles of Context Objects
- System and System Context

[2]

Structure of this Lecture

- **Development Context**
- Requirements Engineering Context
- Different Roles of Context Objects
- System and System Context

[2]

System Context AND Development Context

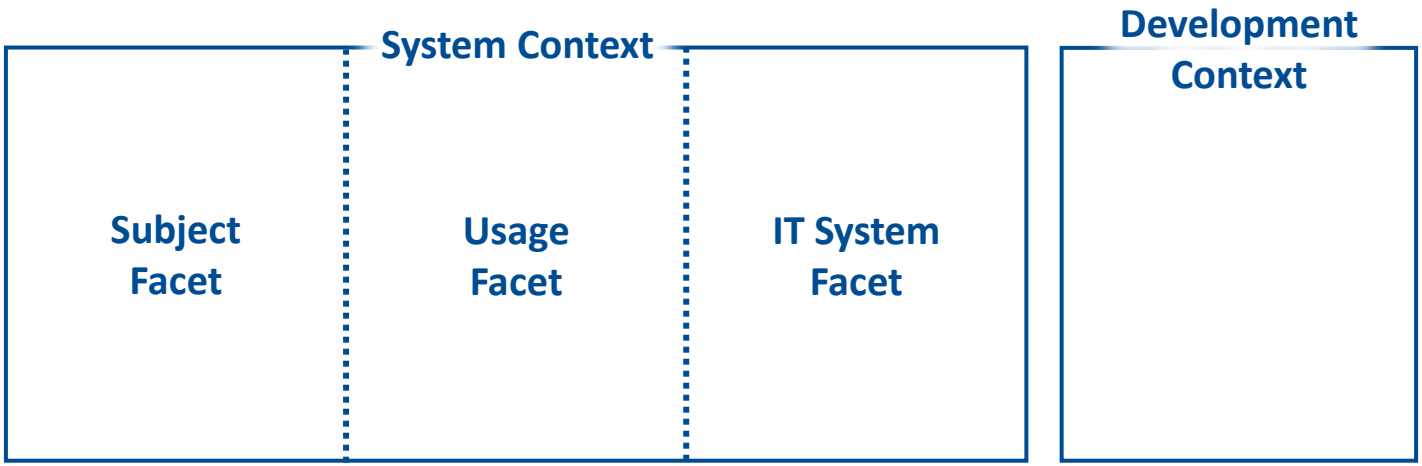
- Just **considering the system context** during requirements engineering **is not sufficient**!
- In addition, requirements and requirements engineering is influenced by the **development process, system development policies** and development **guidelines**.

→ The **development context has to be considered during requirements engineering** in addition to the system context.

Development Context

D The **development context is the part of the context** in which the system is being developed.

D Material or immaterial objects belonging to the development context are called **development context objects**.



Development Context Objects

Examples for objects in the development context:

- Development budget
- Time constraints, e.g., delivery date of the new system
- Qualification of personal, e.g., requirements engineers, architects, …
- Contractual requirements
- Development standards
- Development process guidelines
- Project plan
- Development environments used

Influence of the Development Context

Example: Development of a university library system.

Development Context A:

- Budget: 120.000$
- Max. development time: 2 months
- Development method: Scrum (agile method)
- Available personal: 2 architects, 1 quality assurance engineer
- …

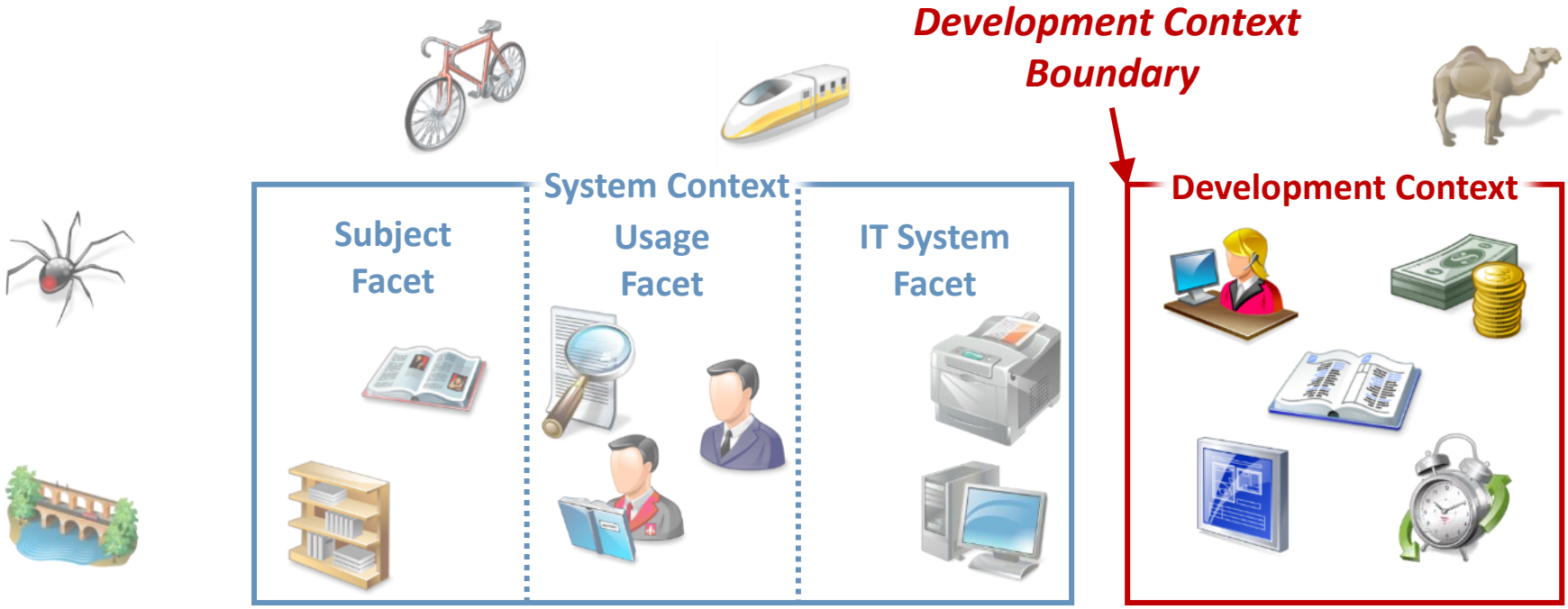
Development Context B:

