T-303-HUGB: Software Engineering

L01: Introduction

Grischa Liebel

This Lecture

- Why Software Engineering?
- (Who am I and why do my slides look weird?)
- Organisation/course overview

Software Engineering

The Role of Software

2009:

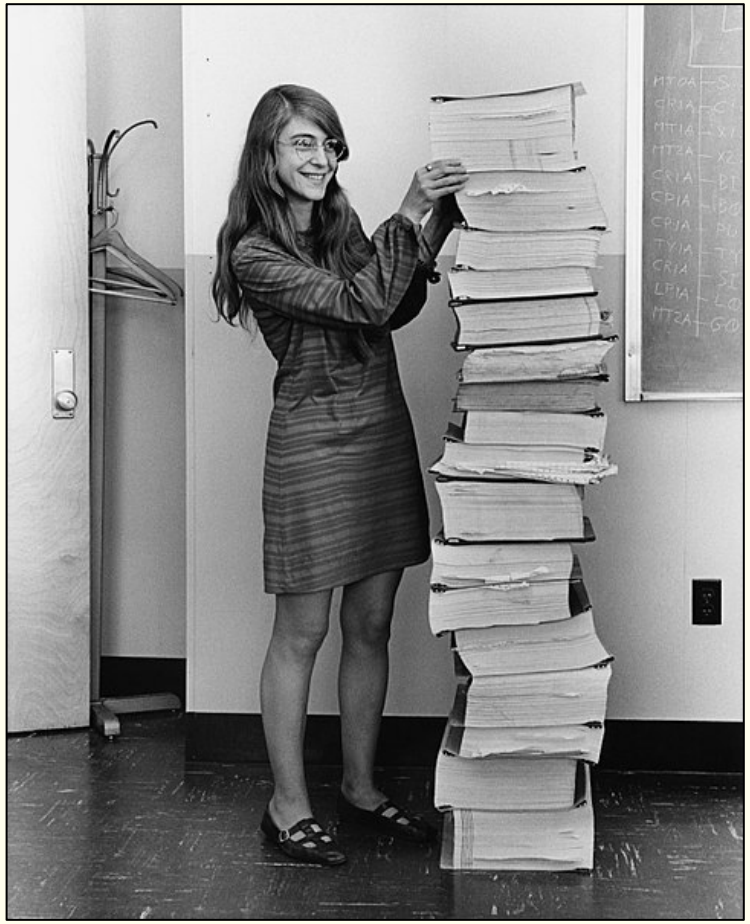
Rank	First quarter [56]	
1	Exxon Mobil	▼336,527
2	PetroChina	▲287,185
3	Wal-Mart	▼204,365
4	ICBC	▲187,885
5	China Mobile	▼174,673
6	Microsoft	▼163,320
7	AT&T	▼148,511
8	Johnson & Johnson	▼145,481
9	Royal Dutch Shell	▼138,999
10	Procter & Gamble	▼138,013

2019:

Rank	First quarter	
1	Microsoft	▲904,860[10]
2	Apple Inc.	▲895,670[11]
3	Amazon.com	▲874,710[12]
4	Alphabet Inc.	▲818,160[13]
5	Berkshire Hathaway	▼493,750[14]
6	Facebook	▲475,730[15]
7	Alibaba Group	▲472,940[16]
8	Tencent	▲440,980[17]
9	Johnson & Johnson	▲372,230[18]
10	ExxonMobil	▲342,170[19]

The Now and Then of Software

- Software gets larger!
  - Apollo 11: 145.000 lines of code (LoC)
  - Google Chrome: 6.7M LoC
  - F-35 fighter jet: 24M LoC
  - Facebook: 62M LoC
  - Modern high-end car: 100M LoC



The Now and Then of Software

- Increasingly distributed and heterogenous
- Changing businesses and society
- Need for security and trust

Software is expensive

Creation effort in person years (estimates):

- Typical ERP system (sales, service, logistics): 100 PY
- Windows 2000: >6000 PY

Software is expensive

Creation effort in person years (estimates):

- Typical ERP system (sales, service, logistics): 100 PY
- Windows 2000: >6000 PY

Costs:

