

# Requirements Engineering I

## Chapter 4

# Elaboration of Requirements

---



# Chapter roadmap



4.1

**Information sources**  
Where to look

4.4

**Eliciting innovative requirements**  
Making stakeholders happy

4.2

**Elicitation practices**  
The toolbox for requirements engineers

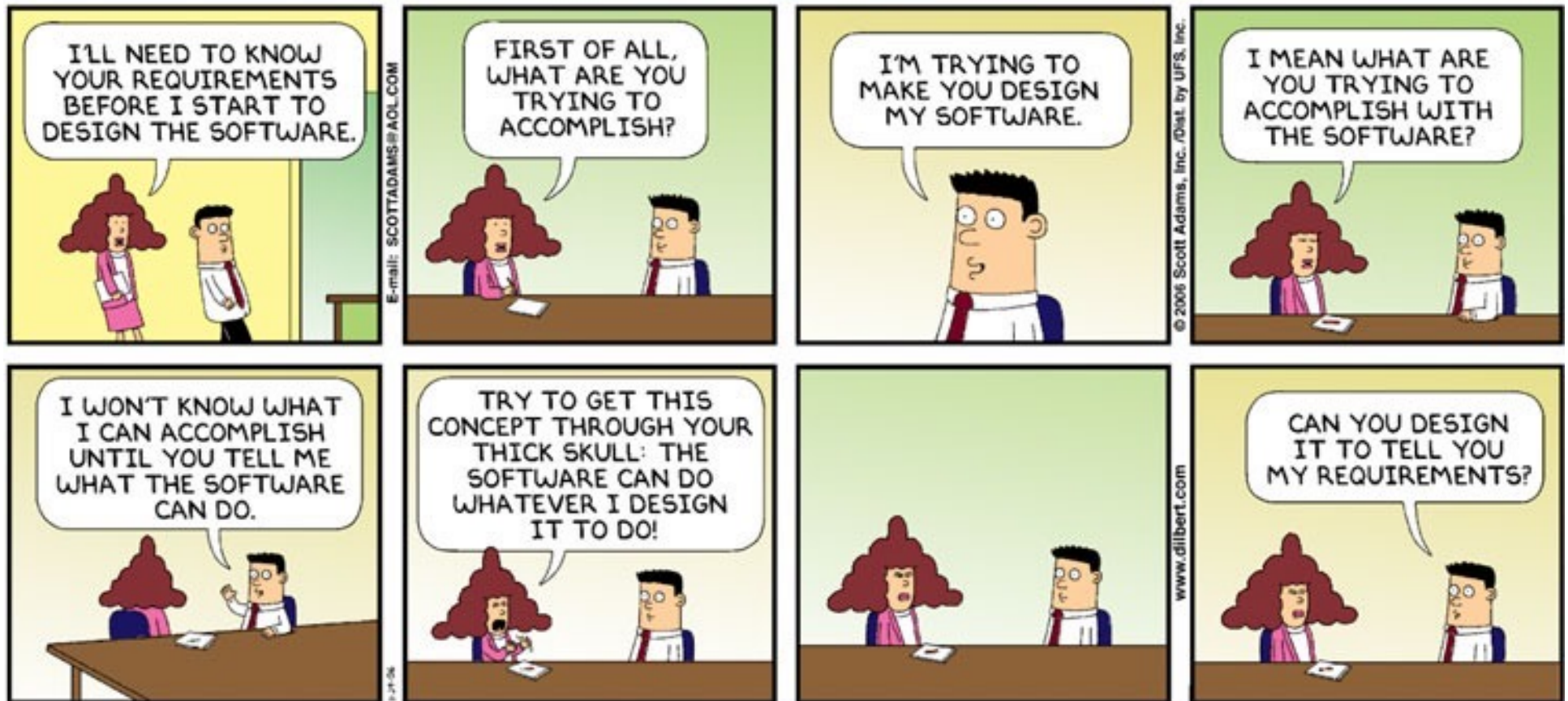
4.5

**Requirements negotiation**  
Resolving conflicts

4.3

**Analyzing and documenting**  
From chaos to structure

# The Problem



# What to do

---

## The prerequisites

- Knowing and tapping the **information sources**
- Knowing the **goals**
- Considering the **context**

## The tasks

- **Elicit, analyze and document** requirements
- **Negotiate**
- **Validate** (→ Chapter 9)