- Budget: 1.000.000$
- Max. development time: 7 month
- Development method: Scrum (agile method)
- Available personal: 2 requirements engineers, 4 architects, 2 quality assurance engineer
- Quality assurance: according to ISO-xxxxx
- …

The different development contexts significantly influence the way requirements engineering is performed (and the system is being developed)!

Development Context Boundary

D The development context boundary defines which **material and immaterial objects belong to the development context**. It thereby separates development context objects from context objects.



All icons from [1]

Structure of this Lecture

- Development Context
- **Requirements Engineering Context**
- Different Roles of Context Objects
- System and System Context

[2]

Need to consider additional Context Objects

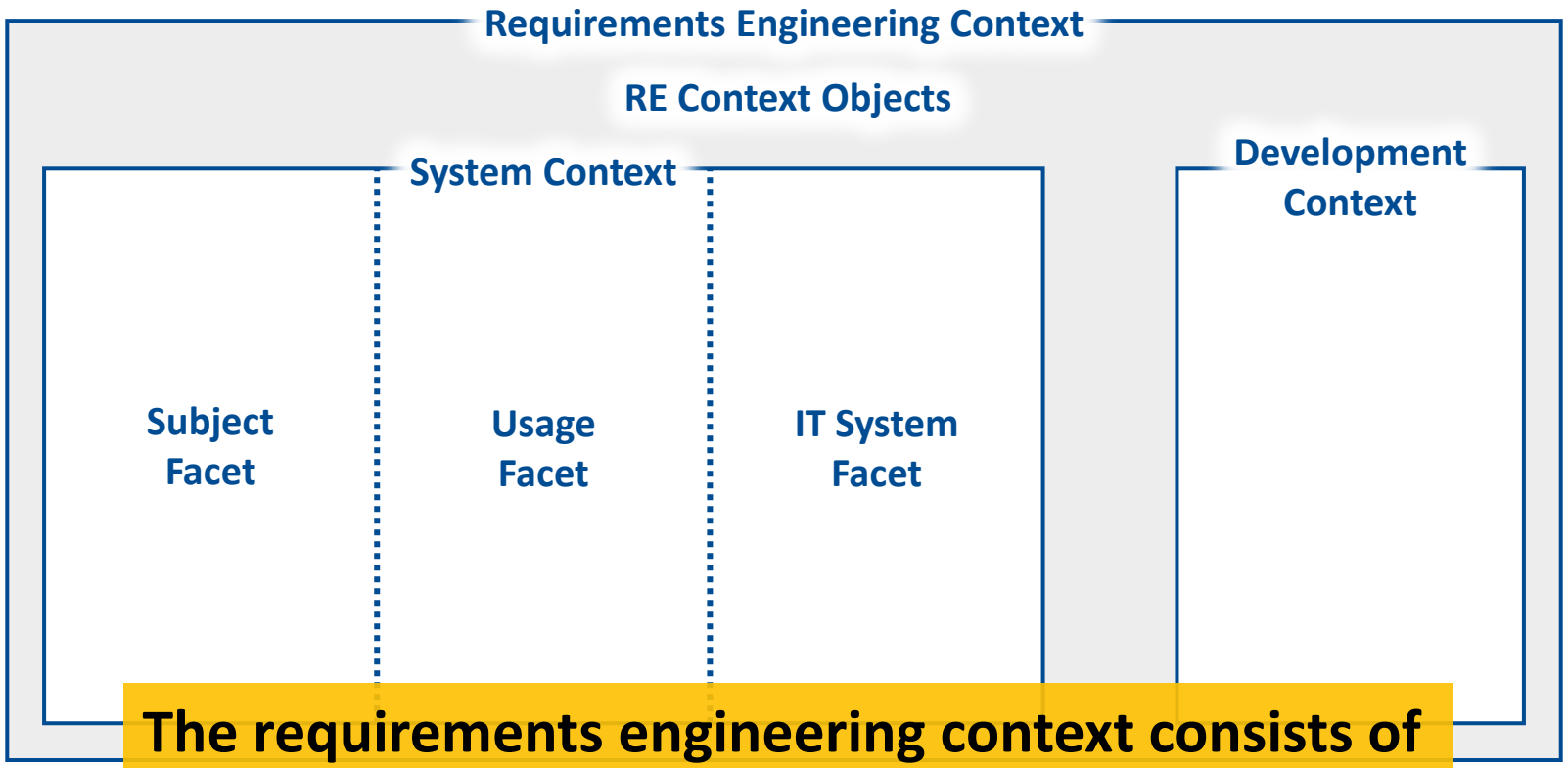
Considering the **system context and the development context** during RE is also **not sufficient**!