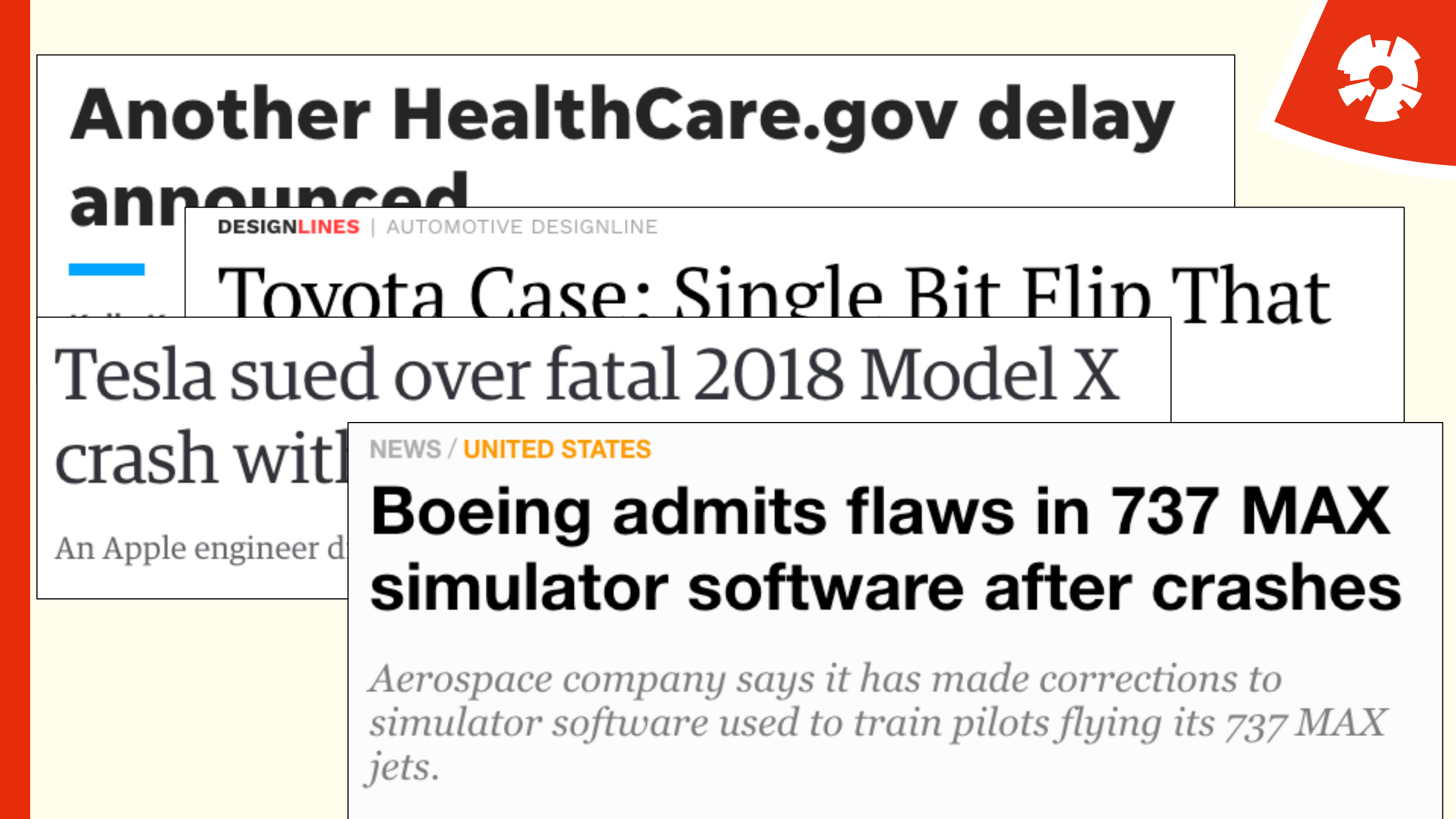
- ERP System: 5-8M €
- Windows 2000: 300-480M €

- Comparison: Harpa cost approx. 165M €

The Now and Then of Software

- Societal impact through software
  - "Cyber-physical systems"
    (transportation, electricity grid, elderly care)
  - "Digitalisation"
    (disruptive business models, consumption patterns)

- Failures have drastic consequences...


