CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Continous Integration, Delivery, Deployment, and DevOps

Dit191/Eda397

Agile Development Processes

Eric Knauss

Agenda today

- Acceptance testing schedule

- Job opportunities (interested? Contact Eric)
  - Summer job:
    Machine Learning/Mining Software Repositories
  - Teaching Assistant:
    Requirements Engineering, Sept-Oct

- Current research challenges

Agile workshop – Acceptance Test No 3.

May-12 13:15-15:45

- Welcome to the third Acceptance Test for the Android App project.
- Each group will have **30 minutes** to show **current status and features of their project** on to discuss the **use of an Agile development approach**.
- Please note the time restrictions, meaning **"Be on time!"**
- **All group members** should bring their laptop where the App should be demonstrated and the development environment (i.e. Trello, Github) should also be accessible.
- If you can't attend, please tell us (Magnus/Terese) via mail **before**.

Acceptance Test - Schedule Group 1-5
Day: Thursday
Focus: Meeting with customer – demo and planning

Room: Room 321, 3th Floor in house Jupiter
Supervisor: **Terese Besker**

Time (30 min/each)	Group	Spokesperson
13.15 – 13.45	5	Wissam Alfreijat
13.45 – 14.15	4	Sofia Edström
14.15– 14.45	3	Sebastian Blomberg
14.45 – 15.15	2	Dominik Muth
15.15 – 15.45	1	Pedram Shirinbolaghi

Acceptance Test - Schedule Group 6-9
Day: Thursday
Focus: Meeting with customer – demo and planning

Room: Room 322, 3th Floor in house Jupiter
Supervisor: Magnus Ågren

Time (30 min/each)	Group	Spokesperson
13.15 – 13.45	9	Fredrik Holmdahl Dominik Muth
13.45 – 14.15	8	Mustafa Hussein
14.15– 14.45	7	Johan Eriksson
14.45 – 15.15	6	Fredrik Holmdahl

Meeting with customer

present status and features of your project

Agile Manifest principle:
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Acceptance test :

- What features did you show the customer during last Acceptance Test?
- What did you and customer agree to on last Acceptance Test?

Today and future:

- What new features can you show customer today?
- Any more planned activities for final acceptance test?
- Customer would like to add…..

Agile Manifest principle:
Working software is the primary measure of progress

Coaching Meeting

using an Agile development approach

"At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly."

Retrospective: the team reflects on what happened in the iteration and identifies actions for improvement going forward:

- What worked well for us?
- What did not work well for us?
- What actions can we take to improve our process going forward?

Current Research Challenges

CONTINUOUS INTEGRATION AND BEYOND

Overview

- Part 1: What and why?
  - Stairway to heaven (from Software Center)
  - Definitions of Continuous X (from NGEA)
- Part 2: RE in Continuous X
  - RE Challenges (from RE 15)
  - Scaling up the planning game (from XP 16)
- Part 3: Testing in Continuous X
  - Test prioritization and selection (RCoSE 15)
  - Non-functional Testing Oracle (ISSTA 16)

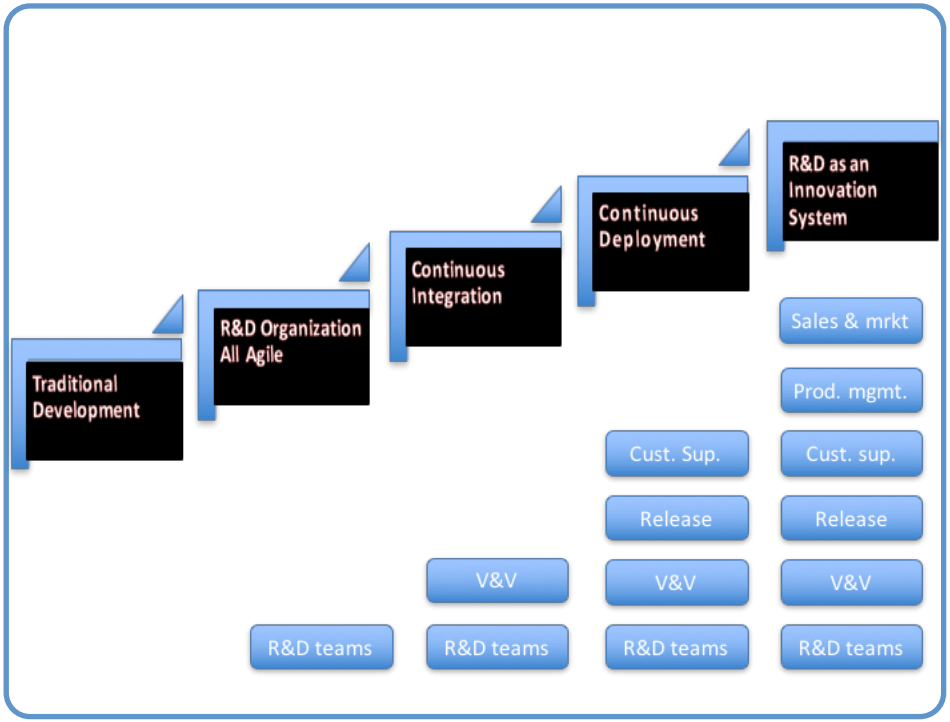
Part 1:
What is Continuous Integration (Delivery, Deployment) and why is it important

DEFINITION OF CONTINUOUS X

STAIRWAY TO HEAVEN

Software Center

The Stairway to Heaven Model describes the stages that companies evolve through when adopting novel approaches to software engineering



- Companies move through a predictable and repeatable pattern over time when evolving software engineering practices
- Each transition has business, architectural, process and organizational implications
- The higher up the stairway an organization climbs, the more organizational units are affected

- H.H. Olsson, H. Alahyari, J. Bosch, Climbing the"" Stairway to Heaven --A Mulitiple-Case Study Exploring Barriers in the Transition from Agile Development towards Continuous Deployment of Software, 38th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA),, pp. 392-399, IEEE, 2012.

For more information please contact jan.bosch@chalmers.se

Organizational Categories

- **Enterprises**: software supports internal processes and external services (e.g. banks, insurance companies)
- **Product oriented organizations**: develop software or software intensive products that are not deployed directly by the development teams
- **Service companies**: develop software that is hosted and deployed directly by the developing team

Matthew Bass (CMU)

Definitions of Continuous X

- Integration
- Delivery
- Deployment

NGEA
Next Generation Electrical Architecture

VINNOVA

Contact:
Magnus Ågren, Rogard Heldal, Eric Knauss, CTH
{magagr,heldal,knauss}@chalmers.se

Context

NGEA

- This is joint work with Agneta Nilsson
  - Synergies with Software Center Project 1 Implications of Continuous Deployment"
  - Based on group interviews with 6 companies + cross-organizational workshops for consolidating results

- Research goal:
  - Many definitions, many concepts (Continuous Integration, Continuous Delivery, Continuous Deployment, …)
  - Understand which aspects are important in practice
  - Agree on shared language = *Common Understanding*

Continuous Integration

NGEA

From Literature

Integrate new or changed code with the mainline codebase at frequent time intervals

From Interview

- Integrate / Test
- Sufficient quality of codebase
- One Mainline / Several
  - As long as each carry customer value
- Frequency Weeks / Days / Instant
- Why do we integrate – to get feedback

Synthesis

Integrate and test new or changed code with a codebase to enable feedback whenever we want

Continuous Deployment

NGEA

The ability to release software whenever we want

- Release and Install in running / Deliver
- Higher quality level of codebase
- Strong quality assurance

Otherwise:
Continuous Delivery

The ability to release and install software in running system whenever we want

Something in between…

NGEA

- Continuous Integration often on team level
- Continuous Deployment implies the whole organization to work in a continuous way

- Enterprise Continuous Integration [Ståhl and Bosch 2015]