E During requirements engineering, **additional context objects** have to be considered such as

- Similar systems of competitors for eliciting requirements.
- An external mediator to foster agreement.
- A domain expert to support validation and QA.
- An expert in formal languages to improve the requirements specifications.
- …

All icons from [1]

Requirements Engineering Context

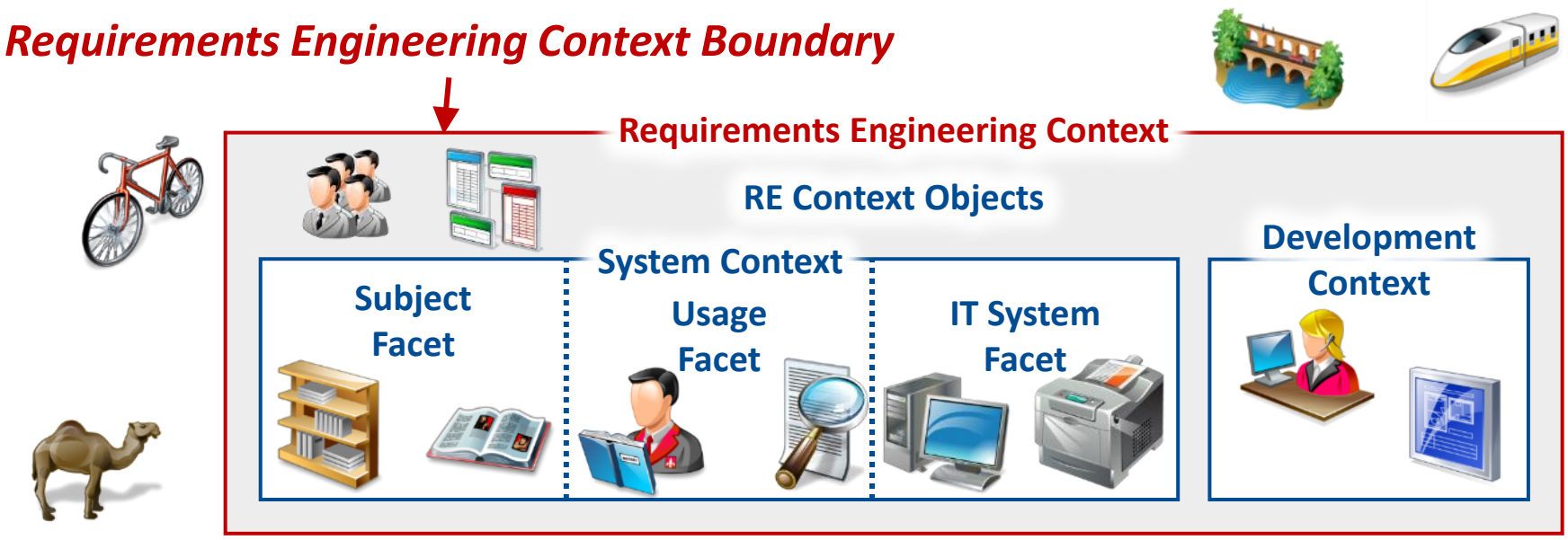


The requirements engineering context consists of

- system context objects
- development context objects
- requirements engineering context objects

Requirements Engineering Context Boundary

D The requirements engineering context boundary defines which **material and immaterial objects belong to the requirements engineering context**. It thereby separates requirements engineering context objects from context objects.



All icons from [1]

Benefits of Structuring the RE Context (1)

- The structure supports stakeholders in identifying important context object and not overlook important ones.
- The structure **facilitates** to focus on one important context aspect at a time.
- A **proven heuristic** for supporting the:
  - Elicitation of requirements.
  - Negotiation of conflicts about requirements.
  - Documentation of requirements.
  - Validation of context consideration.
  - Management of context changes.
  - …

See Lecture “Elicitation”
See Lecture “Negotiation”
See Lecture “Documentation”
See Lecture “Validation”
See Lecture “Management”

Benefits of Structuring the RE Context (2)

- The structured consideration of the RE context leads to a significant **improvement of the quality** of the requirements specification; both in terms of completeness and correctness.
- Fosters the **identification** of important **requirement sources** from all relevant RE context facets.
- Supports to identify **representatives from relevant RE context facets** for validation.
- …

Structure of this Lecture

- Development Context
- Requirements Engineering Context
- **Different Roles of Context Objects**
- System and System Context

[2]

Requirements Engineering Context Objects

- A requirements engineering context object can be **involved in more than one RE activity**.

- The involvement of an RE context object can be

Active	Passive
• Being interviewed • Validating • Documenting • …	• A document which is analysed • A standard which predefines some requirements • …

Passive Involvement: System Documentation of a Predecessor System

E

The manual of a predecessor system includes different information, e.g.,

- Chapter 3 of the manual describes the hardware and software environment of the predecessor system.
- Chapter 4 specifies the database schema of the system.
- Chapter 2 describes the user interface.
- Chapter 11 explains the restrictions to be considered when developing user-defined extensions for the system including the compiler to be used.