Another HealthCare.gov delay announced
DESIGNLINES | AUTOMOTIVE DESIGNLINE
Toyota Case: Single Bit Flip That
Tesla sued over fatal 2018 Model X crash wit
An Apple engineer d
NEWS / UNITED STATES
Boeing admits flaws in 737 MAX simulator software after crashes
Aerospace company says it has made corrections to simulator software used to train pilots flying its 737 MAX jets.

Some software is never completed

- CHAOS report by the Standish group*
  - "31.1% of projects will be cancelled before they ever get completed" (in the US)
  - "52.7% of projects will cost 189% of their original estimates"

*: The methodology has been heavily criticized:
"How large are software cost overruns? A review of the 1994 CHAOS report" (2006), M Jørgensen, K Moløkken-Østvold, Information and Software Technology 48 (4), 297-301)

"Software Engineering"

- 1968 NATO conference on Software Engineering
  - Provocative term
  - Reaction to increasing delays and quality issues

"The application of a **systematic, disciplined, quantifiable** approach to the development, operation, and maintenance of software", IEEE Standard Glossary of Software Engineering Terminology

Engineering Analogy

- Typical analogy: Building houses/bridges/cars/etc.
- Costs and risks can be calculated
- Results are as expected (and in schedule)
- Quality is high

- **But:** Software does not have physical properties

Why do software projects fail?

