Visit **gosocrative.com** and enter room name REQENG

Lecture starts at 13:15

You are welcome to share a short break with us and discuss/ask questions

L8 – Verification and Validation

DAT232/DIT285 Advanced Requirements Engineering

Eric Knauss
eric.knauss@cse.gu.se

October 7, 2025

Outline

1 Housekeeping

2 V&V: Motivation and Terminology

3 V&V Techniques

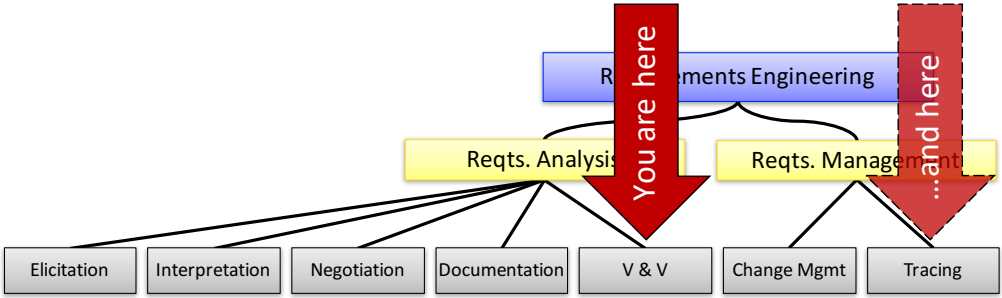
4 Wrapping up

Feedback from Socrative

7. Please name one thing that you liked and one thing that you wished for in this lecture.

Hide Answers | Show Names | 4/28 Students Answered

Maybe define what prioritation really is about and what techniques are there. Not just in the socrative
idk
more context on change management
1



Src: DaimlerChrysler, Dagstuhl-Seminar 1998

Learning Objectives

Knowledge	Skills	Judgement
K1 Identify a common RE challenge in a given software development context.	S1 Plan suitable RE practices in a team with respect to a given software development context.	J1 Assess new requirements engineering knowledge (challenge, principle, practice) and relate them to the framework in this course.
K2 Choose an appropriate RE practice in a given software development context.	S2 Effectively apply a suitable RE practice in a team in a given software development context.	J2 Suggest suitable actions to overcome a lack of requirements knowledge in a software development context.
K3 Compare suitability as well as advantages and disadvantages of given RE practices in a given software development context.	S3 Analyze the effect and quality of the outcome of a set of or individual RE practices in a given software development context.	J3 Consider inter-team, program level and social/ethical implications of a set of RE practices in a given software development context.
K4 Explain the current state of practice and research in requirements engineering.		J4 Critically assess the effectivity of a set of RE practices from the perspective of the student's master program (e.g. Software Engineering & Technology/Management, Interaction Design, Game Design, Data Science, ...)

Learning Objectives

Interpretation for ...Interpretation:

K1,K2,K3	Explain common challenges of quality assuring requirements, describe practices of checking the quality of reqts
K4	Be aware of current research challenges in requirements verification and validation
S1-S3	Apply this knowledge to your project
J1-J3	Look out for opportunities to get beyond course literature
J4	Reflect on how your background changes your view on verification and validation (automatic checks? usability tests?)

Specific Learning Objectives

After this lecture, you should be able to...

- Plan and execute the quality assurance of requirements as well as requirements documents
- Describe the difference between requirements verification and requirements validation
- Suggest and motivate relevant improvements on requirements quality assurance processes

Criteria 3/5 (Quality of Requirements)

Project assessment

ID	*Where checked*	*Criteria*	*F*	*3*	*4*	*5*
q.1	Review Report (process)	Quality assurance is planned and executed using various verification and validation techniques.				
q.2	Review Report (findings)	Assess the quality of requirements and find several relevant problems of several different types.				
q.3	Review Report (findings)	Requirements quality problems are prioritized and discussed, including deeper semantic issues.				
q.4	Review Report (prioritization)	Requirements quality problems are related to contextual risks affecting the project's success.				
q.5	Experience Report (reflection)	Reflects on experiences with quality assurance of requirements.				
q.6	Experience Report (reflection)	Provides critical discussion of quality assurance experiences.				
q.7	Experience Report (reflection)	Reflection connects verification and validation techniques, quality problems, and contextual risks to project success.				