Active Involvement: Interview of an Insurance Merchant

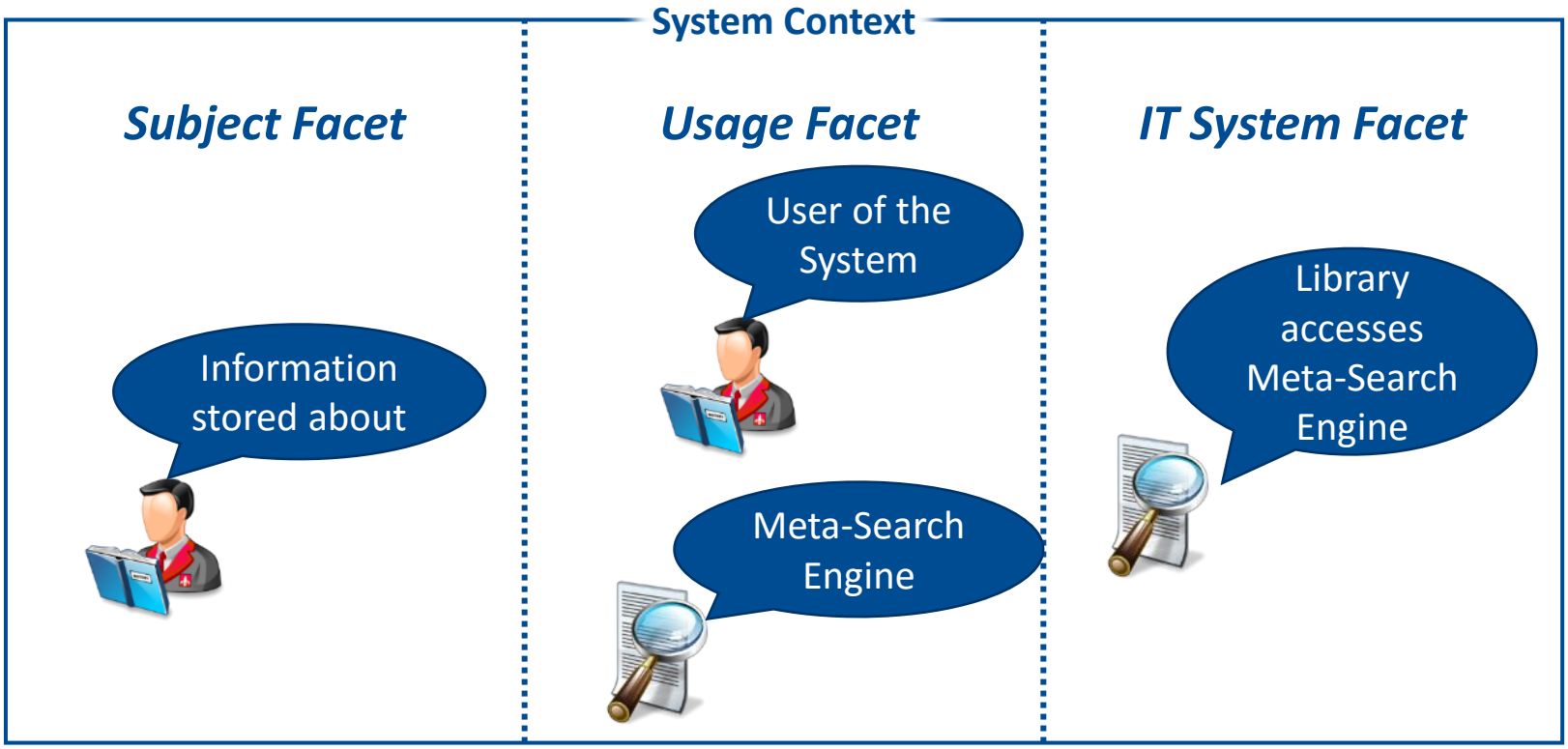
An insurance merchant is interviewed for eliciting requirements for the insurance software. She explains typical workflows in her working environment.

Among others, she describes customer data she enters in the predecessor system. In addition she points out redundant steps in existing processes, which should be eliminated in the future system. She would like to have only one user account for all systems and names additional potential process improvements.

Subject Facet
Usage Facet
IT System Facet

System Context Objects

A system context object can have **different roles in different facets**.