Canvas Course Page
→ Quizzes
→ Lecture 1: Introduction Quiz

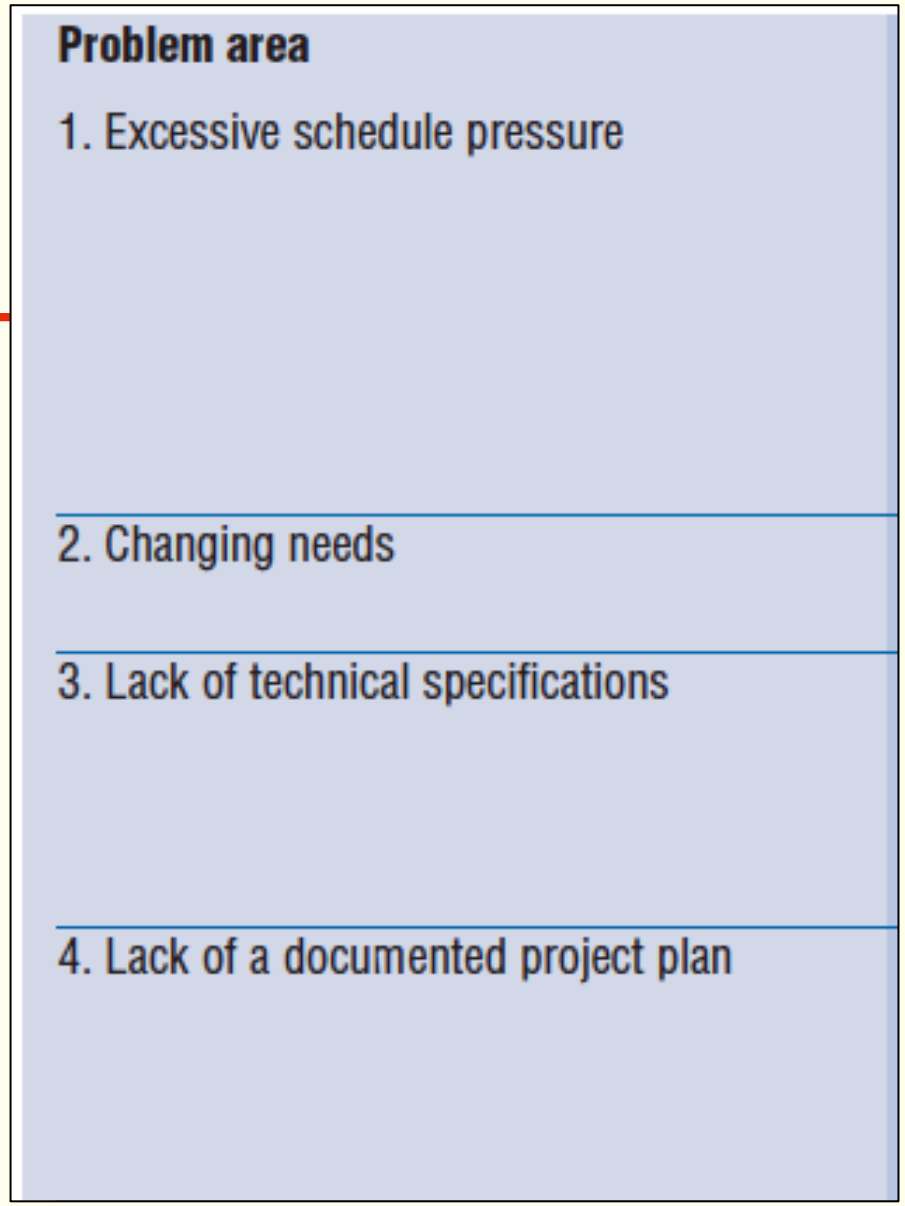
SCAN ME

Why the Vasa Sank

- Almost 400 years ago, the Vasa ship sank in Stockholm harbour after travelling 1300 meters.
- Many reasons
  - Not: "Bad carpenters", "Bad sailmakers", ...
  - Instead: A long list of management failures

Why the Vasa Sank

- Almost 400 years ago, the Vasa ship sank in Stockholm harbour after travelling 1300 meters.
- Many reasons
  - Not: "Bad carpenters", "Bad sailmakers", ...
  - Instead: A long list of management failures
- Software projects today seem to be similar



Fairley, Willshire, 2003. "Why the Vasa sank: 10 problems and some antidotes for software projects", IEEE SW 20 (2)

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)

Software is diverse

- Not every approach works for every system
  - Stand-alone applications
  - Embedded control systems
  - Entertainment/Infotainment systems
  - Data collection systems
  - Systems of systems
- Different customers and contract types

"Software Engineering"

"The application of a **systematic, disciplined, quantifiable** approach to the development, operation, and maintenance of software", IEEE Standard Glossary of Software Engineering Terminology

"Software Engineering"

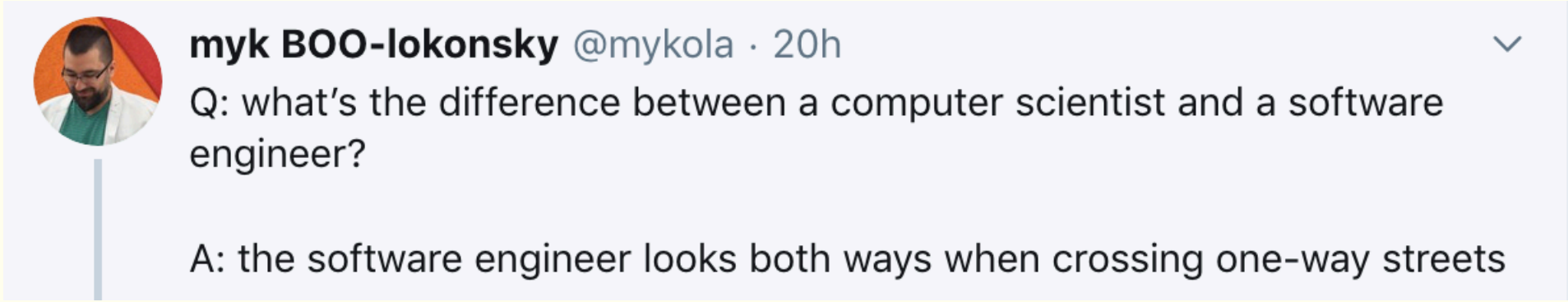
"The application of a **systematic, disciplined, quantifiable** approach to the development, operation, and maintenance of software", IEEE Standard Glossary of Software Engineering Terminology

Meaning: A whole lot about practices, management, activities.
There is some programming, too.

"Software Engineering"

"The application of a **systematic, disciplined, quantifiable** approach to the development, operation, and maintenance of software", IEEE Standard Glossary of Software Engineering Terminology

Meaning: A whole lot about practices, management, activities.
There is some programming, too.



Who am I?

About me

- PhD in Software Engineering
- I do research on Software Engineering
  - Applied, with companies
  - Mainly processes, requirements, modelling
- I sometimes consult
  - E.g., government tenders

I also teach Web Programming I

- No knowledge from VEFF required
- We use something called "WebSockets"
  - but that's nothing covered in VEFF

- Course more theoretical
- Not everything on the slides is covered in the project
- The project is not sufficient to do well on the exam

I also teach Web Programming I

- Single project, limited grading scale
  - Much more attention to individual contribution!

