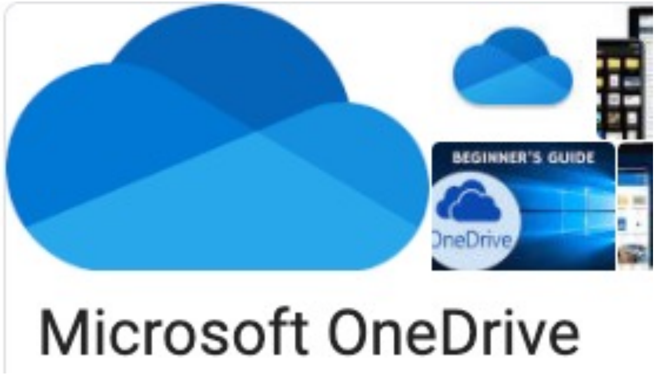
What did we learn from last week?

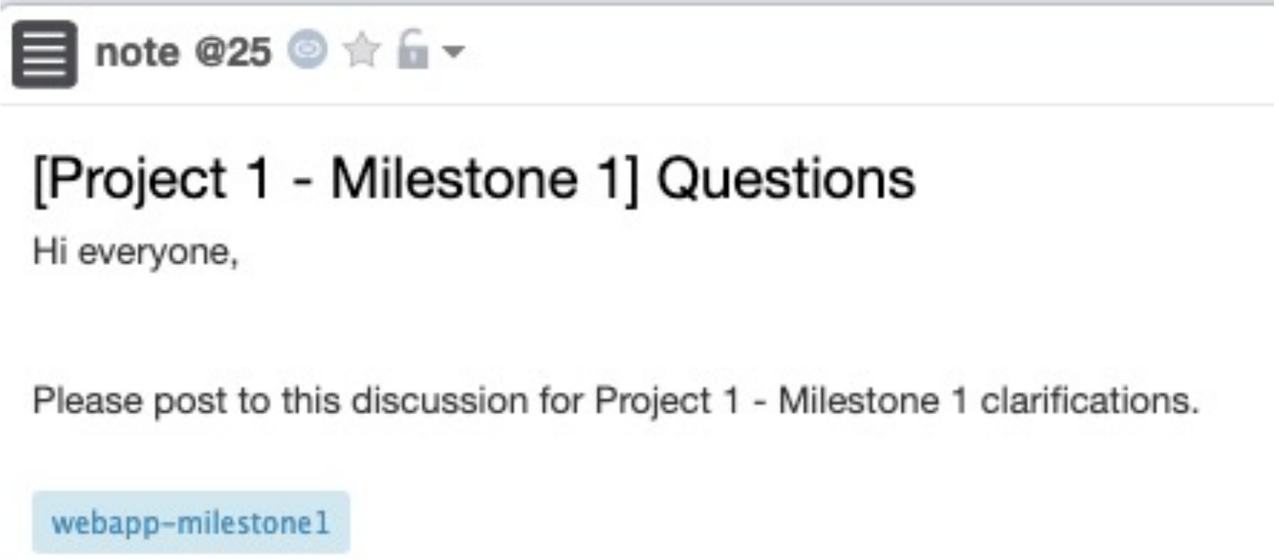
Learning Goals for last lecture

- Introduction of Software Engineering
- Process and Team
  - Recognize the Importance of process
  - Understand the difficulty of measuring progress

Administrivia

- Milestone 1: sharing your OneDrive Folder
- Posting questions on Piazza: One topic ONE thread.



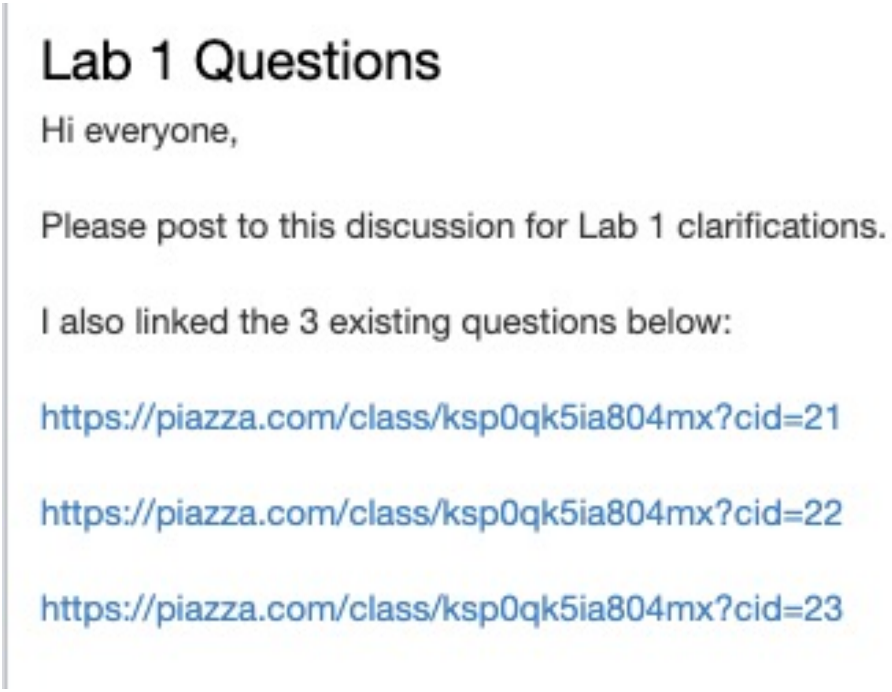


[Project 1 - Milestone 1] Questions

Hi everyone,

Please post to this discussion for Project 1 - Milestone 1 clarifications.

webapp-milestone1



Lab 1 Questions

Hi everyone,

Please post to this discussion for Lab 1 clarifications.

I also linked the 3 existing questions below:

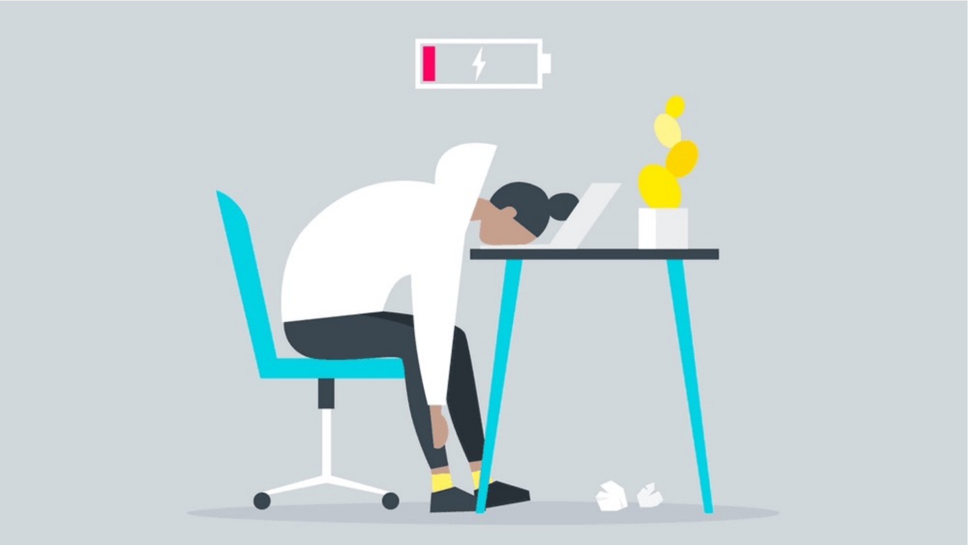
https://piazza.com/class/ksp0qk5ia804mx?cid=21

https://piazza.com/class/ksp0qk5ia804mx?cid=22

https://piazza.com/class/ksp0qk5ia804mx?cid=23

Changes in teams

- Be prepared for changes -- members/roles/responsibilities. Record the changes.
- Ask for accommodation



Announcing CARTE student initiatives:

- 1- **AI/ML research drop-in clinic:** Students can book a free 30-minute slot through here with our research associate, Alex Olson, who would provide guidance on experimental design, literature, and technical tools.
- 2- **Analytics/AI/ML Student CV Bank:** Master of Engineering students looking for M.Eng projects can send their CVs to CARTE's assistant director, Somayeh Sadat, to be added to the CV bank. This CV bank is accessible to CARTE faculty affiliates who might use it to find students for their research projects. MASc and PhD students looking for additional research projects can also send us their CVs, provided that they first ask for permission from their supervisors. Undergraduate students looking for summer projects are welcome to submit their CVs too.
- 3- **MITACS Internship opportunities in Analytics/AI/ML:** We announce new internship opportunities on a bi-weekly basis on our website and through our email list.
- 4- **Information on access to cloud computing resources**: We've compiled relevant information for students here.
- To stay informed about our other services, including industry speaker seminars, conference de-brief sessions, and other research or employment opportunities, please make sure that you subscribe to our email list. You can also follow us on social media (Linkedin, Facebook, or Twitter) for announcements of relevant events across the university.

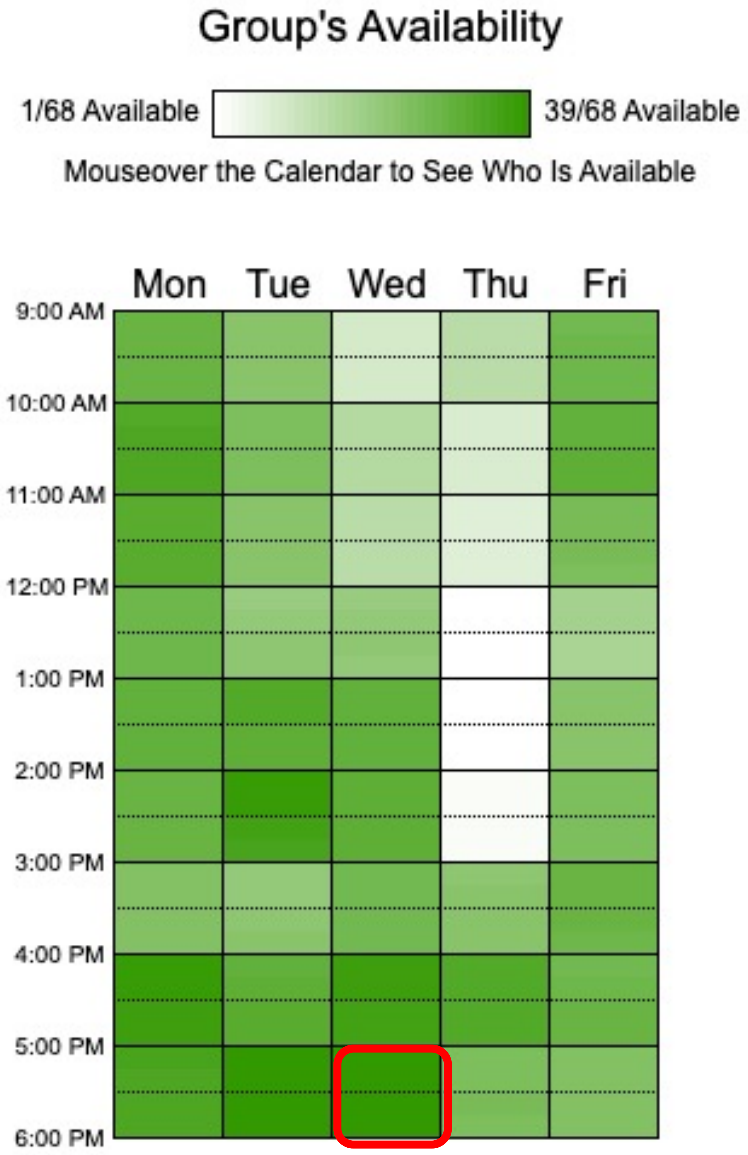
SURVEY
RESULTS!

Logistics -- Office Hours

Wed 5:15-6:15pm

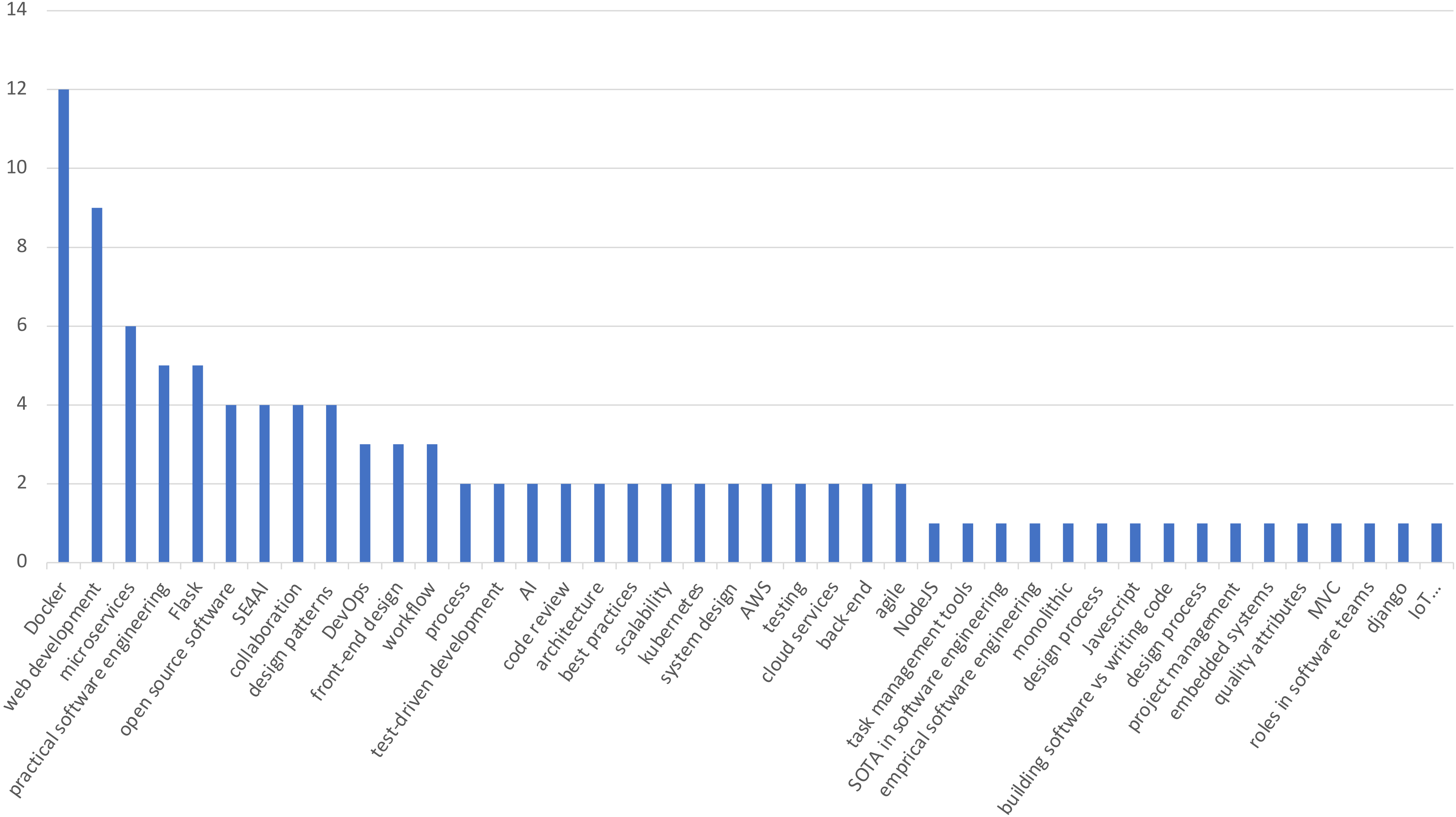
Zoom

Others by appointment



Why did you pick this class?

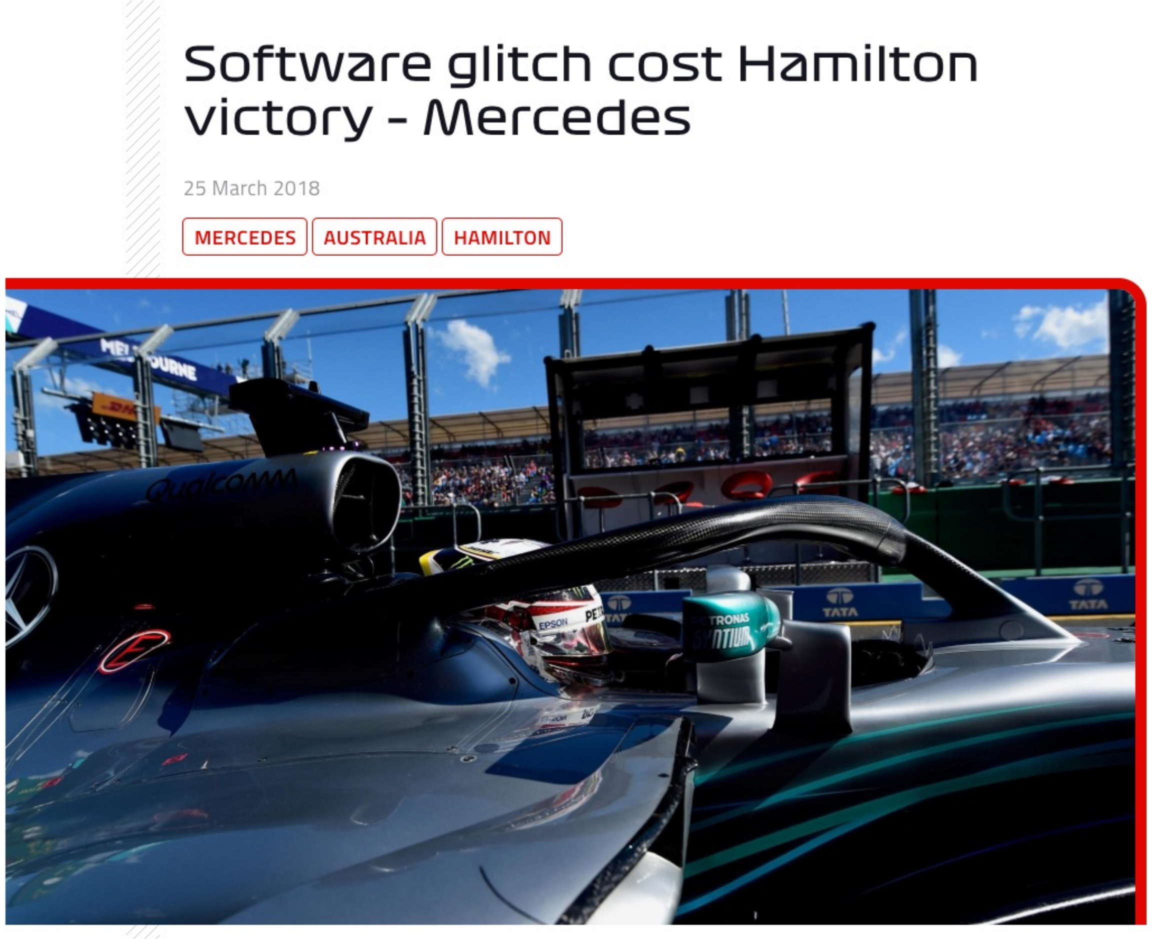
- I am interested in pursuing a career as a software engineer.
- To learn more about software development in real-world application
- To learn more about agile workflow and new skills for fulltime employment
- Learn about best practices in software engineering and web development
- To better understand how to work in larger groups to create and maintain large projects.
- How to build reliable software updated and maintained with convenience.
- Formally learn about design patterns
- Learn about the workflow and architecture of designing a web application
- I love software engineering!
- Interested to learn more about the processes of creating software applications beyond the technical aspects of them.


14
12
10
8
6
4
2
0
Docker
web development
microservices
practical software engineering
Flask
open source software
SE4AI
collaboration
design patterns
DevOps
front-end design
workflow
process
test-driven development
AI
code review
architecture
best practices
scalability
kubernetes
system design
AWS
testing
cloud services
back-end
agile
NodeJS
task management tools
SOTA in software engineering
emprical software engineering
monolithic
design process
Javascript
building software vs writing code
design process
project management
embedded systems
quality attributes
MVC
roles in software teams
django
IoT…

Online lab only for...

- Wednesday 22-Sep
- Wednesday 29-Sep

Software is everywhere

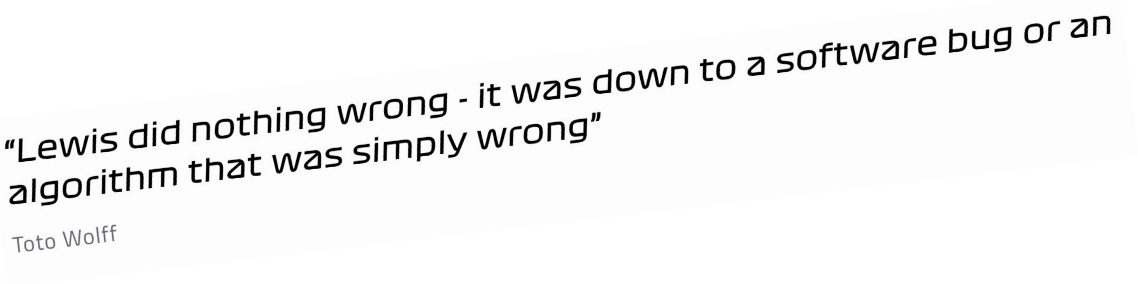


Software glitch cost Hamilton victory - Mercedes

25 March 2018

MERCEDES AUSTRALIA HAMILTON

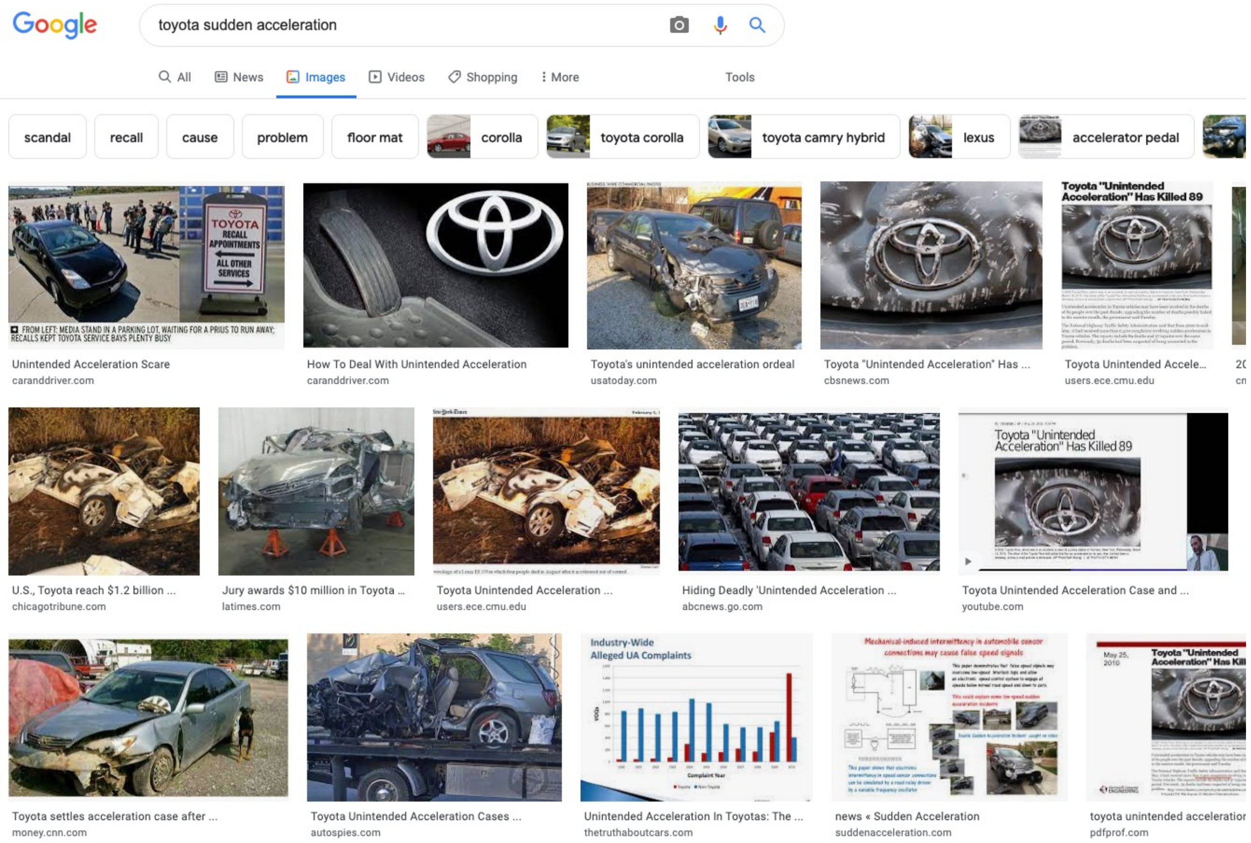
The software the team has used for five years to simulate such scenarios had generated the incorrect figures, consigning Hamilton to a second-place finish behind Vettel’s Ferrari.



> “Lewis did nothing wrong - it was down to a software bug or an algorithm that was simply wrong”
>
> Toto Wolff

https://www.formula1.com/en/latest/article.software-glitch-cost-hamilton-victory-mercedes.6VzyCYpEpaualYsOWYCqYS.html#:~:text=A%20software%20glitch.,season%2Dopening%20race%20in%20Australia.&text=The%20world%20champion%20immediately%20asked,time%20Mercedes%20had%20given%20him.