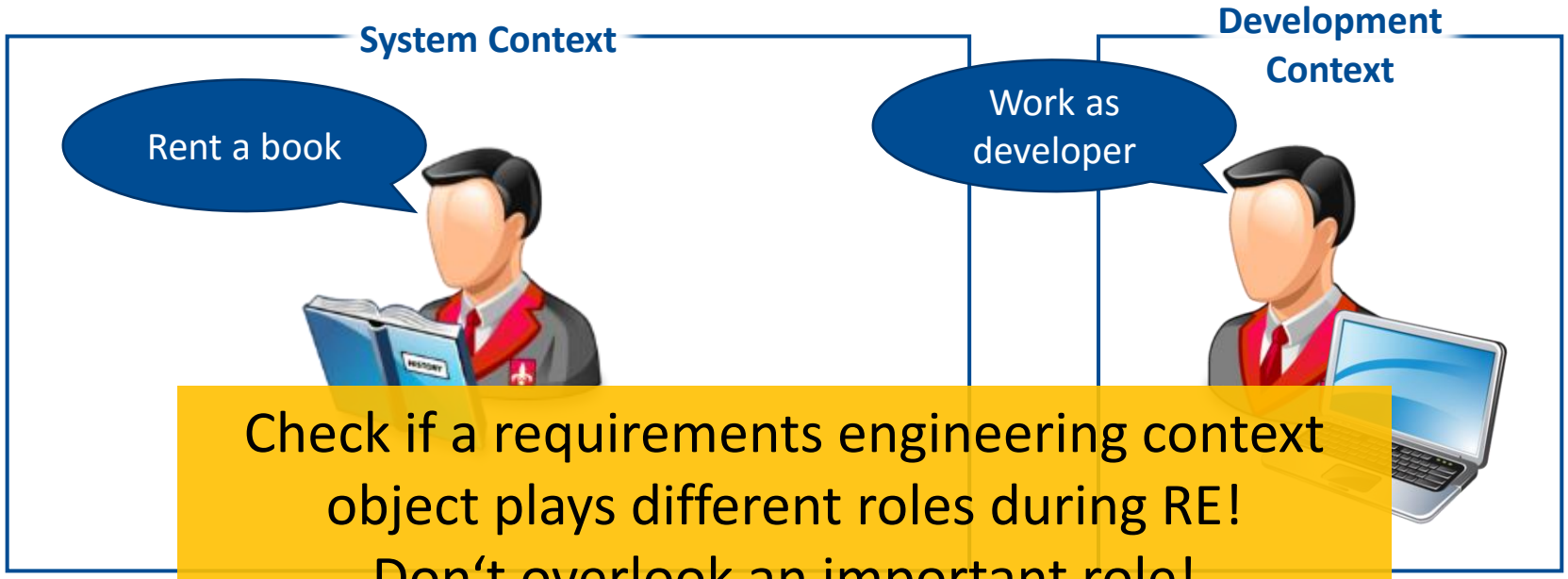


All icons from [1]

System Context & Development Context

A context object can have **different roles in the system context AND the development context**.

The university library system is developed using student assistants as programmers. A student is thus part of the system context and part of the development context.



Check if a requirements engineering context object plays different roles during RE!
Don‘t overlook an important role!

All icons from [1]

Structure of this Lecture

- Development Context
- Requirements Engineering Context
- Different Roles of Context Objects
- **System and System Context**

[2]

System

The system subsumes **all system context objects that can be changed** during the development process.

Typical examples for system context objects which can be changed are

- Functions
- (Business) processes
- Components (Software & Hardware)
- Organizational structures
- Organizational rules and guideline
- Interfaces to other systems
- Interfaces to other organizations
- Sensors and actuators
- …

System and System Context

Example: Development of a university library system.

System Objects

- User data
- Library item data
- User interface
- Check-out terminal
- …

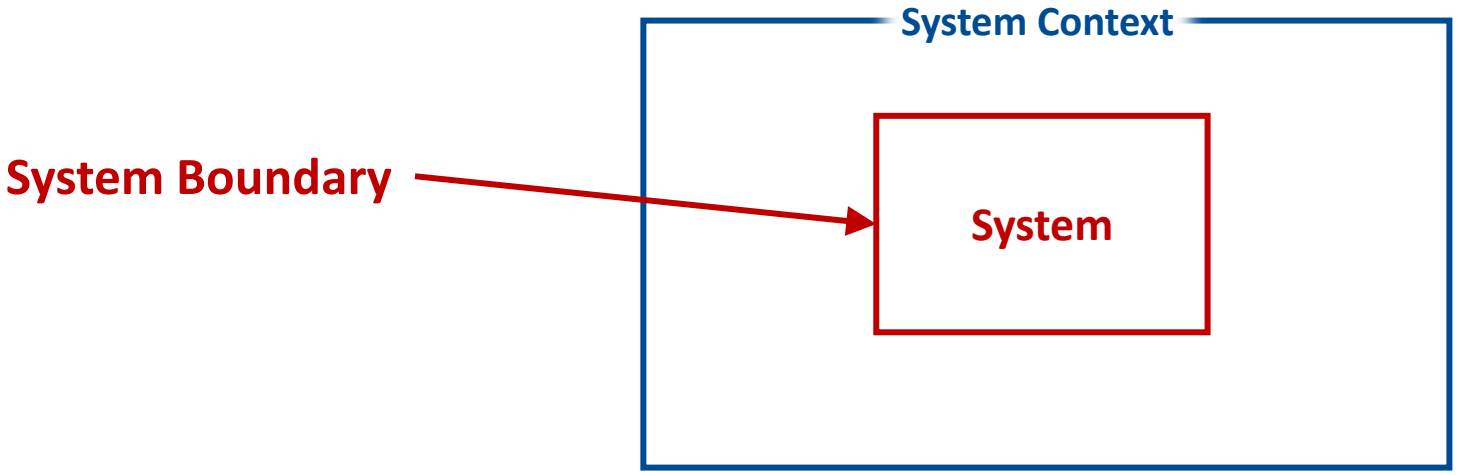
System Context Objects

- Books and magazines
- Marc Genaro (a Student)
- Jennifer Adrian (an employee)
- Workstations
- Students (as user group)
- Employees (as user group)
- Meta-search engine (accessing several library systems)
- User authorization service
- …