Take-aways for your project

- Thanks for submitting R2. We have, after a quick check, distributed it to (an)other group(s)
- If you have received a specification for review, review it in the group
  1. Plan your review (create a check list based on this lecture and course book, short description of your process)
  2. Each member of your group reviews the other group's specification
  3. You meet and merge your changes into one list with criticality. Merge this list with the plan (Step 1).
  4. Send to author groups (if emails are available, otherwise to your supervisor) and upload to canvas.
- If you have not received a specification for review, please contact me.

Outline

1 Housekeeping

2 V&V: Motivation and Terminology

3 V&V Techniques

4 Wrapping up

Warning

- Lots of ambiguous terminology with conflicting definitions (e.g., validation, verification, test).
- Important to understand that different communities use terms differently.

→ `https://socrative.com`, REQENG, Q1

Software Project Failure Factors	Percentage of Projects (%)		
	In-house	Outsourced	Overall
Delivery date impacted the development process	93.9	90.5	92.9
Project under-estimated	83.7	76.2	81.4
Risks were not re-assessed, controlled, or managed through the project	73.4	80.9	75.7
Staff were not rewarded for working long hours	81.6	57.1	74.3
Delivery decision made without adequate requirements information	83.7	47.6	72.9
Staff had an unpleasant experience working on the project	83.7	47.6	72.9
Customers/Users not involved in making schedule estimates	69.4	76.2	71.4
Risk not incorporated into the project plan	65.3	80.9	70.0
Change control not monitored, nor dealt with effectively	63.3	85.7	70.0
Customer/Users had unrealistic expectations	69.4	66.7	68.6
Process did not have reviews at the end of each phase	75.5	47.6	67.1
Development Methodology was inappropriate for the project	71.4	52.4	65.7
Aggressive schedule affected team motivation	69.4	57.1	65.7
Scope changed during the project	67.3	57.1	64.3
Schedule had a negative effect on team member's life	71.4	42.9	62.9
Project had inadequate staff to meet the schedule	63.3	57.1	61.4
Staff added late to meet an aggressive schedule	61.2	61.9	61.4
Customers/Users did not make adequate time available for requirements gathering	61.2	57.1	60.0

Cerpa & Verner, 2009. “Why did your project fail?”, CACM 52 (12)

Here are 10 signs of IS project failure:[3]

1. Project managers don't understand users' needs.
2. The project's scope is ill-defined.
3. Project changes are managed poorly.
4. The chosen technology changes.
5. Business needs change.
6. Deadlines are unrealistic.
7. Users are resistant.
8. Sponsorship is lost.
9. The project lacks people with appropriate skills.
10. Managers ignore best practices and lessons learned.

Reel, 2002. "Critical success factors in software projects", IEEE SW 16 (3)

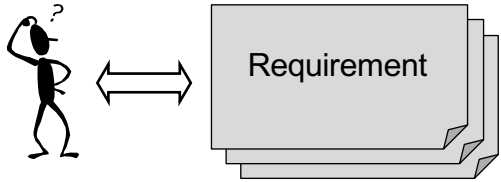
Quality Assurance of Requirements: Why and How?

Discuss!

What is requirements validation?

"The process of checking whether documented requirements match the stakeholders' needs."

– (IREB RE Glossary 1.4)



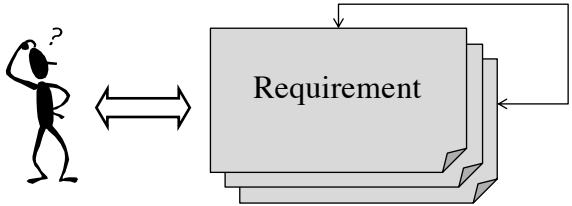
"Are we building the right system?"

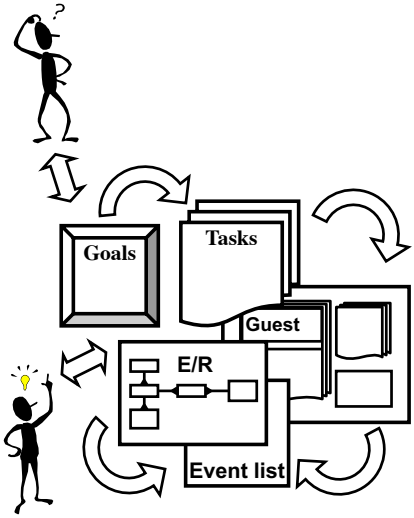
What is requirements verification?