A bad code, a bug could cost more than the victory



Unintended Acceleration Scare
caranddriver.com

How To Deal With Unintended Acceleration
caranddriver.com

Toyota's unintended acceleration ordeal
usatoday.com

Toyota "Unintended Acceleration" Has ...
cbsnews.com

Toyota Unintended Accele...
users.ece.cmu.edu

U.S., Toyota reach $1.2 billion ...
chicagotribune.com

Jury awards $10 million in Toyota ...
latimes.com

Toyota Unintended Acceleration ...
users.ece.cmu.edu

Hiding Deadly 'Unintended Acceleration ...
abcnews.go.com

Toyota Unintended Acceleration Case and ...
youtube.com

Toyota settles acceleration case after ...
money.cnn.com

Toyota Unintended Acceleration Cases ...
autospies.com

Unintended Acceleration In Toyotas: The ...
thetruthaboutcars.com

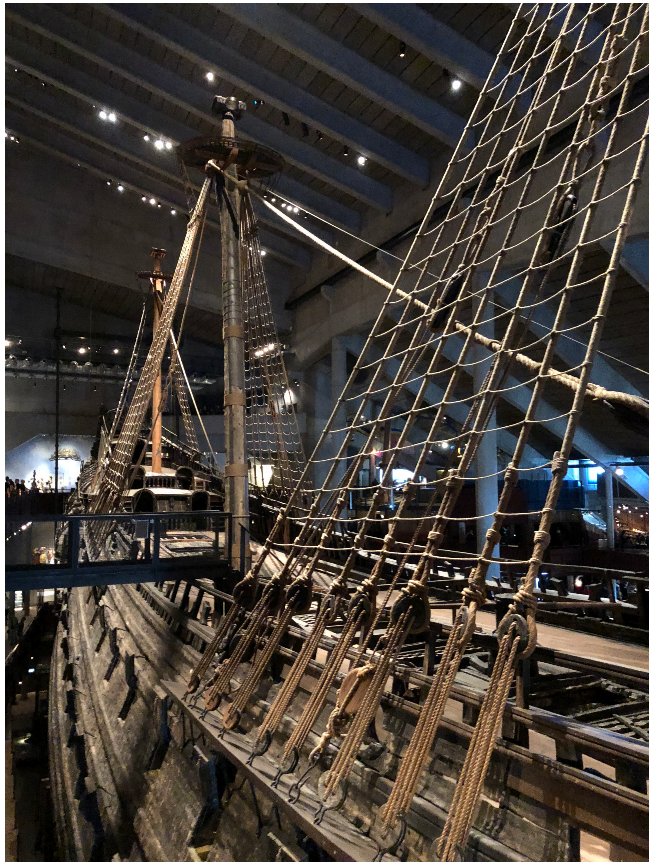
news « Sudden Acceleration
suddenacceleration.com

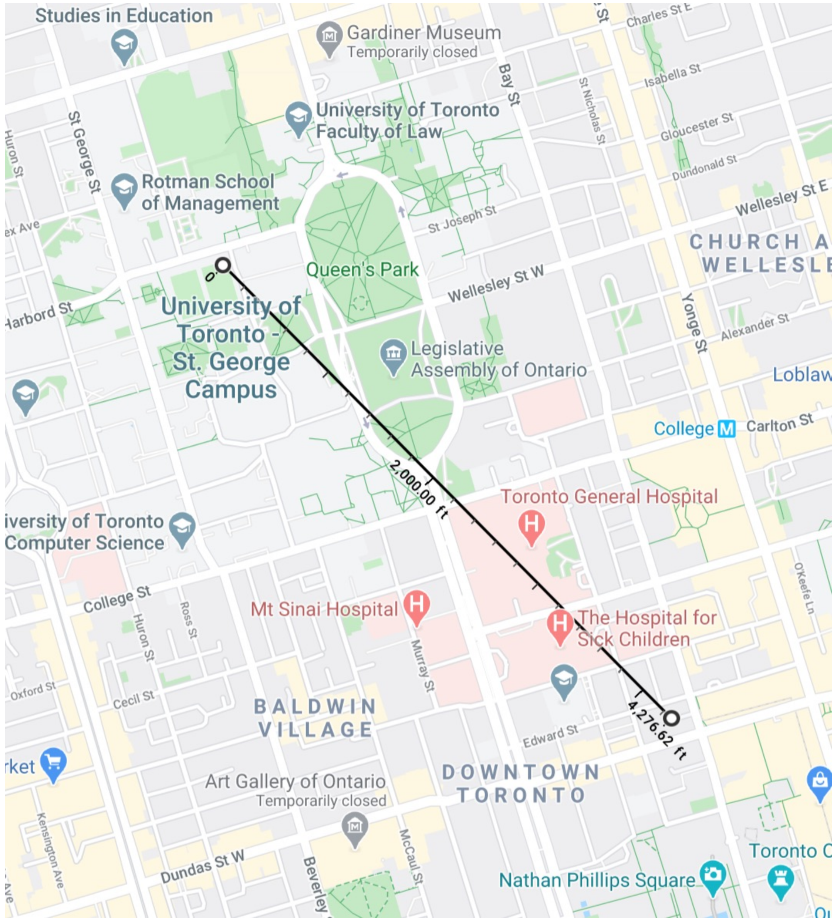
toyota unintended acceleratior
pdfprof.com

Perhaps 89 deaths, hundreds of serious injury lawsuits

- $1.6B class action settlement
- Jury found system defective – Toyota "acted in reckless disregard"
- Many of issues were SW, but also a HW problem

A failure of project management -- Swedish Vasa warship



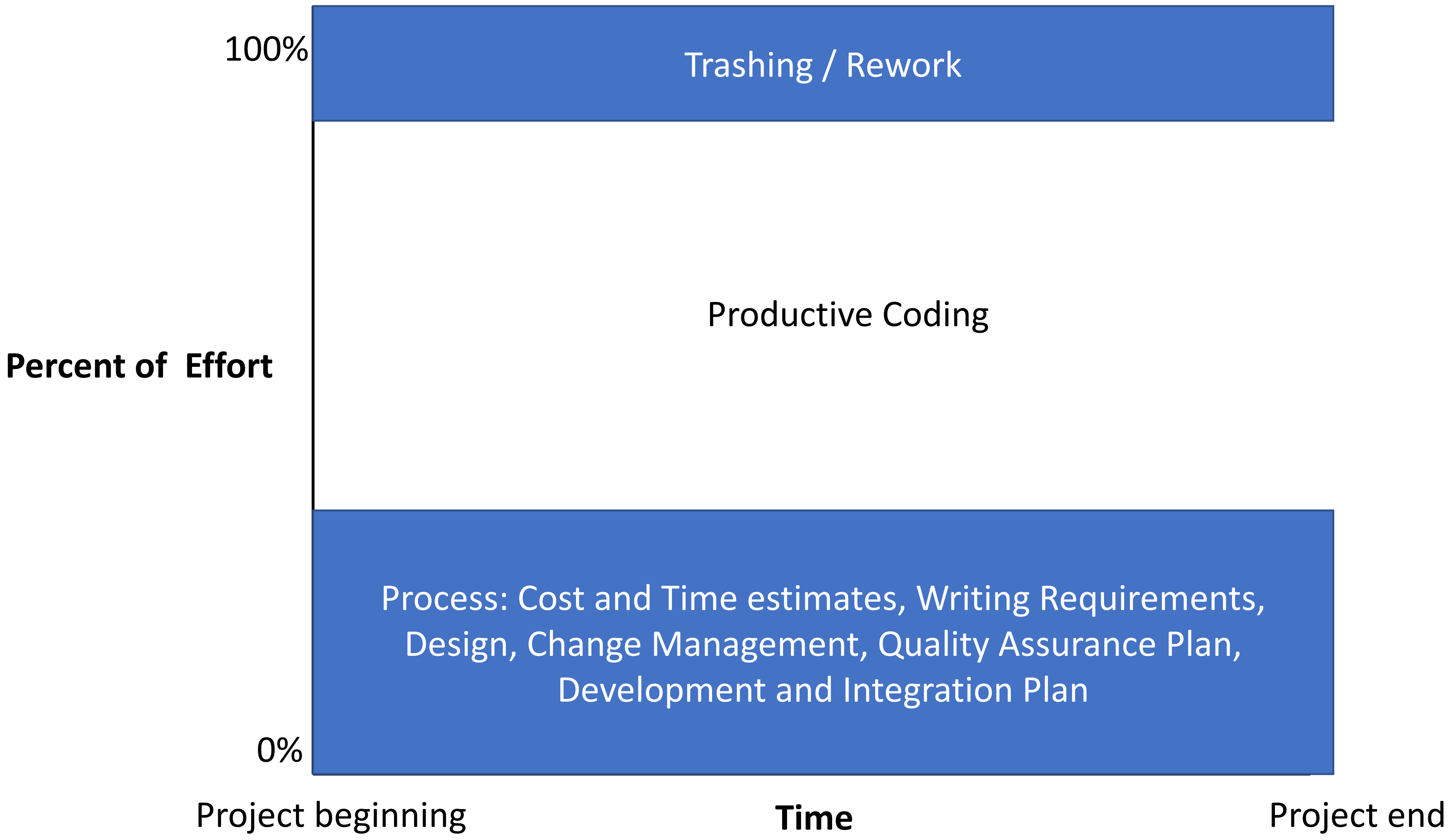


Software Engineering

What is **engineering**? And how is it different from

hacking/programming?

PROCESS IS IMPORTANT



100%

Trashing / Rework

Productive Coding

Percent of Effort

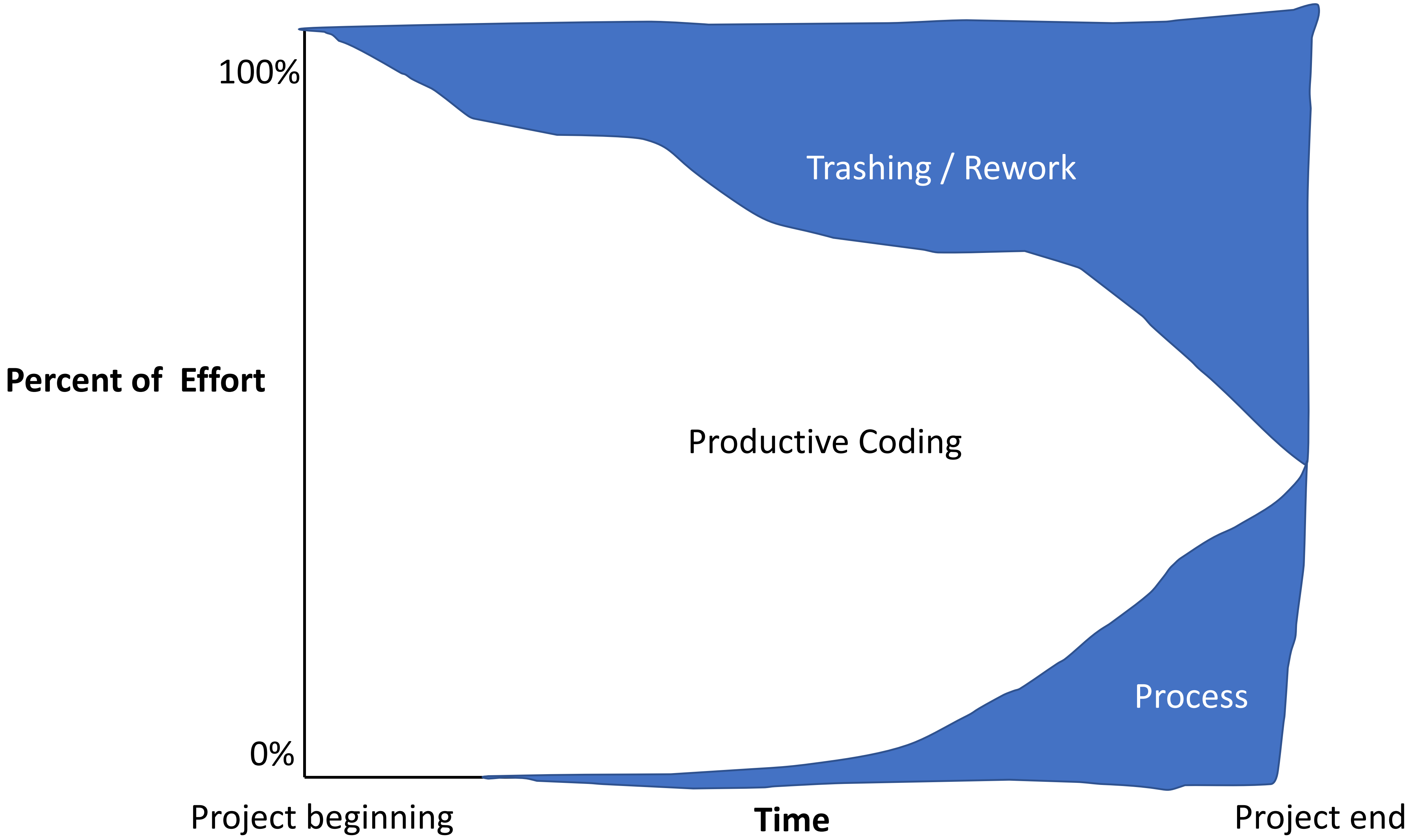
Process: Cost and Time estimates, Writing Requirements, Design, Change Management, Quality Assurance Plan, Development and Integration Plan

0%

Project beginning

Time

Project end

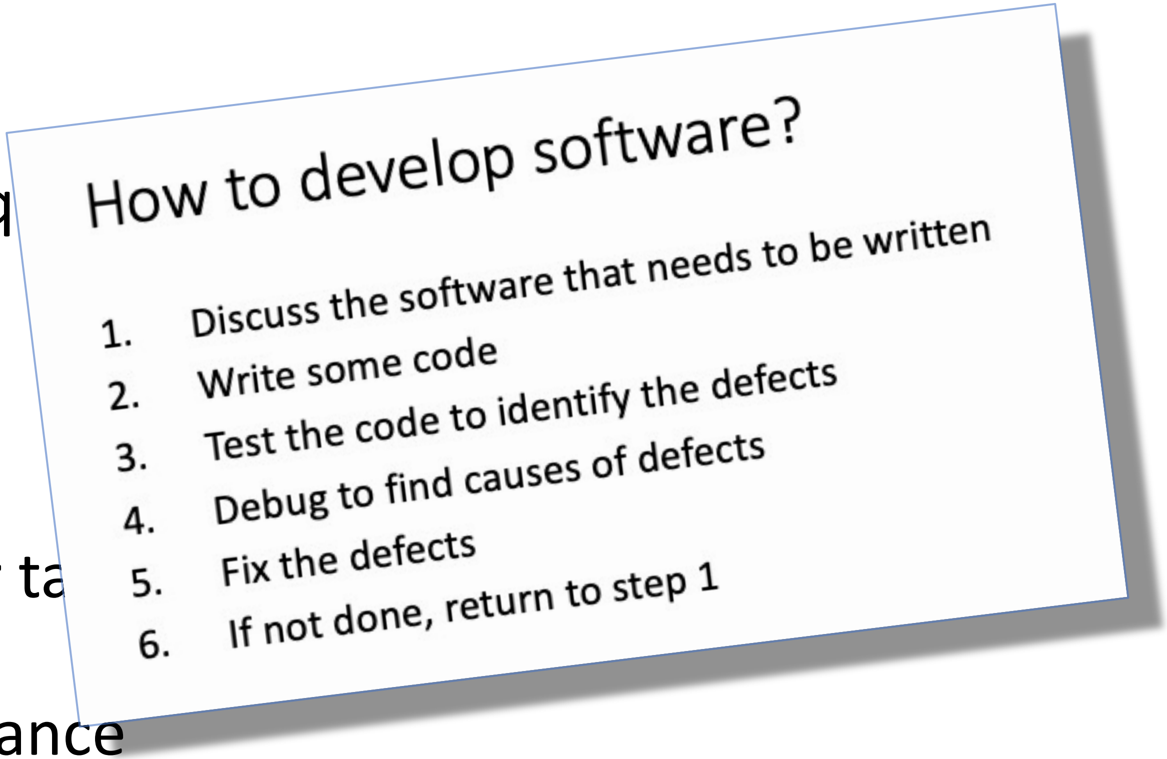

100%
Percent of Effort
0%
Trashing / Rework
Productive Coding
Process
Project beginning
Time
Project end

Survival Mode

- Missed deadlines -> "solo development mode" to meet own deadlines
- Ignore integration work
- Stop interacting with testers, technical writers, managers, …

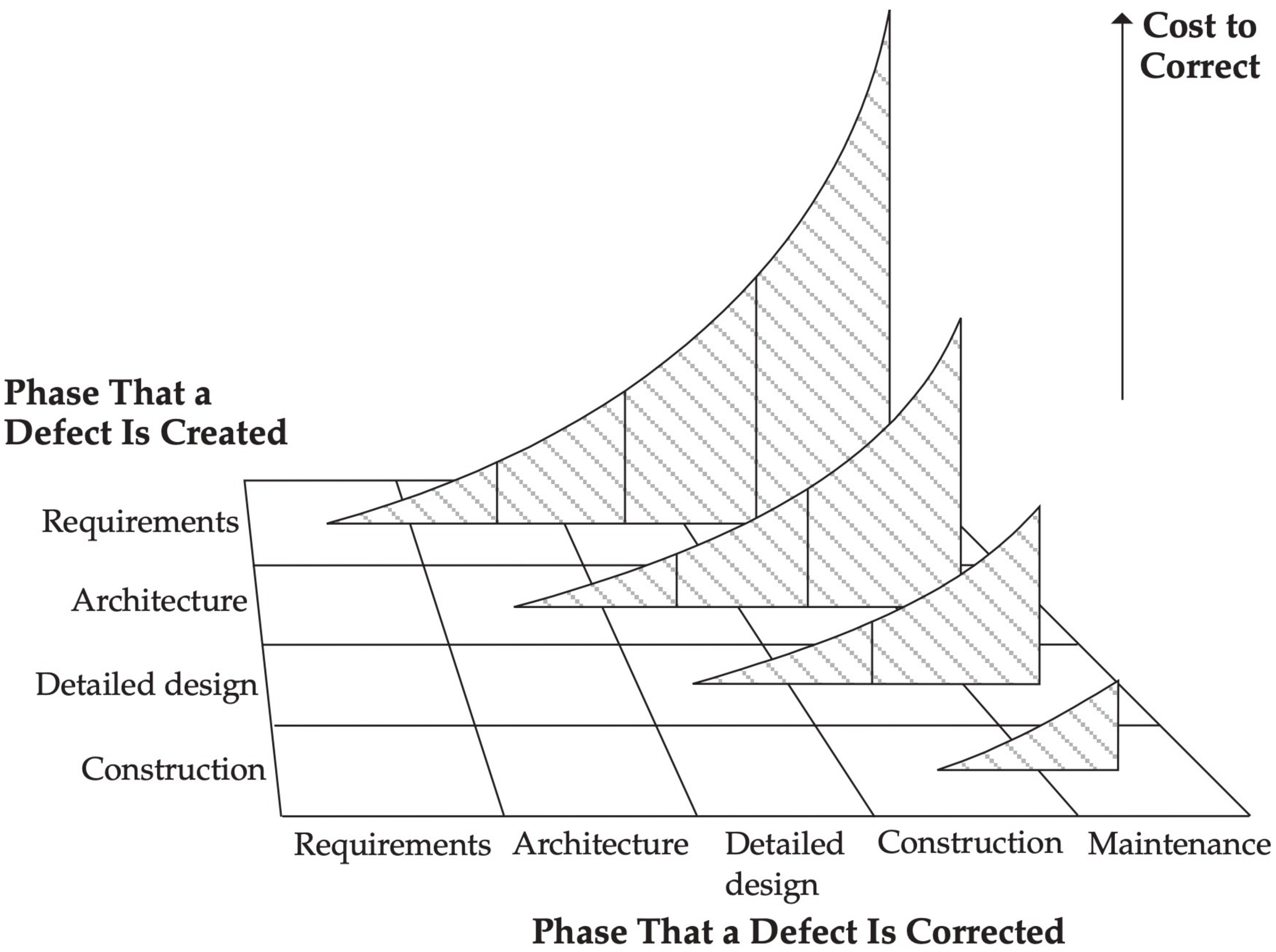
Example of Process Decisions

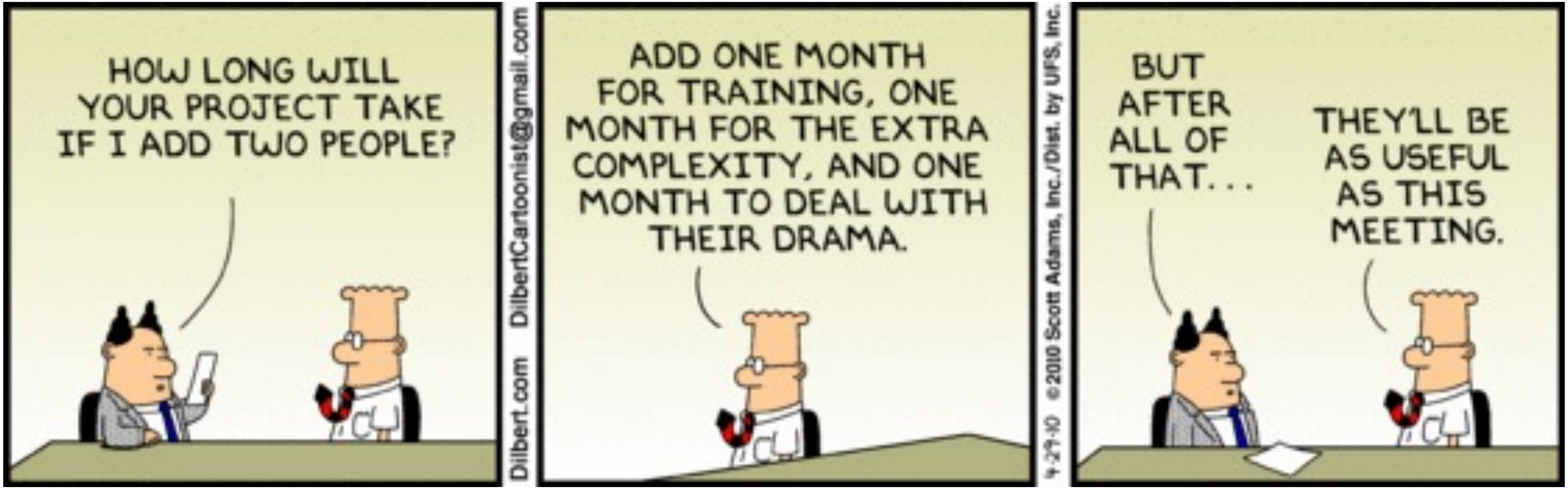
- Writing down all requirements
- Require approval for all changes to req
- Use version control for all changes
- Track all reported bugs
- Review requirements and code
- Break down development into smaller ta them
- Planning and conducting quality assurance
- Have daily status meetings
- Use Docker containers to push code between developers and operation



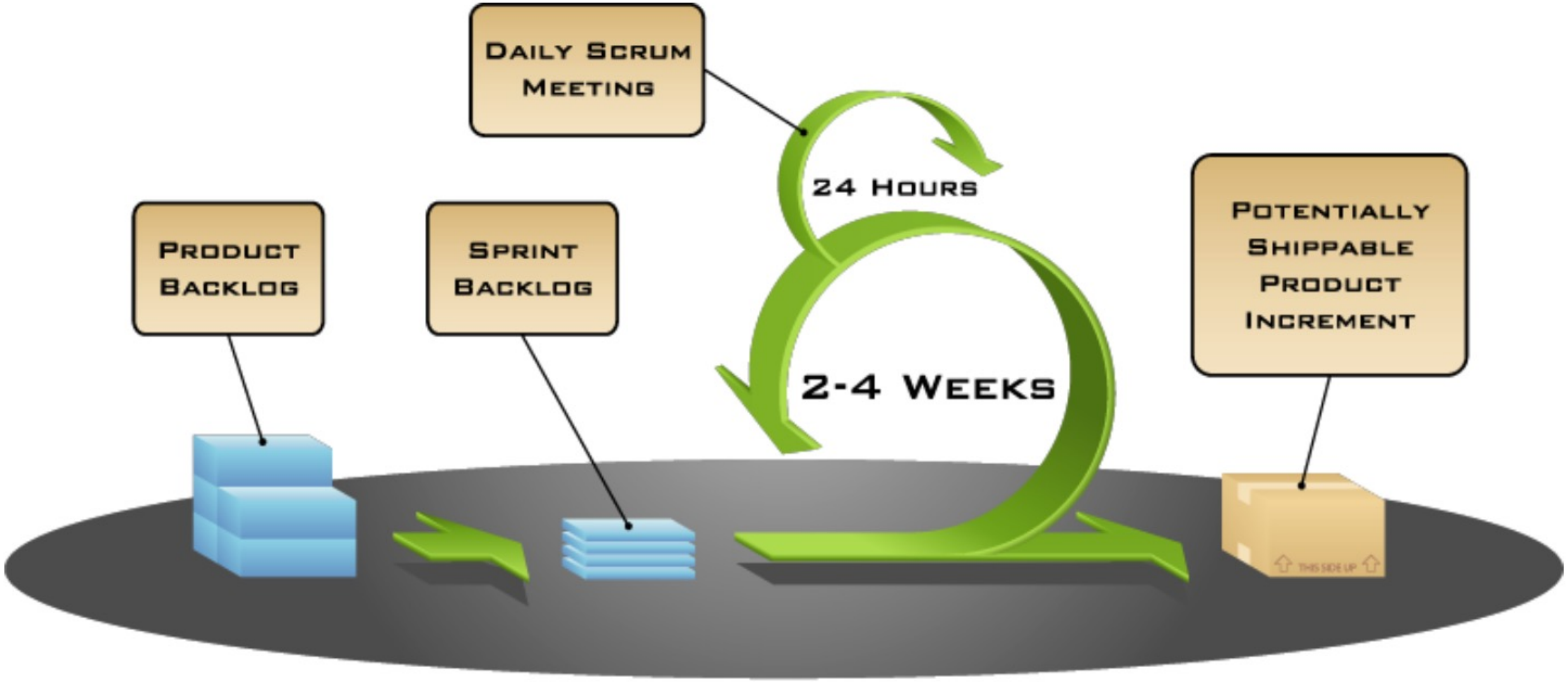
How to develop software?

1. Discuss the software that needs to be written
2. Write some code
3. Test the code to identify the defects
4. Debug to find causes of defects
5. Fix the defects
6. If not done, return to step 1


Cost to Correct
Phase That a Defect Is Created
Requirements
Architecture
Detailed design
Construction
Requirements
Architecture
Detailed design
Construction
Maintenance
Phase That a Defect Is Corrected


HOW LONG WILL YOUR PROJECT TAKE IF I ADD TWO PEOPLE?
ADD ONE MONTH FOR TRAINING, ONE MONTH FOR THE EXTRA COMPLEXITY, AND ONE MONTH TO DEAL WITH THEIR DRAMA.
BUT AFTER ALL OF THAT...
THEY'LL BE AS USEFUL AS THIS MEETING.
Dilbert.com DilbertCartoonist@gmail.com
4-29-10 © 2010 Scott Adams, Inc./Dist. by UFS, Inc.

Brief intro to Scrum



Project 1 – Milestone 1 Team Workflow

- The team workflow is a document that outlines team roles. It also gives us information about organizational issues, like team meeting times. This helps us send course staff to aid you and helps us to follow your progress.
- The main purpose of this document is to give you some rules for team process, management, tracking, and goal setting. As a general rule, groups work pretty well in this course.However, any good working group will have some measurements in place if something goes awry.

ECE444: Software Engineering

Case Study

Shurui Zhou

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Flawed analysis, failed oversight: How Boeing, FAA certified the suspect 737 MAX flight control system

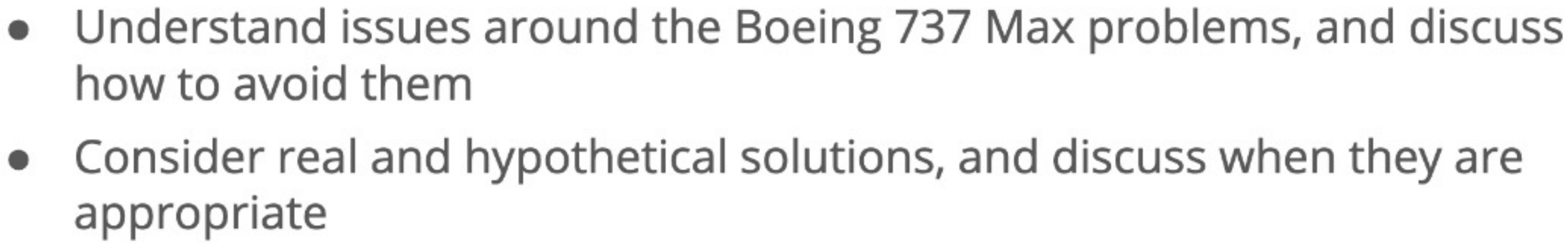
March 17, 2019 at 6:00 am | *Updated March 21, 2019 at 9:46 am*

- One pilot said it was “unconscionable that a manufacturer, the FAA (Federal Aviation Administration), and the airlines would have pilots flying an airplane without adequately training, or even providing **available resources and sufficient documentation** to understand the highly complex systems that differentiate this aircraft from prior models”

https://www.theverge.com/2019/5/2/18518176/boeing-737-max-crash-problems-human-error-mcas-faa

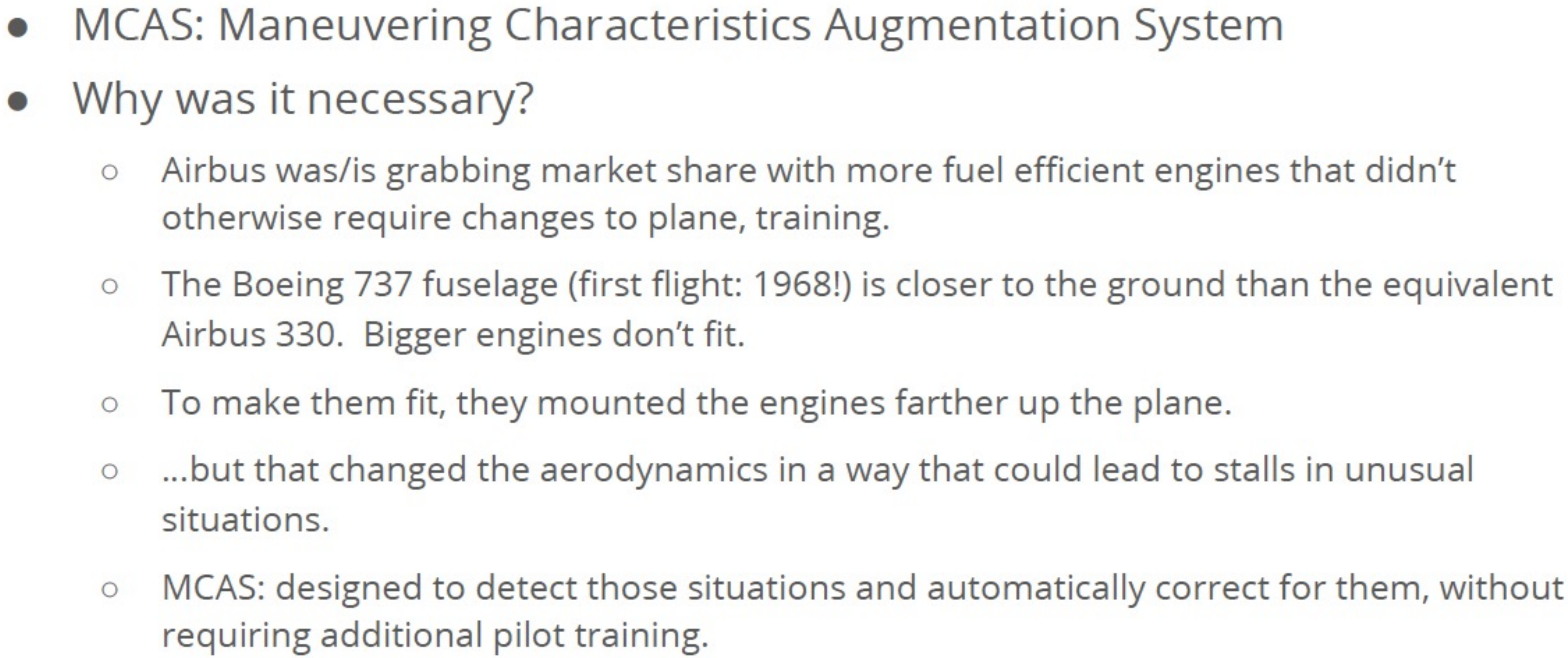
https://www.seattletimes.com/business/boeing-aerospace/failed-certification-faa-missed-safety-issues-in-the-737-max-system-implicated-in-the-lion-air-crash/

Learning goal



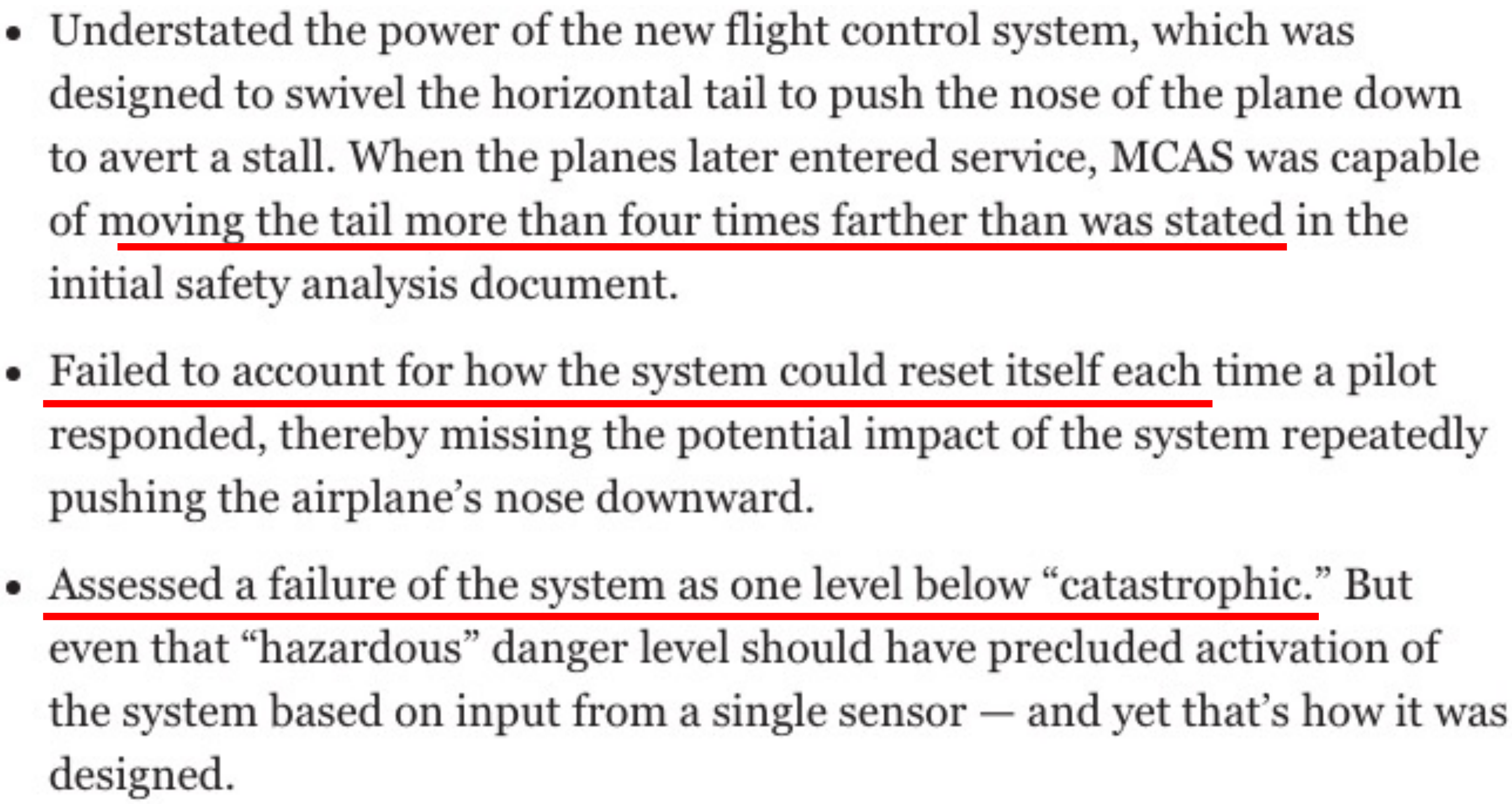
- Understand issues around the Boeing 737 Max problems, and discuss how to avoid them
- Consider real and hypothetical solutions, and discuss when they are appropriate

Some background...