System Boundary

D The system boundary separates

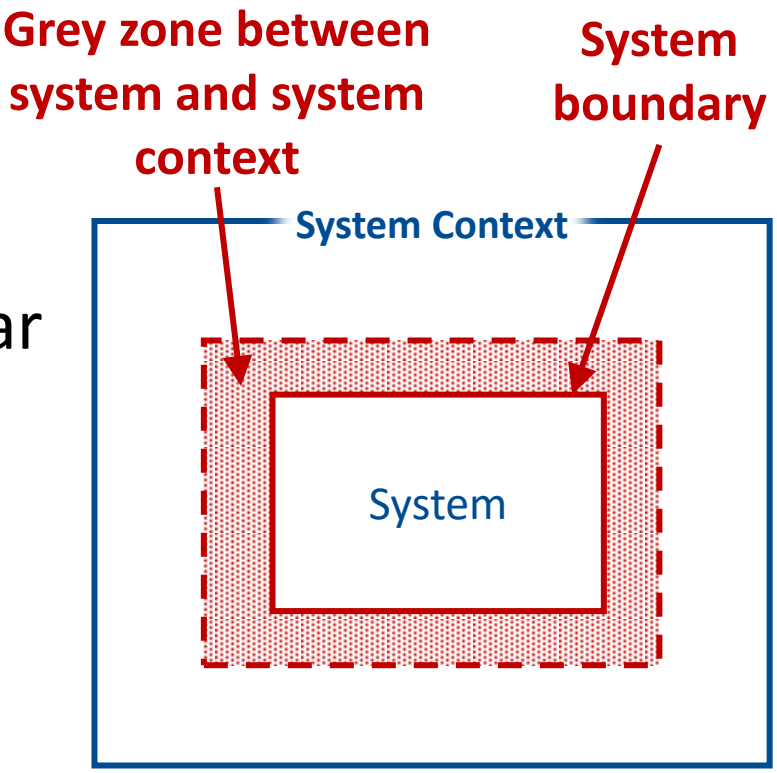
- between **objects that belong to the system**
  → Those objects **can be changed** during the development process!
- and **system context objects**.
  → Those objects **cannot be changed**.



Grey Zone between System and System Context

Typically a grey zone exist between the system and the system context.

The grey zone **subsumes system context objects**, for which it is **not clear** if they **can be changed or not**, i.e., for those object it is not clear if they are system objects or system context objects.



Changes of System Boundary and Grey Zone (1)

During requirements engineering, the **system boundary** and the grey zone boundary are typically unstable and **change frequently**.

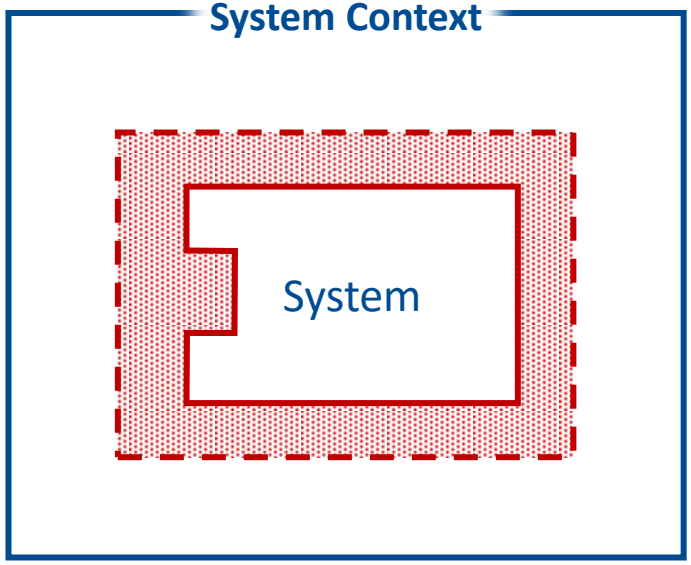
- **System context objects** might become part of the grey zone or the system.
- **Grey zone objects** might become part of the system context or the system.
- **System objects** might become part of the grey zone or the system context.

Changes of System Boundary and Grey Zone (2)

System objects might become part of the **grey zone**.

Example: Development of a university library system.

- To save costs, the **check-out terminals should no longer be developed from scratch**, but reused (if possible).
- It is **unclear if existing terminals can be adjusted** or not.
- The check-out terminals are thus not be considered as system objects anymore.
- They are moved to the grey zone.

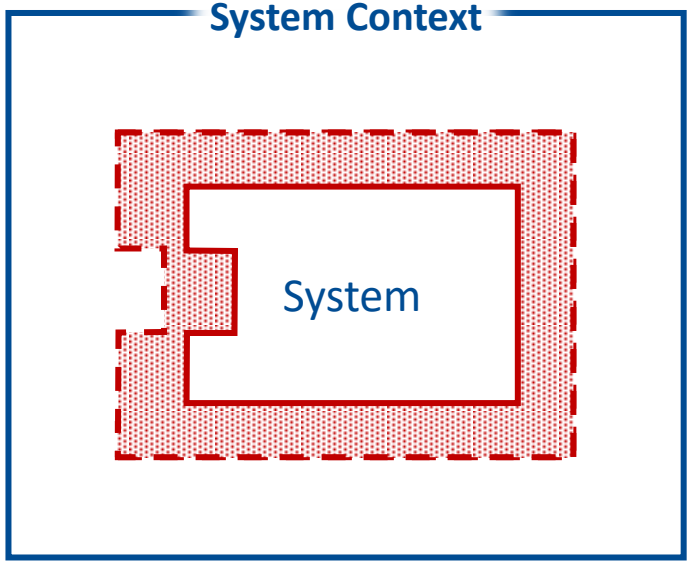


Changes of System Boundary and Grey Zone (3)

System objects might become part of the **system context**.

Example: Development of a university library system.

- To save costs, the **check-out terminals should no longer be developed from scratch**, but, if possible reused.
- The existing terminals **cannot be changed** (their interfaces are fixed).
- In this case the check-out terminals are now considered system context objects instead of system objects.