# 4.1 Information sources

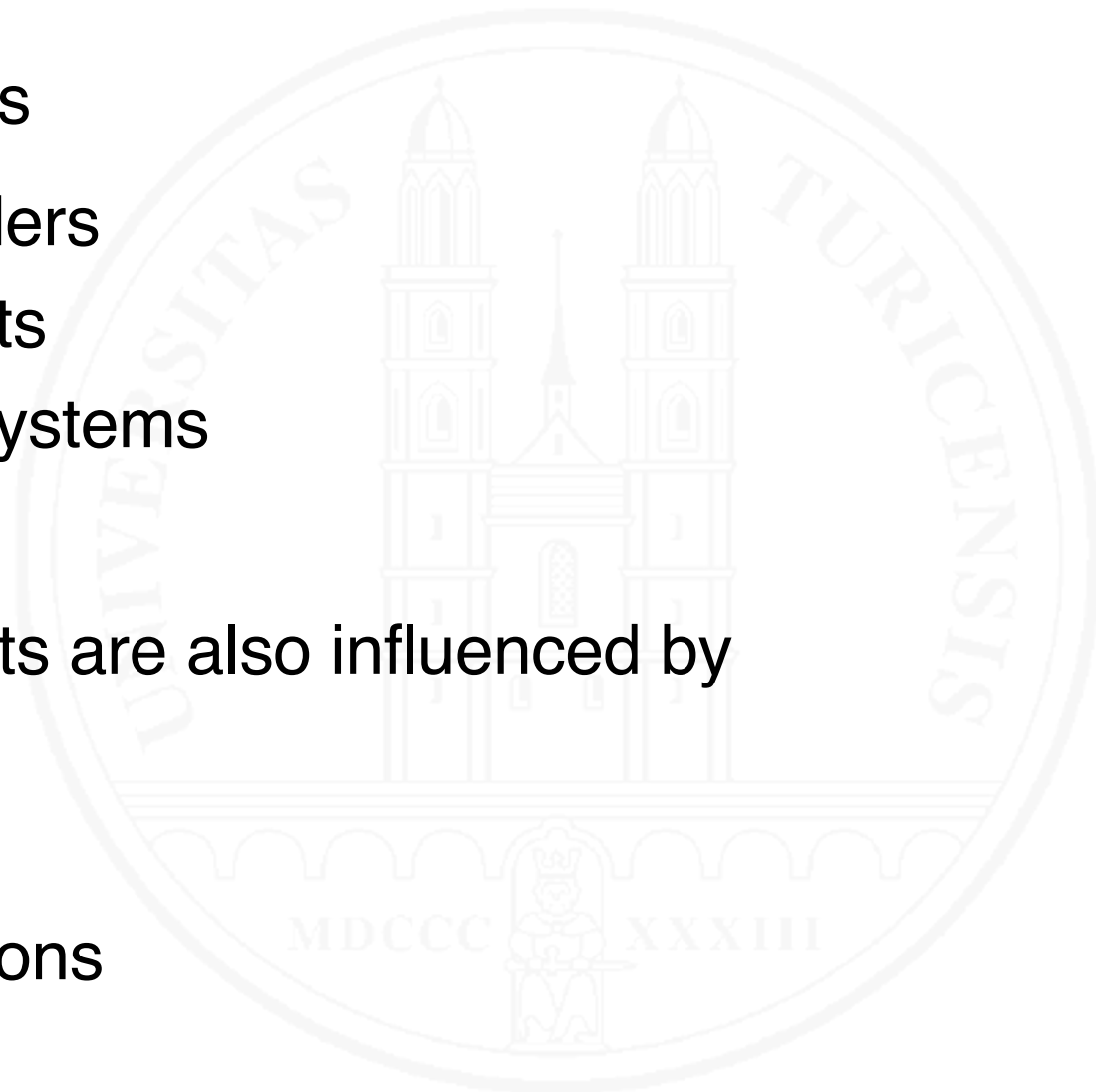
---

## Main sources

- Stakeholders
- Documents
- Existing systems

## Requirements are also influenced by

- Context
- Goals
- Observations



# Stakeholder analysis

---

Identify stakeholder **roles**

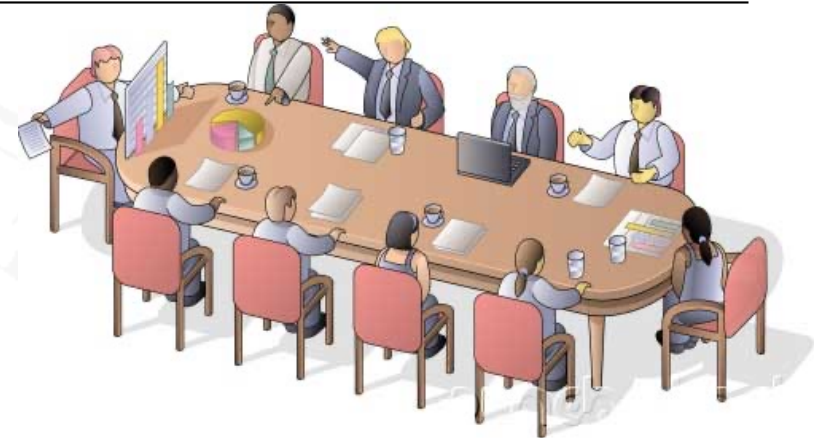
End user, customer, operator,  
project manager, regulator,...

In complex cases: Build model of stakeholder **goals, dependencies** and **rationale**

Classify stakeholders

- **Critical**
- **Major**
- **Minor**

Identify/determine **concrete persons** for each stakeholder role



[Yu 1997]

[van Lamsweerde 2001]

[Glinz and Wieringa 2007]

# Identifying stakeholders / stakeholder roles

---

- Start with the **obvious ones**: end user, customer, ...
- Consider all people, organizations, systems who will **directly interact** with the system
- Consider further people and organizations in the system context who **influence** the system, e.g., regulators
- Ask already identified stakeholders (**snowballing**)
- **Keep a list** of your stakeholders (name, role, contact data, degree of influence, availability)

# Personas

---

What if no concrete persons can be determined for a stakeholder role? → Define **personas**

DEFINITION. **Persona** – A fictitious character representing a **group of users with similar needs, values and habits** who are expected to use a system in a similar way.

A persona typically comprises

- Personal info: Name, photo, age, gender, family, profession,...
- Personality: Introvert/extrovert, conservative/open, ...
- Values, goals, frustrations, motivations



# A sample persona

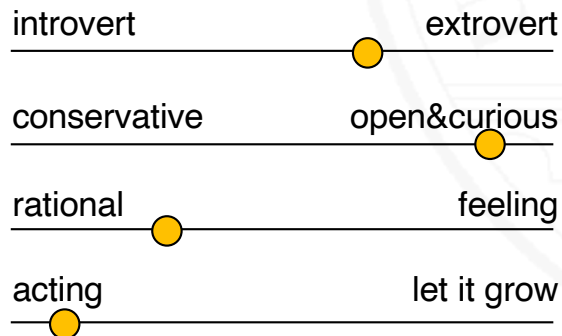
## Personal



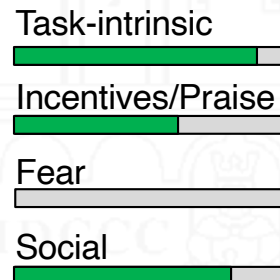
Name: Anna Schneider  
Age: 30  
Gender: female (she/her)  
Family: Married, a son (2 y.)  
Education: MSc in Informatics  
Profession: Software Engineer  
Works: 60%

“Do things well and enjoy it”

## Personality



## Motivation



## Goals

Develop good&useful products  
Become a digital design expert  
Have her own company in 6-8 years  
Good family life, have a 2<sup>nd</sup> child

## Frustrations

Unrealistic expectations  
Ignorant bosses / colleagues  
Routine tasks

## Values

Work-life balance over high income  
Quality over speed  
Analytic thinking over hacking

**Bio** [here](#)

# Mini-Exercise

---

Consider the chairlift access control case study.