- MCAS: Maneuvering Characteristics Augmentation System
- Why was it necessary?
  - Airbus was/is grabbing market share with more fuel efficient engines that didn't otherwise require changes to plane, training.
  - The Boeing 737 fuselage (first flight: 1968!) is closer to the ground than the equivalent Airbus 330. Bigger engines don't fit.
  - To make them fit, they mounted the engines farther up the plane.
  - ...but that changed the aerodynamics in a way that could lead to stalls in unusual situations.
  - MCAS: designed to detect those situations and automatically correct for them, without requiring additional pilot training.

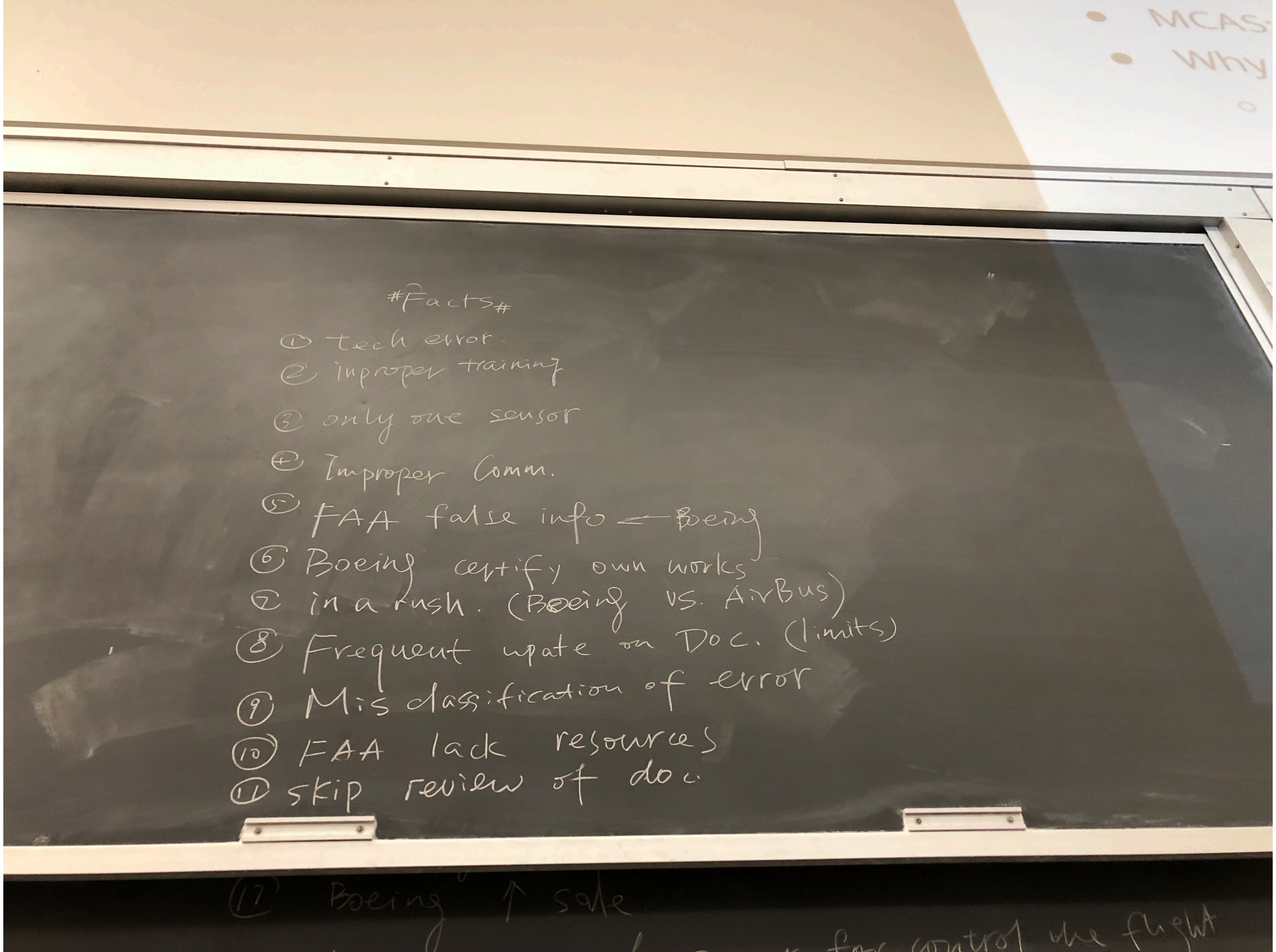
Safety Analysis



- Understated the power of the new flight control system, which was designed to swivel the horizontal tail to push the nose of the plane down to avert a stall. When the planes later entered service, MCAS was capable of moving the tail more than four times farther than was stated in the initial safety analysis document.
- Failed to account for how the system could reset itself each time a pilot responded, thereby missing the potential impact of the system repeatedly pushing the airplane's nose downward.
- Assessed a failure of the system as one level below "catastrophic." But even that "hazardous" danger level should have precluded activation of the system based on input from a single sensor — and yet that's how it was designed.

Activities

- **The Facts**: create a list of as many facts about the Boeing case as you can come up with.
- **Relationship Between Facts**: identify as many relationships between facts as you can
- **Takeaways**

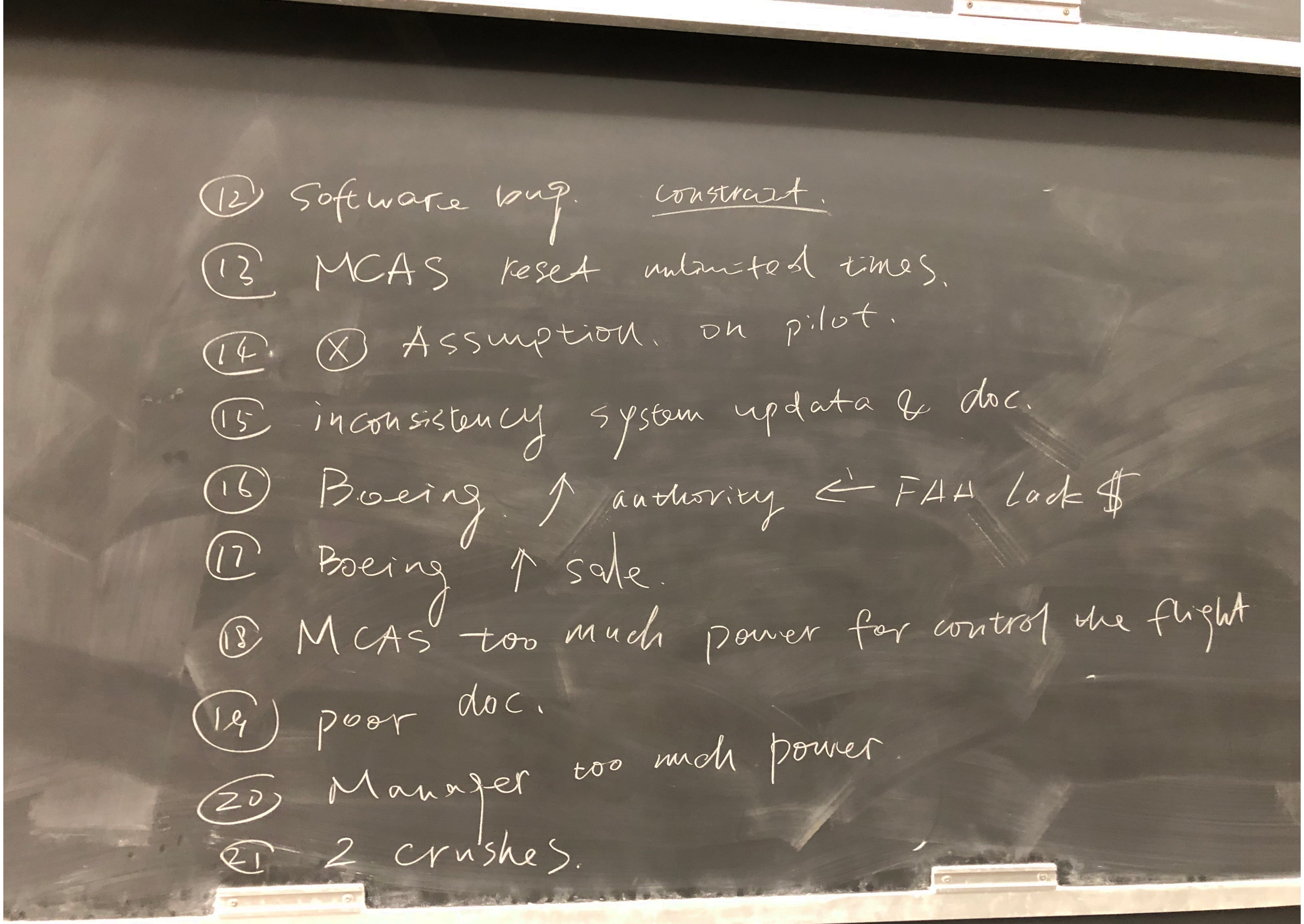


#Facts#

① tech error.
② inproper training
③ only one sensor
④ Improper Comm.
⑤ FAA false info ← Boeing
⑥ Boeing certify own works
⑦ in a rush. (Boeing vs. AirBus)
⑧ Frequent upate on Doc. (limits)
⑨ Mis classification of error
⑩ FAA lack resources
⑪ skip review of doc.

⑰ Boeing ↑ sale

for control the flight



⑫ Software bug. constraint.

⑬ MCAS reset unlimited times.

⑭ ⊗ Assumption. on pilot.

⑮ inconsistency system updata & doc.

⑯ Boeing ↑ authority ← FAA lack $

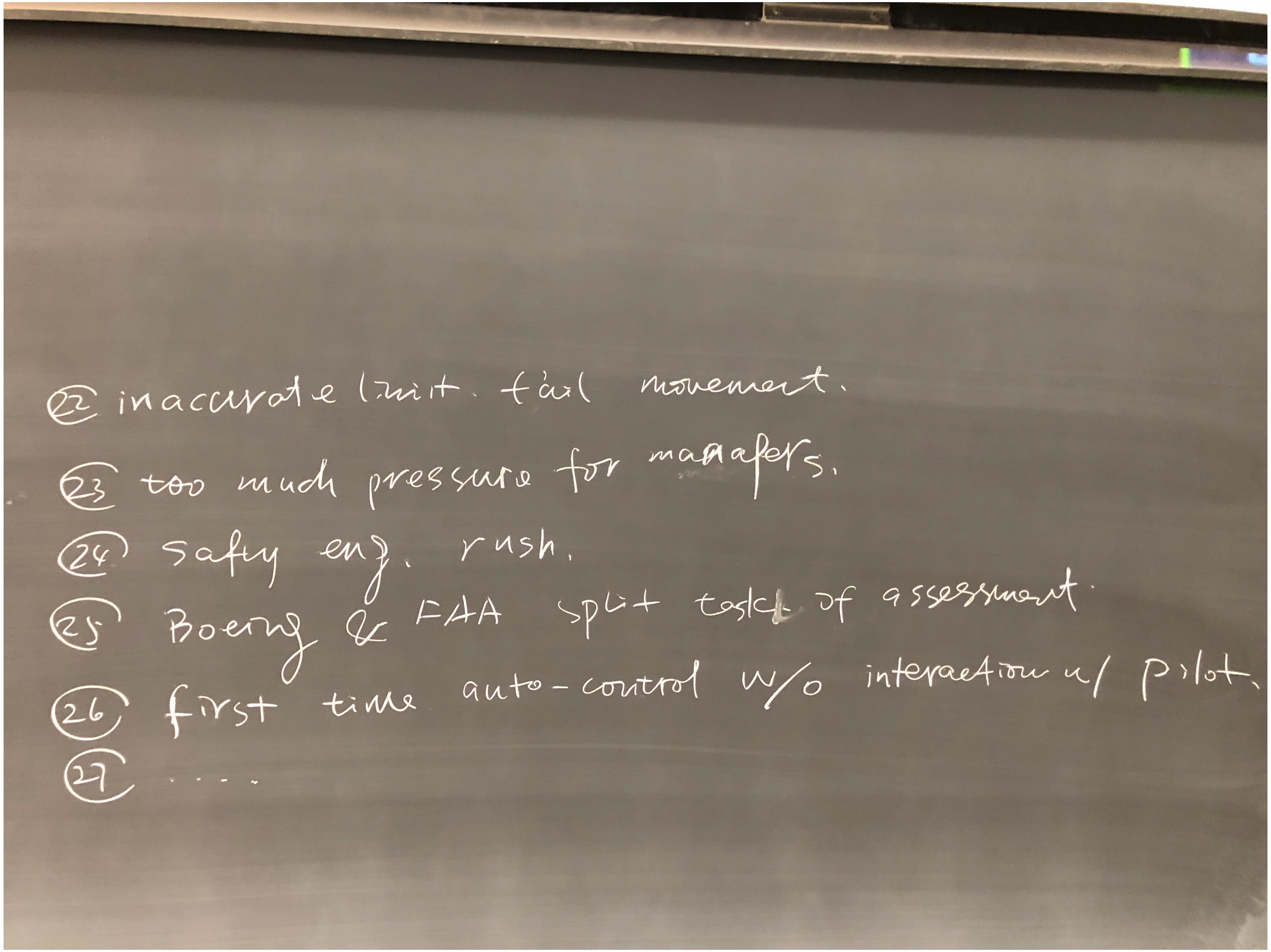
⑰ Boeing ↑ sale.

⑱ MCAS too much power for control the flight

⑲ poor doc.

⑳ Manager too much power.

㉑ 2 crushes.



22. inaccurate limit. fail movement.

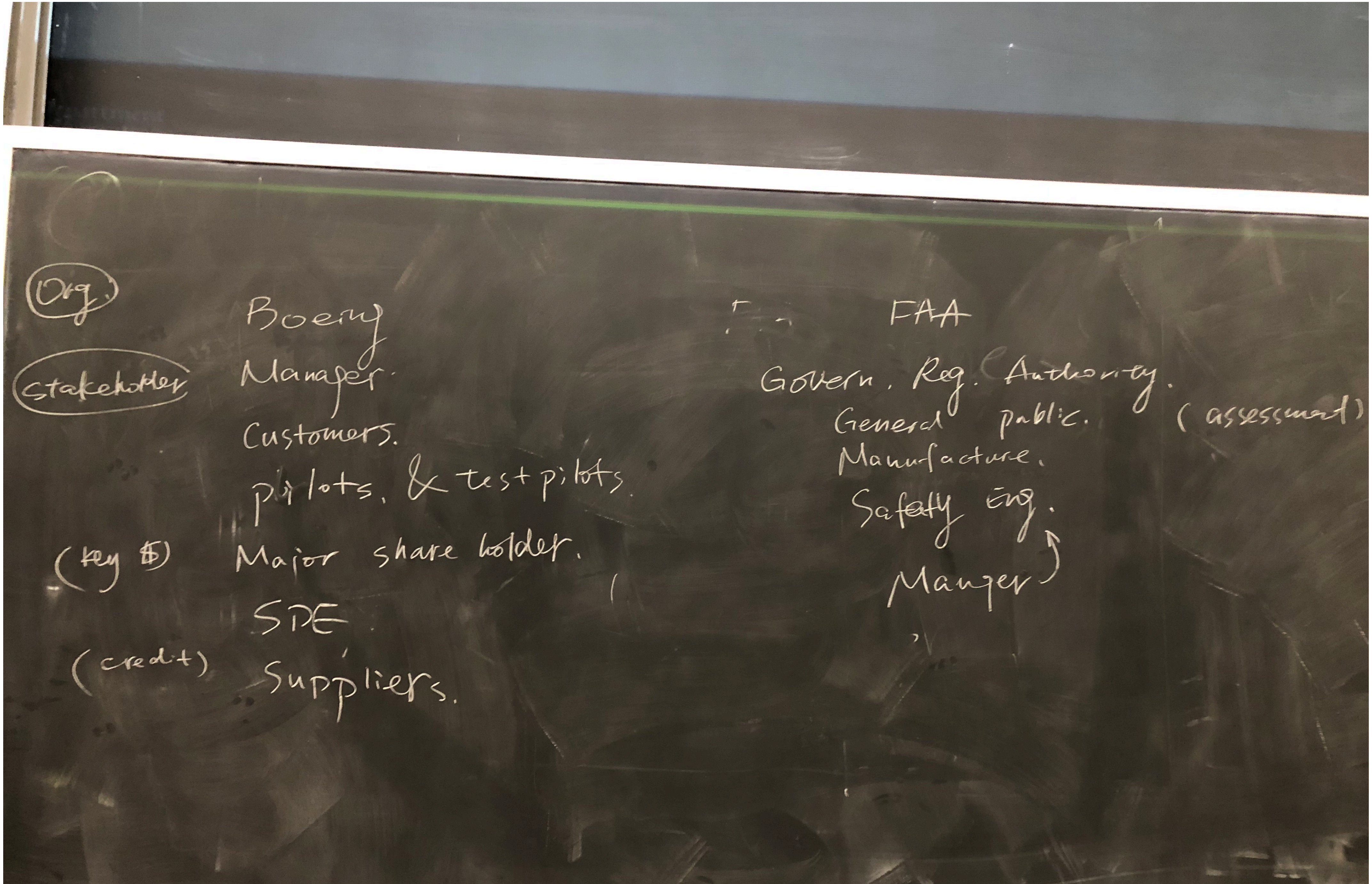
23. too much pressure for managers.

24. safty eng. rush.

25. Boeing & FAA split task of assessment.

26. first time auto-control w/o interaction w/ pilot.

27. ....



Org. Boeing

Stakeholder Manager.

Customers.

pilots. & test pilots.

(key $) Major share holder.

SPE.

(credit) Suppliers.

FAA

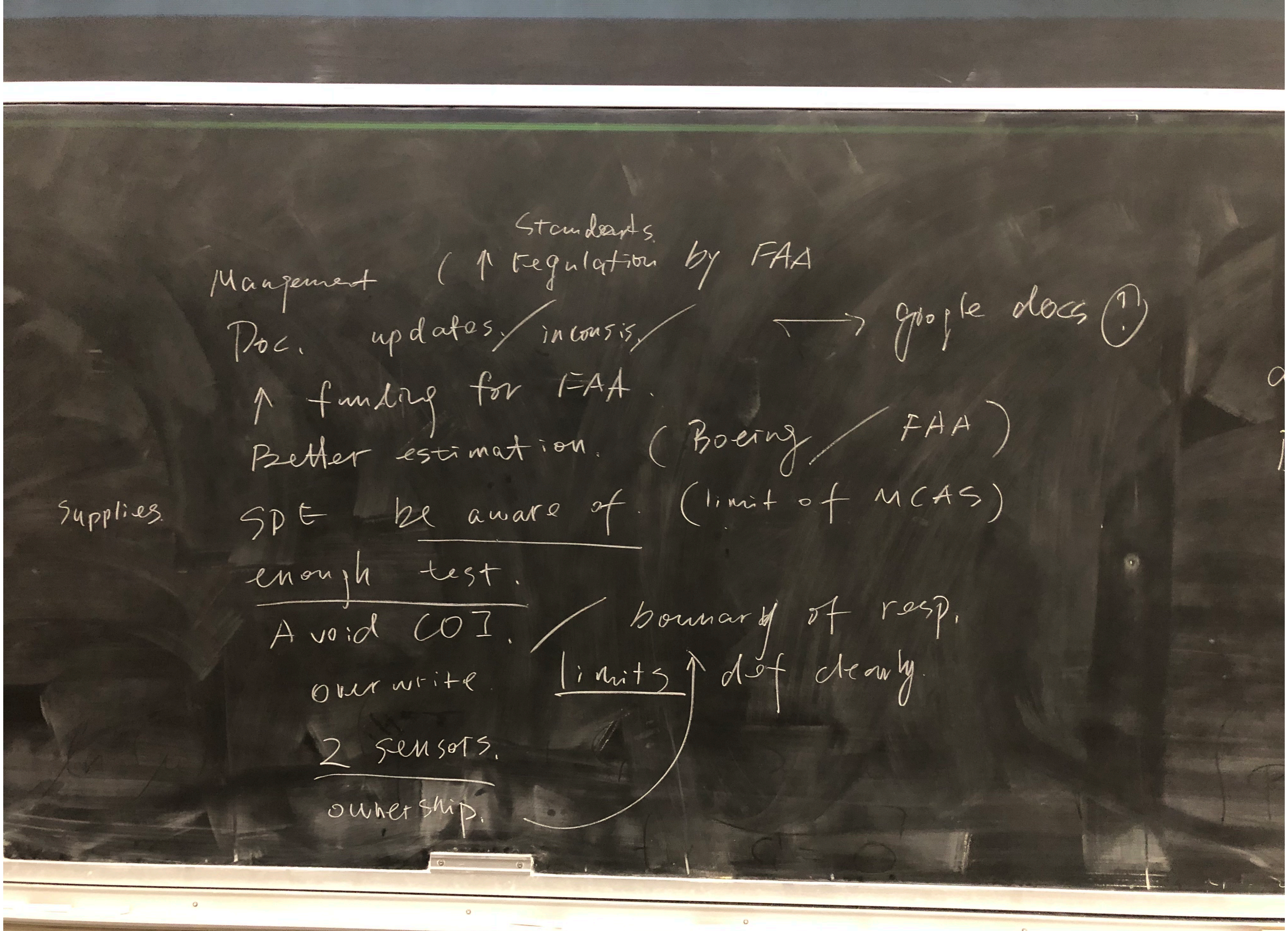
Govern. Reg. Authority.

General public. (assessment)

Manufacture.

Safety eng.

Manger



Standards

Management (↑ regulation by FAA

Doc. updates / inconsis / ⟶ google docs (?)

↑ funding for FAA.

Better estimation. (Boeing / FAA)

Supplies. SPE be aware of. (limit of MCAS)

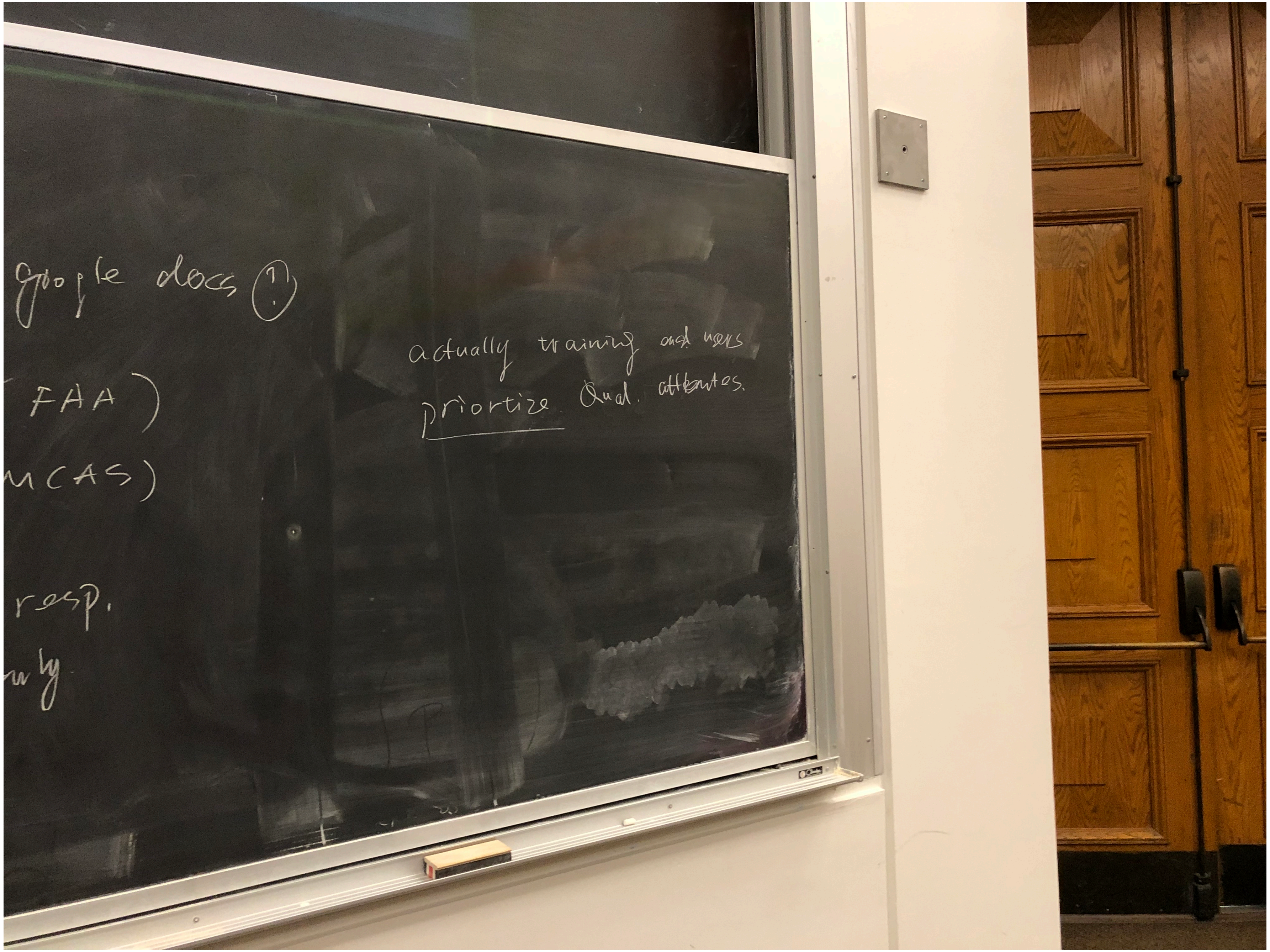
enough test.