Is it enough to check each requirement against a goal/need?
No! The specification as a whole also has to be understood properly → Quality criteria!

Example

Specification = ambiguous or inconsistent
→ large risk to leave stakeholder needs unfulfilled.





From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

- **Check that all parts match & everything is included**
- **Validate that stakeholders are happy (customer, user, developer)**
- **Where are the major risks?**
- **Quality product = meeting the spec?**

Difference between validation and verification?

Common source of disagreement!

Validation / verification of requirements

- *Validation:* Do the requirements fulfil the stakeholder needs/goals? ("Are we specifying the right requirements?")
- *Verification:* Were the requirements specified according to available information? ("Are we specifying the requirements right?")

Validation / verification of system or software product

- *Validation:* Does the system fulfil the stakeholder needs/goals? ("Are we building the right system?")
- *Verification:* Was the system implemented according to the requirements? ("Are we building the system right?")

V&V refers to Validation and Verification; Lauesen refers to requirements verification as requirements check [Lauesen, 2002].

What properties to check?

Classic: A good requirement spec is:

Correct
Each requirement reflects a need.

Complete
All necessary requirements included.

Unambiguous
All parties agree on meaning.

Consistent
All parts match, e.g. E/R and event list.

Ranked for importance and stability
Priority and expected changes per requirement.

Modifiable
Easy to change, maintaining consistency.

Verifiable
Possible to see whether requirement is met.

Traceable
To goals/purposes, to design/code.

Additional:

Traceable from goals to requirements.
Understandable by customer and developer.

Includes Atomicity
(if a req. is not atomic, you cannot clearly say whether it is met or not)

From: Soren Lauesen: Software Requirements © Pearson / Addison-Wesley 2002

Outline

1 Housekeeping

2 V&V: Motivation and Terminology

3 V&V Techniques

4 Wrapping up

V&V Techniques

- Inspections
  - Checklists
  - Ad-hoc
  - Perspective-based reading
- Tests
  - Usability tests
  - Prototypes
  - Simulation
- Mathematical/Formal proofs

L8 – Verification and Validation
E. Knauss
Housekeeping
V&V: Motivation and Terminology
V&V Techniques
Wrapping up

Inspections

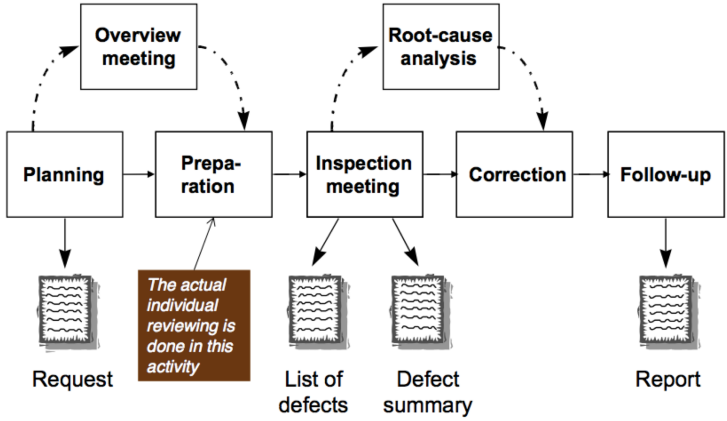
Fagan, 1976. “Design and Code Inspections to Reduce Errors in Program Development“, IBM Systems Journal 15 (3)

- A systematic method for manually reading through specifications (and other artefacts)
- Defined roles (e.g., moderator, designer)
- Aims: Defect detection, knowledge dissemination, process improvement, decision making, …

- Very 1970s (i.e., heavyweight)

The inspection process

Roles:
Moderator
Author
Reviewer
Secretary



L8 – Verification and Validation
E. Knauss
Housekeeping
V&V: Motivation and Terminology
V&V Techniques
Wrapping up

Inspections

- Different methods to guide the actual reading
  - Ad-hoc: No specific guidelines.
  - Checklists: Lists of questions/items to focus the reading.
  - Perspective-based reading: Read the specification from a specific perspective (role).
  - N-fold inspection: n different groups go through the inspection process in parallel

Break & Quiz

https://socrative.com

Room name: REQENG, Q2 - Q4

Fig 9.2A Contents check

From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

Does the spec contain:

- Customer, sponsor, background
- Business goals + evidence of tracing

- Data requirements (database, i/o formats, …)

- System boundaries & interfaces
- Domain-level, product-level, design-level reqs
- Specification of non-trivial functions
- Stress cases & special events & task failures

- Quality reqs (performance, usability, security . . .)