- Some people like VEFF and not HUGB, and vice versa

- HUGB is a whole lot easier

  (Prerequisites: PROG and GHOH)

Why are my slides so weird?

- More accessible to neurodiverse people
  - ADHD, Dyslexia, Autism Spectrum

- We provide all slides in this style
  - … and project descriptions in two styles

- We will follow up on this

Organisation

COVID-19

- There is currently little/no information
- We have done this course remote and in person

- Currently, not enough space in V201 (1m rule)
- For now, we stay remote. I'll keep you posted.

Maybe move Twitch and Echo360 to Youtube as well

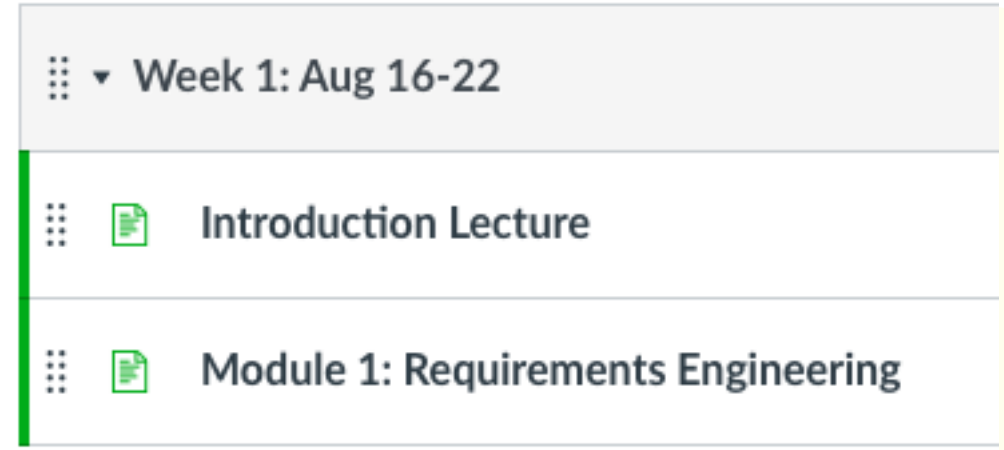
Overview: Websites

- Canvas for overview and information
- Piazza for discussions/questions (Linked in Canvas)
- Echo360 for live recordings (Linked in Canvas)
- Youtube for pre-recorded videos (Linked in Canvas)
- Twitch for live streams (Linked in Canvas)
- GitLab for project work (https://gitlab.com)

  Note: Not GitHub

Canvas

- Course overview
- Individual module material
  - Modules → weekly topics, required reading, links to videos, slides
- Grading
- Individual submissions (more later)
- Project survey (more later)



Piazza

- Heavy use of Piazza
  - You are expected to be active regularly
  - Important notifications
  - **Clarify to each other**
  - What is on Piazza counts!
- Avoid private messages to me if it concerns everybody

Echo360

- Live streams are recorded
  - And will be uploaded through Echo360
- Module material is pre-recorded
- Sometimes smaller videos on Echo360 for clarifications/detailed explanations

Twitch

- For now: live streams on Twitch (https://twitch.tv/grischal)
- To comment/chat: Verified email needed
  - To avoid spam
  - Write "question" (or similar) before formulating your question

(Streams will be recorded and uploaded on Echo360)

Modules

- One topic per week (= one "module")
- Pre-recorded videos (roughly 1.5 hours/week)
- Discussed in the next live stream (1 week delayed)

- Example:
  - This week, you watch **Requirements Engineering**
  - Next Monday, we discuss **Requirements Engineering**

Foundations and recap:	M1: Requirements Eng.	([Sommerville] Ch.4)
	M3: Modelling	([Sommerville] Ch.5)
Processes:	M2: Processes	([Sommerville] Ch.2, 3)
Testing:	M4: Testing	([Sommerville] Ch.8)
Architecture & Design:	M5: Architecture	([Sommerville] Ch.6)
	M6: Design	([Sommerville] Ch.7)
	M7: Security	([Sommerville] Ch.14)
Human factors:	M8: Human factors	(Research literature)

Extended/current topics: M9 & M10 (and buffers)

E.g.: Global SE, Model-based Engineering, Measurement, Empirical SE, Business models/Open source, DevOps

Learning Outcomes (LOs): Knowledge and comprehension

LOs in the slide sets!