Avoid COI. / bounary of resp.

overwrite limits ↑ def clearly.

2 sensors.

ownership.



google docs ?

FAA)

MCAS)

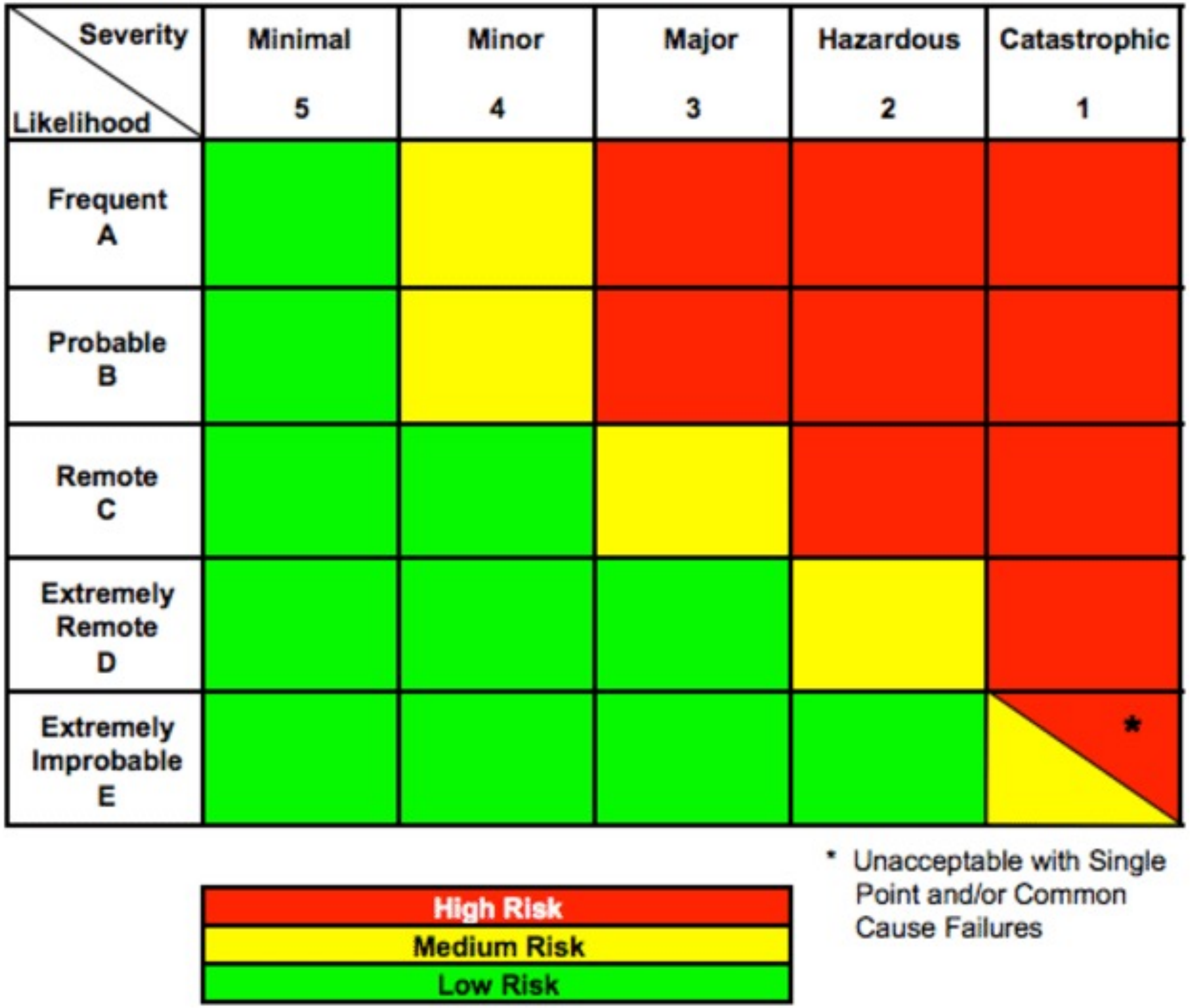
resp.

nly

actually training and users

priortize Qual. attributes.

Risk Matrix



Severity / Likelihood	Minimal 5	Minor 4	Major 3	Hazardous 2	Catastrophic 1
Frequent A	🟩	🟨	🟥	🟥	🟥
Probable B	🟩	🟨	🟥	🟥	🟥
Remote C	🟩	🟩	🟨	🟥	🟥
Extremely Remote D	🟩	🟩	🟩	🟨	🟥
Extremely Improbable E	🟩	🟩	🟩	🟩	🟨/🟥 *

🟥	High Risk
🟨	Medium Risk
🟩	Low Risk

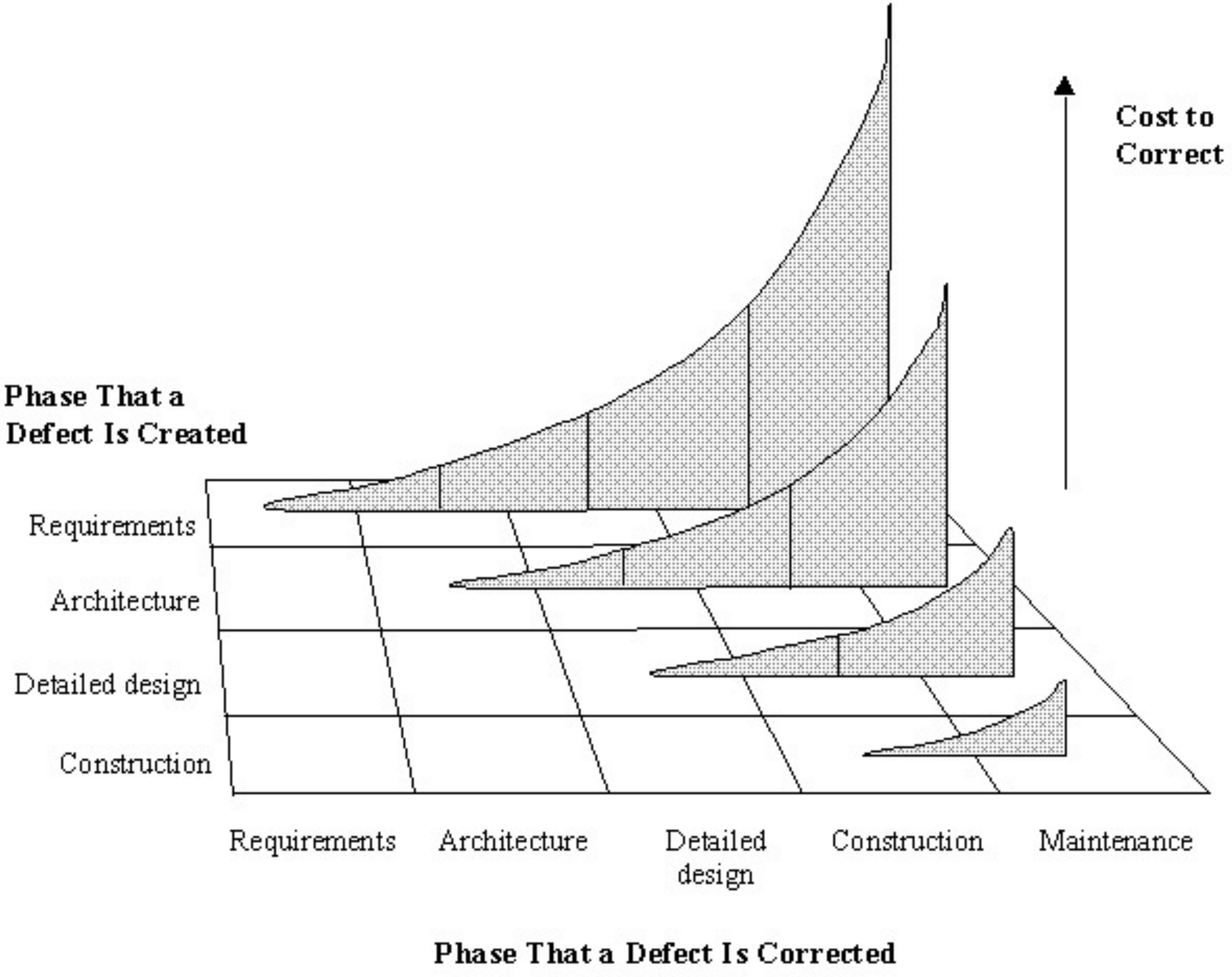
* Unacceptable with Single Point and/or Common Cause Failures

Requirements 1:

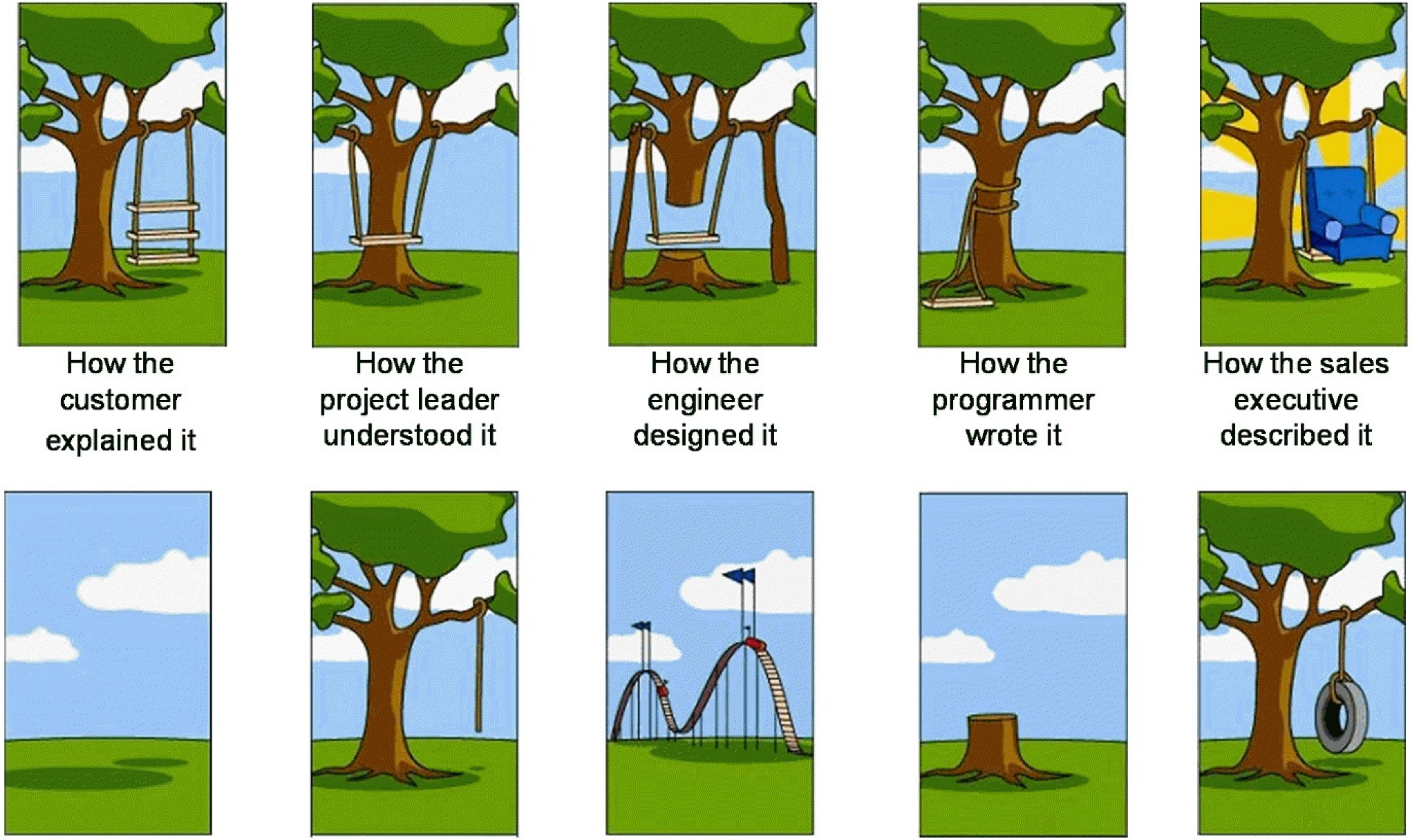
Overview and Concepts

ECE444 Software Engineering (Fall 2021)

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO



Copyright 1998 Steven C. McConnell. Reprinted with permission from *Software Project Survival Guide* (Microsoft Press, 1998).



How the customer explained it

How the project leader understood it

How the engineer designed it

How the programmer wrote it

How the sales executive described it

How the project was documented

What operations installed

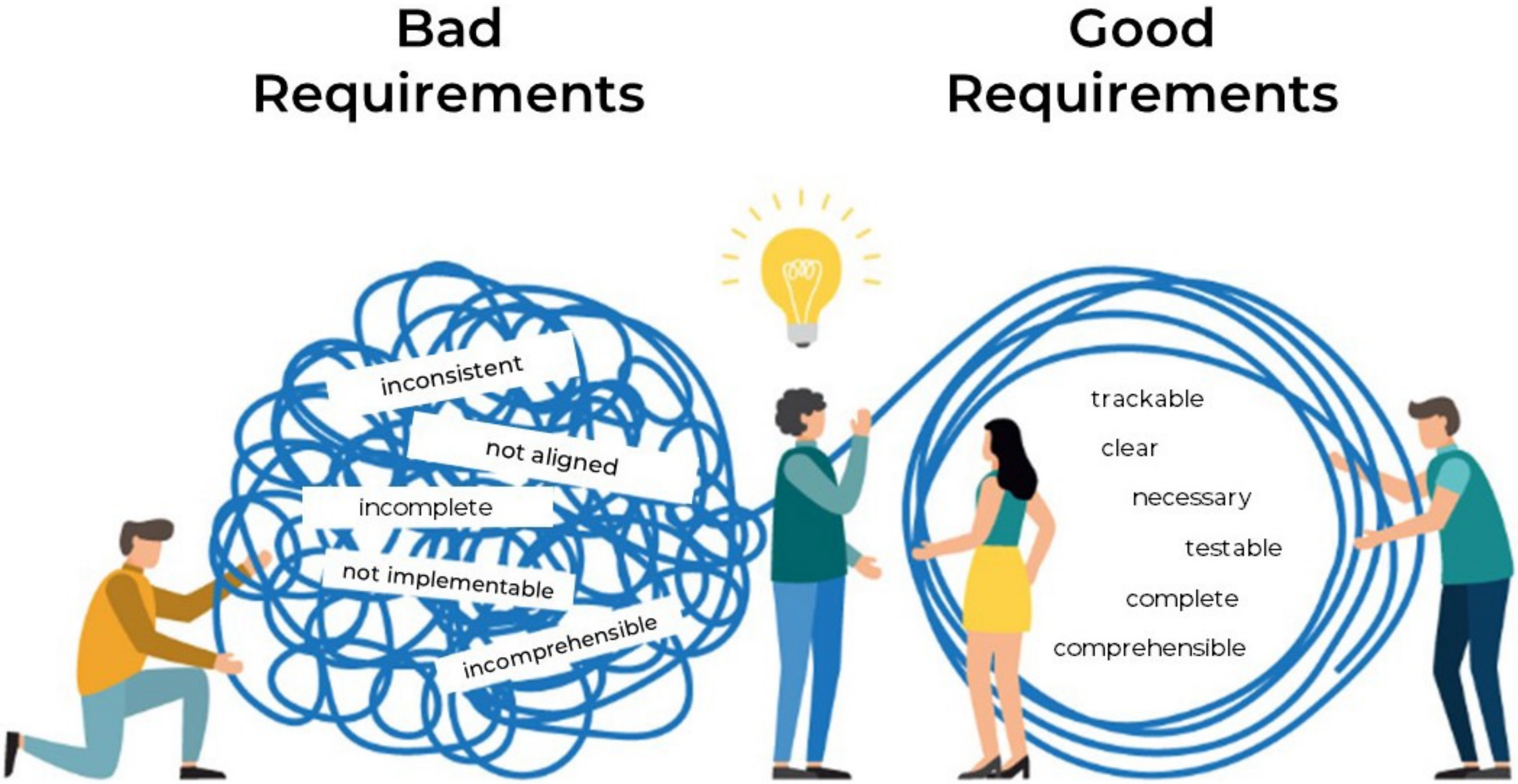
How the customer was billed

How the help desk supported it

What the customer really needed

Learning Goals

- Explain the importance and challenges of requirements in software engineering.
- Explain how and why requirements articulate the relationship between a desired system and its environment.
- Identify assumptions.
- Distinguish between and give examples of: functional and quality requirements; informal statements and verifiable requirements.
- State quality requirements in measurable ways



https://www.softwareone.com/en-ca/blog/articles/2020/03/11/equirements-engineering-documentation-of-requirements

Overly simplified definition...

Requirements say what the system will do (and not how it will do it).

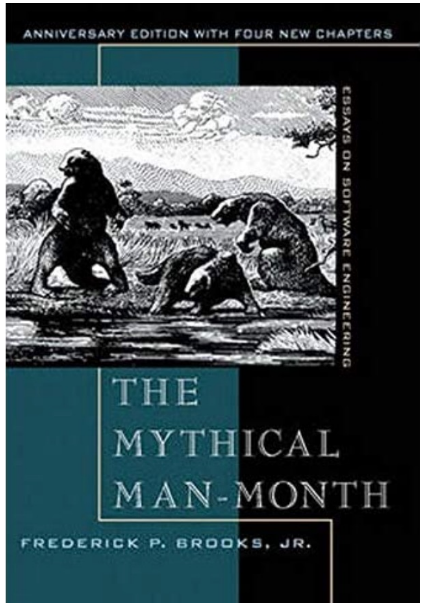
Communication problem

Goal: figure out what should be built.

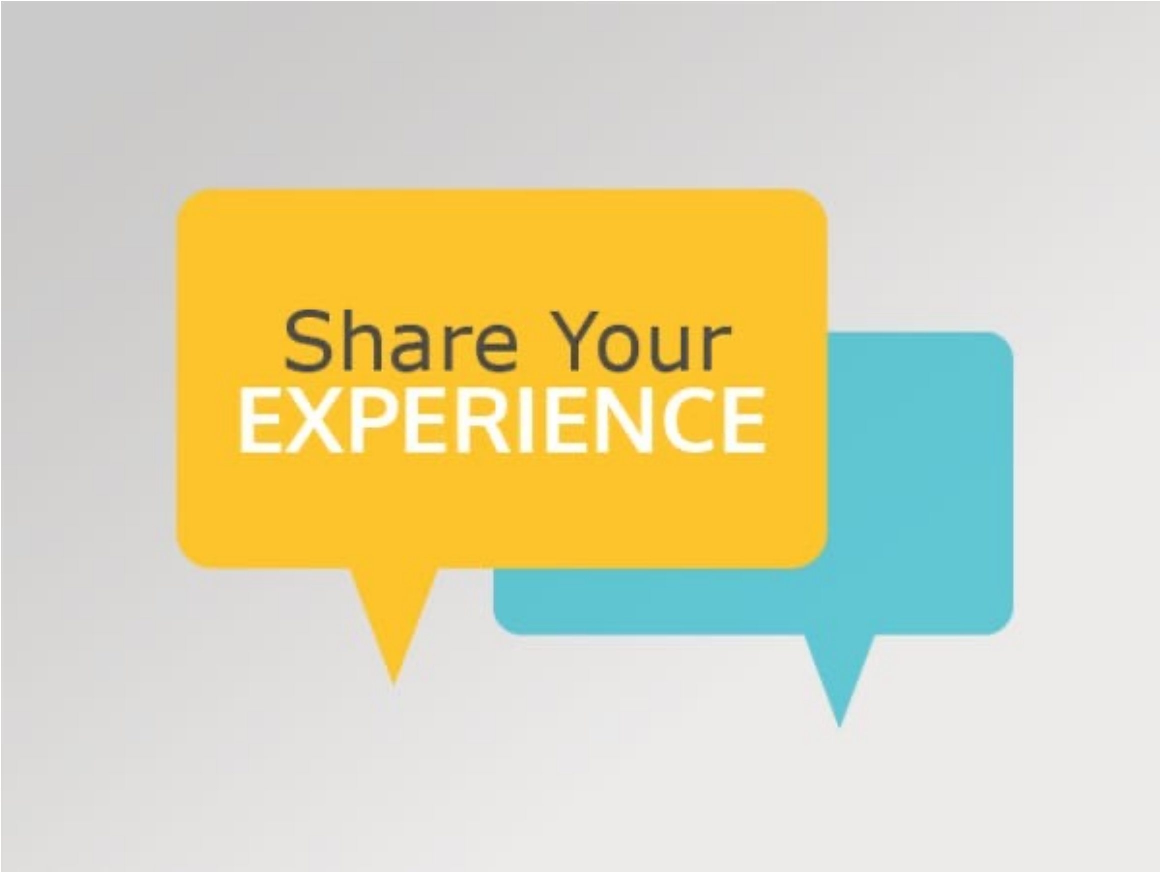
Express those ideas so that the correct thing is built.

Fred Brooks, on requirements

- *The hardest single part of building a software system is deciding precisely* ***what to build****.*
- *No other part of the conceptual work is as difficult as establishing the detailed technical requirements ...*
- *No other part is as difficult to rectify later.*


ANNIVERSARY EDITION WITH FOUR NEW CHAPTERS
ESSAYS ON SOFTWARE ENGINEERING
THE
MYTHICAL
MAN-MONTH
FREDERICK P. BROOKS, JR.

Why is this hard?



What is requirement engineering?

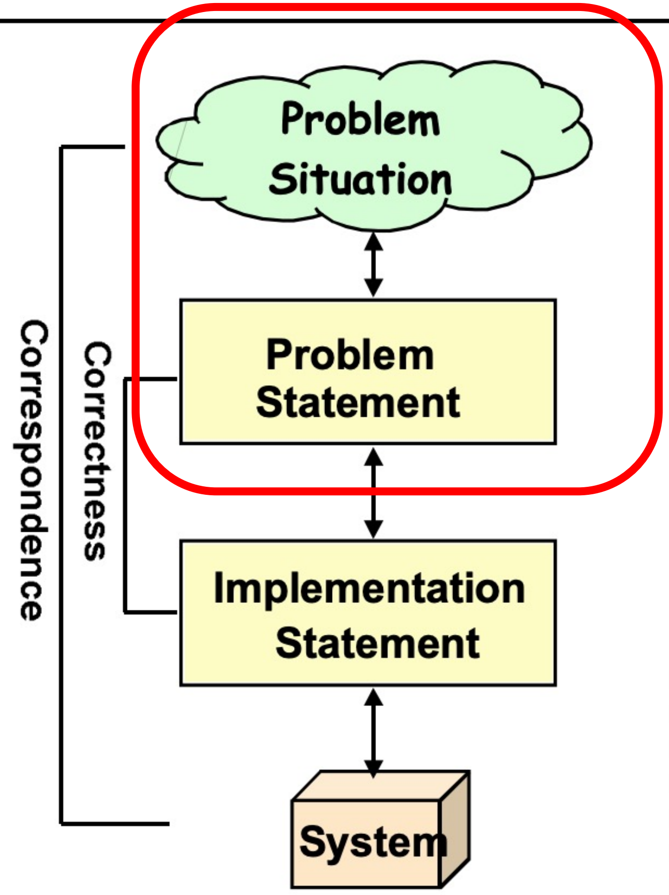


Figure 1: Separating the problem statement from the solution statement (adapted from Blum 1992)

Problem: ?

Solution: code, design drawings, system architecture, user manuals, etc

- http://www.cs.toronto.edu/~sme/CSC340F/readings/FoRE-chapter02-v7.pdf

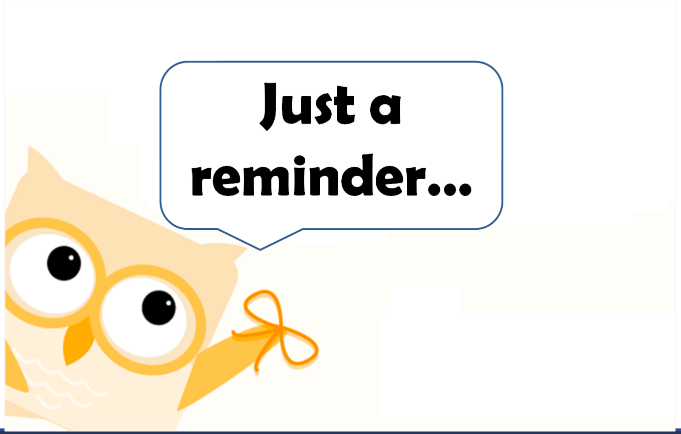
What is requirement engineering?

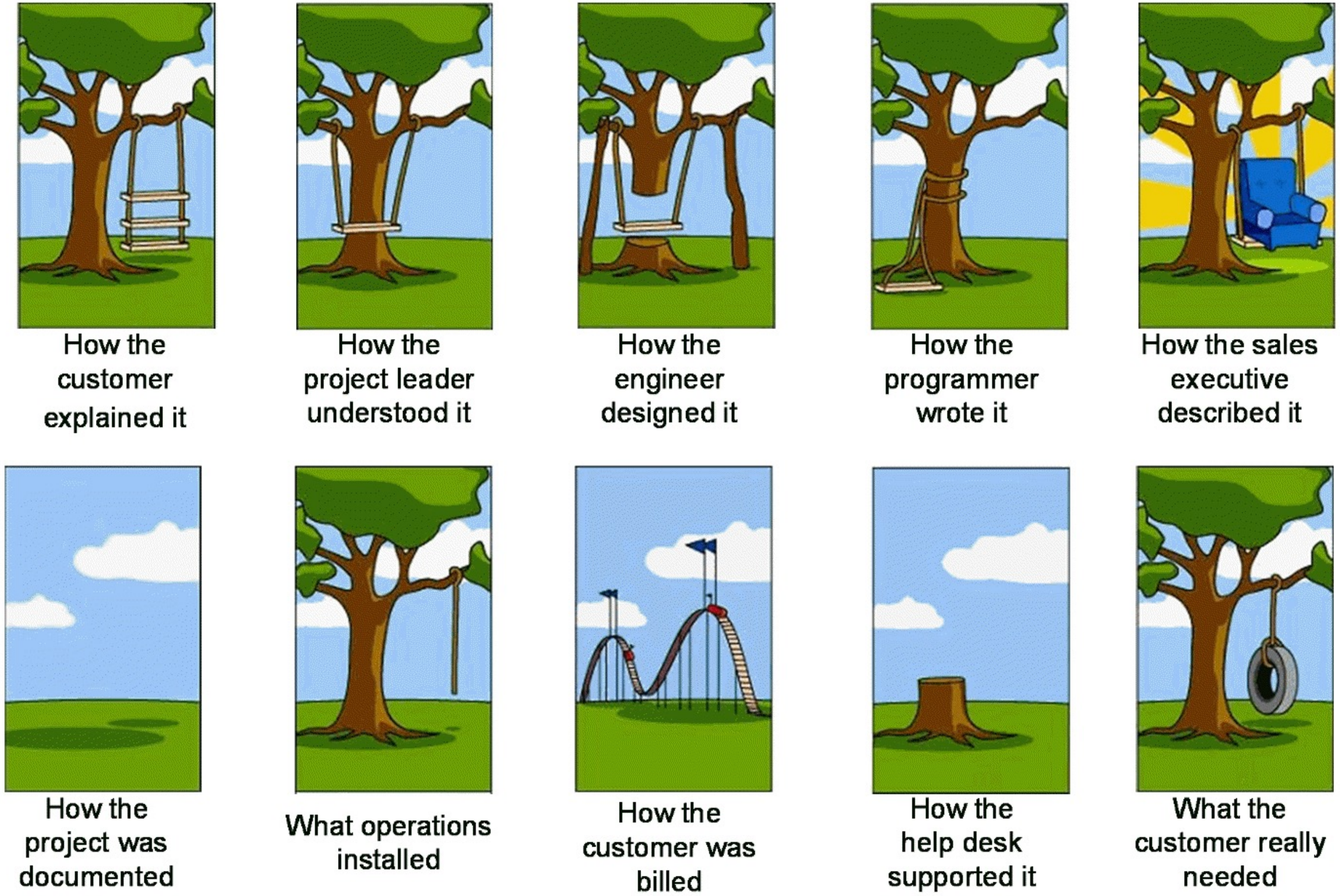
- Knowledge **acquisition** – how to capture relevant detail about a system?
  - Is the knowledge complete and consistent?
- Knowledge **representation** – once captured, how do we express it most effectively?
  - Express it for whom?
  - Is it received consistently by different people?