Perform the following elements of a stakeholder analysis.

- Identify stakeholder roles.
- Assess the degree of influence of the stakeholder roles found.
- For which roles will it be possible to identify concrete people as representatives – and where do you need to create personas?

# Documents as sources for requirements

---

Documents can be a rich source for requirements

- Artifacts produced or consumed in business processes
- Process or procedure descriptions
- Regulatory documents
- Company guidelines
- ...
  
- Keep a list of identified documents, their relevance and where to find them

# Existing systems as sources for requirements

---

- Existing systems as a source of requirements:
  - **Renewing/renovating** legacy systems
  - **Copying/mimicking** successful parts of an existing system (“The user interface shall look and feel the same as the order processing interface of system xxx”)
  - Developing a new product shall **beat** an existing product of a **competitor**
- Existing system as partial **specifications by example**
- Beware: Existing legacy systems can also be a source of **negative requirements**: what stakeholders do **not** want

# Context analysis

---

Determine the system's **context** and the context **boundary**

**Identify context constraints**

- Physical, legal, cultural, environmental
- Embedding, interfaces



Photo © Universitätsklinikum Halle (Saale)

**Identify assumptions** about the context of your system and make them **explicit**

Map real world phenomena adequately on the required system properties and capabilities (and vice-versa)

Determine the **system scope** (cf. Chapter 2.4)

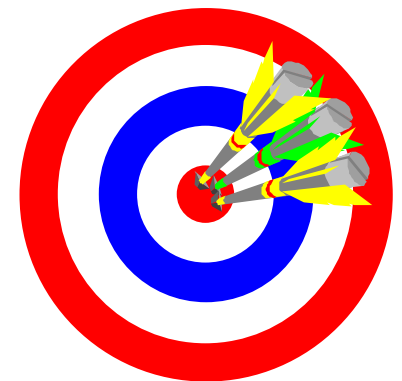
# Goal analysis

---

*Knowing your destination is more important than the details of the timetable.*

Before eliciting detailed requirements, the general **goals** and **vision** for the system to be built must be clear

- What are the main goals?
- How do they relate to each other?
- Are there goal conflicts?



# The role of (informal) observations

---

When interacting with stakeholders, keep an open eye on **informal observations**, such as

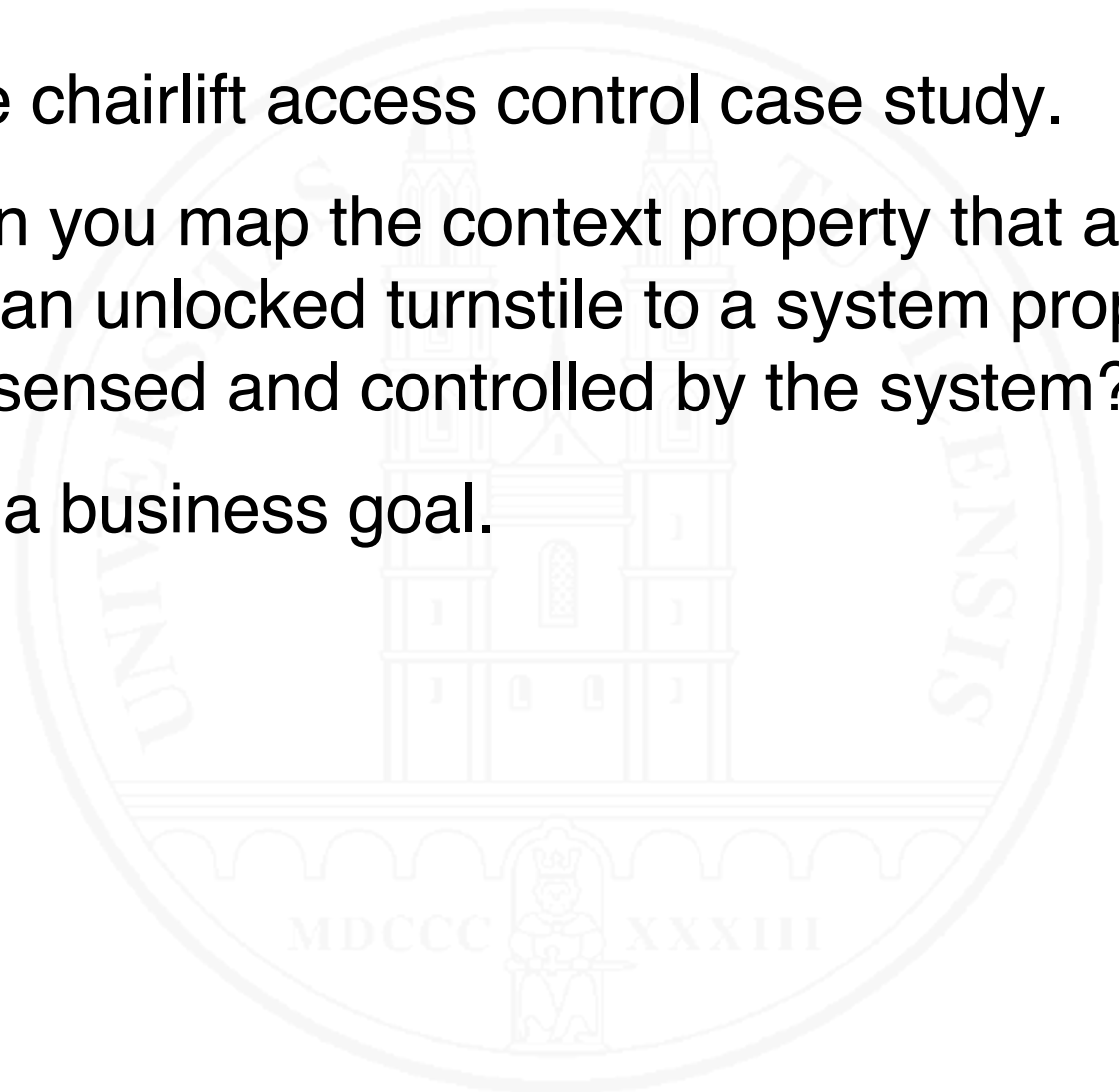
- Stakeholder **behavior** (who is open for change, who is afraid of change, who is central in the social network, who dominates,...)
  - **Conflicts** and power **relationships** between stakeholders
  - Coffee break **chats**
- Can be a **source** for requirements and also helps when **negotiating requirements conflicts**

# Mini-Exercise

---

Consider the chairlift access control case study.