1. Contrast software engineering techniques required for different types of software systems.
2. Discuss ethical issues arising in the context of modern software engineering projects.
3. Explain what software engineering is and why it is needed.
4. Illustrate the term stakeholder in relation to different types of software systems.
5. Summarise different techniques for performing requirements validation.
6. Discuss how system modeling can be used in different ways to address the needs of modern software systems.
7. Discuss the need for systematic processes in software engineering.
8. Compare plan-driven and agile processes in relation to different types of software systems.
9. Explain several common agile practices.
10. Discuss the issues of applying agile processes in large-scale and regulated environments.
11. Explain the different stages and scopes of testing.
12. Discuss different testing coverage criteria.
13. Discuss how architectural decisions can affect different system qualities.
14. Illustrate key architectural patterns.
15. Explain key design patterns of object-oriented design.
16. Contrast security and safety in the context of software systems.
17. Summarise design guidelines to achieve security in software systems.
18. Illustrate the key ideas of model-based engineering.
19. Summarise recent trends in software engineering.

LOs: Application and analysis

1. Classify different kinds of requirements needed in software engineering.
2. Apply system modeling to provide an overview of a software system.
3. Demonstrate understanding of different parts of the Scrum process.
4. Conduct unit and system testing in a test-first matter.
5. Make use of architectural styles/patterns to create a basic system architecture.

LOs: Synthesis and evaluation

1. Formulate functional and quality requirements using different techniques.
2. Adapt a process to the specific needs of a software system.
3. Examine the role of human factors in the development of software systems.