Capturing vs Synthesizing

- Engineers acquire requirements **from many sources**
  - Elicit from stakeholders
  - Extract from policies or other documentation
  - Synthesize from above + estimation and invention

- Because stakeholders do not always "know what they want"*, engineers must…
  - Be faithful to stakeholder needs and expectations
  - Anticipate additional needs and risks
  - Validate that "additional needs" are necessary or desired





How the customer explained it

How the project leader understood it

How the engineer designed it

How the programmer wrote it

How the sales executive described it

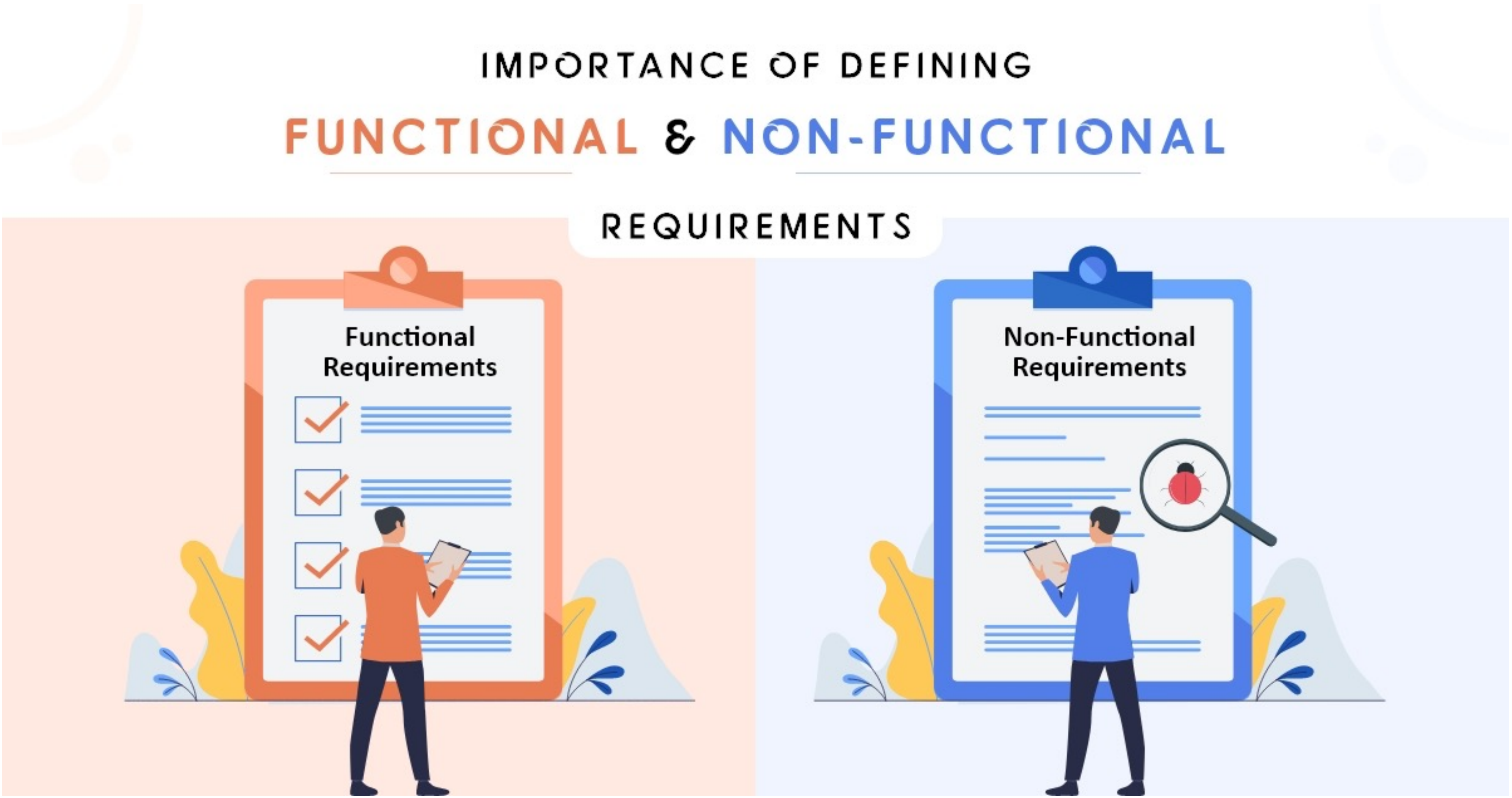
How the project was documented

What operations installed

How the customer was billed

How the help desk supported it

What the customer really needed



https://kodytechnolab.com/functional-requirements-vs-non-functional-requirements

Functional & Non-Functional Requirements

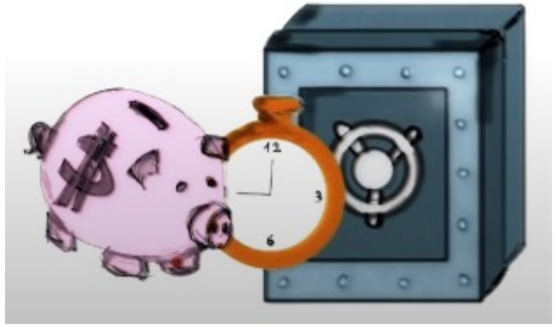
Functional Requirements: covers the main functions.

- *user* requirement
- *system* requirement

Non-Functional Requirements

These are the constrains on the functions provided by the system.

e.g., performance & security &…



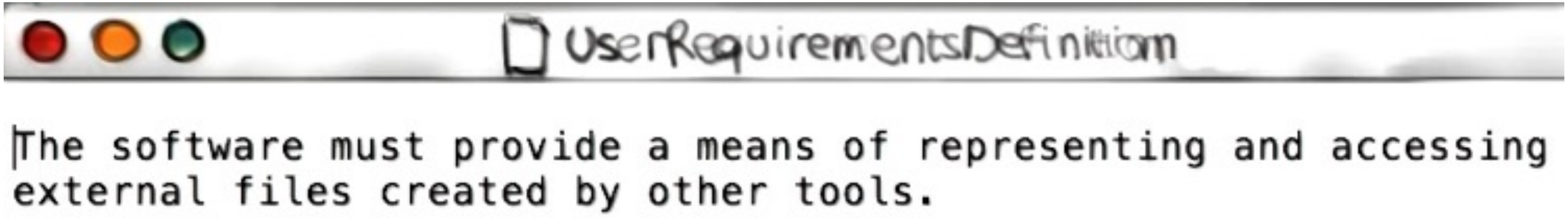
User and System Requirements

User Requirements

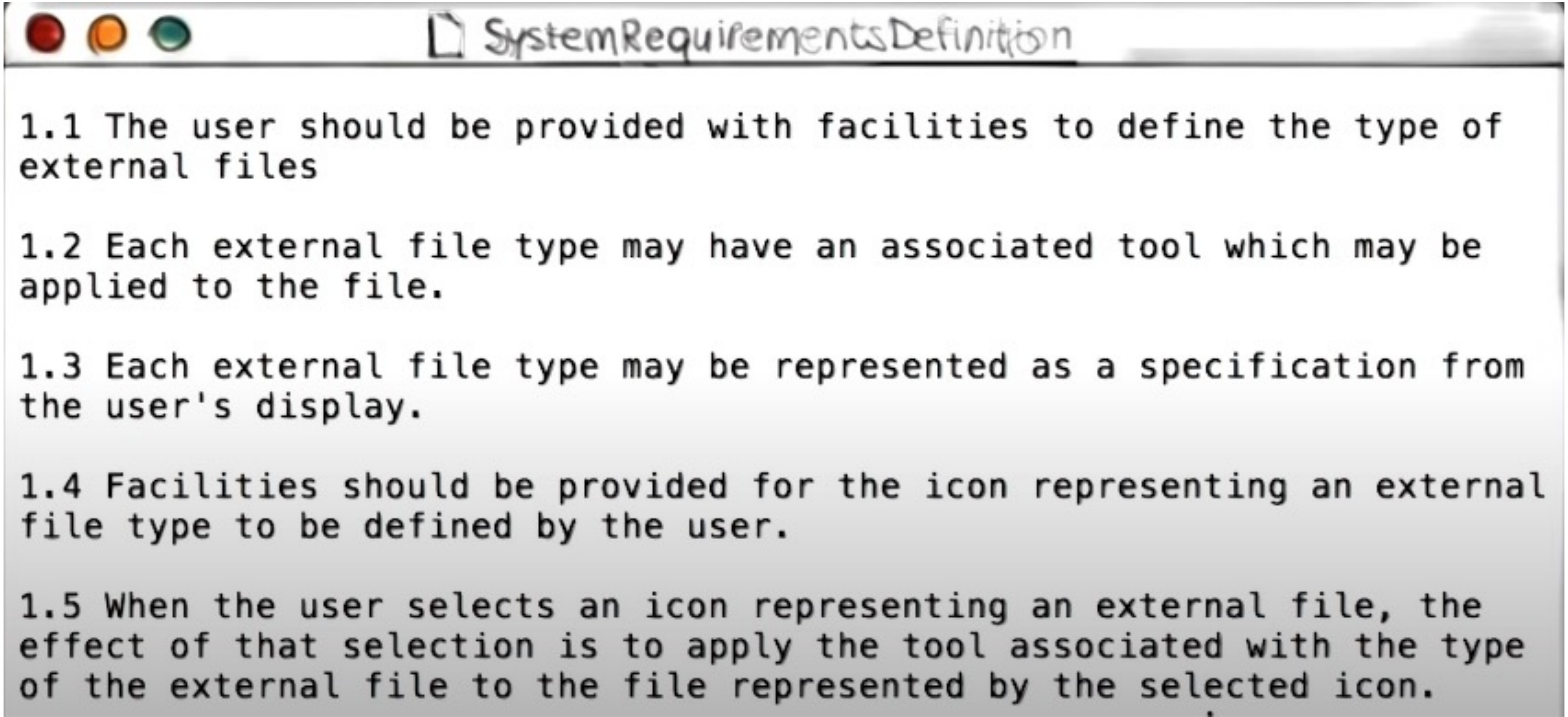
- Written for customers
- Usually written in a natural language
- No technical details

System Requirements

- Audiences: engineers, system architects, testers, etc.
- Clearly and more rigorously specified
- What the machine should do: Input , Output, Interface, Response to events, ...



The software must provide a means of representing and accessing external files created by other tools.



1.1 The user should be provided with facilities to define the type of external files

1.2 Each external file type may have an associated tool which may be applied to the file.

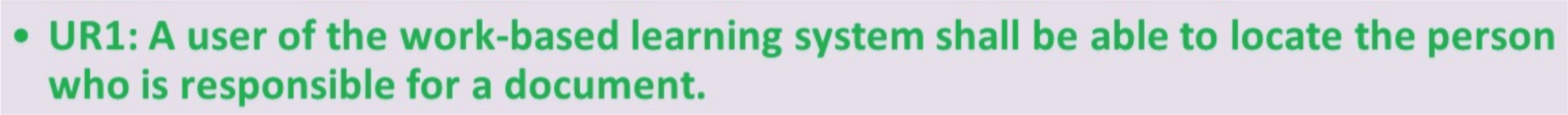
1.3 Each external file type may be represented as a specification from the user's display.

1.4 Facilities should be provided for the icon representing an external file type to be defined by the user.

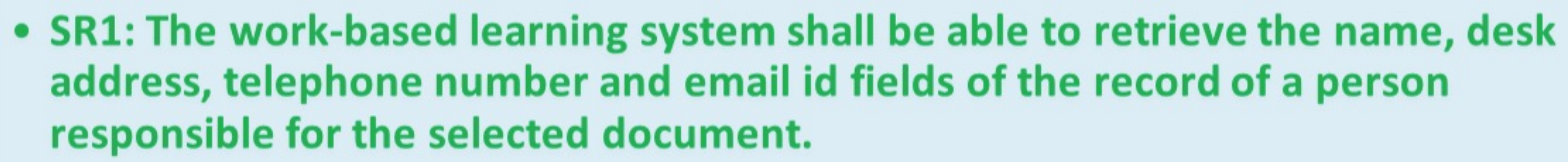
1.5 When the user selects an icon representing an external file, the effect of that selection is to apply the tool associated with the type of the external file to the file represented by the selected icon.

https://www.youtube.com/watch?v=vpNnZDwC_vs

Example of User Req. and System Req.



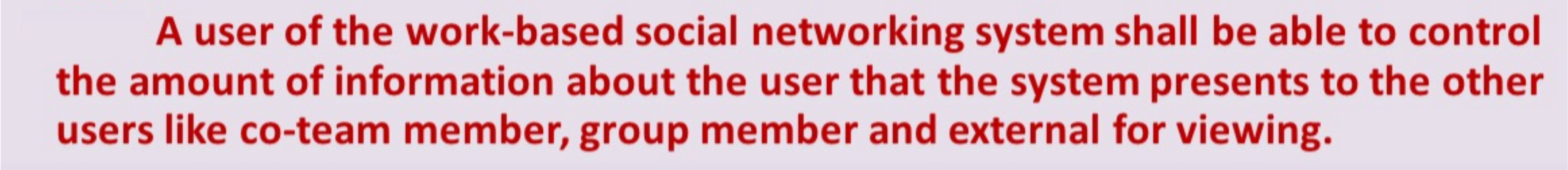
- UR1: A user of the work-based learning system shall be able to locate the person who is responsible for a document.



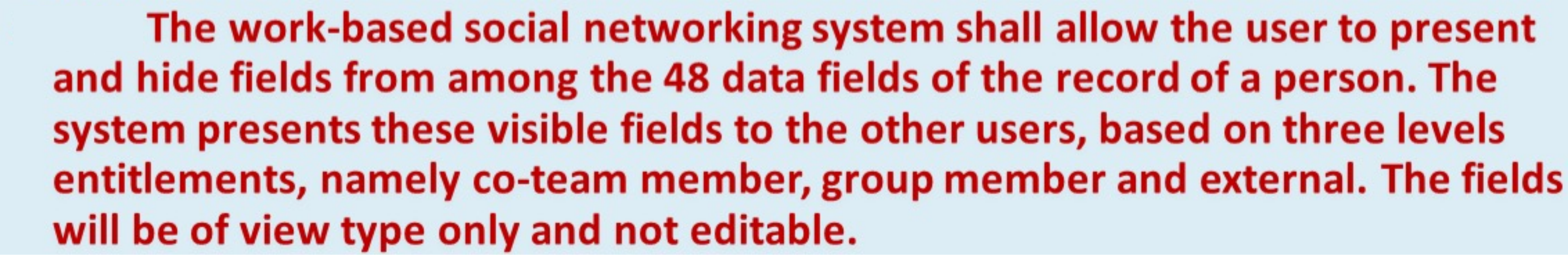
- SR1: The work-based learning system shall be able to retrieve the name, desk address, telephone number and email id fields of the record of a person responsible for the selected document.

https://tech-talk.org/2015/02/06/requirement-analysis-in-software-design/

Which one is UR? Which one is SR?

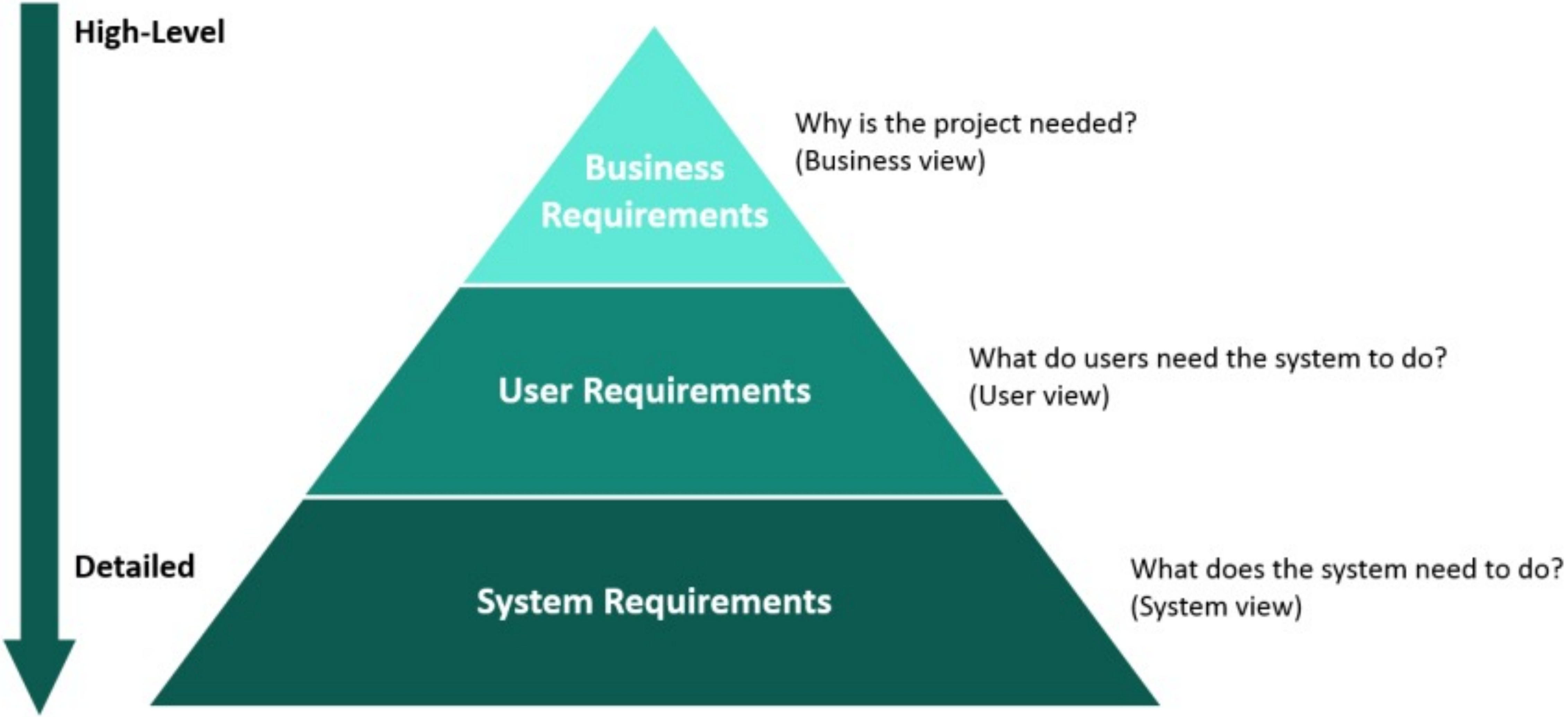


A user of the work-based social networking system shall be able to control the amount of information about the user that the system presents to the other users like co-team member, group member and external for viewing.

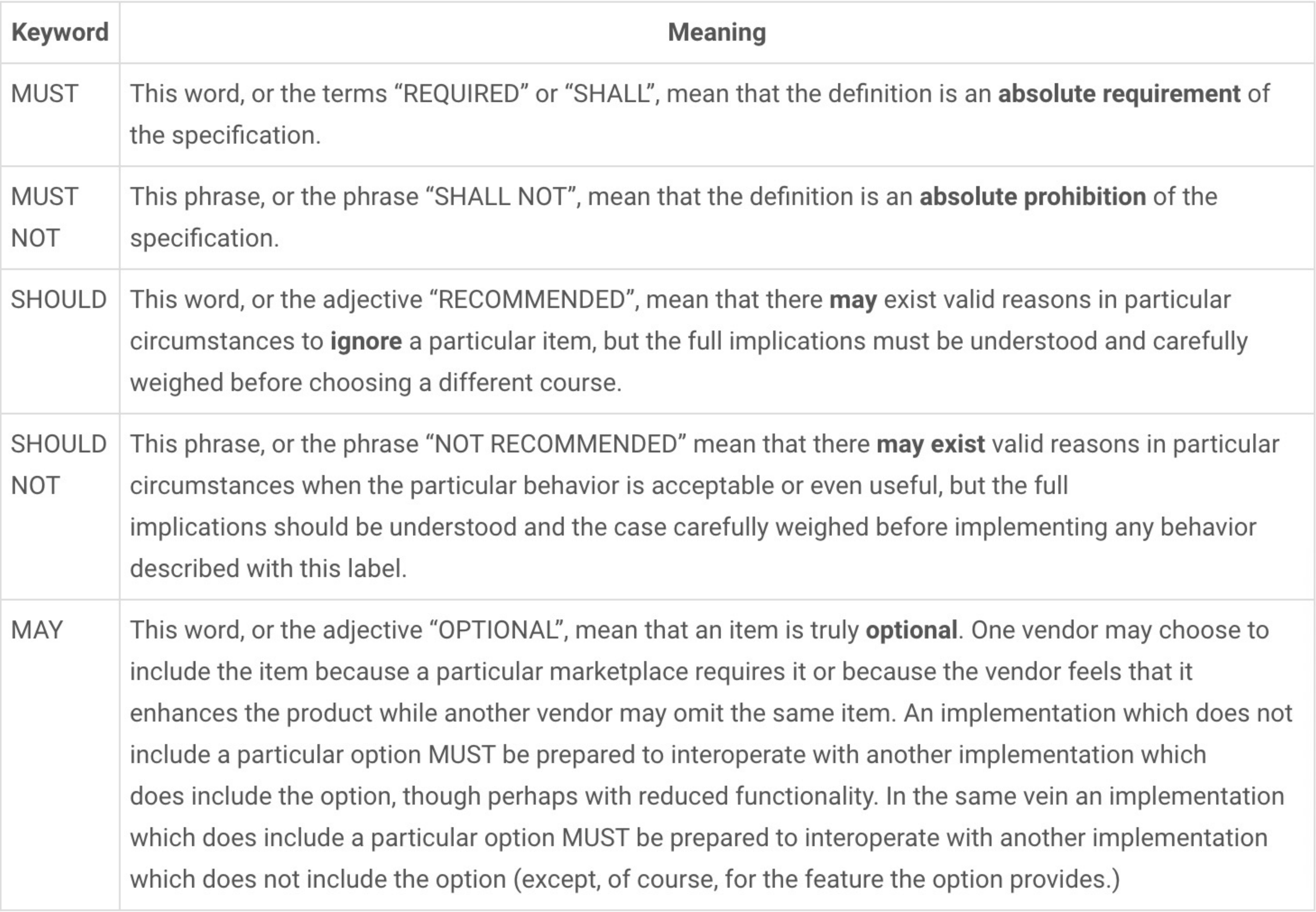


The work-based social networking system shall allow the user to present and hide fields from among the 48 data fields of the record of a person. The system presents these visible fields to the other users, based on three levels entitlements, namely co-team member, group member and external. The fields will be of view type only and not editable.

https://tech-talk.org/2015/02/06/requirement-analysis-in-software-design/



http://www.crvs-dgb.org/en/activities/analysis-and-design/8-define-system-requirements/

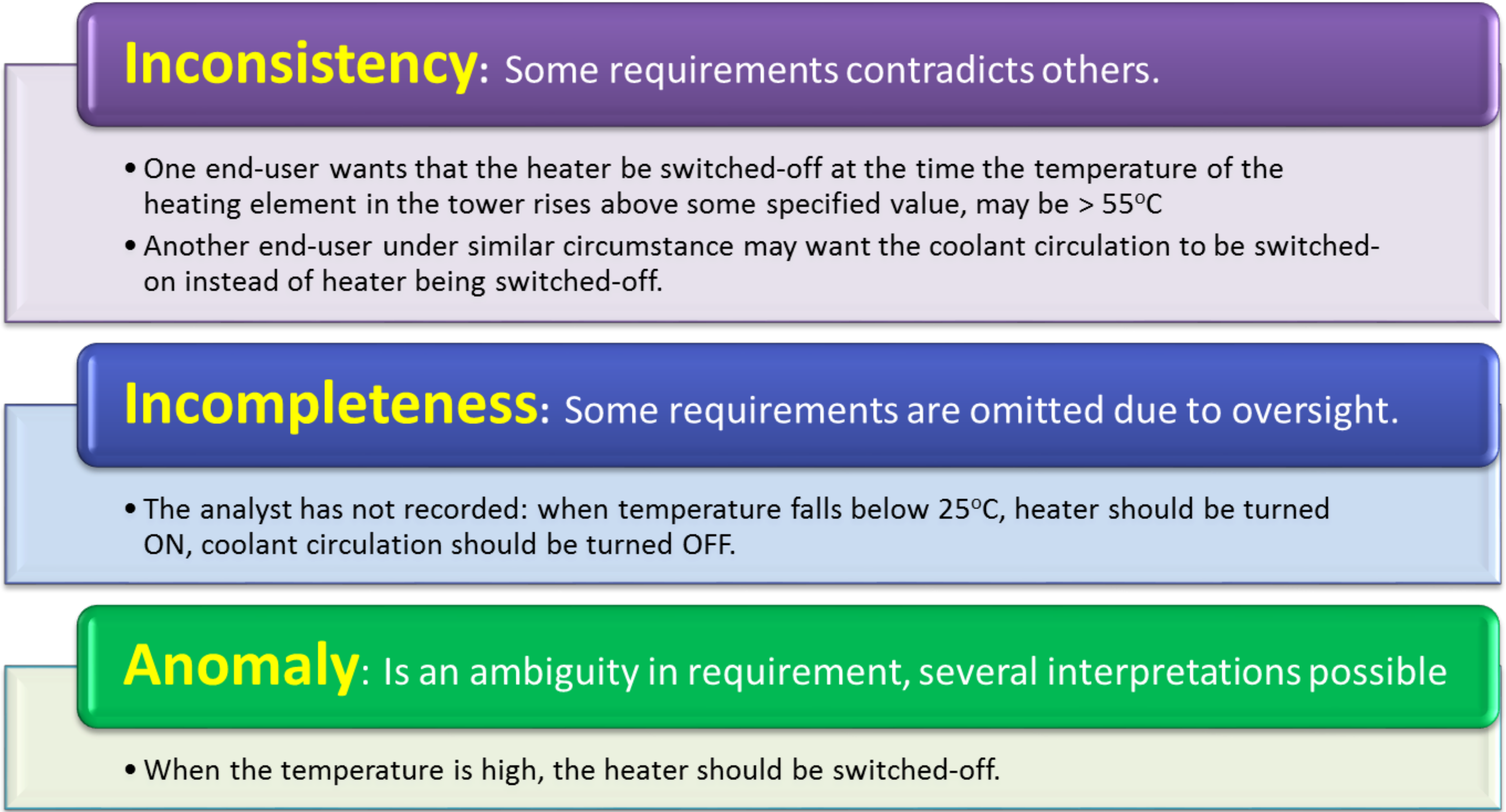


Keyword	Meaning
MUST	This word, or the terms “REQUIRED” or “SHALL”, mean that the definition is an **absolute requirement** of the specification.
MUST NOT	This phrase, or the phrase “SHALL NOT”, mean that the definition is an **absolute prohibition** of the specification.
SHOULD	This word, or the adjective “RECOMMENDED”, mean that there **may** exist valid reasons in particular circumstances to **ignore** a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
SHOULD NOT	This phrase, or the phrase “NOT RECOMMENDED” mean that there **may exist** valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
MAY	This word, or the adjective “OPTIONAL”, mean that an item is truly **optional**. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

- https://www.ietf.org/rfc/rfc2119.txt

https://techtalkdotorg.files.wordpress.com/2015/02/requirements_analysis_focus.png

Criteria



Inconsistency: Some requirements contradicts others.

- One end-user wants that the heater be switched-off at the time the temperature of the heating element in the tower rises above some specified value, may be > 55^{o}C
- Another end-user under similar circumstance may want the coolant circulation to be switched-on instead of heater being switched-off.

Incompleteness: Some requirements are omitted due to oversight.

- The analyst has not recorded: when temperature falls below 25^{o}C, heater should be turned ON, coolant circulation should be turned OFF.

Anomaly: Is an ambiguity in requirement, several interpretations possible

- When the temperature is high, the heater should be switched-off.

Functional requirements and implementation bias

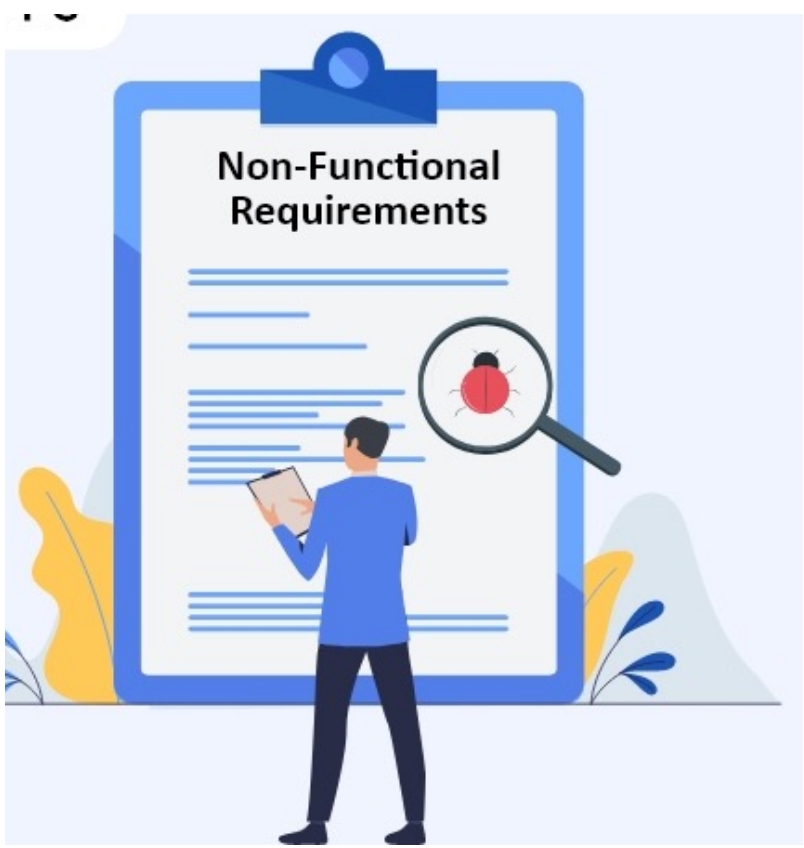
Requirements say what the system will do (**and not how it will do it**).