- (a) How can you map the context property that a skier passes an unlocked turnstile to a system property which can be sensed and controlled by the system?
- (b) Identify a business goal.





## 4.2 Elicitation practices

---

DEFINITION. **Requirements elicitation** – The process of **seeking**, **capturing** and **consolidating** requirements from available sources, potentially including the **re-construction** or **creation** of requirements.

- Determine the stakeholders' **desires** and **needs**
- Elicit information from all available **sources** and **consolidate** it into **well-documented requirements**
- Make stakeholders **happy**, not just satisfy them
- Every elicited and documented requirement must be **validated** and **managed**
- Work **value-oriented** and **risk-driven**

# Elicitation techniques

---

## Ask

- ▣ Interview stakeholders
- Use questionnaires and polls
- Reply/follow-up to user feedback

## Collaborate

- Hold requirements workshops
- Provide community platforms

## Build and play

- ▣ Build, explore and discuss prototypes (cf. Chapter 3)
- Perform role playing



[Zowghi and Coulin 2005]  
[Dieste, Juristo, Shull 2008]  
[Gottesdiener 2002]  
[Hickey and Davis 2003]  
Kolpondinos and Glinz 2019]  
[Goguen and Linde 1993]

# Elicitation techniques – 2

---

## Observe

- Observe stakeholders in their work context

## Analyze

- Analyze work products
- Analyze user feedback
  - Direct feedback: problem/bug reports, app reviews, tweets, explicit feedback channels, ...
  - Indirect feedback: user forums, system usage monitoring, ...
- Conduct market studies
- Perform benchmarking

# Which technique for what?

---

Technique	Suitability for			
	Express needs	Demonstrate opportunities	Analyze system as is	Explore market potential
Interviews	+	-	+	0
Questionnaires and polls	0	-	+	+
Workshops, Community platforms	+	0	0	0
Explorative prototypes	0	+	-	0
Role play	+	0	0	-
Stakeholder observation	0	-	+	0
Work product analysis	0	-	+	-
User feedback analysis	+	-	-	0
Market studies	-	-	0	+
Benchmarking	0	+	-	+

# Typical problems

---

Inconsistencies among stakeholders in

- needs and expectations
- terminology

Stakeholders who know their needs, but **can't express** them

Stakeholders who **don't know** their needs

Stakeholders with a **hidden agenda**

Stakeholders thinking in **solutions** instead of problems

Stakeholders frequently **neglect quality requirements** and **constraints**

→ Elicit them explicitly

# Who should elicit requirements?

---

- Stakeholders must be involved
- Domain knowledge is essential
  - Stakeholders need to have it (of course)
  - Requirements engineers need to know the main domain concepts
  - A “smart ignoramus” can be helpful [Berry 2002, Sect. 7]
- Don’t let stakeholders specify themselves without professional support
- Best results are achieved when stakeholders and requirements engineers collaborate