- Continuous X includes a number of dimensions

Summary / Proposed Terminology

NGEA

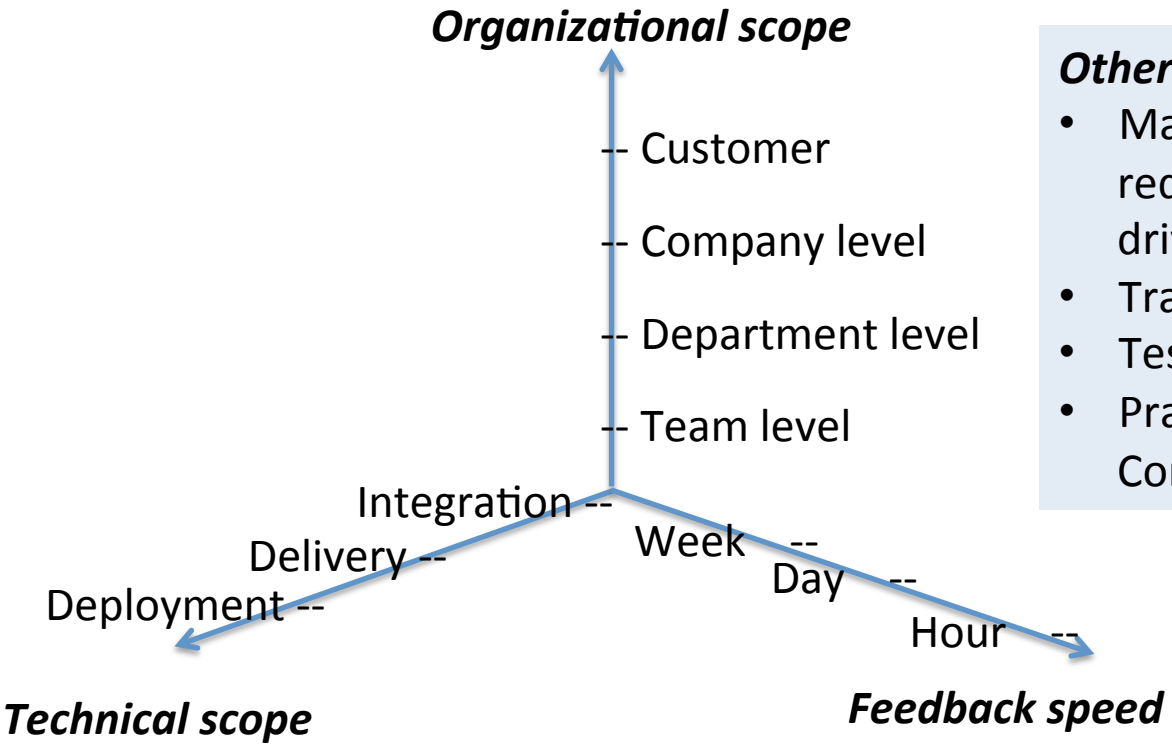
		Target/Receiver			
		Other team/ unit	Internal Customer	Development partner	Consumer
Range of Flow	1. Integration (Integrate to test and learn about the quality and fit of the delta)				
	2. Delivery (Deliver a delta to a target/receiver (no active test and learning))				
	3. Deployment (Deploy to a target/receiver's runtime)				

"There is no *best practice*, only *most suited*!"

Current work

NGEA

- Systematic Literature Review to understand different dimensions of these terms



***Other dimensions*:**

- Managing architecture, requirements, business drivers
- Transparency
- Testing
- Practical scope: Vision, Constructive, Experience

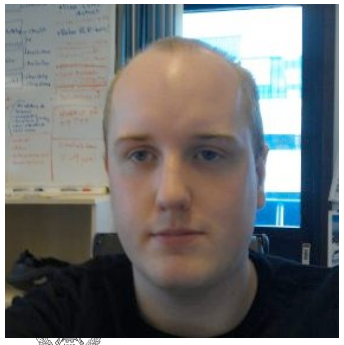
Part 2:

Managing Requirements and Related Knowledge – continuously

REQUIREMENTS ENGINEERING IN CONTINUOUS SOFTWARE ENGINEERING

The Need of Complementing Plan-Driven Requirements Engineering with Emerging Communication: Experiences from Volvo Car Group

Ulf Eliasson, Rogardt Heldal,
Eric Knauss, Patrizio Pelliccione



23rd IEEE International Requirements Engineering Conference
August 24-28, 2015.
Ottawa, Canada.

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Research Problem:

To change plan-driven organisation
when need for **faster innovation** and
shorter time to market
increases

Research Questions

- RQ1: How do requirements flow between stakeholders of automotive (software) engineering?
- RQ2: Is there a need for shorter feedback loops for requirements validation?

Reqts. flow in automotive ecosystem

Challenge	Practices	Recommendations
Balancing under-specification and over-specification of requirements	• Personal network • Oral communication • Assumptions	• Networking • On demand / Just-in-time RE • Infrastructure for communication and feedback
Synchronizing cross-function and cross-system requirements	• Personal network	• Continuous integration and deployment
Friction for changing requirements increases over time	• Workarounds	• Defer commitment
Avoid premature commitment	• Workarounds • Oral communication	• On demand / Just-in-time RE
Requirements need context	• Personal network • Oral communication	• Rational and context as by-product
Slow feedback cycle on requirements	• Personal network • Oral communication	• Short iterations / agile • Virtual verification early
Balancing the need for oral communication and thorough documentation of requirements	• Personal network • Oral communication	• On demand / Just-in-time RE • Rationale as by-product
Find the right person for getting or giving feedback or information	• Personal network • Expert seeking	• Facilitate networking within company and supply-chain
Sufficiently fast supplier interaction	• Personal network	• New business cases / opportunities for suppliers

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Context

mplex system

The most advanced cars have more/comparable software than fighter airplanes …

… said by Alfred Katzenbach, the former director of information technology management at Daimler.

Modern cars are complex distributed software in moving, safety-critical, mechatronic systems.

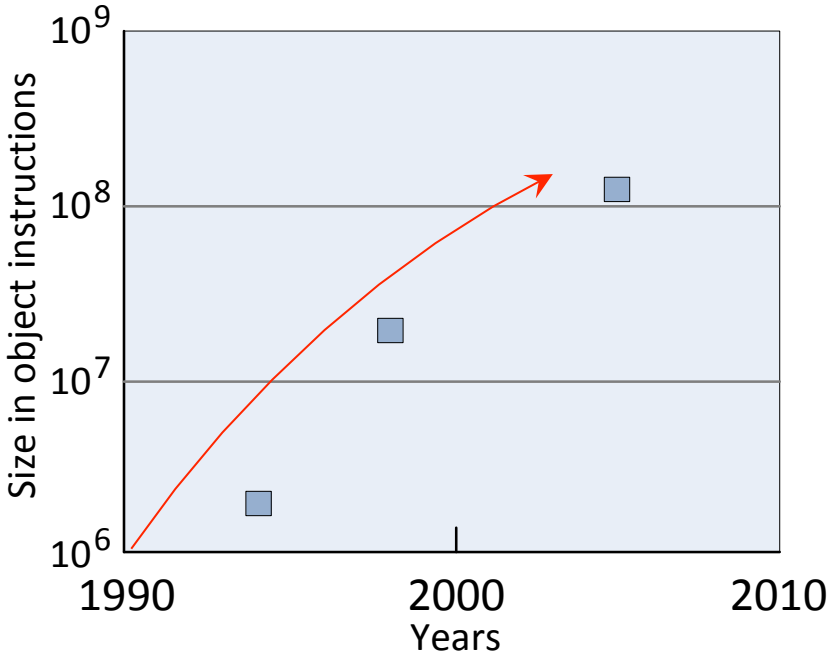
Innovation

- Reduced energy consumption
- Connected cars
- Autonomous cars, basically creating robots
- …

- … All of this requires a lot of Software!