Why not “how”?

Quality/Non-functional requirements

- Specify not the functionality of the system, but the quality with which it delivers that functionality.

- Can be more critical than functional requirements
  - Can work around missing functionality
  - Low-quality system may be unusable

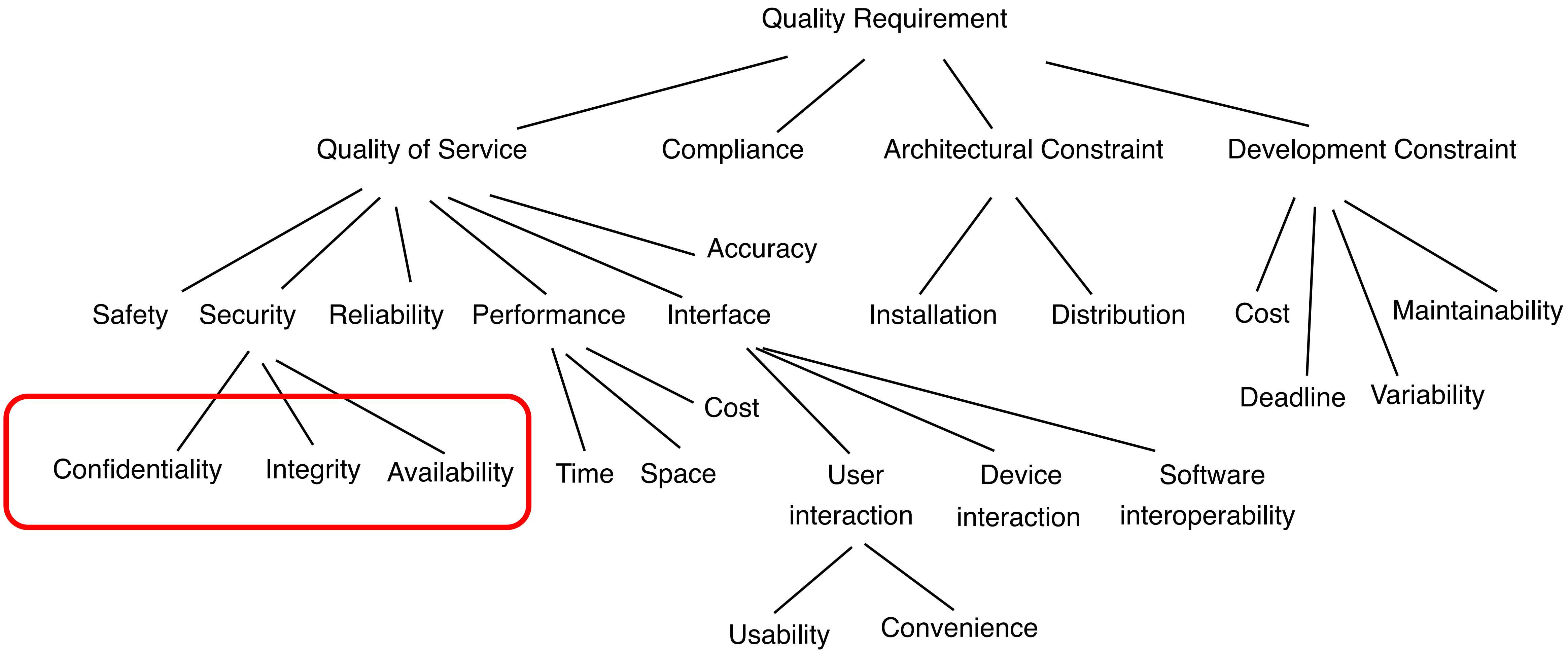


Quality (non-funct.) requirements

- Specify not the functionality of the system, but the quality with which it delivers that functionality.
- Can be more critical than functional requirements
  - Can work around missing functionality
  - Low-quality system may be unusable
- Examples?

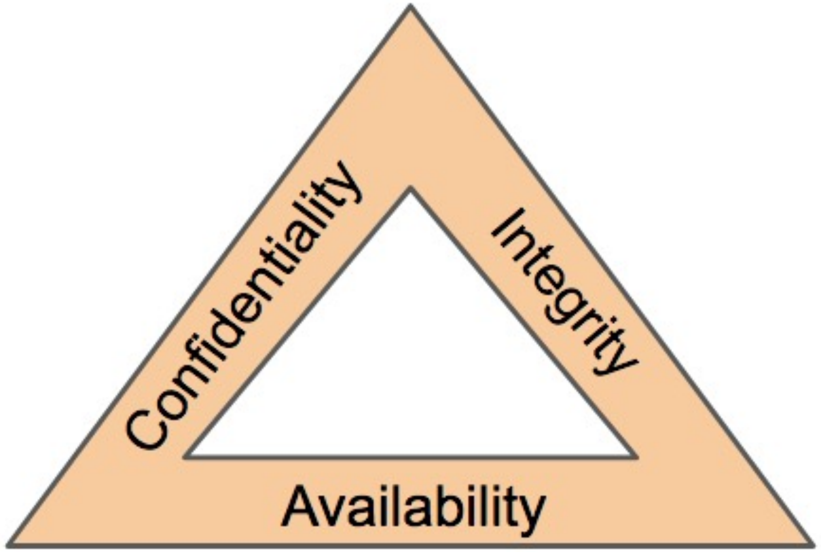
Here’s the thing…

- Who is going to ask for a slow, inefficient, unmaintainable system?
- A better way to think about quality requirements is as *design criteria to help choose between alternative implementations.*
- Question becomes: to what extent must a product satisfy these requirements to be acceptable?


Quality Requirement
Quality of Service
Compliance
Architectural Constraint
Development Constraint
Accuracy
Safety
Security
Reliability
Performance
Interface
Installation
Distribution
Cost
Maintainability
Cost
Deadline
Variability
Confidentiality
Integrity
Availability
Time
Space
User interaction
Device interaction
Software interoperability
Usability
Convenience

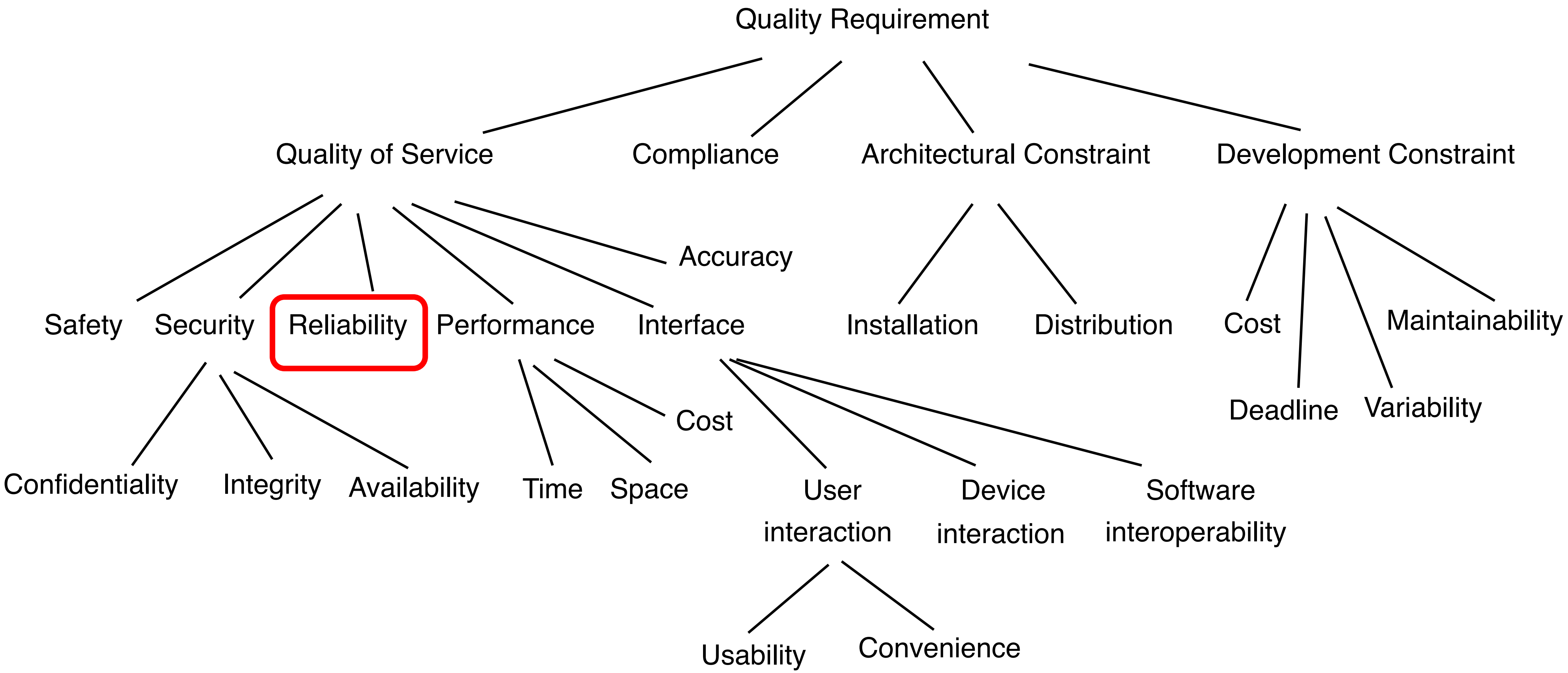
Security

- **Confidentiality** -- data, objects and resources are protected from unauthorized viewing and other access.
- **Integrity** -- data is protected from unauthorized changes to ensure that it is reliable and correct.
- **Availability** -- authorized users have access to the systems and the resources they need.



Security Req (Example)

- **Confidentiality requirement**: A non-staff patron may never know which books have been borrowed by others.
- **Integrity req**: The return of book copies shall be encoded correctly and by library staff only.
- **Availability req**: A blacklist of bad patrons shall be made available at any time to library staff. Information about train positions shall be available at any time to the vital station computer.

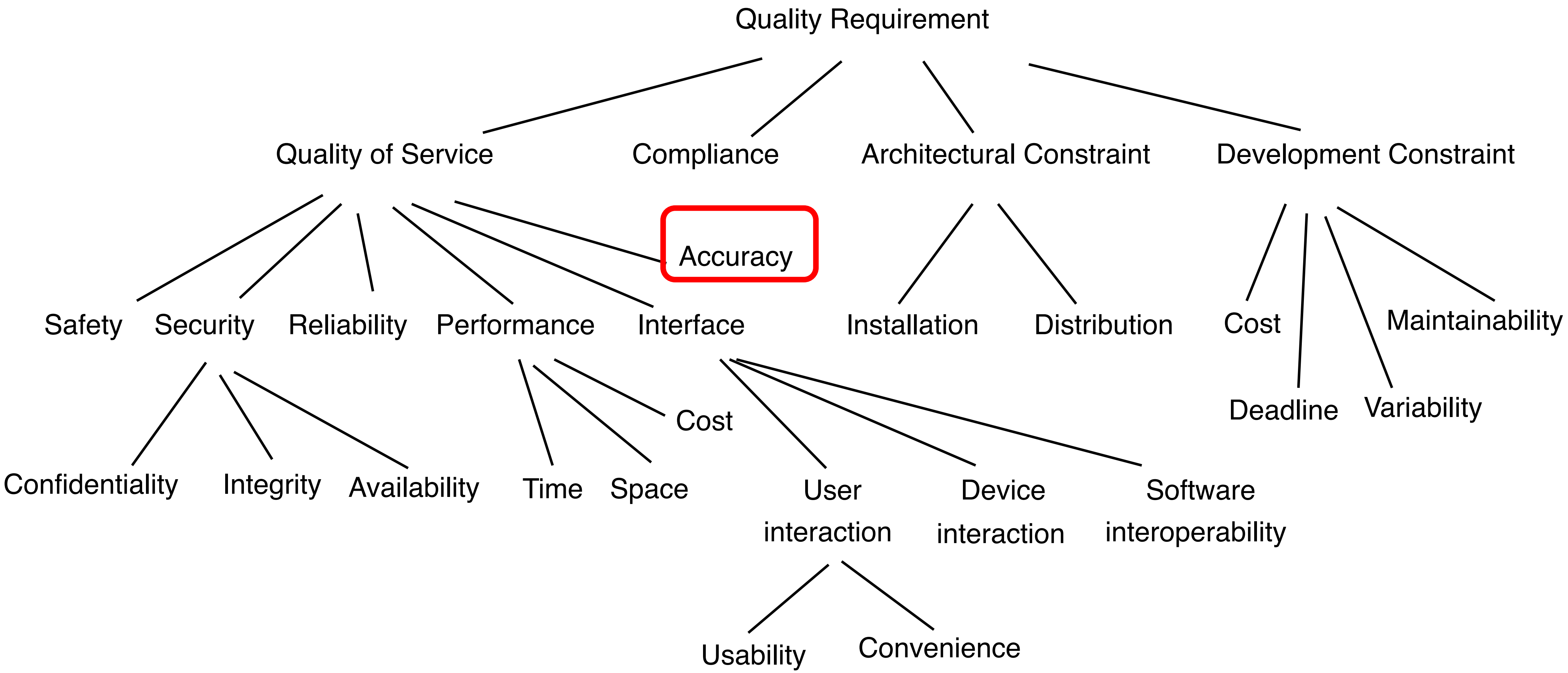

Quality Requirement
Quality of Service
Compliance
Architectural Constraint
Development Constraint
Accuracy
Safety
Security
Reliability
Performance
Interface
Installation
Distribution
Cost
Maintainability
Deadline
Variability
Cost
Confidentiality
Integrity
Availability
Time
Space
User interaction
Device interaction
Software interoperability
Usability
Convenience

Reliability req.

The probability that a product will operate without failure for a specified number of uses (transactions) or for a specified period of time.

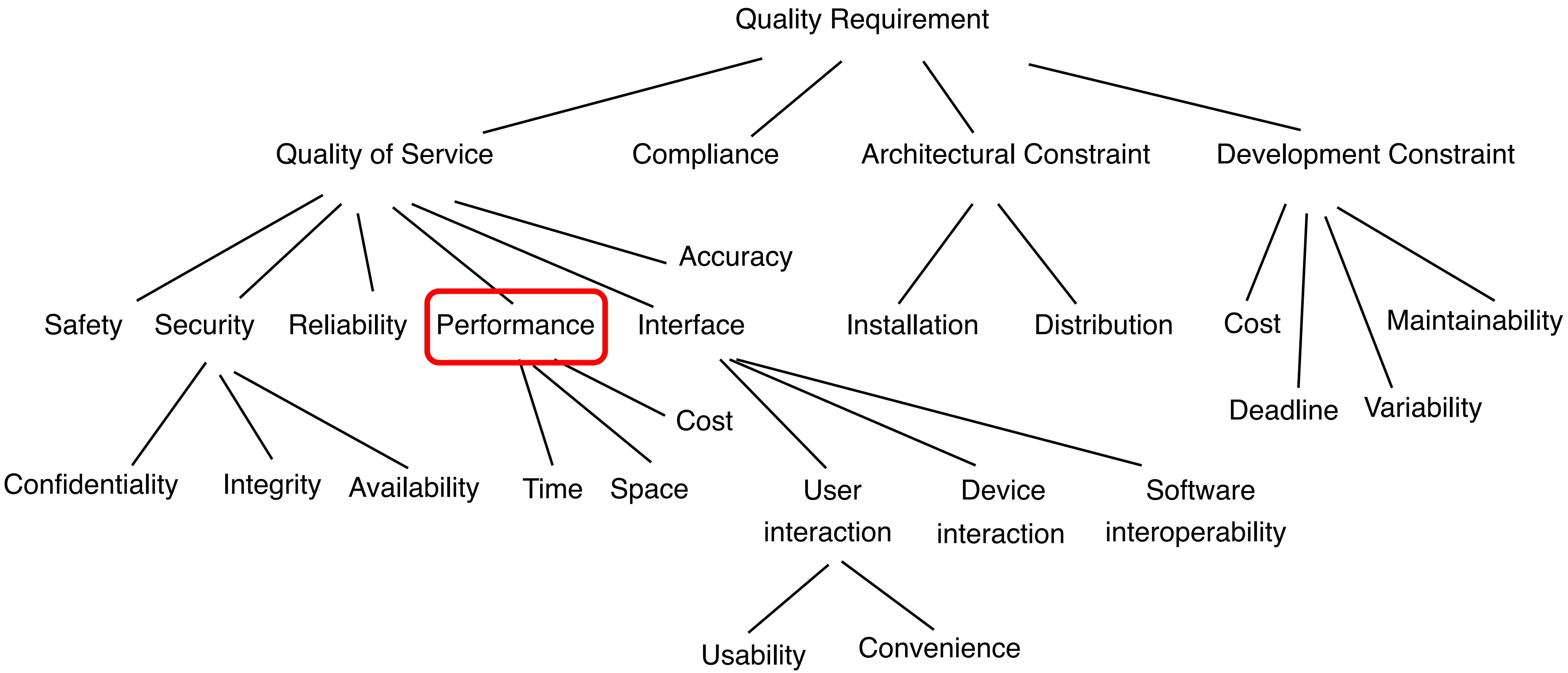
e.g. The train acceleration control software shall have a mean time between failures of the order of 109 hours.

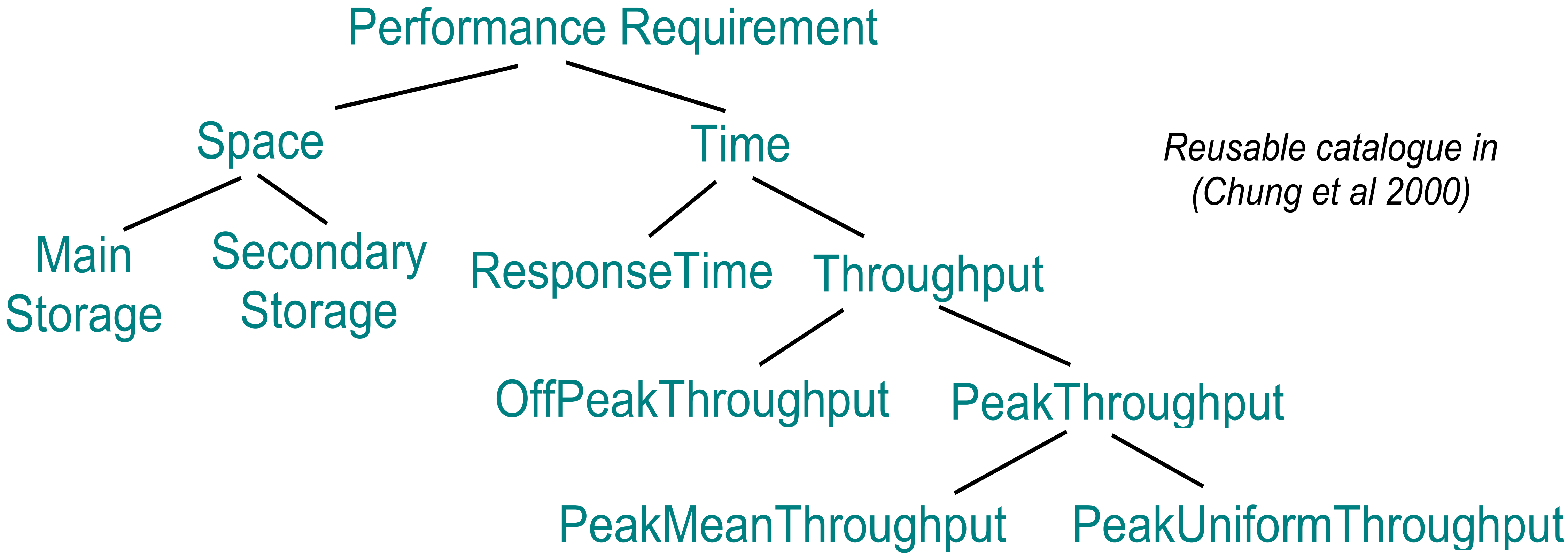
Reliability
/re-ly-a-bi-li-ti/
The ability of the software to function under the given conditions


Quality Requirement
Quality of Service
Compliance
Architectural Constraint
Development Constraint
Accuracy
Safety
Security
Reliability
Performance
Interface
Installation
Distribution
Cost
Maintainability
Deadline
Variability
Cost
Confidentiality
Integrity
Availability
Time
Space
User interaction
Device interaction
Software interoperability
Usability
Convenience

Accuracy req. (Examples)

- The solution needs to be 100% accurate
- A copy of a book shall be stated as available by the loan software if and only if it is actually available on the library shelves.
- The information about train positions used by the train controller shall accurately reflect the actual position of trains up to X meters at most.
- The constraints used by the meeting scheduler should accurately reflect the real constraints of invited participants.


Quality Requirement
Quality of Service
Compliance
Architectural Constraint
Development Constraint
Accuracy
Safety
Security
Reliability
Performance
Interface
Installation
Distribution
Cost
Maintainability
Deadline
Variability
Cost
Confidentiality
Integrity
Availability
Time
Space
User interaction
Device interaction
Software interoperability
Usability
Convenience



Performance Requirement

Space

Time

Main Storage

Secondary Storage

ResponseTime

Throughput

OffPeakThroughput

PeakThroughput

PeakMeanThroughput

PeakUniformThroughput

Reusable catalogue in (Chung et al 2000)

Performance req.

- Def: how well the software system accomplishes certain functions under specific conditions.
- e.g.
  - Responses to bibliographical queries shall take less than 2 seconds.
  - Acceleration commands shall be issued to every train every 3 seconds.
  - The meeting scheduler shall be able to accommodate up to x requests in parallel.
  - The new e-subscription facility should ensure a 30% cost saving.

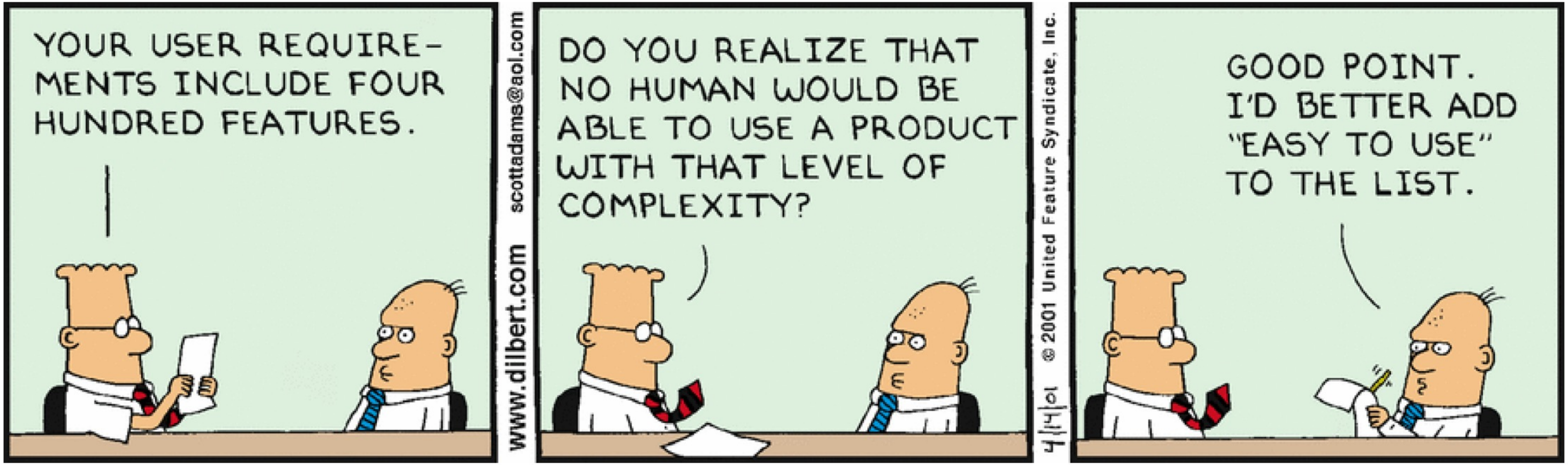
More Examples

- **Interface** req: The format for bibliographical queries and answers shall be accessible to students from any department. To ensure smooth and comfortable train moves, the difference between the accelerations in two successive commands sent to a train should be at most x. To avoid disturbing busy people unduly, the amount of interaction with invited participants for organizing meetings should be kept as low as possible.
- **Interoperability** req: The meeting scheduling software should be interoperable with the wss Agenda Manager product.
- **Compliance** req: The value for the worst-case stopping distance between successive trains shall be compliant with international railways regulations. The meeting scheduler shall by default exclude official holidays associated with the target market.

Examples 4

- Architectural req: The on-board train controllers shall handle the reception and proper execution of acceleration commands sent by the station computer. The meeting scheduling software should run on Windows version X.x and Linux version Y.y.
- **Development** req.: The overall cost of the new UWON library software should not exceed x. The train control software should be operational within two years. The software should provide customized solutions according to variations in type of meeting (professional or private, regular or occasional), type of meeting location (fixed, variable) and type of participant (same or different degrees of importance).

Expressing quality requirements


YOUR USER REQUIREMENTS INCLUDE FOUR HUNDRED FEATURES.
DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?
GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.
scottadams@aol.com
www.dilbert.com
© 2001 United Feature Syndicate, Inc.
4/14/01

Expressing quality requirements

- Requirements serve as contracts: they should be **testable/falsifiable**.
- **Informal** goal: a general intention, such as ease of use.
  - May still be helpful to developers as they convey the intentions of the system users.
- **Verifiable** non-functional requirement: A statement using some measure that can be objectively tested.

Examples

- **Informal goal: "**the system should be easy to use by experienced controllers, and should be organized such that user errors are minimized."
- **Verifiable non-functional requirement: "**Experienced controllers shall be able to use all the system functions after a total of two hours training. After this training, the average number of errors made by experienced users shall not exceed two per day, on average."

ECE444: Software Engineering

Requirements 2: Requirements Elicitation

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Learning Goals

- Basic proficiency in executing effective requirements interviews
- Understand that requirements are just “design data”, the information you will use to support your design
- Understand what/why/how about personas
- Recognize and resolve conflicts with priorities

Requirements Elicitation

Typical Steps

- Identify stakeholders
- Understand the domain
  - Analyze artifacts, interact with stakeholders
- Discover the real needs
  - Interview stakeholders
- Explore alternatives to address needs

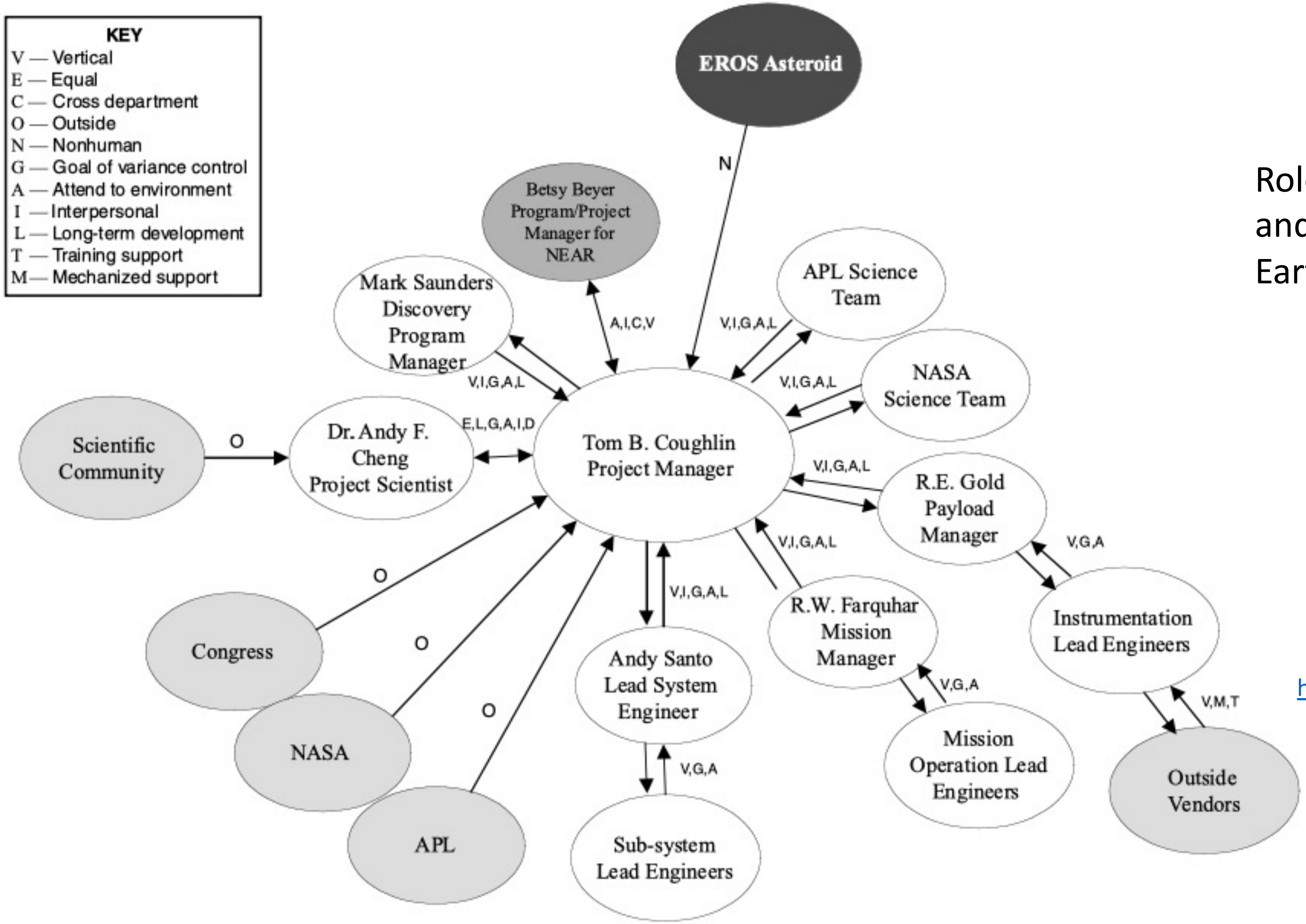
Questions

- Who is the system for?
- Stakeholders:
  - End users
  - System administrators
  - Engineers maintaining the system
  - Business managers
  - …who else?