- Other deliverables (documentation, training . . .)
- Glossary (definition of domain terms . . .)

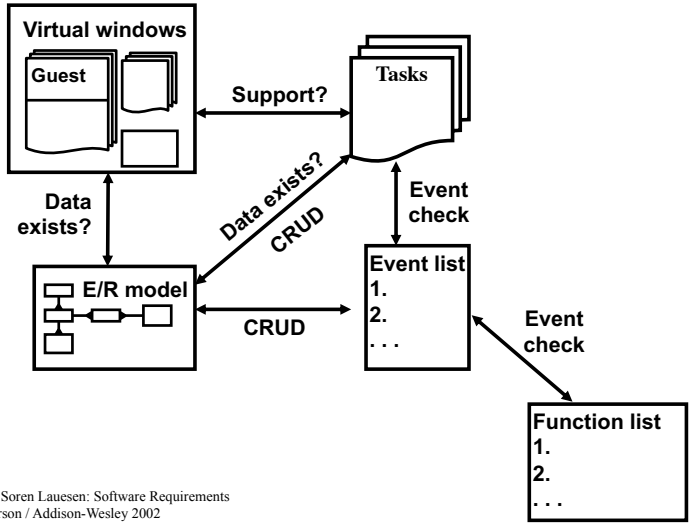
Fig 9.2B Structure check

Does the spec contain:

- Number or Id for each requirement
- Verifiable requirements
- Purpose of each requirement
- Examples of ways to meet requirement
- Plain-text explanation of diagrams, etc.
- Importance and stability for each requirement
- Cross refs rather than duplicate information
- Index
- An electronic version

From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

Fig 9.2C Consistency checks



From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

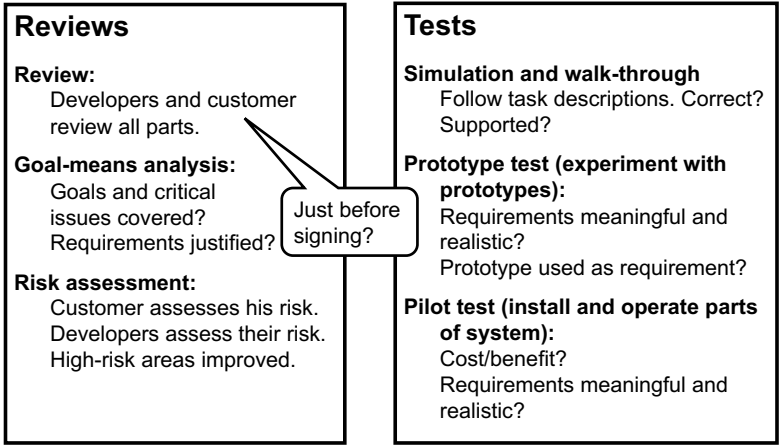
Fig 9.2D CRUD matrix

Create, Read, Update, Delete + Overview

Task \ Entity	Guest	Stay	Room	RoomState	Service	ServiceType
Book	C U O	C	O	U O		
CheckinBooked	RU	U O	O	U O		
CheckinNonbkd	C U O	C	O	U O		
Checkout	U	U O	R	U		
ChangeRoom	R	R	O	U O		
RecordService			O		C	R
PriceChange			C UDO			C UDO
Missing?	D	D		C?UD?	UD	

From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

Fig 9.3 Checks against surroundings



From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

Fig 9.4(A) Check list at work

Project:	Noise Source Location, NSL vers. X	**Date, who:** 99-03-15, JPV
Contents check	**Observations - found & missing**	**Problem?**
Customer & sponsor	Missing, OK	
. . .		
Data: Database contents	Class model as intermediate work product	
. . .		
Initial data & states	Missing	Seems innocent, but caused many problems particularly when screen windows were opened.
Functional reqs: Limits & interfaces		
Product-level events and functions	Mostly as features	
. . .		
Special cases: Stress cases		
Power failure, HW failure, config.	Missing	**Problem.** Front-end caused many problems

From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

Project:	Noise Source Location, NSL vers. X	**Date, who:** 99-03-15, JPV
Contents check (2)	**Observations - found & missing**	**Problem?**
Quality reqs: Performance	Missing, also in parts not shown here.	**Problem.** Response time became important.
Capacity, accuracy	Missing, also in parts not shown here.	**Problem.** Data volume, etc. became important.
Usability	Missing	Would have been useful
Interoperability	Missing	External dataformats, robot role, etc. caused problems
. . .		
Other deliverables: Documentation	Missing	Unimportant. Company standards exist.
. . .		