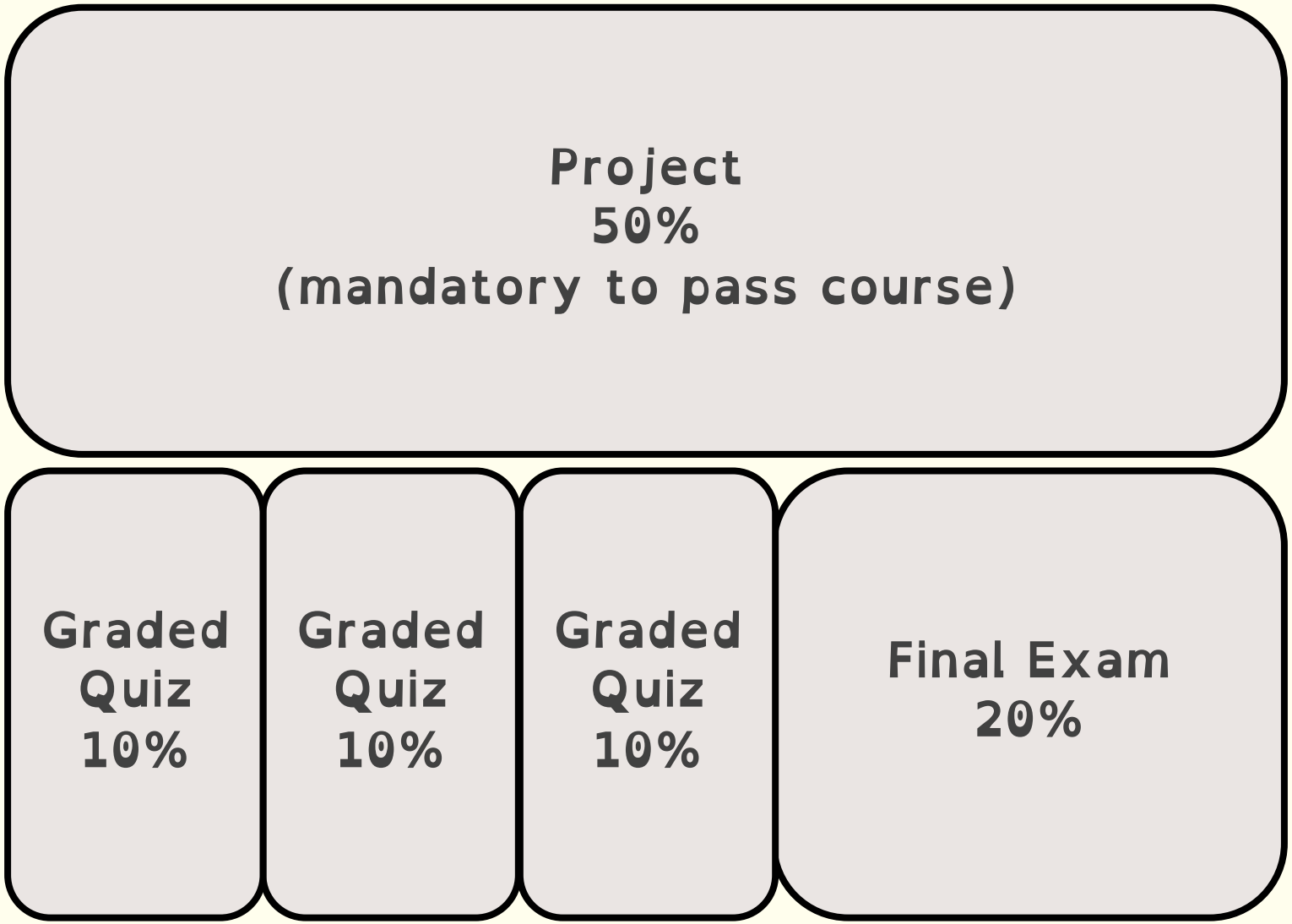
Literature

- Ian Sommerville, "Software Engineering", 9th Edition (Addison-Wesley) (**[Sommerville]**)
  - Older and newer editions work as well (maybe not <8th)
  - When I refer to chapter numbers, it's for 9th edition
- Additionally some articles
  - Marked in modules what is **optional!**

Other Resources

- This course borrows from two other courses
  - Softwaretechnik I
    (Ulm University, Germany, Matthias Tichy)
  - Foundations of Software Engineering
    (CMU 2018/19, Christian Kästner and Michael Hilton,
    https://www.cs.cmu.edu/~ckaestne/17313/2018/)

Grade Overview



Final Exam (20%)

- Date: between 8th - 19th November
- Re-exam: 4th - 7th January 2022

- All lecture content + required readings
- Focused on free-text answers
- Required to pass the exam
- Not decided whether in person or home exam

Graded quizzes (3x10%)

- Date: In course week 3, 7, and 10
- Covering 3 modules each
- Similar question style as final exam
- Option to repeat once

- Detailed dates/explanations coming

Project Assignment (mandatory)

- One single project assignment
- Spans the entire course
  (starting Tue, 24th August)
- For now:
  - Make sure you have time during your assigned slot
    (H2: Tue 14:20, H1: Wed 10:10, HMV: Thu 16:50)
  - If not, change section (mail **td@ru.is**)
  - Watch project introduction video on Echo360
  - Fill in survey on Canvas (**required for all students**)

How much time?

- Reading: 1 hr/week
- Watching recordings: 1.5hrs/week
- Live stream: 1.5 hrs/week
- Labs/Dæmatímar: 1.5 hrs/week
- Group work (in addition to Dæmatímar): 5 hrs/week

- Keep track of your hours!

Communication and Rules

- Public posts on Piazza are preferable
- Private posts if private issues
- Avoid email as much as possible
- If project-related: First talk to TA, then to Shalini (see project intro), then to me

- Stay professional in your tone

Plagiarism

- Plagiarism of any form is not accepted
- Directly copying material without referencing is plagiarism
- Sharing your solution for others to copy is as well
- This does not mean you're not allowed to talk to each other!
- If in doubt, read the RU rules on studying & assessment

ToDo

- Watch Project Introduction Video
- Fill in Project Survey
  (until Wednesday, 23:59)
- Watch Requirements Engineering videos

Next Topic

- Requirements Engineering
  - Recap from T-216-GHOH + additions
  - Literature: [Sommerville] Ch. 4

- https://www.youtube.com/playlist?list=PLCTWqbu-D5bPpTPZgTjRONyzx38pgZ8bP
- Slight echo/reverb in this module


HÁSKÓLINN Í REYKJAVÍK
REYKJAVÍK UNIVERSITY

Sources

Maps screenshots: Google
Wow: Designed by winkimages / Freepik
Education: Designed by Freepik
Exam: Designed by Freepik
Tools: Designed by Terdpongvector / Freepik
Confusion: Designed by Freepik
Vasa: CC BY-SA 3.0, JavierKohen

Assignment icons: Designed by ibrandify / Freepik
Todo: Designed by Makyzz / Freepik
Megaphones: Designed by Freepik