# Mini-Exercise

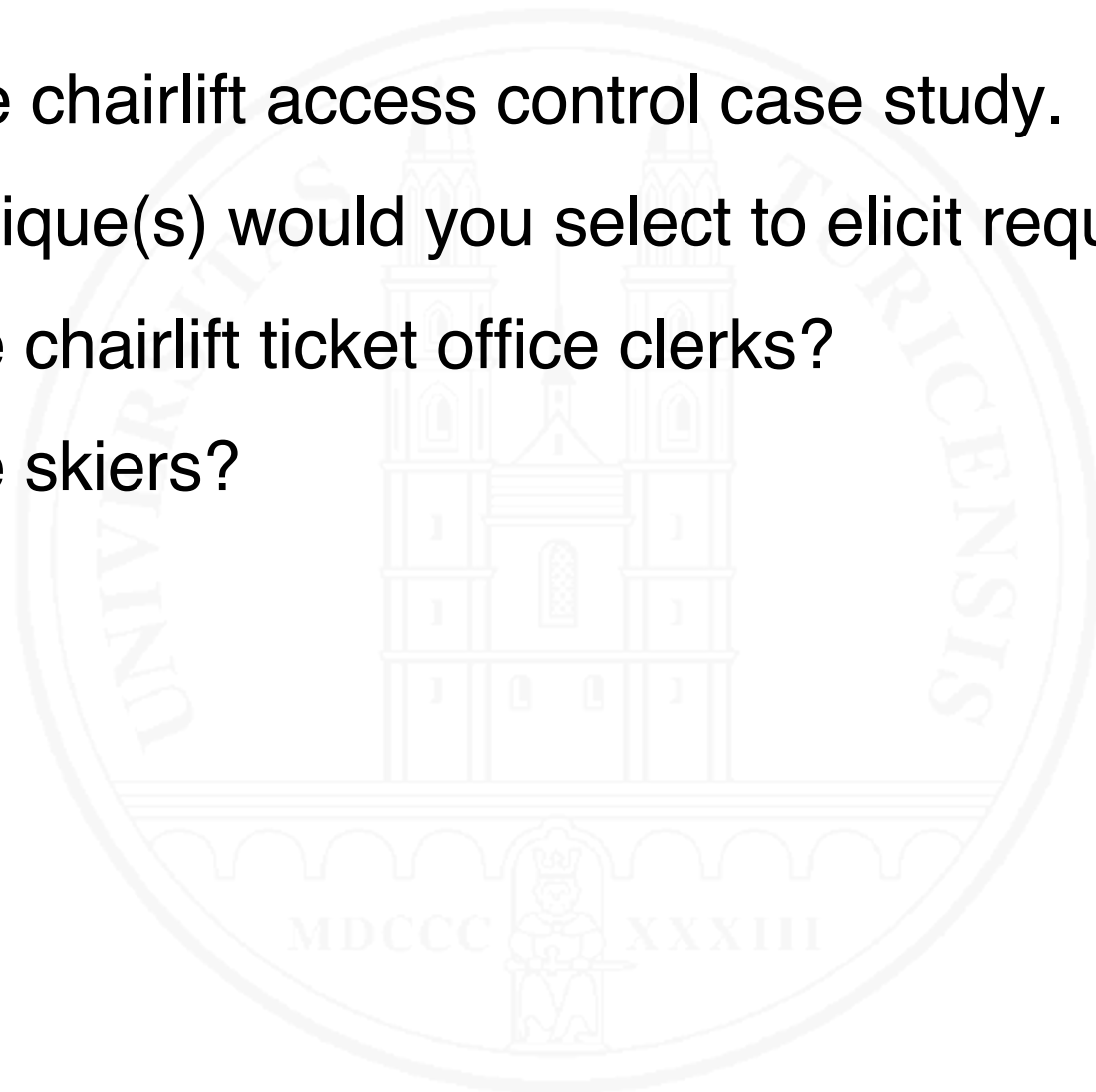
---

Consider the chairlift access control case study.

Which technique(s) would you select to elicit requirements

(a) from the chairlift ticket office clerks?

(b) from the skiers?



# Eliciting functional requirements

---

- Who wants to achieve what with the system?
- For every identified function
  - What's the desired result and who needs it?
  - Which transformations and which inputs are needed?
  - In which state(s) shall this function be available?
  - Is this function dependent on other functions?
- For every identified behavior
  - In which state(s) shall the system have this behavior?
  - Which event(s) lead(s) to this behavior?
  - Which event(s) terminate(s) this behavior?
  - Which functions are involved?



# Eliciting functional requirements – 2

---

- For every identified **data** item
  - What are the required **structure** and the **properties** of this item?
  - Is it **static** data or a data **flow**?
  - If it's static, must the system keep it **persistently**?
- Analyze **mappings**
  - How do real world functions/behavior/data map to system functions/behavior/data and vice-versa?
- Specify **normal and exceptional** cases

# Eliciting quality requirements

---

Stakeholders frequently state quality requirements in qualitative form:

“The system shall be fast.”

“We need a secure system.”

Problem: Such requirements are

- Ambiguous
  - Difficult to achieve and verify
- Classic approach:
- Quantification → ⊕ measurable ⊖ maybe too expensive
  - Operationalization → ⊕ testable ⊖ implies premature design decisions

# New approach to eliciting quality requirements

---

[Glinz 2008]

Represent quality requirements such that they deliver **optimum value**

**Value** of a requirement = **benefit** of development risk reduction  
**minus cost** for its specification

- Assess the criticality of a quality requirement
- Represent it accordingly
- Broad range of possible representations

# The range of adequate representations

---

<b>Situation</b>	<b>Representation</b>	<b>Verification</b>
1. Implicit shared understanding	Omission	Implicit
2. Need to state general direction Customer trusts supplier	Qualitative	Inspection
3. Sufficient shared understanding to generalize from examples	By example	Inspection, (Measurement)
4. High risk of not meeting stake- holders' desires and needs	Quantitative in full	Measurement
5. Somewhere between 2 and 4	Qualitative with partial quantification	Inspection, partial measurement

# Eliciting performance requirements

---

## Things to elicit

- **Time** for performing a task or producing a reaction
- **Volume** of data
- **Throughput** (data transmission rates, transaction rates)
- **Frequency** of usage of a function
- **Resource consumption** (CPU, storage, bandwidth, battery)
- **Accuracy** (of computation)

# Eliciting performance requirements – 2

---

- What's the meaning of a performance value:
  - Minimum?
  - Maximum?
  - On average?
  - Within a given interval?
  - According to some probability distribution?
- How much deviation can be tolerated?

# Eliciting specific quality requirements

---

- Ask stakeholders explicitly
- A quality model can be used as a checklist
- Quality models also help when a specific quality requirement needs to be quantified

The ISO/IEC 25010:2011 quality model (formerly ISO/IEC 9126) has major issues that limit its usability

There is ongoing work to improve the situation

[Glinz et al. 2023]

# Eliciting constraints

---

- Ask about **restrictions** of the potential **solution space**
  - **Technical**, e.g., given interfaces to neighboring systems
  - **Legal**, e.g., restrictions imposed by law, standards or regulations
  - **Organizational**, e.g. organizational structures or processes that must not be changed by the system
  - **Cultural, environmental, ...**
- Check if a requirement is **concealed** behind a constraint
  - Constraint stated by a stakeholder: **“When in exploration mode, the print button must be grey.”**
  - Actual requirement: **“When the system is used without a valid license, the system shall disable printing.”**