Stakeholder

- Any person or group who will be affected by the system, directly or indirectly.
- Stakeholders may disagree.
- Requirements process should trigger negotiation to resolve conflicts.

Stakeholders, a NASA example



KEY
V — Vertical
E — Equal
C — Cross department
O — Outside
N — Nonhuman
G — Goal of variance control
A — Attend to environment
I — Interpersonal
L — Long-term development
T — Training support
M — Mechanized support

EROS Asteroid
N
Betsy Beyer Program/Project Manager for NEAR
A,I,C,V
Mark Saunders Discovery Program Manager
V,I,G,A,L
APL Science Team
V,I,G,A,L
NASA Science Team
V,I,G,A,L
Scientific Community
O
Dr. Andy F. Cheng Project Scientist
E,L,G,A,I,D
Tom B. Coughlin Project Manager
V,I,G,A,L
R.E. Gold Payload Manager
V,G,A
Instrumentation Lead Engineers
V,M,T
Outside Vendors
V,I,G,A,L
R.W. Farquhar Mission Manager
V,G,A
Mission Operation Lead Engineers
V,I,G,A,L
Andy Santo Lead System Engineer
V,G,A
Sub-system Lead Engineers
O
Congress
O
NASA
O
APL

Role network for National Aeronautics and Space Administration (NASA's) Near Earth Asteroid Rendezvous project.

https://web.ssu.ac.ir/Dorsapax/userfiles/Sub55/849.pdf

Stakeholder analysis: criteria for identifying relevant stakeholders

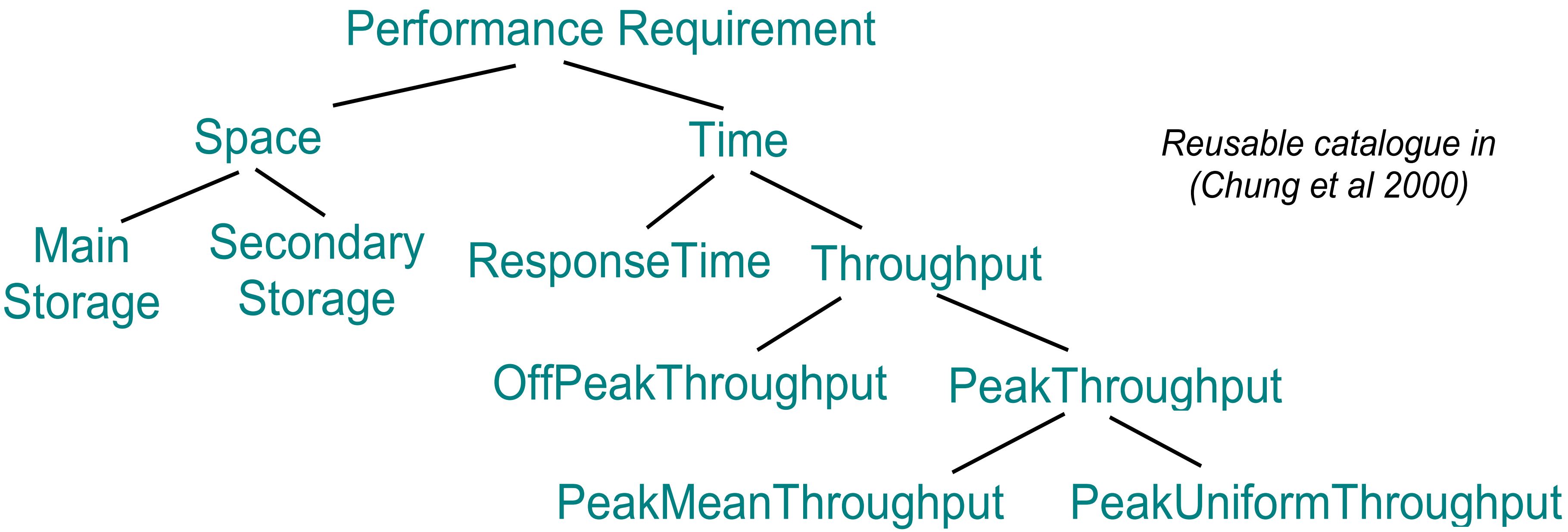
- Relevant positions in the organization
- Effective role in making decisions about the system
- Level of domain expertise
- Exposure to perceived problems
- Influence in system acceptance
- Personal objectives and conflicts of interest

Studying Artifacts (Content Analysis)

- Learn about the domain
  - Books, articles, wikipedia
- Learn about the system to be replaced
  - How does it work? What are the problems? Manuals? Bug reports?
- Learn about the organization
- Knowledge reuse from other systems?

Checklists (Domain-independent knowledge)

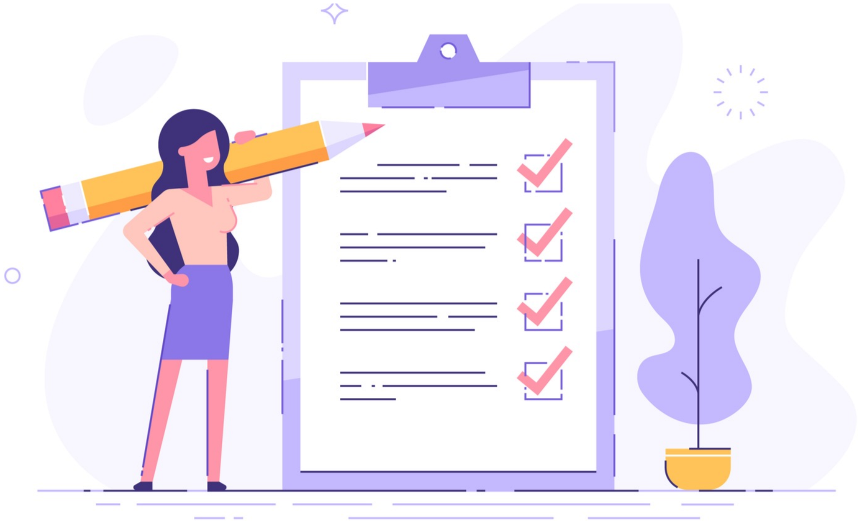
- Consider list of qualities for relevance, e.g. privacy, security, reliability, …



Reusable catalogue in (Chung et al 2000)

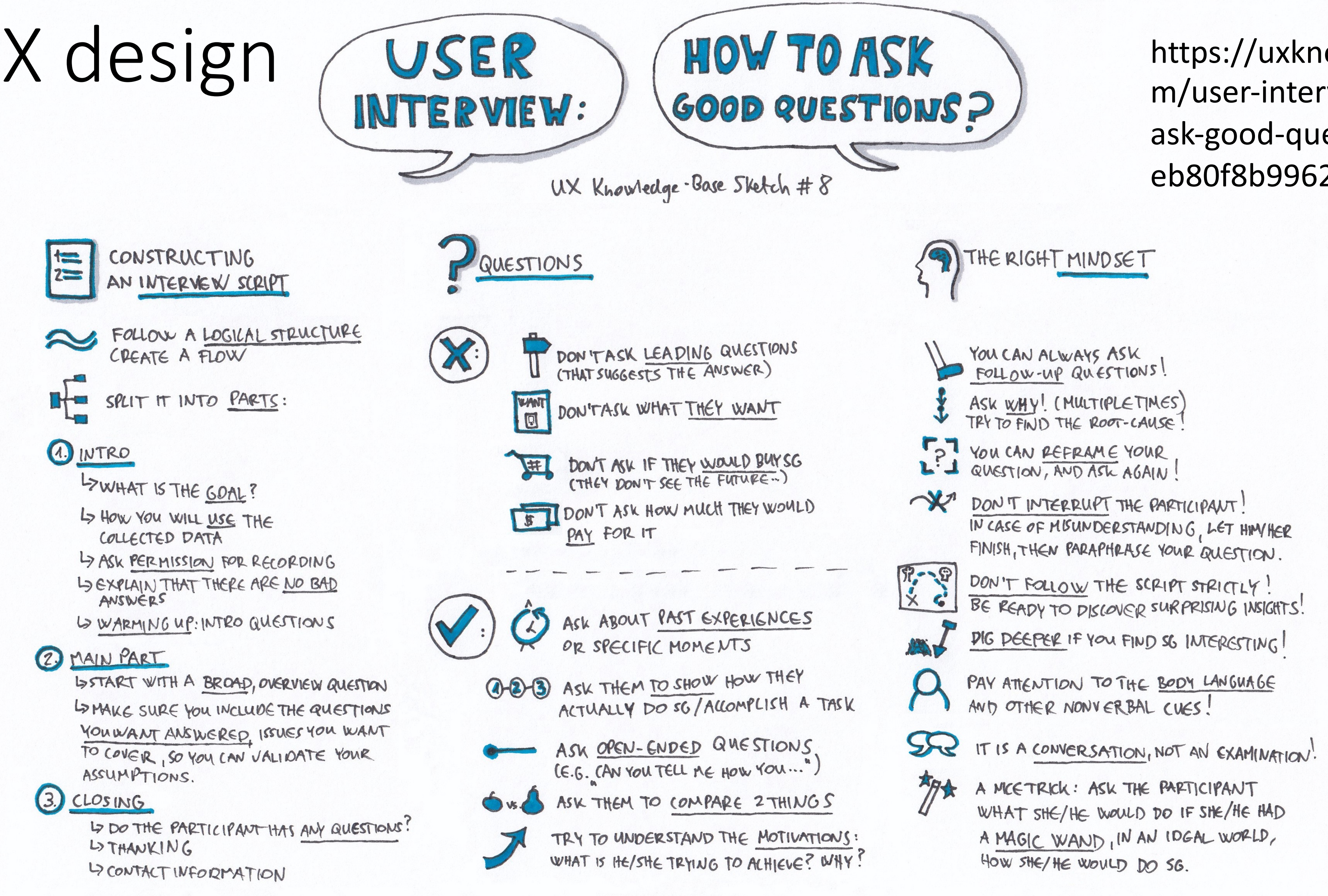
Collecting requirements: Elicit from stakeholders

- **Survey**: measure topics of interest in a controlled, consistent manner; easy to administer across large groups
  - Identify target population, their attitudes and preferences
  - Validate assumptions or facts
- **Interview:** More expensive, but could have follow-up questions to resolve ambiguity



UX design

https://uxknowledgebase.com/user-interview-how-to-ask-good-questions-eb80f8b99627



USER INTERVIEW: HOW TO ASK GOOD QUESTIONS?

UX Knowledge-Base Sketch # 8

CONSTRUCTING AN INTERVIEW SCRIPT

FOLLOW A LOGICAL STRUCTURE
CREATE A FLOW

SPLIT IT INTO PARTS:

1. INTRO
   - ↳ WHAT IS THE GOAL?
   - ↳ HOW YOU WILL USE THE COLLECTED DATA
   - ↳ ASK PERMISSION FOR RECORDING
   - ↳ EXPLAIN THAT THERE ARE NO BAD ANSWERS
   - ↳ WARMING UP: INTRO QUESTIONS
2. MAIN PART
   - ↳ START WITH A BROAD, OVERVIEW QUESTION
   - ↳ MAKE SURE YOU INCLUDE THE QUESTIONS YOU WANT ANSWERED, ISSUES YOU WANT TO COVER, SO YOU CAN VALIDATE YOUR ASSUMPTIONS.
3. CLOSING
   - ↳ DO THE PARTICIPANT HAS ANY QUESTIONS?
   - ↳ THANKING
   - ↳ CONTACT INFORMATION

QUESTIONS

✗:

DON'T ASK LEADING QUESTIONS
(THAT SUGGESTS THE ANSWER)

DON'T ASK WHAT THEY WANT

DON'T ASK IF THEY WOULD BUY SG
(THEY DON'T SEE THE FUTURE...)

DON'T ASK HOW MUCH THEY WOULD PAY FOR IT

✓:

ASK ABOUT PAST EXPERIENCES OR SPECIFIC MOMENTS

ASK THEM TO SHOW HOW THEY ACTUALLY DO SG/ACCOMPLISH A TASK

ASK OPEN-ENDED QUESTIONS
(E.G. "CAN YOU TELL ME HOW YOU...")

ASK THEM TO COMPARE 2 THINGS

TRY TO UNDERSTAND THE MOTIVATIONS:
WHAT IS HE/SHE TRYING TO ACHIEVE? WHY?

THE RIGHT MINDSET

YOU CAN ALWAYS ASK FOLLOW-UP QUESTIONS!

ASK WHY! (MULTIPLE TIMES)
TRY TO FIND THE ROOT-CAUSE!

YOU CAN REFRAME YOUR QUESTION, AND ASK AGAIN!

DON'T INTERRUPT THE PARTICIPANT!
IN CASE OF MISUNDERSTANDING, LET HIM/HER FINISH, THEN PARAPHRASE YOUR QUESTION.

DON'T FOLLOW THE SCRIPT STRICTLY!
BE READY TO DISCOVER SURPRISING INSIGHTS!

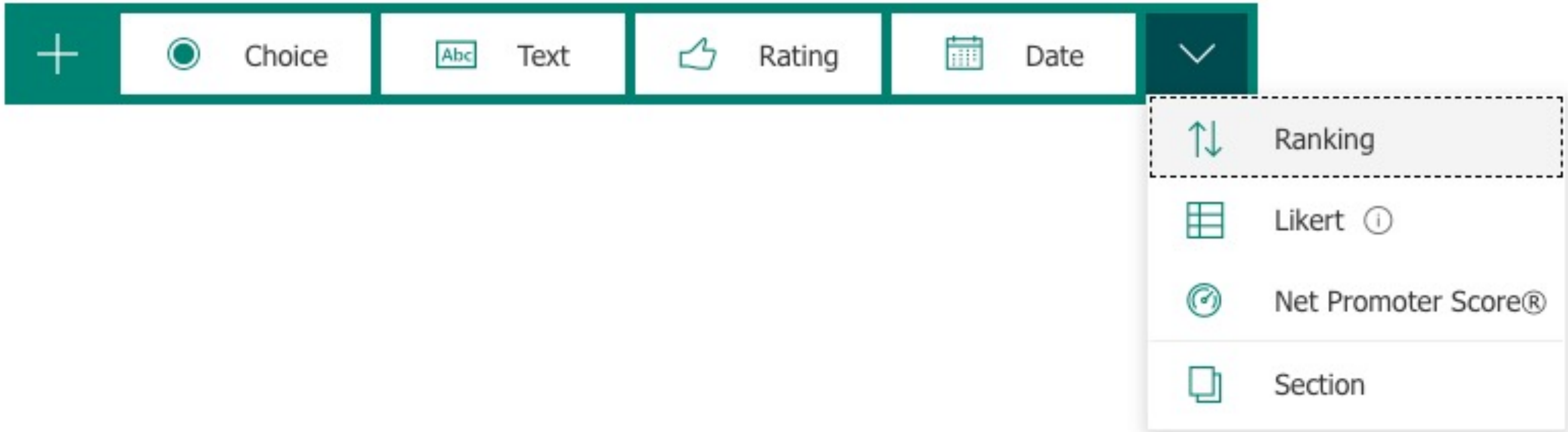
DIG DEEPER IF YOU FIND SG INTERESTING!

PAY ATTENTION TO THE BODY LANGUAGE AND OTHER NONVERBAL CUES!

IT IS A CONVERSATION, NOT AN EXAMINATION!

A NICE TRICK: ASK THE PARTICIPANT WHAT SHE/HE WOULD DO IF SHE/HE HAD A MAGIC WAND, IN AN IDEAL WORLD, HOW SHE/HE WOULD DO SG.

Types of questions: depend on your goals



Closed-ended Questions

- **Nominal scales** provide interviewees with a list of categories from which to select their answer (e.g., White, Black or African American, American Indian, Asian, Native Hawaiian or Pacific Islander)
- Good practices –
  - Solicit response options in a pilot study
  - Randomize order, if concerned about order effects
  - Avoid bias from unequal response options
  - Check all that apply vs. forced-choice

Example: Unequal response options

How likely are you to share your location to meet friends after work?

- Absolutely never
- Sometimes
- Occasionally
- Once or more a week
- Everyday

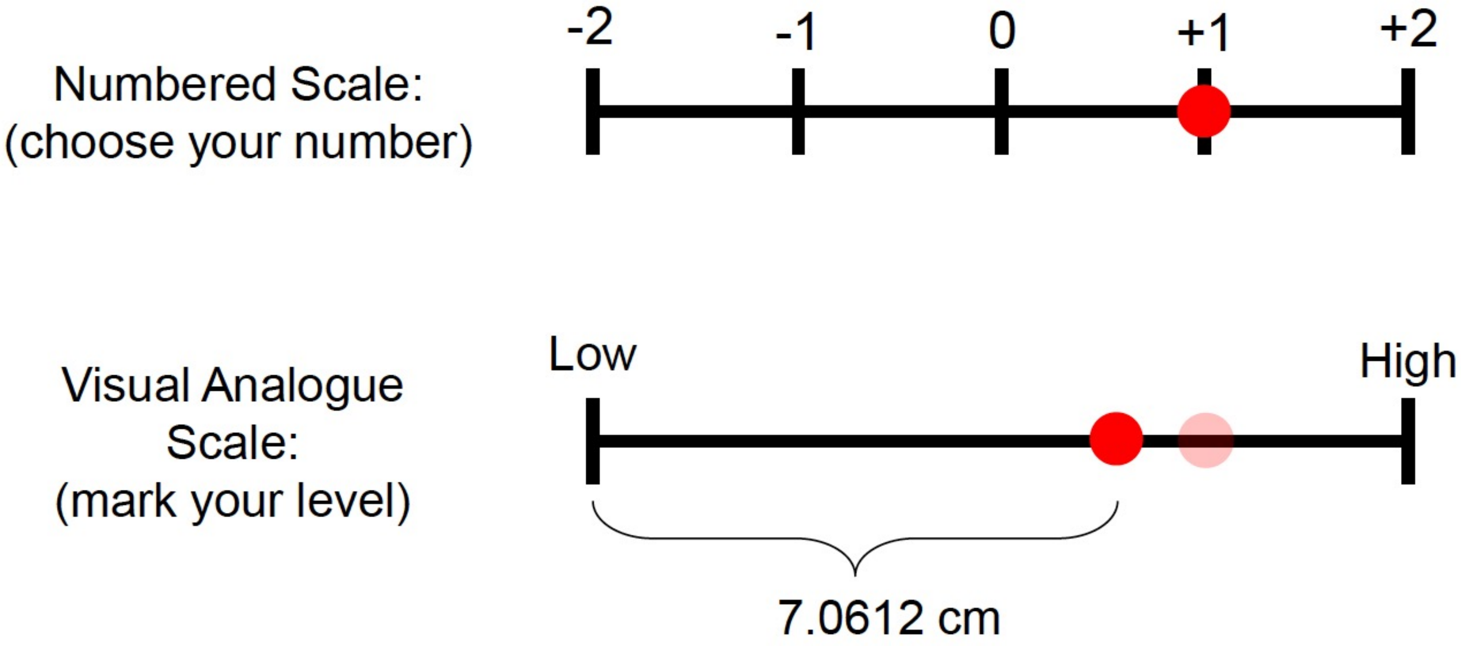
Is it easy or difficult to distinguish between these three categories?

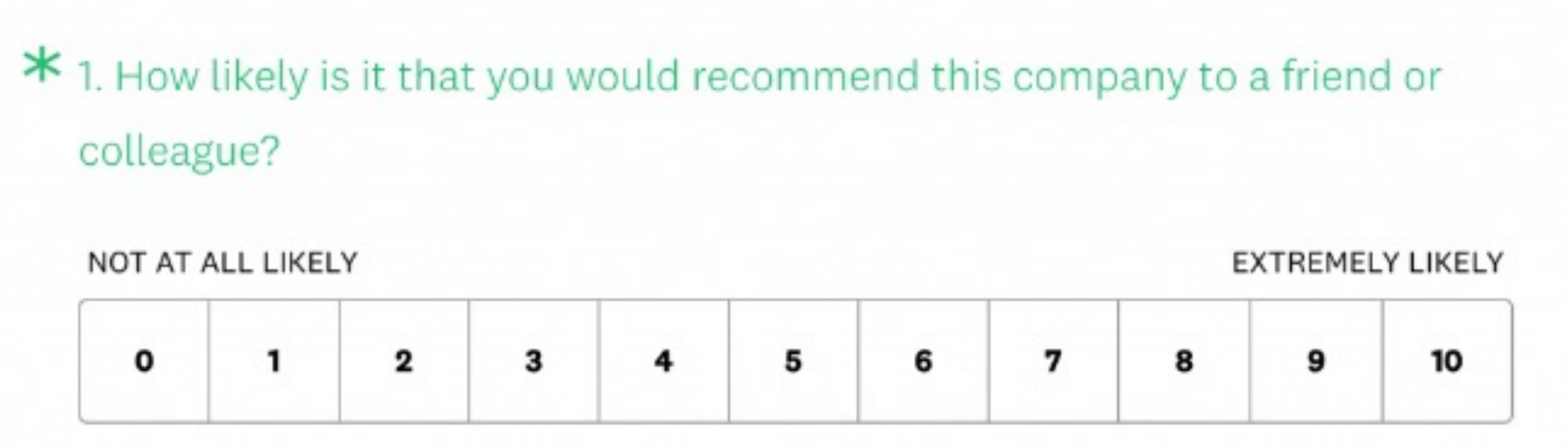
If difficult, why?

Cowley, Youngblood. “Subjective response differences between visual analogue, ordinal and hybrid response scales,” *Human Factors and Ergonomics Society Annual Meeting*, 53(25): 1883-1887, 2009.

Ordinal Scales

- Ordinal or interval scales ask interviewees to choose a “level” of the variable of interest

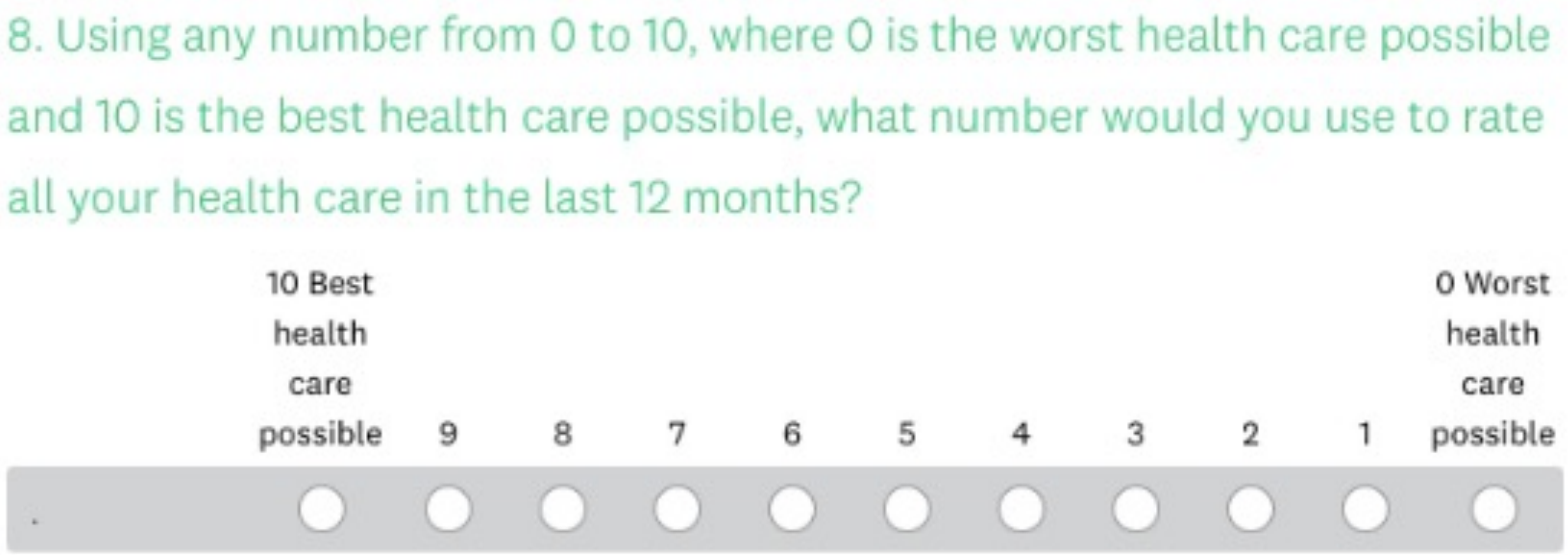




* 1. How likely is it that you would recommend this company to a friend or colleague?

NOT AT ALL LIKELY   EXTREMELY LIKELY

0	1	2	3	4	5	6	7	8	9	10



8. Using any number from 0 to 10, where 0 is the worst health care possible and 10 is the best health care possible, what number would you use to rate all your health care in the last 12 months?

10 Best health care possible	9	8	7	6	5	4	3	2	1	0 Worst health care possible
○	○	○	○	○	○	○	○	○	○	○

Open-ended Questions

- **Definition and designation questions**

  What-is asks to develop definitions of things

  Who identifies the responsible agent

  What-kinds-of ask for possible types and exemplars

- **Process, event and exception questions**

  How-to ask how an action is performed

  When asks about timing constraints, pre-and post-conditions

  What-if asks about failures or unexpected events

  Follow-on questions result from answers from previous questions

Follow-up questions

Do you mean in general?

Can you recall a specific example?

Did you participate in this example?

Do you remember any events before or after?

What time of day was it?

Who was present?

What happened next?

Survey

- https://www.surveymonkey.com/mp/how-to-use-an-interval-scale-in-your-survey-questions/

Survey Organization & Execution

- Begin with salient questions that respondents can easily answer
- Group questions by topic
- Keep in mind ordering effects and biases

  Acquiescence: the tendency to agree

  Social desirability: the need to present oneself in a desirable light
- During open-ended responses in interviews:
  - Jot down "sign posts" and "way points" in your notes to guide the conversation back to important points
  - Limit tangents and distractions, but be willing to explore unexpected ideas
- Limit interviews and surveys to 30-45 minutes
- Pilot the survey on a friend or colleague!

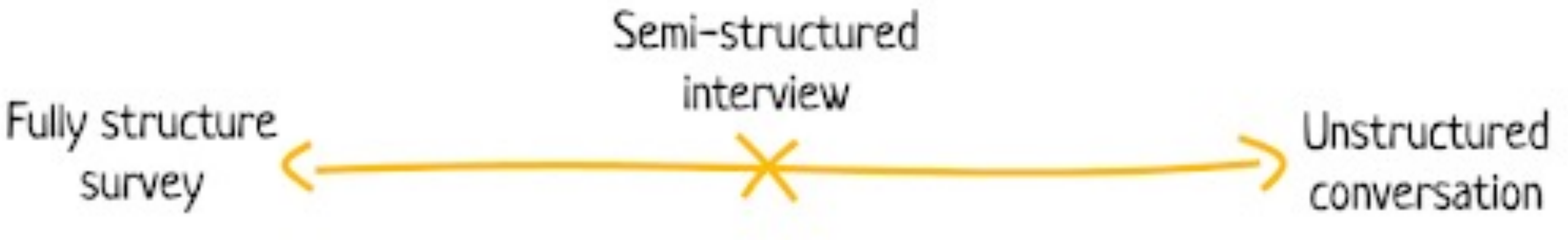
Interview

Interviews

- (Most) common method of data gathering in qualitative research
- A variety of forms of qualitative research interview

(and assumptions that underlie their use)

- Types of interviews:
  - ‘semi-structured’ -- list of questions (open-ended and closed-ended) or topics
  - ‘un-structured’ -- list of prompts



Interview Process

- Identify stakeholder of interest and target information to be gathered.
- Conduct interview.
  - (structured/unstructured, individual/group)
- Record + transcribe interview
- Report important findings.
- Check validity of report with interviewee.

