

# Requirements Engineering I

## Chapter 12

# Conclusions

---



# 12.1 How much RE?

[Glinz 2016]

---

We **no longer believe** in big, unambiguous, and complete requirements specifications as the standard result of good Requirements Engineering.

- Although many standards and textbooks still do
- **Modern RE is value-driven**: the effort invested into RE is determined by the value that the requirements create
- Depends on domain and project context, driven by various factors, in particular
  - Shared understanding
  - Risk
  - Customer-supplier relationship

# 12.2 RE under time pressure

---

- Risk-oriented specification
  - The risk determines the needed effort, not the available time frame!
- Don't specify in uniform depth
  - Only the risky stuff in full detail
  - The rest coarsely or not at all
- Employ iterative processes
- Don't strive for perfection; good enough suffices



# What is indispensable?

---

- Know and involve the **critical stakeholders**
- Know the **problem**
- Identify the key **goals**
- Define the **key terms** (of the domain and the system) in a **glossary**
- Identify and document the system's **main functions and use cases**
- Identify and document critical **quality requirements, constraints and risks**
- Identify critical **domain assumptions and domain constraints**

# What makes it harder? (implies higher effort)

---

- High **complexity** of the domain
- Team is **not familiar** with the domain
- **Many stakeholders**
- **Distributed** development and/or stakeholders
- **Long feedback cycles**
- **Safety-critical** requirements
- High **project risks**

# Mini-Exercise



What do you reply to your boss?

## 12.3 RE topics not covered in this course

---

- Advanced techniques for
  - Requirements elicitation
  - Requirements modeling
  - Requirements management
- Advanced RE processes, e.g. RE for scaled agile projects
- RE for AI (machine learning and robotics systems)
- Crowd RE
- RE for sustainability

# 12.4 Requirements Engineering in a nutshell

---

- Stakeholders are key
- Validate your requirements early and frequently
- Work value-oriented:
  - Cost and benefit of requirements need to be in balance
  - Concentrate on the essential – don't just collect tons of detailed requirements
- Work risk-driven: the more risk, the more extensive and precise requirements specifications are necessary
- Intertwining of requirements and design is natural – you'll need to live with it



# Requirements Engineering in a nutshell – 2

---

- **Situate your system in its context**
  - Value is only created when using systems in their real world context – so you need to know this context
  - Elicit and document domain assumptions and constraints
- **No discovery:** Requirements must be elicited with serious endeavor, they can't be just discovered
- **Strive for innovation:** just automating what we have today is not enough
- **You are not the stakeholders' voice recorder** – elicit and *design* requirements that make stakeholders excited

# Requirements Engineering in a nutshell – 3

---

- **Control requirements evolution** – otherwise requirements evolution will control you
- **No universal language or method:** You'll need to use a variety of practices and languages
- **Specifying is not programming:** Skip all technical details which are not part of the problem
- **Finally: make it fun.** Nobody likes boring tasks. Make RE
  - *a fascinating expedition into the unknown,*
  - *to places where the desirable and the doable meet*
  - *and eventually merge into exciting new opportunities.*

# The final word

---

Follow the principles.

Practice the practices.

Be guided by the risk.

Strive for value.

**Requirements Engineering** – doing things right ...

...from the very beginning