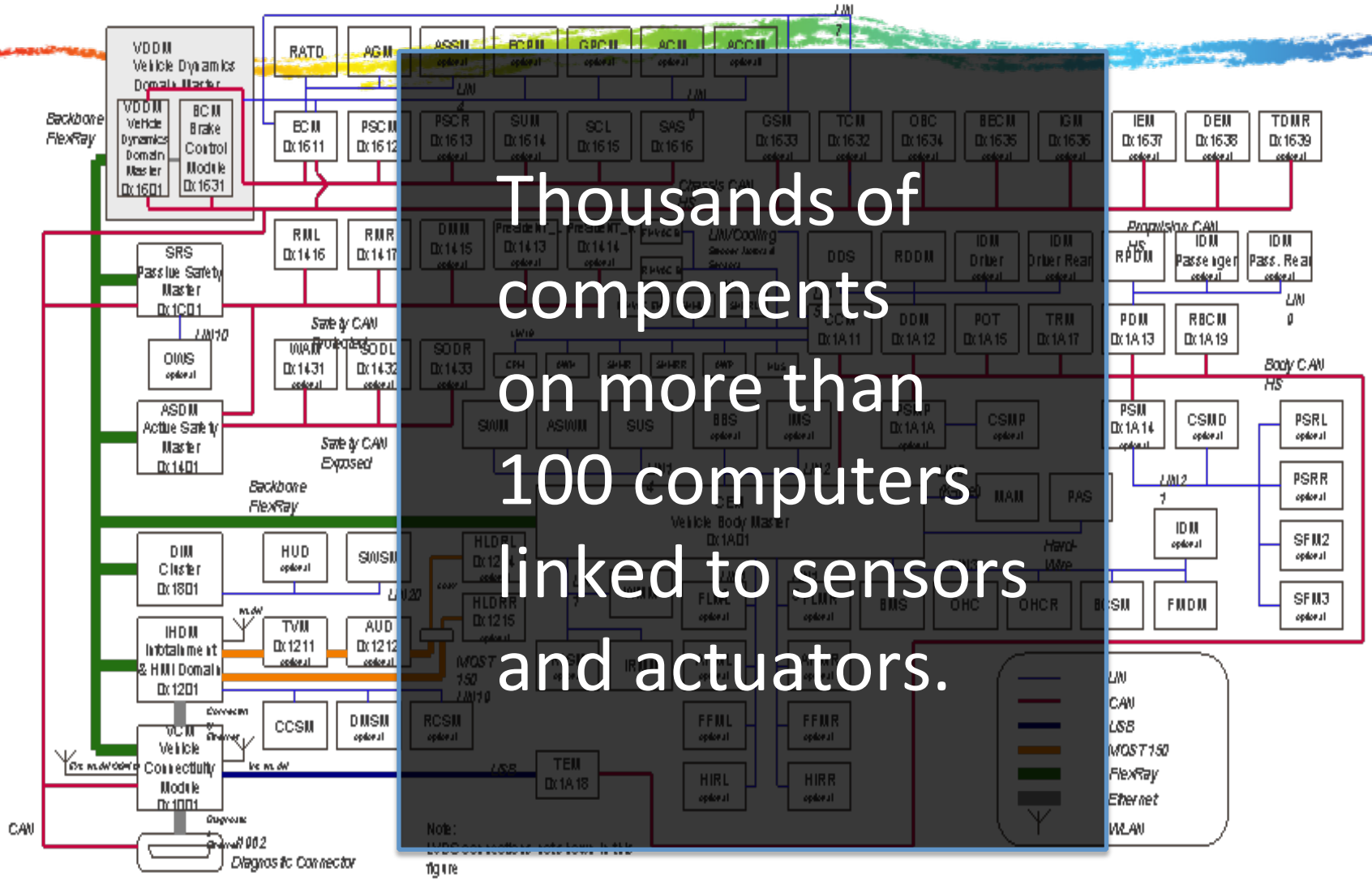
Automotive Software

- 90% of innovation from electronics
  - B. Hardung, T. Kölzow, and A. Krüger
- 80% of that from software
  - B. Hardung, T. Kölzow, and A. Krüger
- The amount and complexity of software in cars increase
  - C. Ebert and C. Jones,



- New hybrid vehicles add even more, safety critical software!

100+ ECUs


Thousands of components on more than 100 computers linked to sensors and actuators.

Function

Function owner

F1 F2 F3 F500+

Function Realization

Function Realiz. Resp.

FR1 FR2 FR3 FR500+

Driven by regulations, complexity,….!

Sub System

System Design

SS1 SS2 SS3 SS50+

Suppliers/ In-house

SW and HW Design

2 3 4 100+

Development Teams

Method

Method

- 11 semi-structured interviews (four groups):
  - 8 interviews on information flow (Volvo Car)
  - 3 interviews focused on the area of RE (Volvo Car)
- 4 semi-structured interviews for triangulation:
  - 2 interviews on information (Volvo Truck)
  - 2 interviews focused on the area of RE (Volvo Truck)
- Ca. 45 min pr. interview

Subjects on all levels

Function

Function owner

F1

Function Realization

Function Realiz. Resp.

FR1

FR2

Sub System

System Design

SS1

SS2

Suppliers/ In-house

SW and HW Design

2

3

Development Teams

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Findings

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

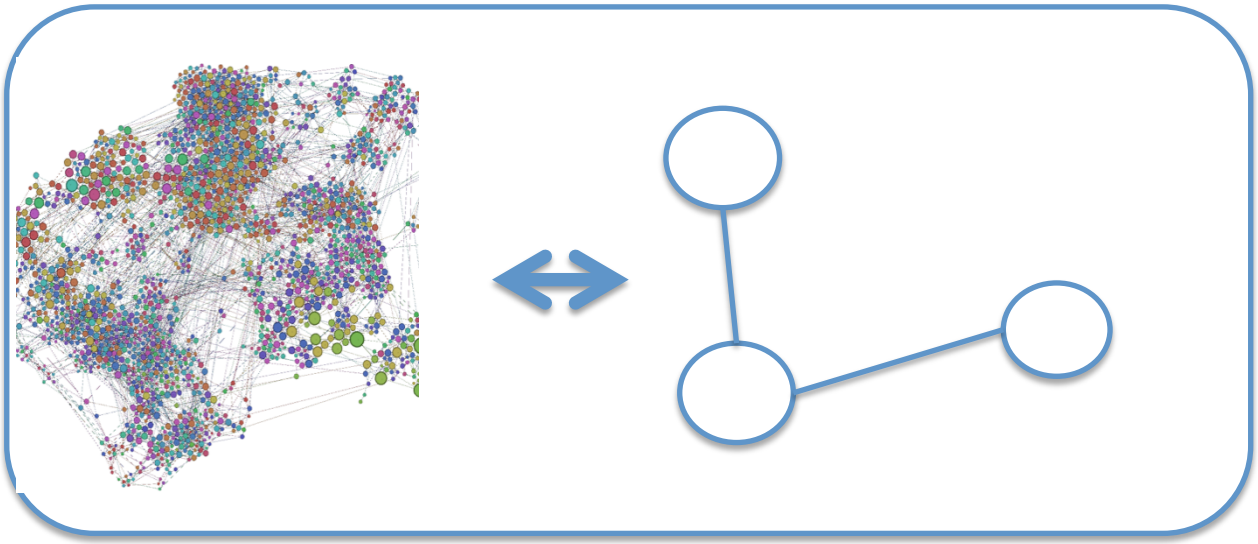
Research Question 1

How do requirements flow between stakeholders of automotive (software) engineering?