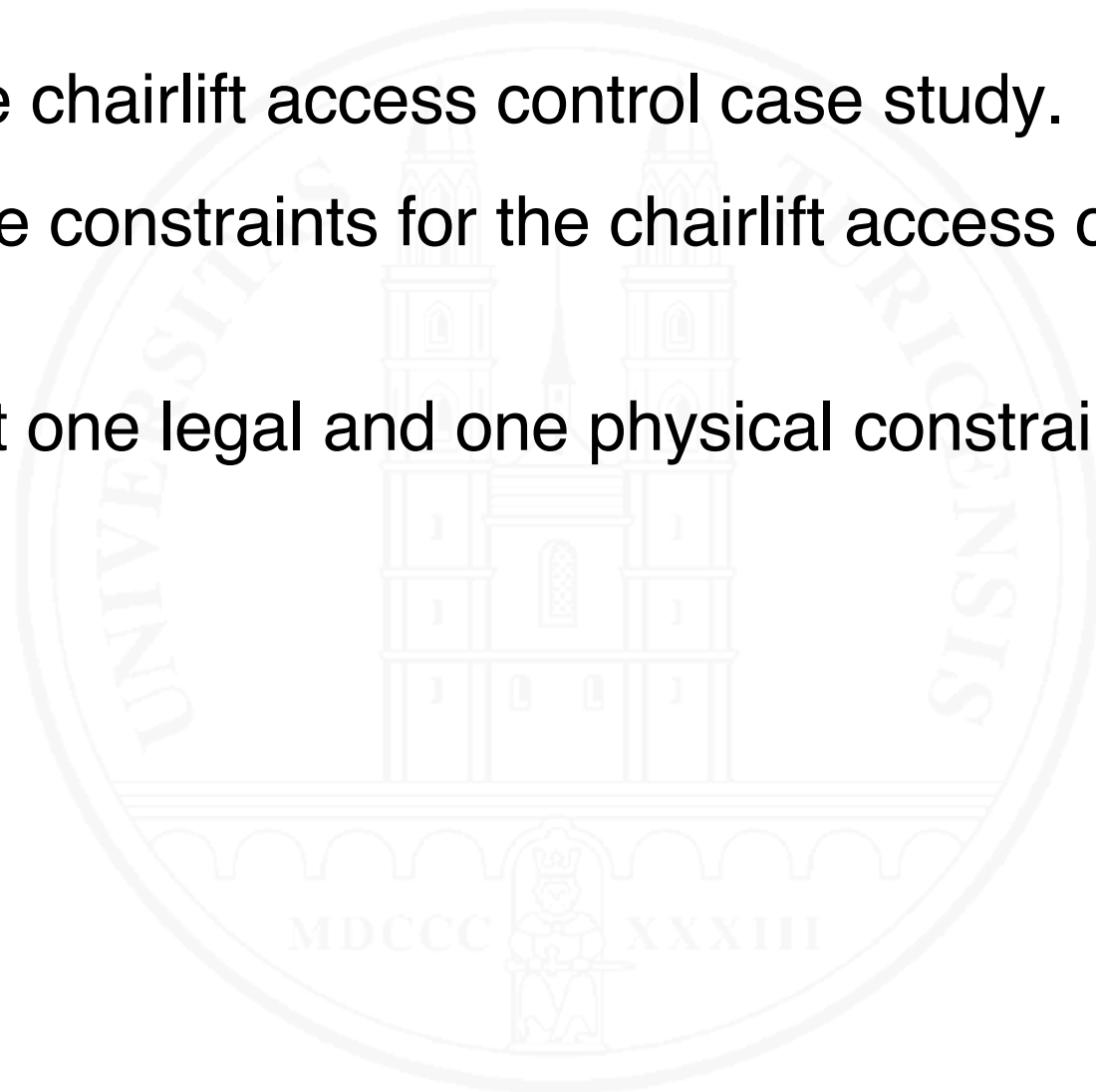
# Mini-Exercise

---

Consider the chairlift access control case study.

Identify some constraints for the chairlift access control system.

Elicit at least one legal and one physical constraint.



## 4.3 Analyzing and documenting

---

Elicited information is often **unstructured**

→ Analysis techniques help

- **shape collected information** into proper requirements
- **document** requirements properly
- **inform** and **structure** information **gathering**

# Analyzing elicited information

## Structure-oriented

Analyze terminology /  
domain properties  
Build glossary

Analyze business  
and data objects  
Build object and  
class models

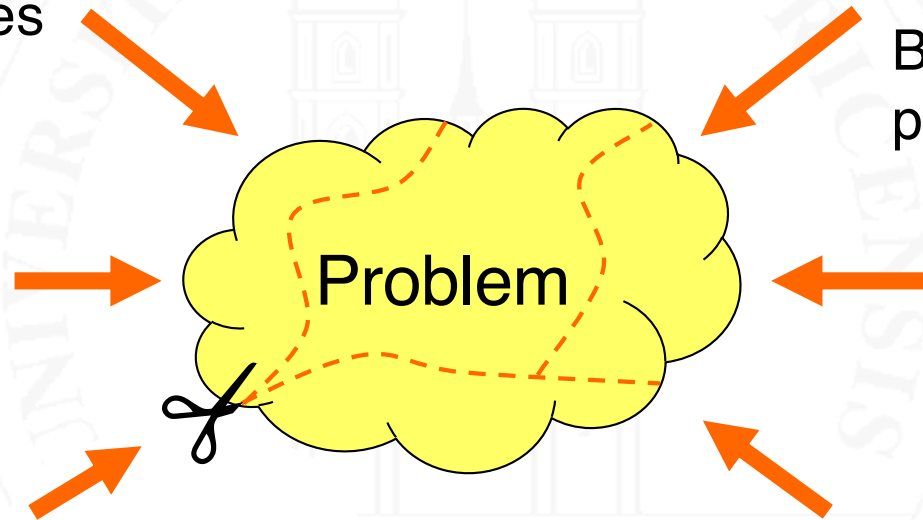
Decompose problem  
Build hierarchical structure

## Process-oriented

Analyze processes /  
workflows  
Build activity /  
process models

Analyze dynamic  
system behavior  
Build behavior  
model

Analyze actor-system interaction  
Build scenarios / use cases



Note: requirements are about a future state of affairs; analyze the current state only when necessary