Example: Identifying Problems

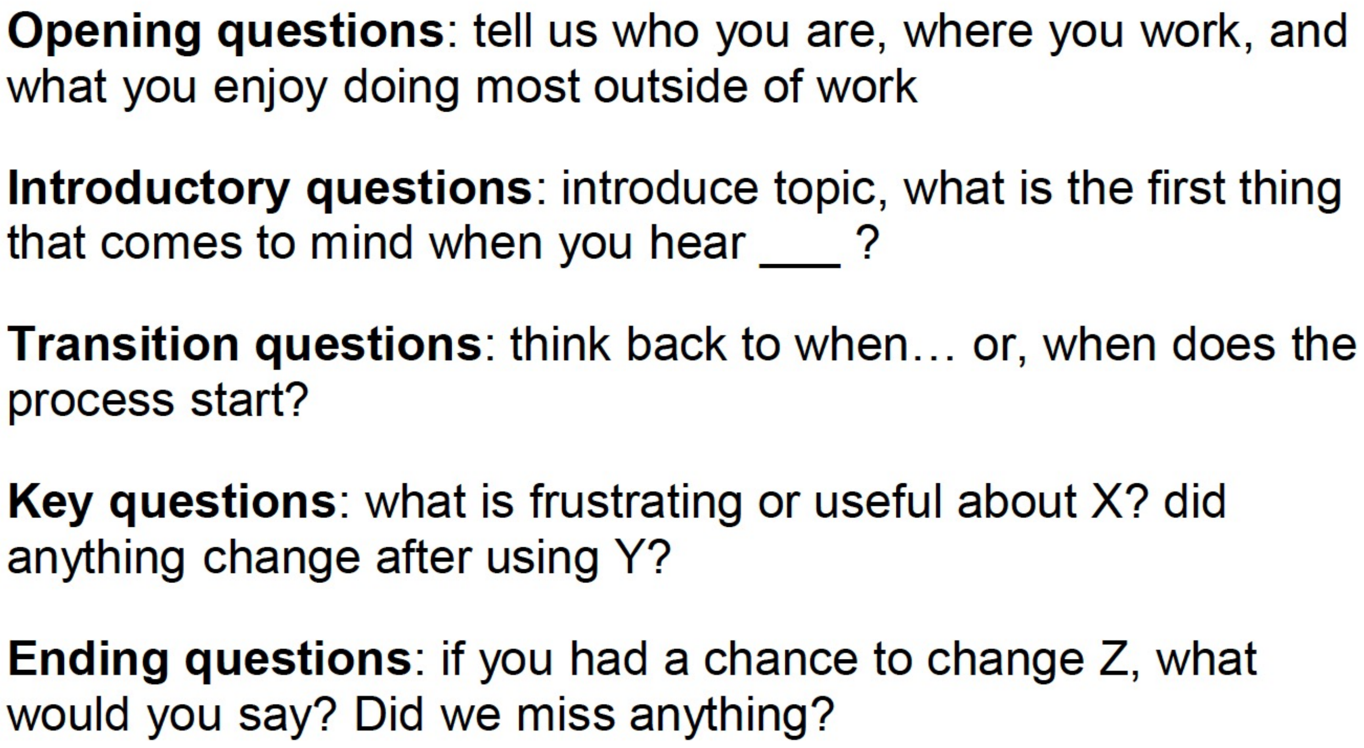
- What problems do you run into in your day-to-day work? Is there a standard way of solving it, or do you have a workaround?
    - Why is this a problem? How do you solve the problem today? How would you ideally like to solve the problem?
- Keep asking follow-up questions ("What else is a problem for you?", "Are there other things that give you trouble?") for as long as the interviewee has more problems to describe.

- So, as I understand it, you are experiencing the following problems/needs (describe the interviewee's problems and needs in your own words – often you will discover that you do not share the same image. It is very very common to not understand each other even if at first you think you do).
- Just to confirm, have I correctly understood the problems you have with the current solution?
- Are there any other problems you're experiencing? If so, what are they?

Example Questions:
The User Environment

- Who will be the users of the system?
- What level of education or training do the users have?
- What computer skills do the users have?
- Are users familiar with this type of IT system?
- What technical platforms do they use today?
- Do you know of any plans for future systems or platforms?
- What other IT systems does the organization use today that the new system will need to link to?
- What are your expectations regarding system usability?
- What training needs do you expect for the future system?
- What kind of documentation do you expect?

Kinds of questions



Opening questions: tell us who you are, where you work, and what you enjoy doing most outside of work

Introductory questions: introduce topic, what is the first thing that comes to mind when you hear ___ ?

Transition questions: think back to when… or, when does the process start?

Key questions: what is frustrating or useful about X? did anything change after using Y?

Ending questions: if you had a chance to change Z, what would you say? Did we miss anything?

Interview Advice

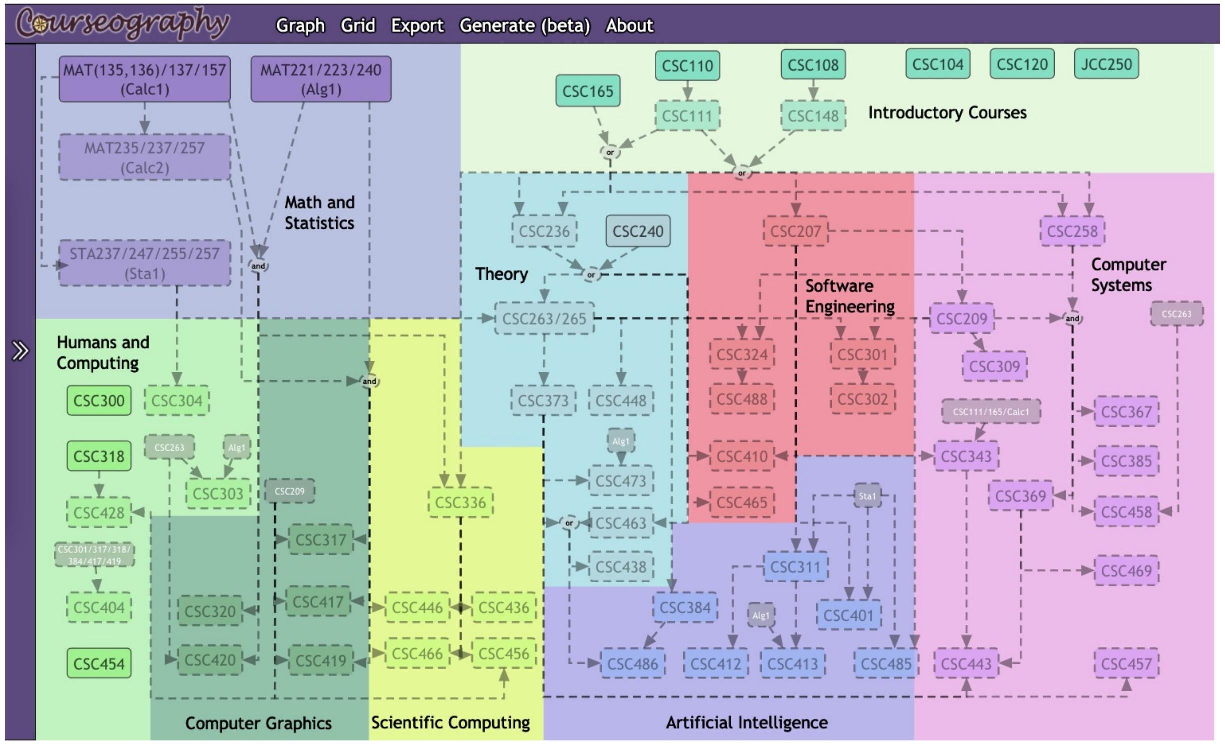
- Get basic facts about the interviewee before (role, responsibilities, …)
- Review interview questions before interview
- Begin concretely with specific questions, proposals; work through prototype or scenario
  - Relate to current system, if applicable.
- Be open-minded; explore additional issues that arise naturally, but stay focused on the system.
- Contrast with current system/alternatives. Explore conflicts and priorities
- Plan for follow-up questions

Another two examples (Requirements for Chef Copilot)

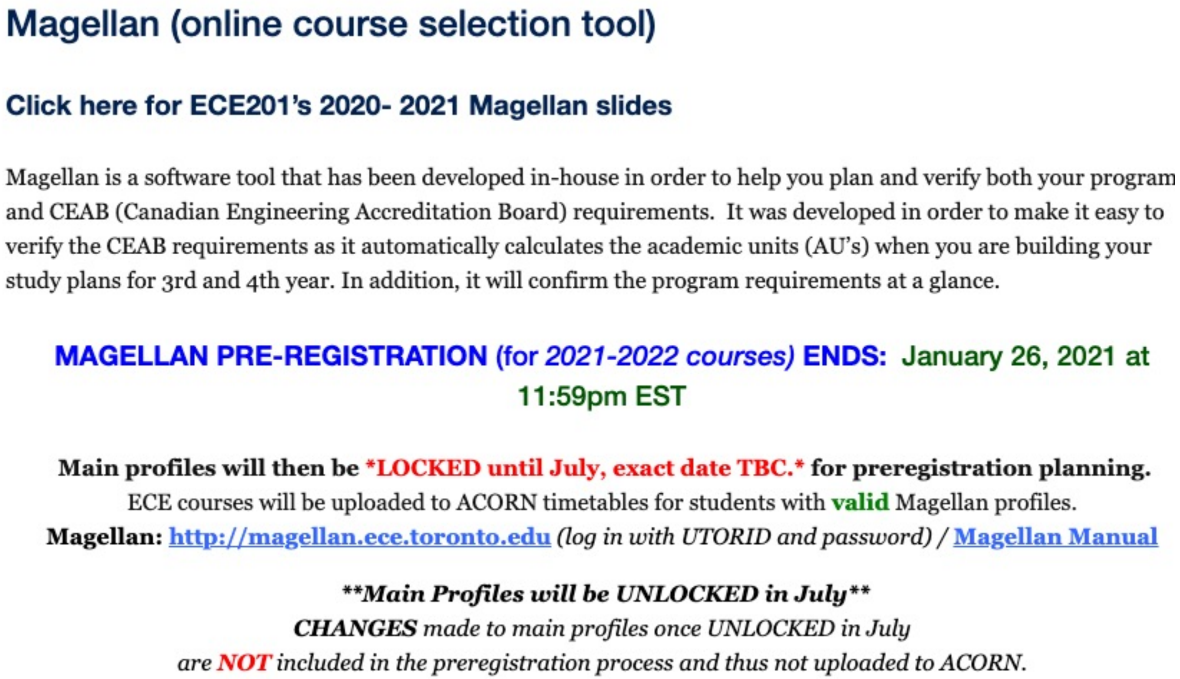
Which one is Better?

Exercise: Project 1

Education Pathways



http://courseography.cdf.toronto.edu/graph



Magellan (online course selection tool)

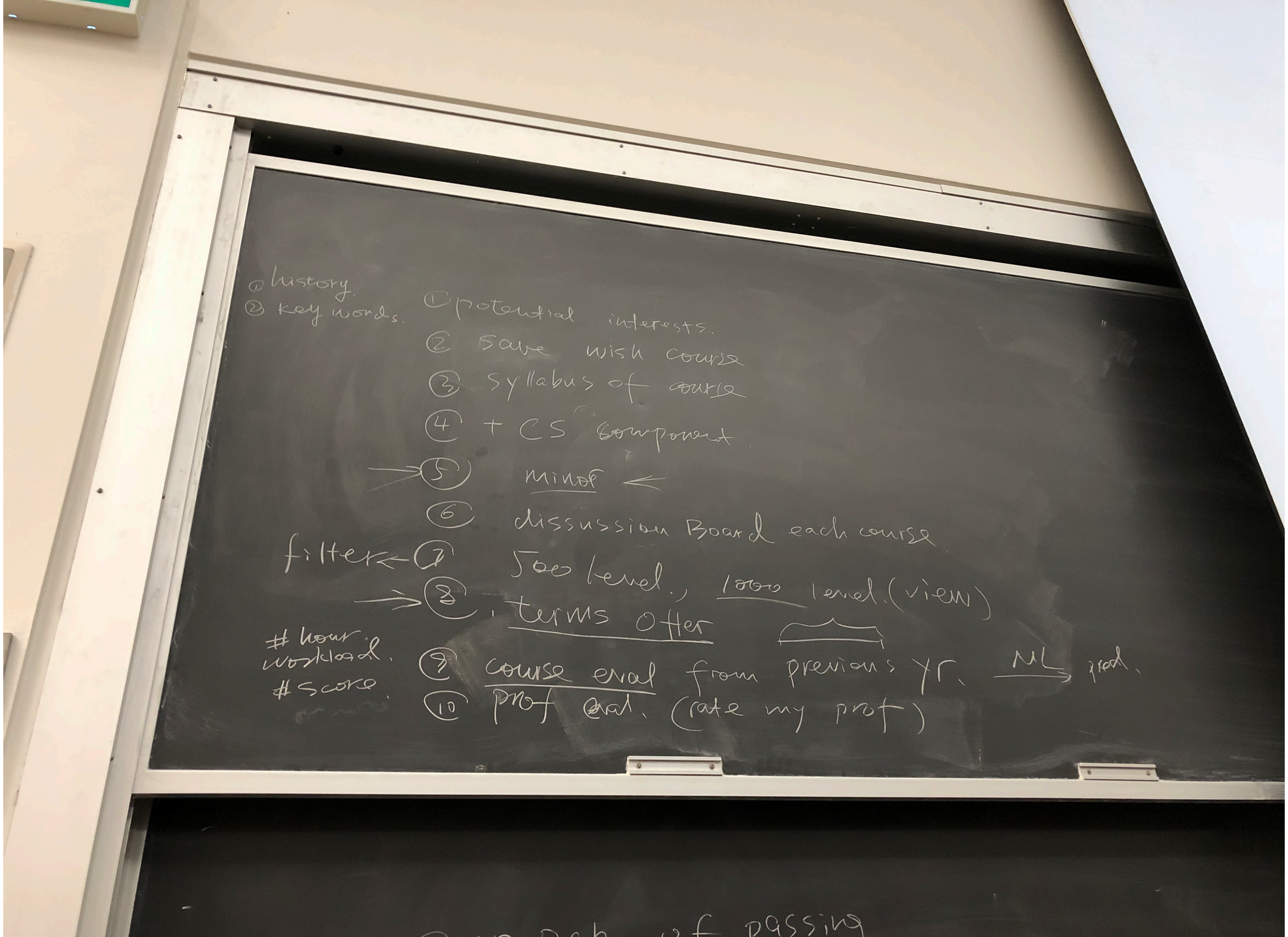
Click here for ECE201's 2020- 2021 Magellan slides

Magellan is a software tool that has been developed in-house in order to help you plan and verify both your program and CEAB (Canadian Engineering Accreditation Board) requirements. It was developed in order to make it easy to verify the CEAB requirements as it automatically calculates the academic units (AU's) when you are building your study plans for 3rd and 4th year. In addition, it will confirm the program requirements at a glance.

MAGELLAN PRE-REGISTRATION (for *2021-2022 courses*) ENDS: January 26, 2021 at 11:59pm EST

Main profiles will then be *LOCKED until July, exact date TBC.* for preregistration planning.
ECE courses will be uploaded to ACORN timetables for students with **valid** Magellan profiles.
Magellan: http://magellan.ece.toronto.edu *(log in with UTORID and password)* / Magellan Manual

*****Main Profiles will be UNLOCKED in July*****
***CHANGES** made to main profiles once UNLOCKED in July*
*are **NOT** included in the preregistration process and thus not uploaded to ACORN.*



① history

② key words.

① potential interests.

② save wish course

③ syllabus of course

④ + CS component

→ ⑤ minor ←

⑥ dissussion Board each course

filter ← ⑦ 500 level., 1000 level (view)

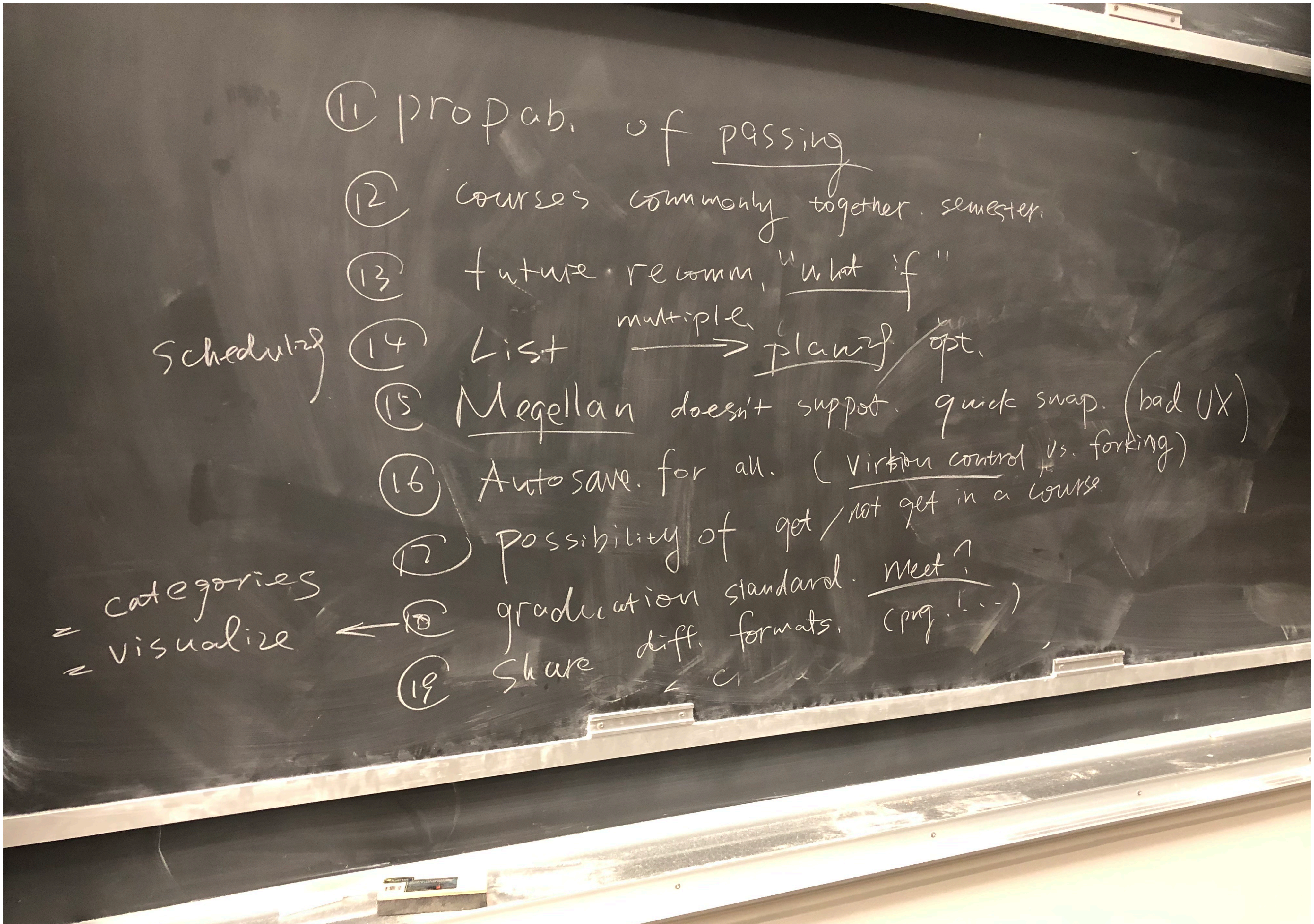
→ ⑧. terms Offer

hour.
workload.
# score

⑨ course eval from previous yr. ML → pred.

⑩ prof eval. (rate my prof)

passing



11 propab. of passing

12 courses commonly together. semester.

13 future recomm, "what if"

Scheduling 14 List —multiple→ planing / opt.

15 Megellan doesn't suppot. quick snap. (bad UX)

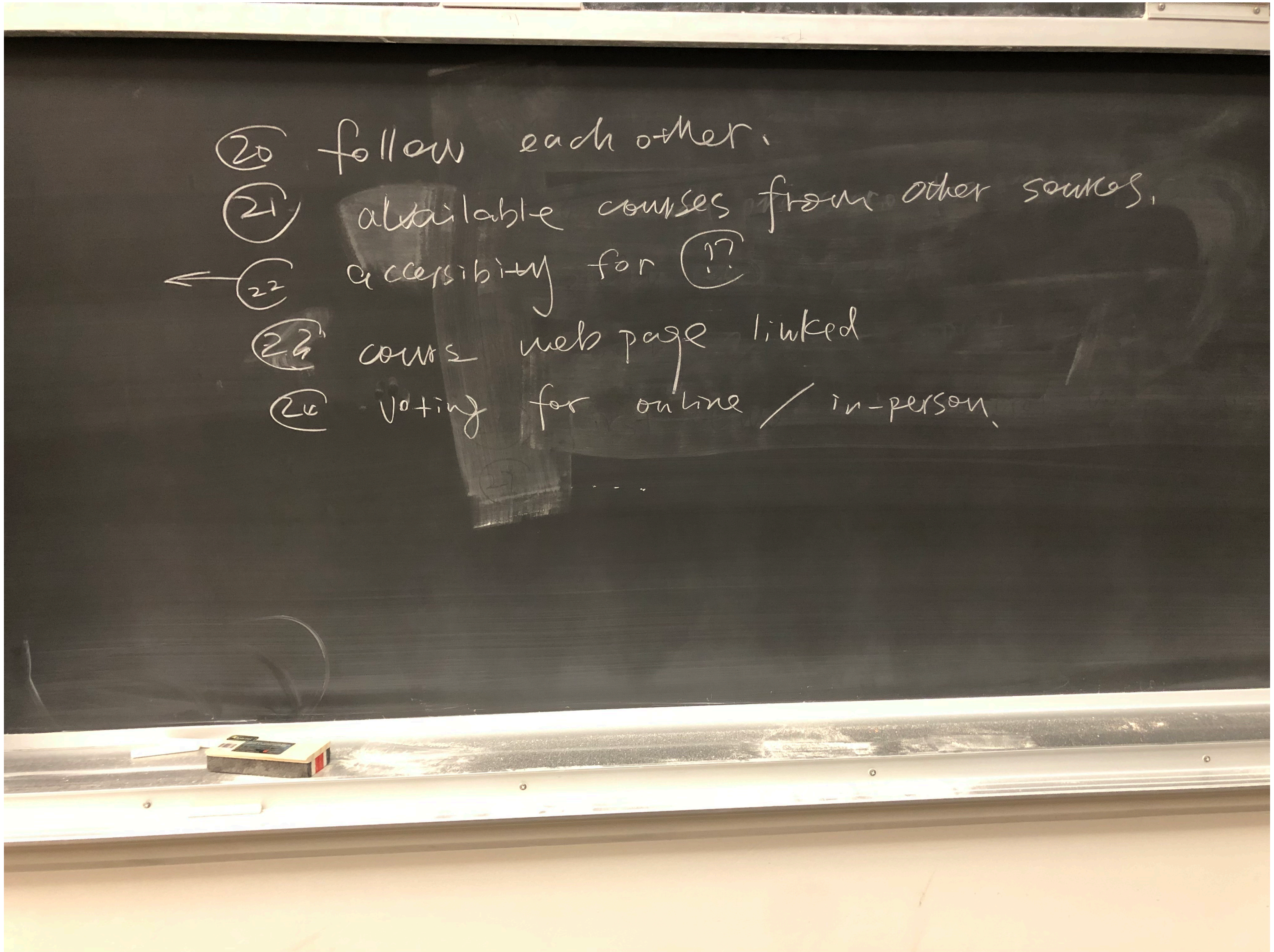
16 Autosave. for all. (Virsion control vs. forking)

17 possibility of get / not get in a course

18 graduation standard. meet?

= categories
= visualize ← 18

19 Share diff. formats. (png. ! ...)



20 follow each other.

21 available courses from other sources.

← 22 accessibility for !?

23 cours web page linked

24 voting for online / in-person.

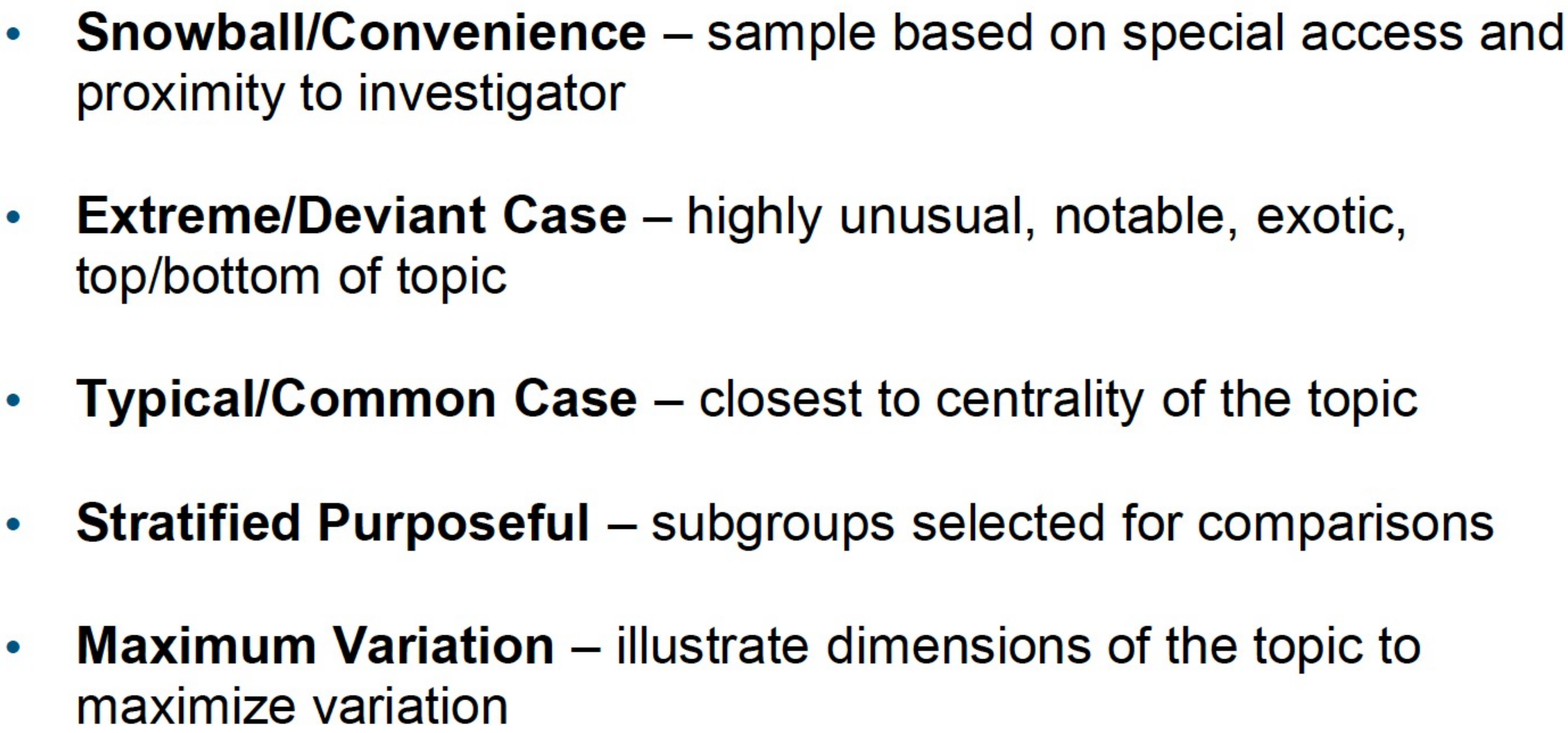
INTERVIEWS
SURVEYS

Interview Tradeoffs

- Strengths
  - What stakeholders do, feel, prefer
  - How they interact with the system
  - Challenges with current systems
- Weaknesses
  - Subjective, inconsistencies
  - Capturing domain knowledge
  - Familiarity
  - Technical subtlety
  - Hinges on interviewer skill

Patton, M. Q. (1990). *Qualitative Evaluation and Research Methods, 2nded.* Newbury Park, CA: Sage Publications.

Sampling Strategies



- **Snowball/Convenience** – sample based on special access and proximity to investigator
- **Extreme/Deviant Case** – highly unusual, notable, exotic, top/bottom of topic
- **Typical/Common Case** – closest to centrality of the topic
- **Stratified Purposeful** – subgroups selected for comparisons
- **Maximum Variation** – illustrate dimensions of the topic to maximize variation