Research Question 1:
How do requirements flow between stakeholders

Challenge	Practices	Recommendations
Balancing under-specification and over-specification of requirements	• *Personal network* • *Oral communication* • *Assumptions*	• *Networking* • *Infrastructure for communication and feedback*
Synchronizing cross-function and cross-system requirements	• *Personal network*	• *Continuous integration and deployment*
The Cost for changing requirements increases over time	• *Workarounds*	• *Defer commitment*
Avoid premature commitment	• *Workarounds* • *Oral communication*	• *On demand / Just-in-time RE*
Requirements need context	• *Personal network* • *Oral communication*	• *Rationale and context as by-product*

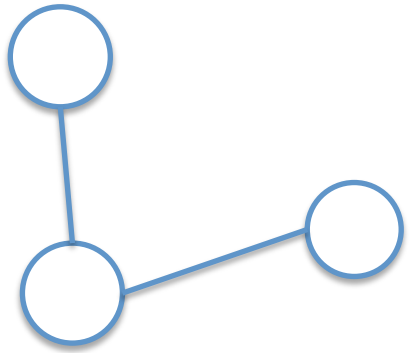
Contrasting opinions: Under-Specification and Over-Specification



Under-Specification

System responsible:

”… a use case is very limited, it describes, a chain of event, what should happen. And then another chain of events. But there exist a **lot of other subsets** of all these steps in the chain, which have **no requirements stated**”



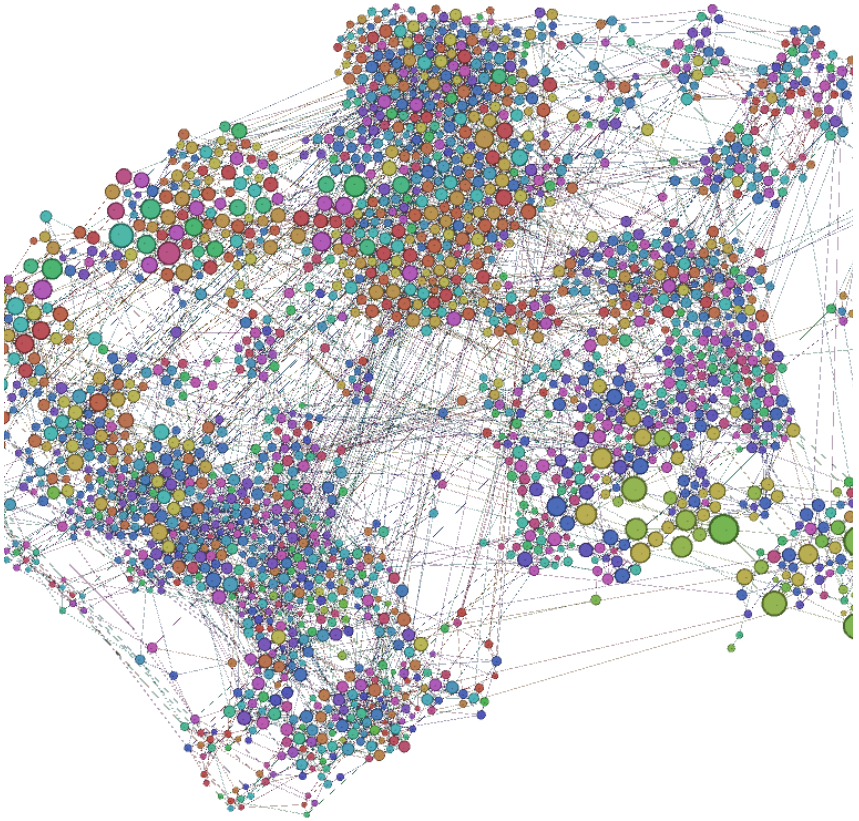
Over-Specification

System responsible:

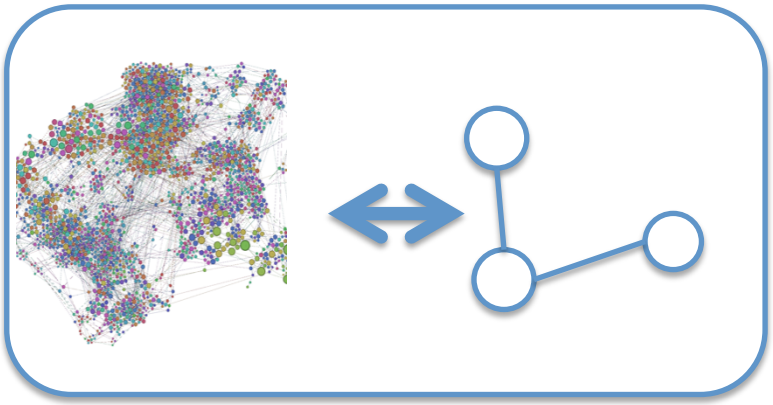
"… we would have liked to just say: When this error occurs the battery should act like this …

Then they can decide (SW and HW designers), what does it mean to open …

Req send to their suppliers. So then it needs to be at a certain level of detail, **considering that there will be no further refinement of the requirement"**



Balancing Under-Specification and Over-Specification



- Practices
  - Personal network
  - Oral communication
  - Assumptions
- Recommendations
  - Networking
  - Infrastructure for communication and feedback

Top down flow

Function

Function owner

F1

Function Realization

Function Realiz. Resp.

FR1

FR2

Sub System

System Design

SS1

SS2

Suppliers/ In-house

SW and HW Design

2

3

Development Teams

Synchronizing Cross-Function and Cross-System Requirements

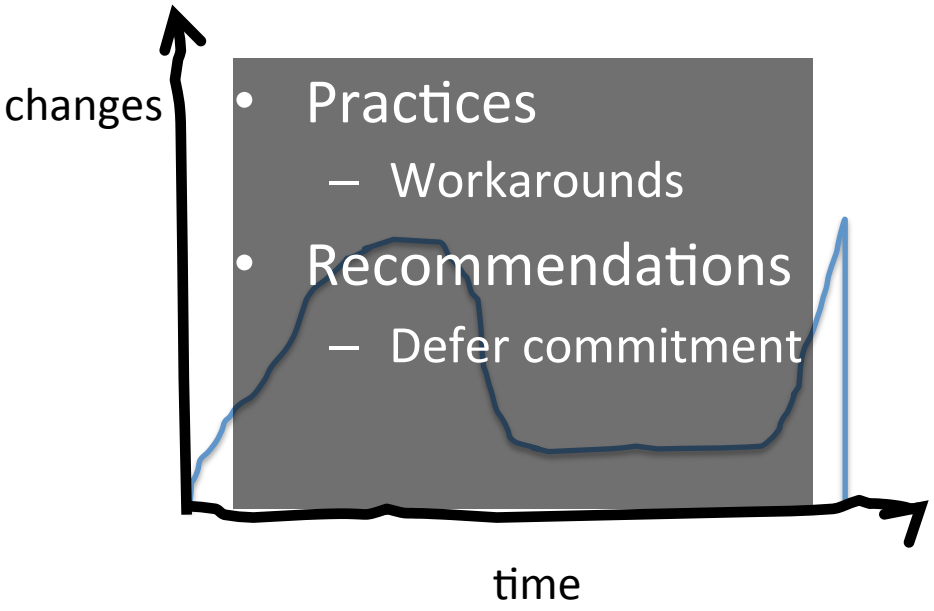
System responsible:

"Then **requirement can come from other directions** as well, e.g., base technology, **attribute requirements** can directly impact us, even architectural requirement and **cross-requirements** from other sub-systems, or rather from other functions."

The Cost for Changing Requirements Increases Over Time

System Responsible:

"as the project **progress**, new gatekeepers and decisions forum where a **change must be approved** are introduced. [...] And the further the project progress the suppliers start to demand payment for changes, and that also introduces additional friction."



Avoid Premature Commitment

Sub-system responsible

”… in nine cases out of ten we want to have the **requirement on the ECU in itself**, then it is the responsibility of the supplier to divide the requirements between software or hardware."

- Practices
  - Workarounds
  - Oral communication
- Recommendations
  - On demand / Just-in-time RE

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Research Questions 2

Is there a need for shorter feedback loops for requirements validation?