Structure check	**Observations - found & missing**	**Problem?**
ID for each req.	OK	
Purpose of each requirement	Good. Domain described.	

Consistency checks	**Observations - found & missing**	**Problem?**
CRUD check: Create, read, update, delete all data?	Have been made	

Tests	**Observations - found & missing**	**Problem?**
Prototype test	Not done, nor during development.	**Should have been done**. Caused many problems later.

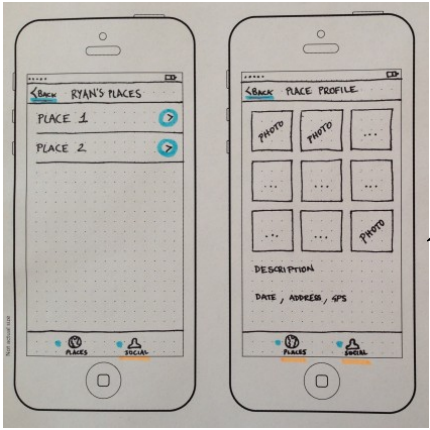
L8 – Verification and Validation
E. Knauss
Housekeeping
V&V: Motivation and Terminology
V&V Techniques
Wrapping up

Validation through tests

- Manual simulation based on scenarios/use cases/task descriptions
- Paper prototypes: mock-ups
- Executable prototypes (formal models)
- Pilot tests

L8 – Verification and Validation
E. Knauss
Housekeeping
V&V: Motivation and Terminology
V&V Techniques
Wrapping up

Paper prototypes: mock-ups



What do we require for R2/R3?

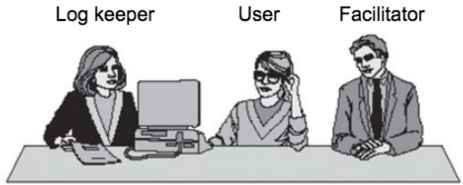
Source: http://www.raywenderlich.com

L8 – Verification and Validation
E. Knauss
Housekeeping
V&V: Motivation and Terminology
V&V Techniques
Wrapping up

Fig 6.6C Usability test & heuristic evaluation

Usability test
Realistic introduction
Realistic tasks

Note problems
- Observe only or
- Think aloud & ask



From: Soren Lauesen: Software Requirements
© Pearson / Addison-Wesley 2002

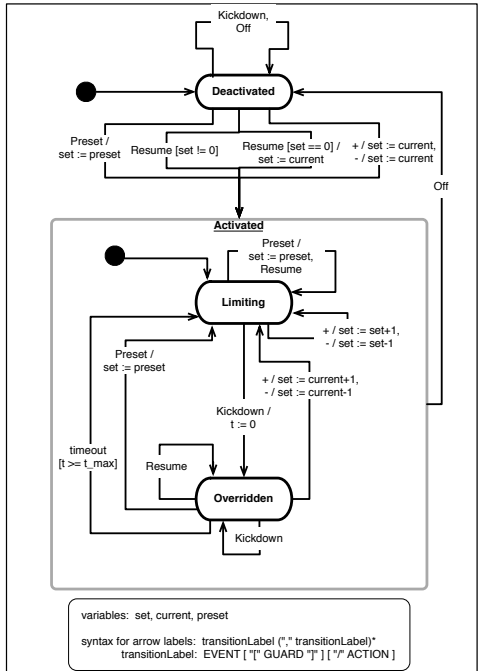
Executable prototypes (formal models)

- Require a formal language (e.g., timed automata, xtUML, Simulink)
- High training effort
- High creation effort (but benefits?)
- Can be used to detect inconsistencies
- Generation of test cases
- Use as test oracles
- Demonstration to stakeholders → early feedback
  - No real product available (yet)
  - V&V on real product expensive (e.g., an aeroplane)

Executable prototypes (Case I)