# Documenting elicited requirements

---

Build specification **incrementally** and **continuously**

Document requirements in **small units**

**End over means**: Result → Function → Input

Consider the **unexpected**: specify non-normal cases

**Quantify** critical attributes

Document critical **assumptions explicitly**

Avoid **redundancy**

Build a **glossary** and stick to terminology defined in the glossary

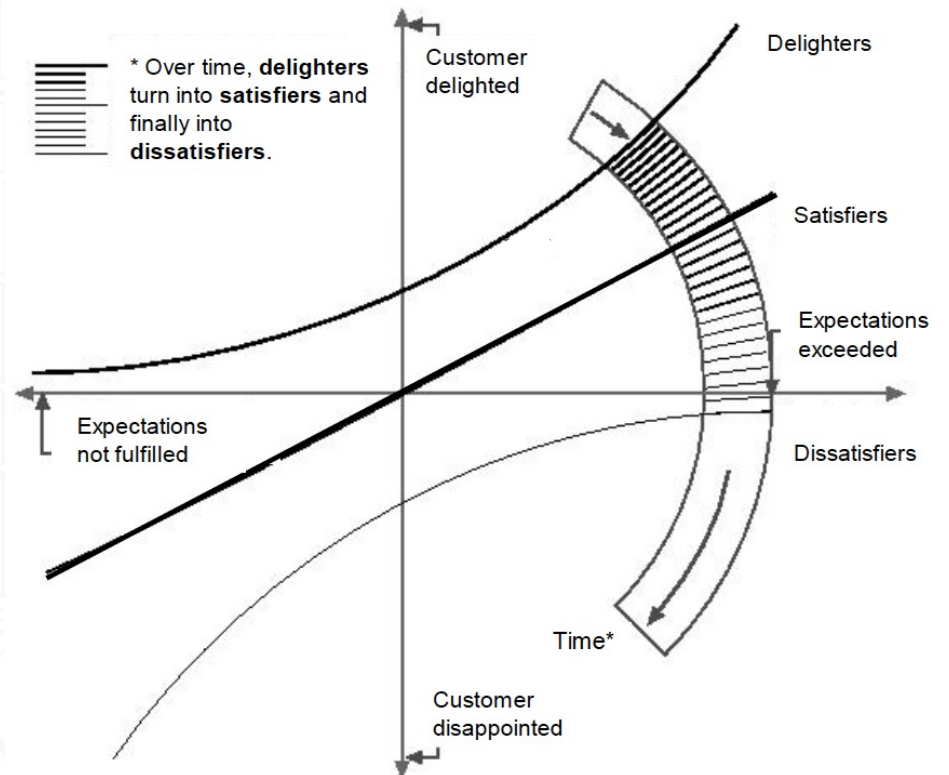
## 4.4 Eliciting innovative requirements

Satisfying stakeholders is not enough  
(see Principle 8 in Chapter 2)

Kano's model helps identify...

- what is **implicitly expected** (dissatisfiers)
- what is **explicitly required** (satisfiers)
- what the stakeholders don't know, but will **delight** them if they get it: **innovative requirements**

[Kano et al. 1984]



Caution: Over time, delighters **degrade** to plain expectations

# How to create innovative requirements?

---

Encourage **out-of-the-box thinking**

- Stimulate the stakeholders' **creativity**
  - Imagine/ make up scenarios for possible futures
  - Imagine a world without constraints and regulators
  - Find and explore metaphors
  - Study other domains
- **Involve solution experts** and **explore what's possible** with available and future technology
- Involve **smart people without domain knowledge**



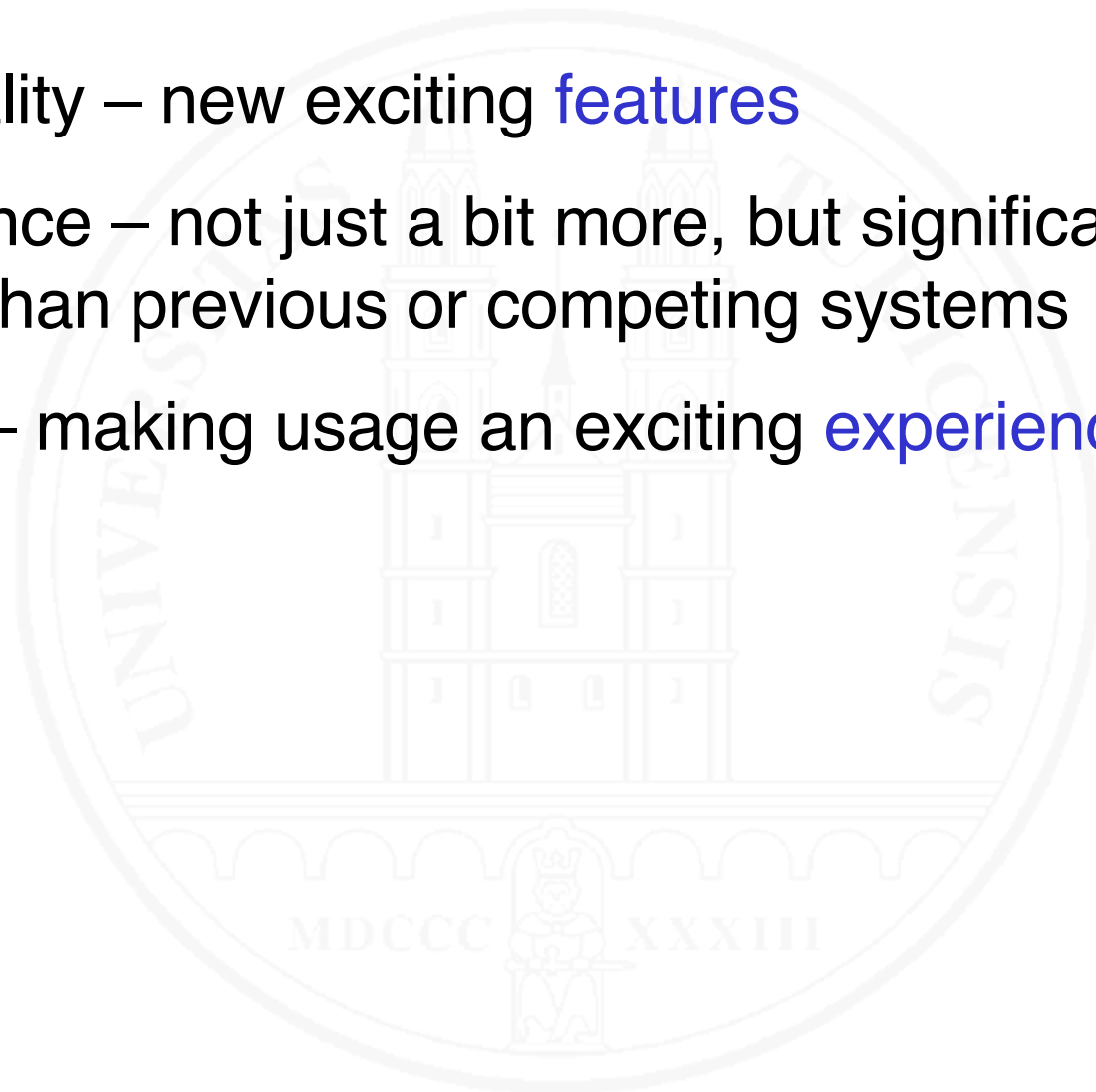
[Maiden, Gitzikis and Robertson 2004]