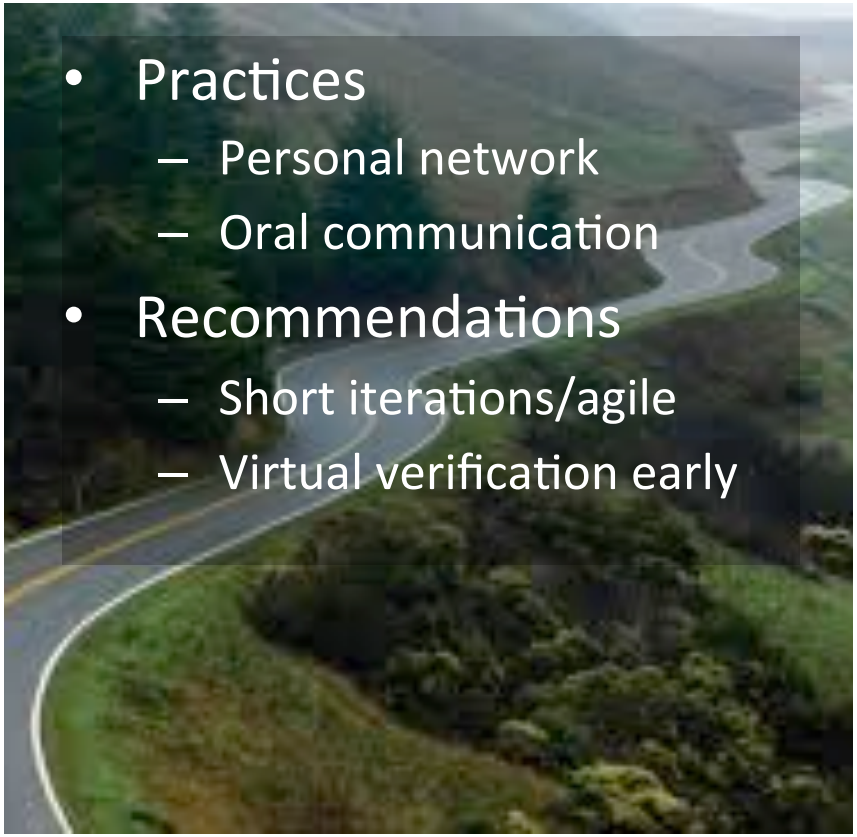
RQ2: Is there a need for shorter feedback loops for requirements validation?

Challenge	Practices	Recommendations
Slow feedback cycle on requirements	• *Personal network* • *Oral communication*	• Short iterations / agile • Virtual verification early
Balancing the need for oral communication and thorough documentation of requirements	• *Personal network* • *Oral communication*	• On demand / Just-in-time RE • Rationale as by-product
Find the right person for getting or giving feedback or information	• Personal network • Expert seeking	• Facilitate networking within company and supply-chain
Sufficiently fast supplier interaction	• Personal network	• New business cases / opportunities for suppliers

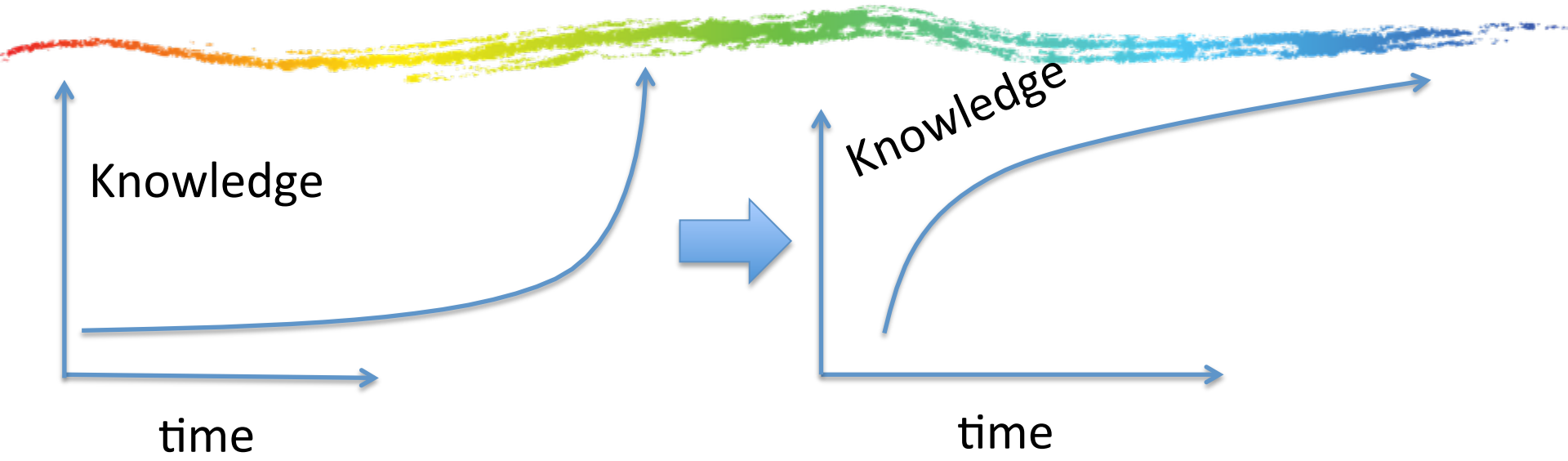
Slow Feedback Cycle on Requirement

Sub-system responsible

- ”It takes a long time [...] and it usually goes full circle, that they have the software ready, it is **put in the car**, it is **tested together** in the environment it should be in, and then you **discover that this and this is probably** wrong."



Experimenting on requirement level!



More knowledge -> less assumptions, lower risk

U. Eliasson, R. Heldal, P. Pelliccione, and J. Lantz. Architecting in the Automotive Domain: Descriptive vs Prescriptive Architecture. In 12th Working IEEE/IFIP Conference on Software Architecture (WICSA 2015), Montreal, Canada, 2015. IEEE

Find the Right Person for Getting or Giving Feedback or Information

Sub-system responsible:

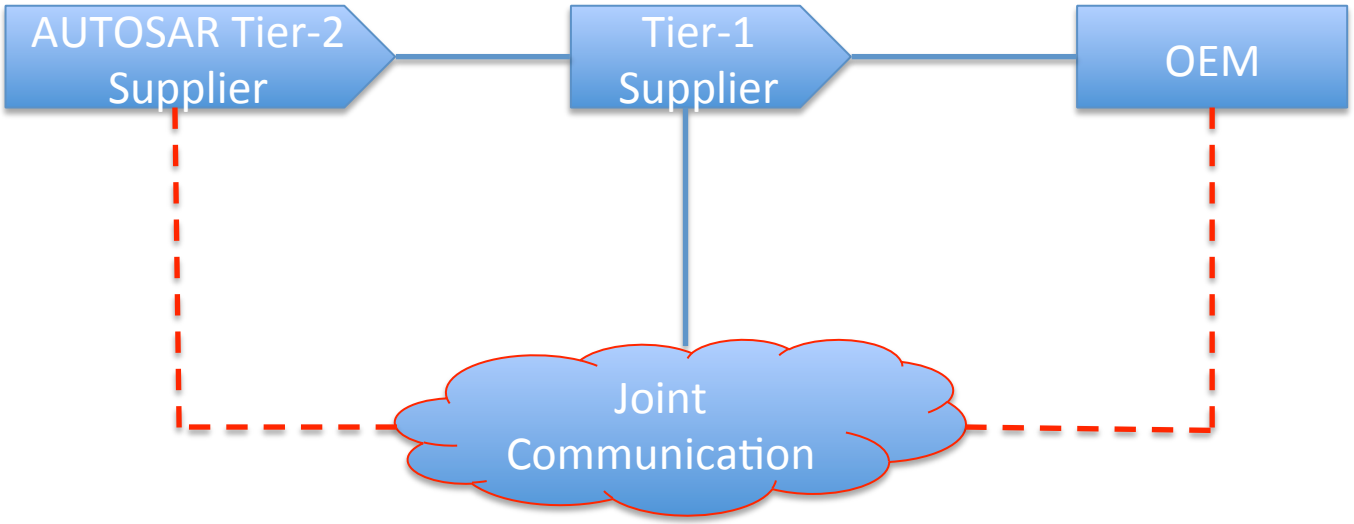
"… So I start with someone, and they might answer that they do not know, so I ask who do you think know? Then I try with the next one and **keep on doing that until I get what I need**…."

Sufficiently Fast Supplier Interaction

Sub-system responsible:

”I think that sometimes the supplier needs to **understand the overall behaviour** to be able to implement a good function."