[Liebel et al., 2017]

- 3-year project with automotive OEM
- Sample specification (50 pages) of safety-critical function
  - Very low abstraction level!
- Similar to a car wiper
- Three different executable models
  - High complexity (13 automata, 45 states)
  - Medium complexity (8 automata, 15 states)
  - Low complexity (1 automaton, 3 states)


Kickdown, Off
Deactivated
Preset / set := preset
Resume [set != 0]
Resume [set == 0] / set := current
+ / set := current, - / set := current
Off
Activated
Preset / set := preset, Resume
Limiting
+ / set := set+1, - / set := set-1
Preset / set := preset
+ / set := current+1, - / set := current-1
Kickdown / t := 0
timeout [t >= t_max]
Resume
Overridden
Kickdown
variables: set, current, preset
syntax for arrow labels: transitionLabel (“,” transitionLabel)*
transitionLabel: EVENT [“[“ GUARD “]”] [“/” ACTION]

Executable prototypes (Case 1)

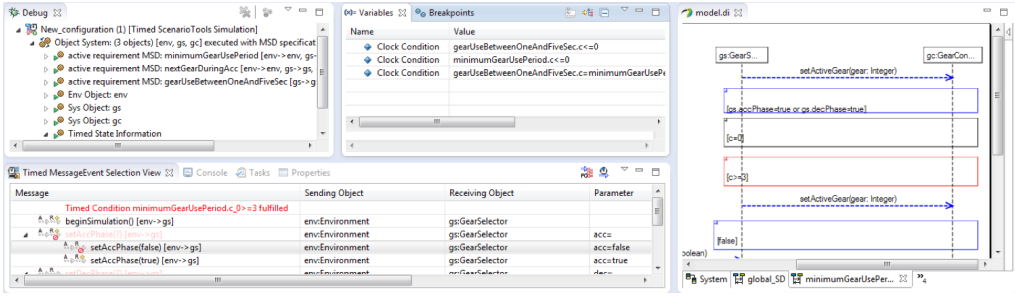
[Liebel et al., 2017]

- Several inconsistencies or errors found in the specification
  → Mainly while building the model!
- Domain experts judged low complexity automata model to be most useful (for validation).
- Low complexity model could be used for test case generation and for simulation/demonstration.

Executable prototypes (Case II)

[Brenner et al., 2014]

- Executable Sequence Diagrams from Specification (Emission Testing Standard)
- Each (behaviour) requirement = 1 diagram
- Whole model can be executed
  - Validation (demonstration, simulation)
  - Verification (does the system behave as specified?)



Post lecture quiz

https://socrative.com

Room name: REQENG, Q5 to Q8

After L8, you can answer:

(Should be able to...)

1. What is requirements validation?
2. What is the difference to requirements verification?
3. What are consequences of poor requirements validation?
4. What properties should be checked during requirements verification vs. during requirements validation?
5. What requirements verification/validation techniques exist?

Outline

1 Housekeeping

2 V&V: Motivation and Terminology

3 V&V Techniques

4 Wrapping up

Todo

- Thanks for submitting R2. We have, after a quick check, distributed it to (an)other group(s)
- If you have received a specification for review, review it in the group
  1. Plan your review (create a check list based on this lecture and course book, short description of your process)
  2. Each of you reviews the document
  3. You meet and merge your changes into one list with criticality. Merge this list with the plan (Step 1).
  4. Send to author groups (if emails are available, otherwise to your supervisor) and upload to canvas by Friday, 8am.
- If you have not received a specification for review, please contact me.
- Student representative next week Thursday, 15:15
- Next week RE Agile and Scale (L9) as well as RE Ethics (L10) (may switch order)

References I

Brenner, C., Greenyer, J., Holtmann, J., Liebel, G., Stieglbauer, G., and Tichy, M. (2014).
Scenariotools real-time play-out for test sequence validation in an automotive case study.
In *Proc. of the 13th International Workshop on Graph Transformation and Visual Modeling Techniques (GTVMT 2014)*.

Lauesen, S. (2002).
Software Requirements.
Pearson / Addison-Wesley.
the course book (Lau).

Liebel, G., Anjorin, A., Knauss, E., Lorber, F., and Tichy, M. (2017).
Modelling behavioural requirements and alignment with verification in the embedded industry.
In *Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development*, pages 427–434, Porto, Portugal.