[Maiden and Robertson 2005]

# Where to innovate

---

- Functionality – new exciting **features**
- Performance – not just a bit more, but significantly **more powerful** than previous or competing systems
- Usability – making usage an exciting **experience**

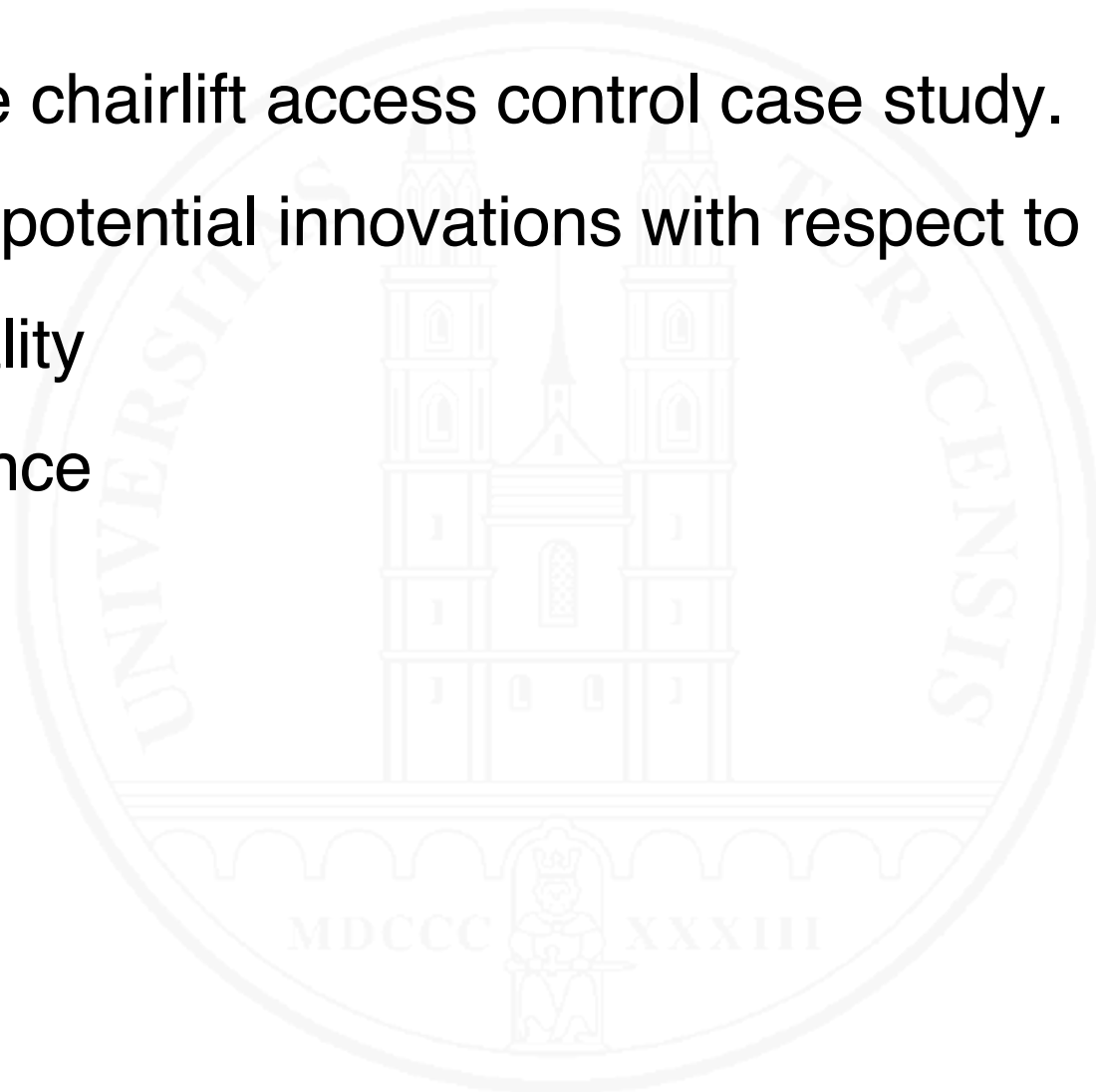


# Mini-Exercise

---

Consider the chairlift access control case study.  
Think about potential innovations with respect to

- **Functionality**
- **Performance**
- **Usability**





# 4.5 Requirements negotiation

---

Stakeholders may have conflicting requirements.

→ Negotiation of requirements needed

○ Requirements negotiation implies

- Identification of conflicts
- Conflict analysis
- Conflict resolution
- Documentation of resolution

○ Requirements negotiation can happen

- While eliciting requirements
- When validating requirements



# Conflict analysis

---

Identifying the underlying reasons of a conflict helps select appropriate resolution techniques

Typical underlying reasons are

- **Subject matter** conflict (divergent factual needs)
- **Data** conflict (different interpretation of data, inconsistent data)
- **Interest** conflict (divergent interests, e.g., cost vs. function)
- **Value** conflict (divergent values and preferences)
- **Relationship** conflict (emotional problems in personal relationships between stakeholders)
- **Organizational** conflict (between stakeholders on different hierarchy and decision power levels in an organization)

# Conflict resolution

---

- Various strategies / techniques
- Conflicting stakeholders must be involved in resolution
- Win-win techniques
  - Agreement
  - Compromise
  - Build variants
- Win-lose techniques
  - Overruling
  - Voting
  - Prioritizing stakeholders (important stakeholders override less important ones)

# Conflict resolution – 2

---

- Decision support techniques
  - PMI (Plus-Minus-Interesting) categorization of potential conflict resolution decisions
  - Decision matrix (Matrix with a row per interesting criterion and a column per potential resolution alternative. The cells contain relative weights which can be summarized per column and then compared)

# Mini-Exercise

---

Consider the chairlift access control case study.

How, for example, can you achieve consensus among the ski resort management, the technical director of chairlifts, the ticket office clerks, and the service employees about their requirements?