New business cases / opportunities for suppliers



Process developer: "This in practice means that we pay our subcontractors for development time."

Mozhan Soltani and Eric Knauss. Cross-Organizational Challenges of Requirements Engineering in the AUTOSAR Ecosystem: An Exploratory Case Study. In Proceedings of 5th IEEE International Workshop on Empirical Requirements Engineering at RE '15, Ottawa, Canada, 2015

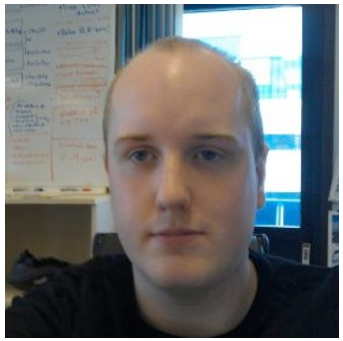
Summary

Challenge	Practices	Recommendations
Balancing under-specification and over-specification of requirements	• Personal network • Oral communication • Assumptions	• Networking • On demand / Just-in-time RE • Infrastructure for communication and feedback
Synchronizing cross-function and cross-system requirements	• Personal network	• Continuous integration and deployment
Friction for changing requirements later in time	• Workarounds	• Defer commitment
Avoid premature commitment	• Workarounds • Oral communication	• On demand / Just-in-time RE • Rationale as by-product
Slow feedback cycle on requirements	• Personal network • Oral communication	• Short iterations / agile • Virtual verification early
Balancing the need for oral communication and thorough documentation of requirements	• Personal network • Oral communication	• On demand / Just-in-time RE • Rationale as by-product
Find the right person for getting or giving feedback or information	• Personal network • Expert seeking	• Facilitate networking within company and supply-chain
Sufficiently fast supplier interaction	• Personal network	• New business cases / opportunities for suppliers

Document-centric, plan-driven processes are too slow

- Engineers rely on their personal network and direct communication to overcome those limitations
- Existing tooling and documentation structure do not fully support that (e.g. need to decide too early)

The Need of Complementing Plan-Driven Requirements Engineering with Emerging Communication: Experiences from Volvo Car Group



Ulf Eliasson

Rogardt Heldal

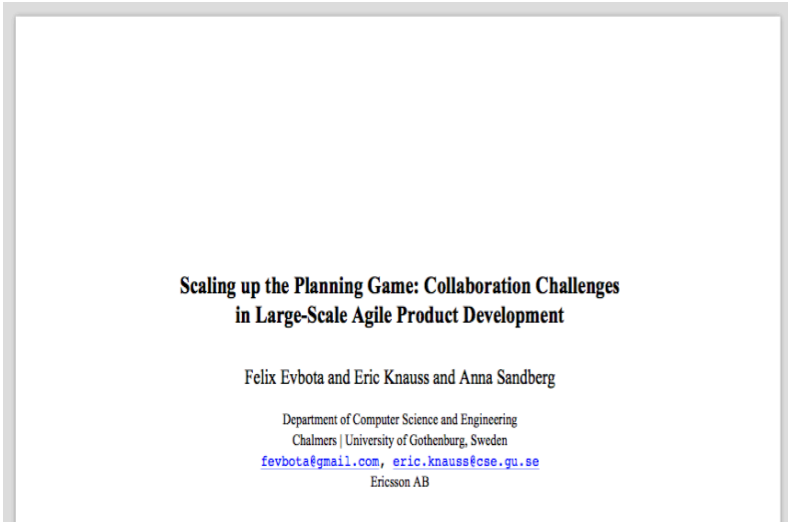
Eric Knauss

Patrizio Pelliccione

Questions?

LARGE-SCALE PLANNING GAME

Felix Evbota, Eric Knauss, Anna Sandberg: Scaling up the Planning Game: Collaboration Challenges in Large Scale Agile Product Development. In: Proc. of 17th Int. Conf. on Agile Software Development (XP 2016), Edinburgh, UK, 2016



Scaling up the Planning Game: Collaboration Challenges in Large-Scale Agile Product Development

Felix Evbota and Eric Knauss and Anna Sandberg

Department of Computer Science and Engineering
Chalmers | University of Gothenburg, Sweden
fevbota@gmail.com, eric.knauss@cse.gu.se
Ericsson AB

CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Understanding Customers and End-users

Large-Scale agile

- Huge distance to customers and end-users
- Hierarchy of product owners
- Coordinate efforts and dependencies of many teams
- Information silos vs. overload

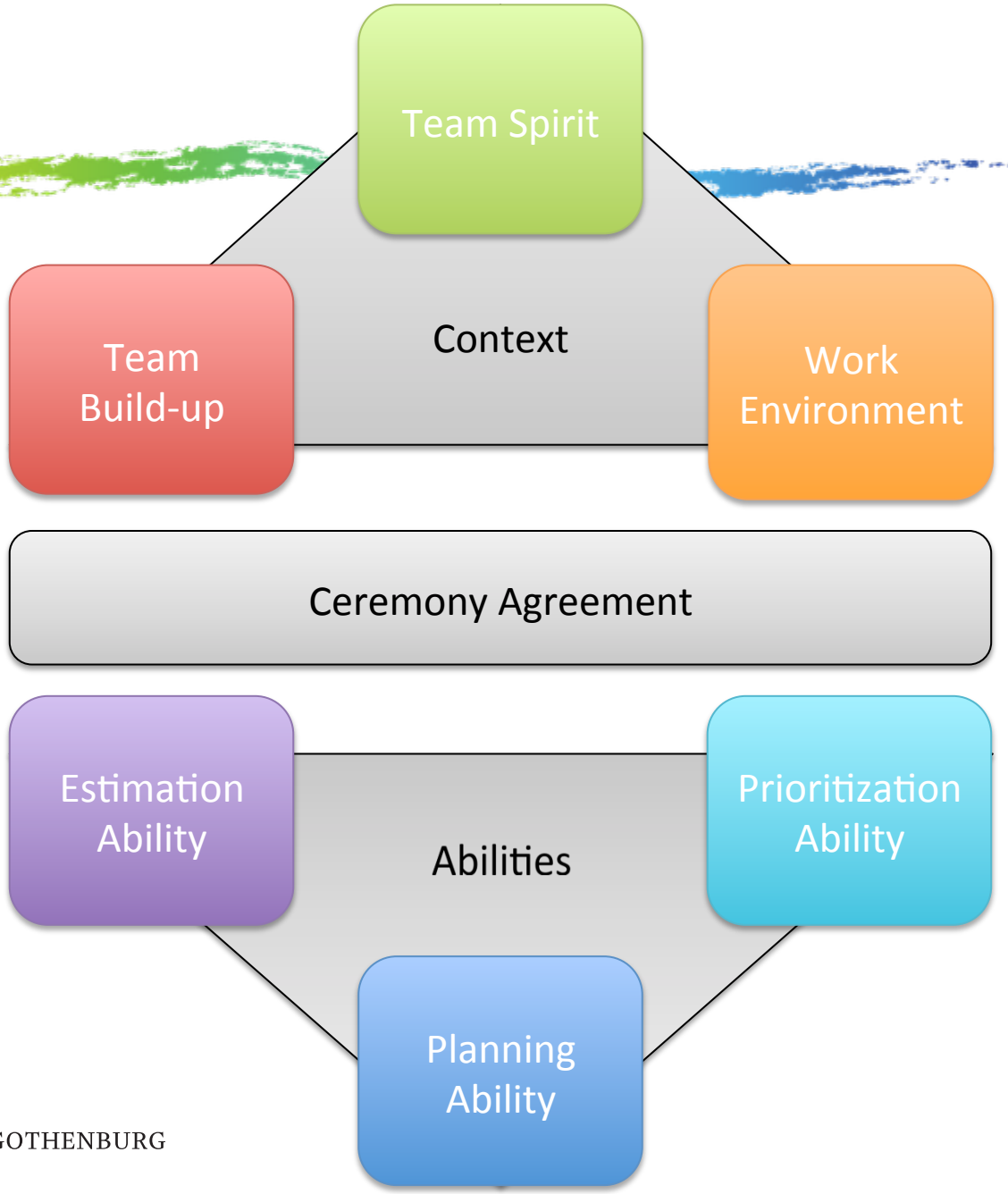
Agile

- Onsite customer / product owner
- User stories facilitate discussions
- Planning game to facilitate dialogue between customer and supplier
- Cross-functional team offers rich perspective