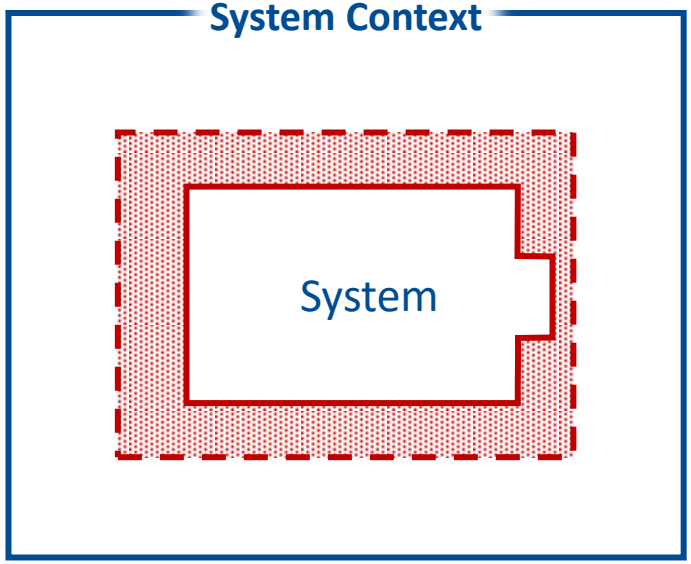


Changes of System Boundary and Grey Zone (4)

Grey zone objects might become part of the **system**.

Example: Development of a university library system.

- It turns out that the **re-used check-out terminals can be and should be changed**.
- The terminals now become system object again.

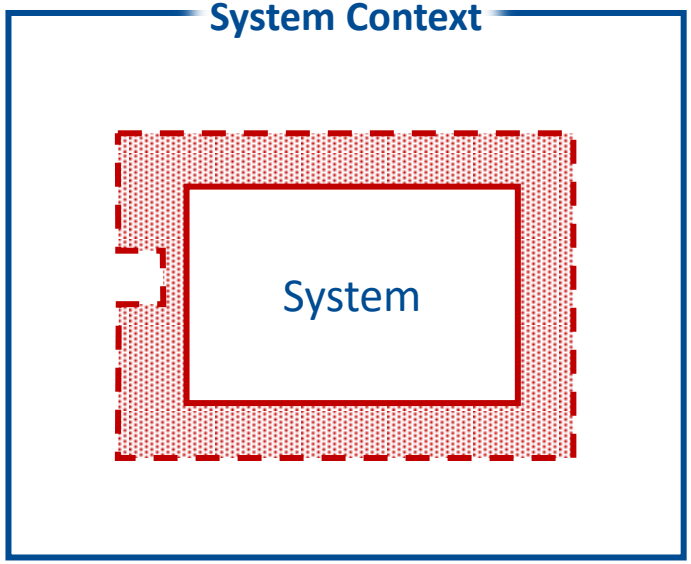


Changes of System Boundary and Grey Zone (5)

Grey zone objects might become part of the **system context**.

Example: Development of a university library system.

- After checking the technical parts of the **old workstations**, it turns out that they **can be reused as they are**.
- Therefore the old workstations which belongs to the grey zone becomes a system context object.

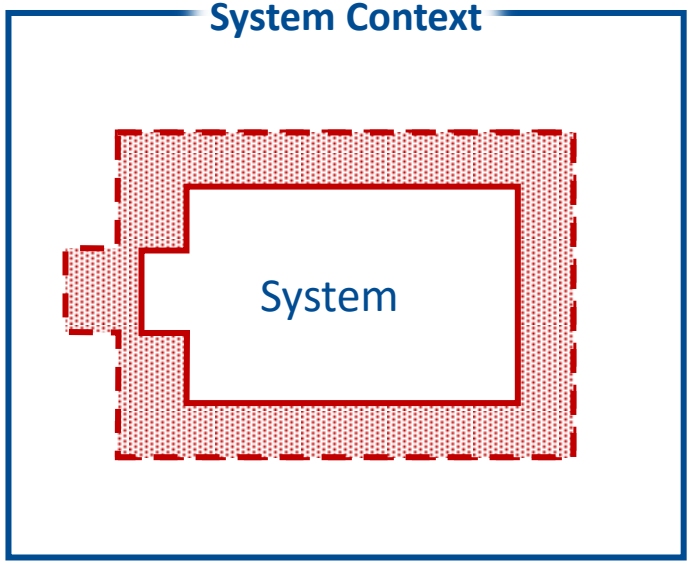


Changes of System Boundary and Grey Zone (6)

System context objects might become part of the **system**.

Example: Development of a university library system.

- It turns out that some **very old work-stations** of librarians **would need to be updated to be reusable** for the system.
- Fortunately, it turns out that the **work-stations can be updated**.
- Therefore the system context object old librarian work-station becomes a system object.

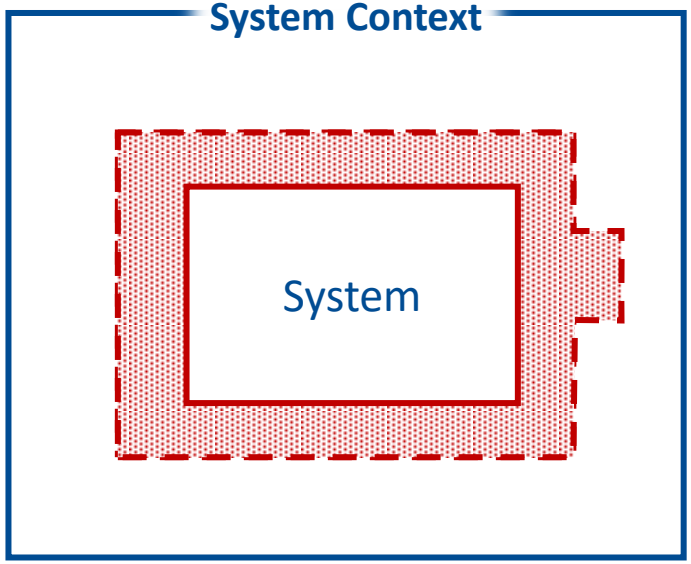


Changes of System Boundary and Grey Zone (7) E

System context objects might become part of the **grey zone**.