Findings

Research Method

- Qualitative Case Study (Ericsson)
- 10 Semi-Structured interviews with
  - op. product owner,
  - line manager,
  - program leader,
  - project leader,
  - release leader,
  - team leader and
  - developer

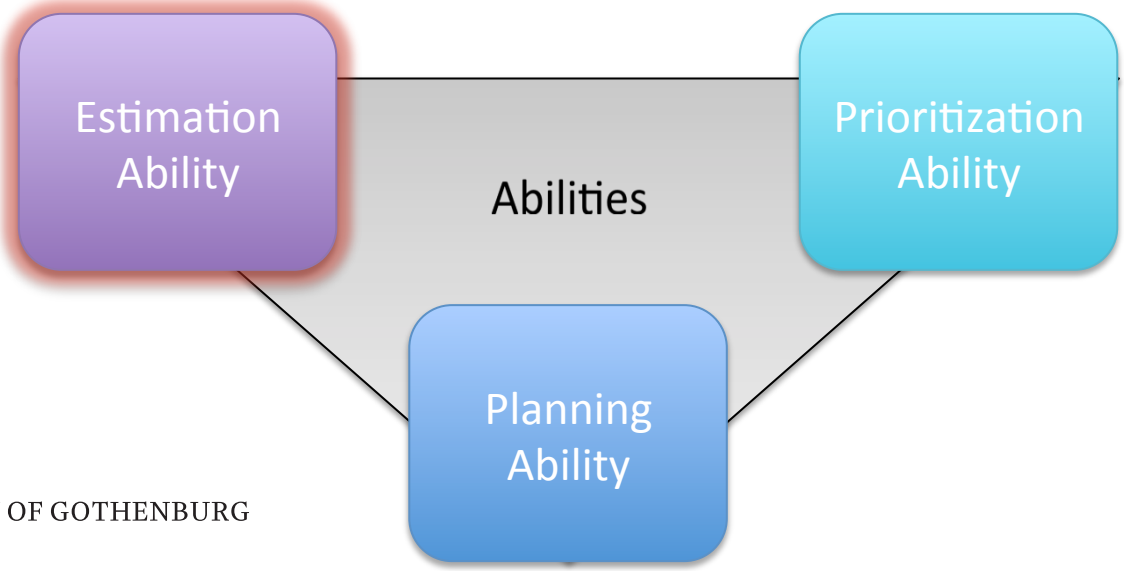


Findings

"[Previously we] estimated on available days in the sprint, that is not a good way because you do not include the unexpected things" [OPO]

"[Sometimes we have a] tester estimating design tasks and a designer estimating test tasks. It is important to know whose estimation should be looked at". [Line mgr]

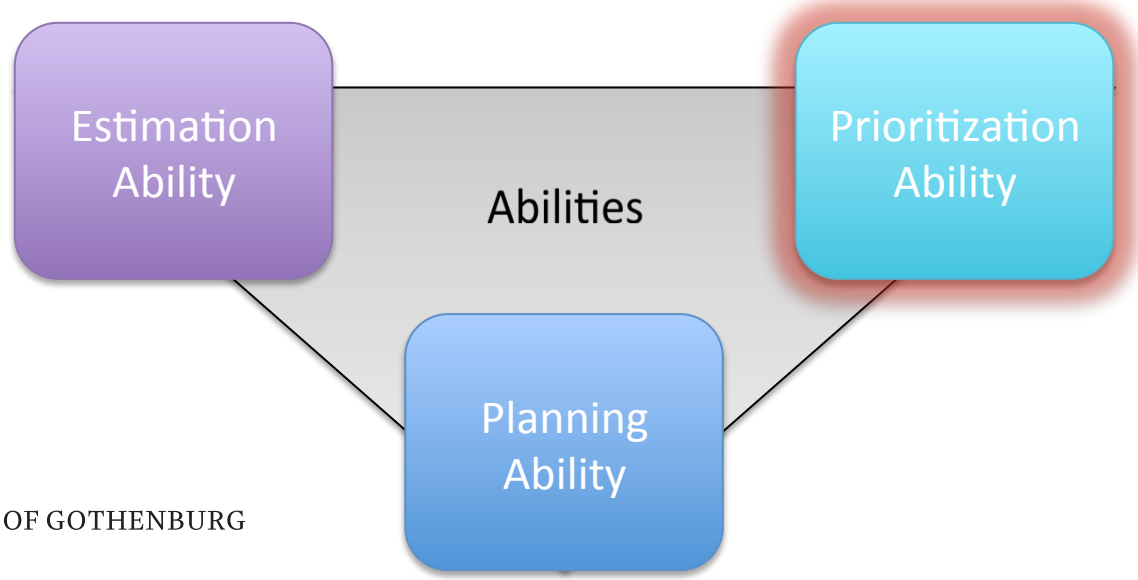
- Skeptical about estimations
- Need to monitor discussions
- *Estimate tasks that do not fit role*



Findings

"The challenge is if you have a lot of small backlog you are not in control at all be- cause if you have one common backlog and you decide on a program level, that is how we work [...] if not everything is visible on the common backlog program and only visible in the XFTs backlog then you maybe having a mismatch." [OPO]

- Complex structure of product owners and backlogs
- Inconsistencies between backlogs
- Lack of transparency