Example: Development of a university library system.

- It turns out that **old workstations would need to be adapted to be reused** with the system.
- It is **not clear if** those workstations **can be changed** accordingly.
- Therefore the system context object old workstation becomes part of the grey zone.



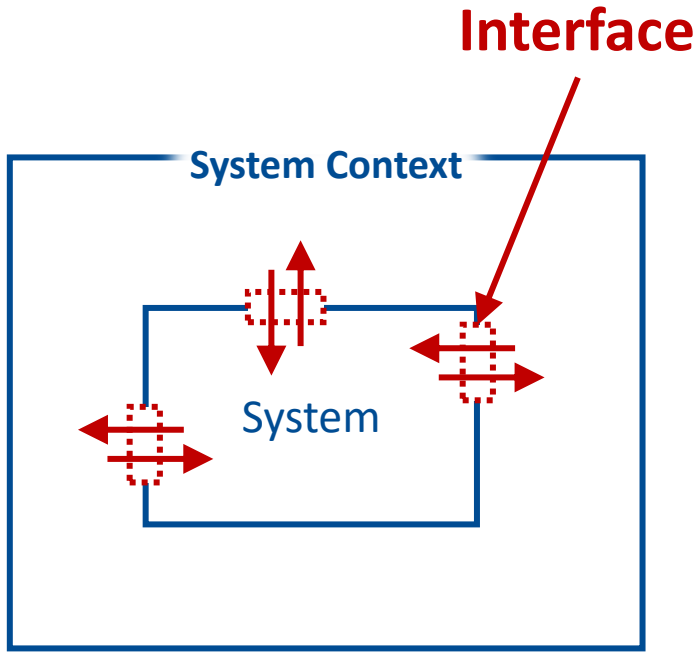
Size of the Grey Zone

- If at a given point in time it cannot be decided if a context object belongs to the system context or the system, the object should be part of the grey zone.
- **Don't move grey zone objects**, for which it is **unclear** if they **can be changed (or not)**, to the system (or the system context).
- **Keep the grey zone as small as possible**, i.e. check frequently for each grey zone object if it can be decided if it belongs to the system or the system context.
- **Briefly**: the grey zone should not be too big but also not too small.

System Interfaces (1)

- The **system interacts via interfaces** with the **system context objects** and vice versa.

- The system **interfaces typically change during requirements engineering** and system development.



System Interfaces (2)

Example: Development of a university library system.

Typical system interfaces of a library system are:

- **User interfaces**, e.g., for library users or library staff.
- Interfaces **to other systems**, e.g., authorization systems, order management system.
- Interfaces **to peripheral devices**, e.g., book scanner, printers.
- …

System Interfaces and System Context Facets

- The **system** can **interact via a system interface** with system context objects from **each of the three system context facets**:
  - **Subject facet**: e.g., receive data, update data, …
  - **Usage facet**: e.g., provide output to the user, …
  - **IT system facet**: e.g., interfaces to the IT infrastructure, …

- A single interface can define **interactions of the system with system context objects from all three system context facets**!

Summary

- The development context has to be additionally considered during requirements engineering.
- The requirements engineering context consists of system context objects, development context objects and additional requirements engineering objects.
- Requirements engineering context objects might have different roles. Do not overlook an important role!
- The system boundary separates the system from the system context.
- System objects can be changed during development; system context objects cannot be changed.
- A grey zone exists between the system and the system context.
- The system interacts via interfaces with the system context.

[2]

Questions for Self-Assessment

- Why should the development context be considered during RE?
- How is the requirements engineering context structured?
- What are the benefits of structuring the RE context?
- Name typical examples for additional requirements engineering context objects?
- Which role can a RE context objects have?
- Which RE context object can be changed during RE and development?
- Why is there a grey zone between the system and the system context?
- Can this grey zone change? If so, how?
- With which type of context object does the system interact?

[2]

Literature

[Pohl 2010] K. Pohl: Requirements Engineering - Fundamentals, Principles and Techniques. 1st edition, Springer, 2010.

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

[2]

Literature for Further Reading

[Gunter et al. 2000] C. A. Gunter, E. L. Gunter, M. Jackson, P. Zave: A Reference Model for Requirements and Specifications. IEEE Software, Vol.17, NO. 3, IEEE Press, Los Alamitos, 2000, pp. 37-43.

[McMenamin and Palmer 1984] S. M. McMenamin, J. F. palmer: Essential Systems Analysis. Prentice Hall, London, 1984.

[Parnas and Madey 1995] D. L. Parnas, J. Madey: Functional Documents for Computer Systems. Science of Computer Programming, Vol. 25, No. 1, Elsevier, North-Holland, Amsterdam, 1995, pp. 41-61.

[Sutcliffe 2002] A. Sutcliffe: The Domain Theory – Patterns for Knowledge and Software Reuse. L. Erlbaum, Mahwah, 2002.

Image References

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

[2] Provided by Microsoft Office

This Lecture Material was produced by

Prof. Dr. Klaus Pohl *Dr. Thorsten Weyer*

Torsten Bandyszak *Philipp Bohn* *Jennifer Brings*

Johanna Buchner *Marian Daun* *Felix Föcker*

Andrea Salmon *Philipp Schmidt* *Bastian Tenbergen*

paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen
Germany

UNIVERSITÄT DUISBURG ESSEN

Open-Minded

PALUNO
The Ruhr Institute for Software Technology
https://www.paluno.uni-due.de