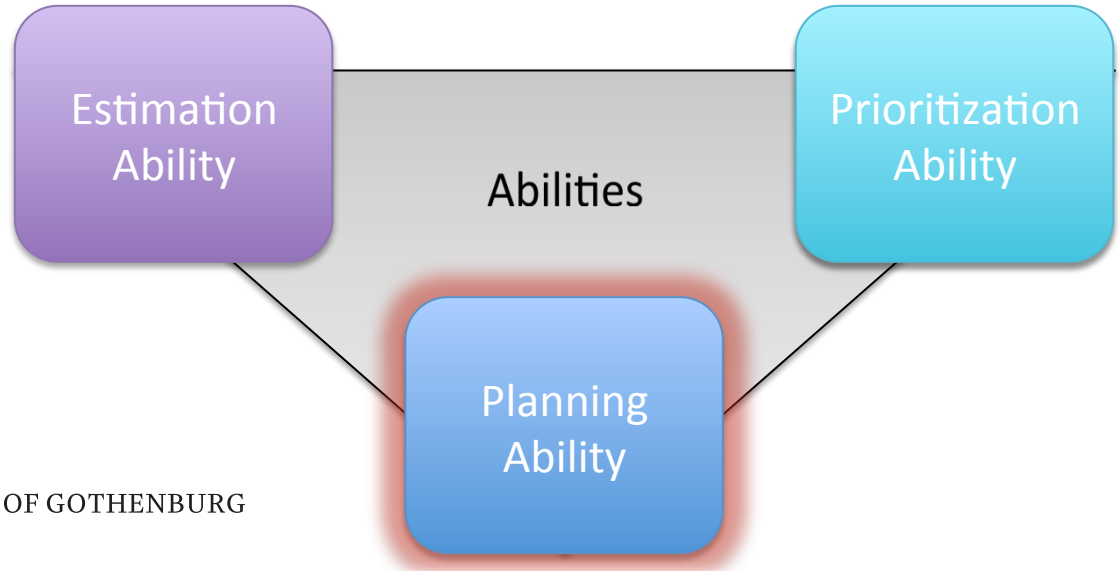


Findings

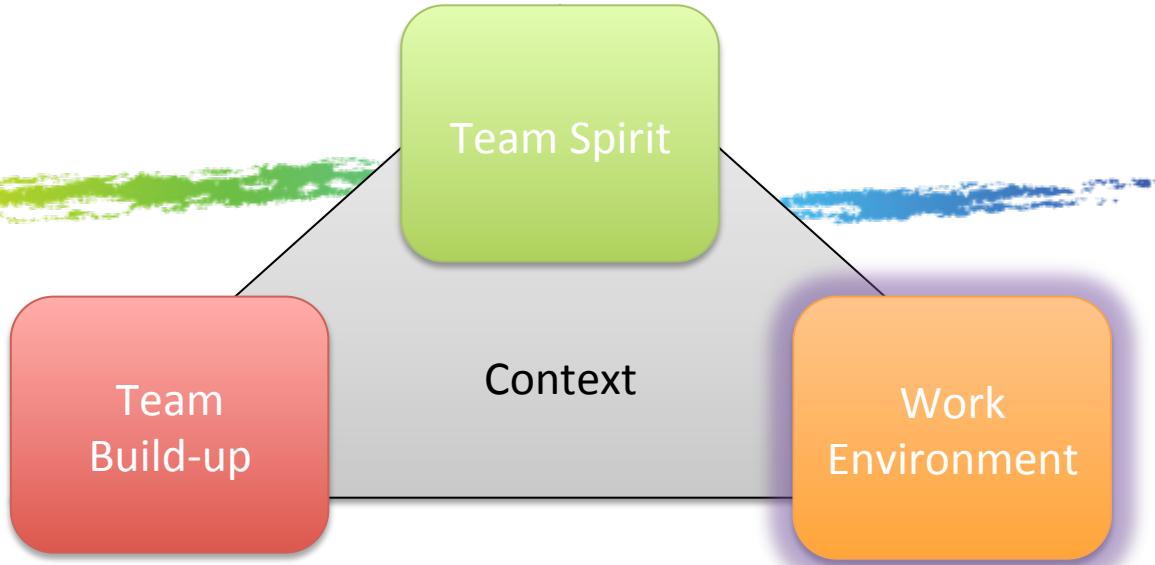
"... we have been working like this for two and half years now, and we are still struggling finding our roles". *[OPO]*

"[planning is done by] me as the manager and perhaps with the help of the team leaders sometimes... [unclear] just how much should the team be involved in the planning phase and stuff like that..."
[Line mgr]

- Unclear requirements
- Unclear role of operational product owner
- Involvement of teams



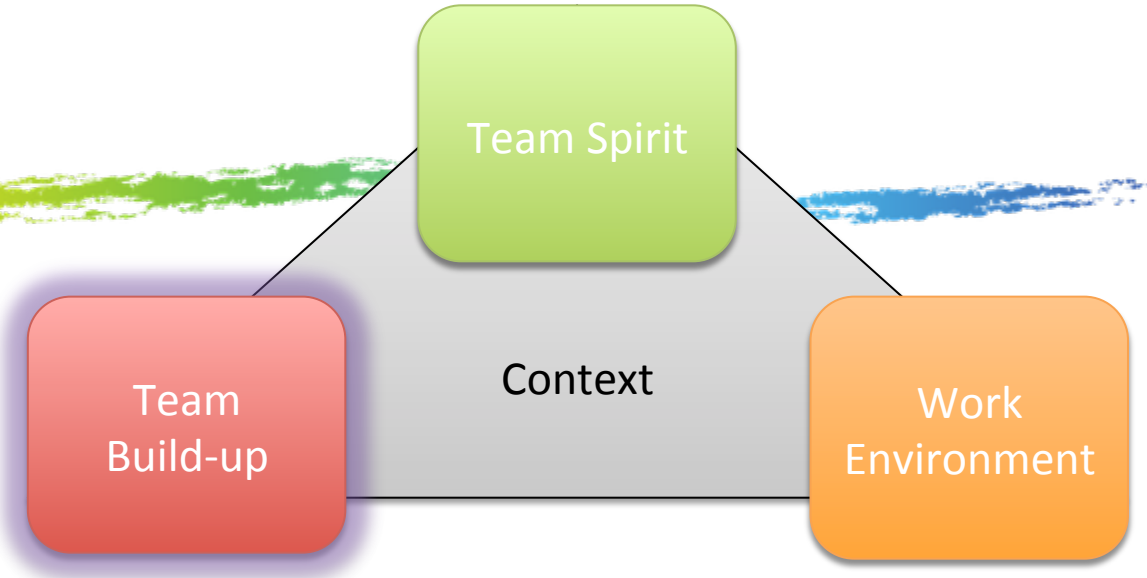
Findings



- Open Space
- Disturbing other teams
- By-passing operative product owner

"...we have scrum meetings in open office space [...]. You kind of get disturbed when other teams are having their scrum meeting in the open setting. It is better [if] every team has their different rooms." [Dev.]

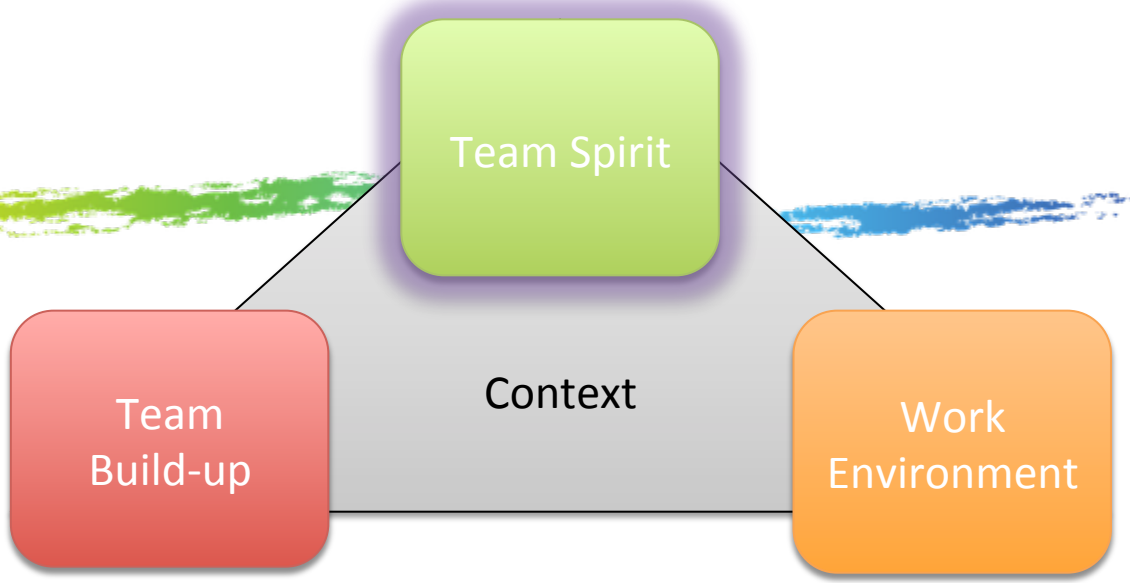
Findings



- Capabilities of team
- Moving team members
- Need for broader competence

"[Sometimes] operative product owner has one view and the team has another view [… It] can be a challenge to have the operative product owners to understand what capability a certain team has and he wants much more than they are capable of doing."

Findings



- Shared concept of agile
- Team maturity

"moving somebody from one team to the other, then you are impacting the team spirit in both teams, you might go back to the team development stage when you get a new team member or lose a new team member".

Findings

- Lack of suitable information channels
- Coordination meetings boring or too short
- Inadequate anatomy of features

Ceremony Agreement

"The biggest challenge I pick is the coordination of the feature portfolio, [...] on top of getting out features in our program fast and efficient, we need to collaborate on a portfolio basis to align the features over two programs". [OPO]

Conclusion on Large-Scale Agile Planning

- Qualitative Model of what influences agile planning
  - Overview of key aspects of collaborative planning in large-scale agile development.
  - Large-scale agile planning not only depends on team abilities or skill, but also on the context in which those teams operate.
  - Ceremonies and practices on inter-team and inter-product level are currently missing and invite further research.

- Outlook: We encourage constructive research to provide improvement for one or several aspects. Our vision is a collection of best, or at least good, practices for each area in out model.

Part 3:

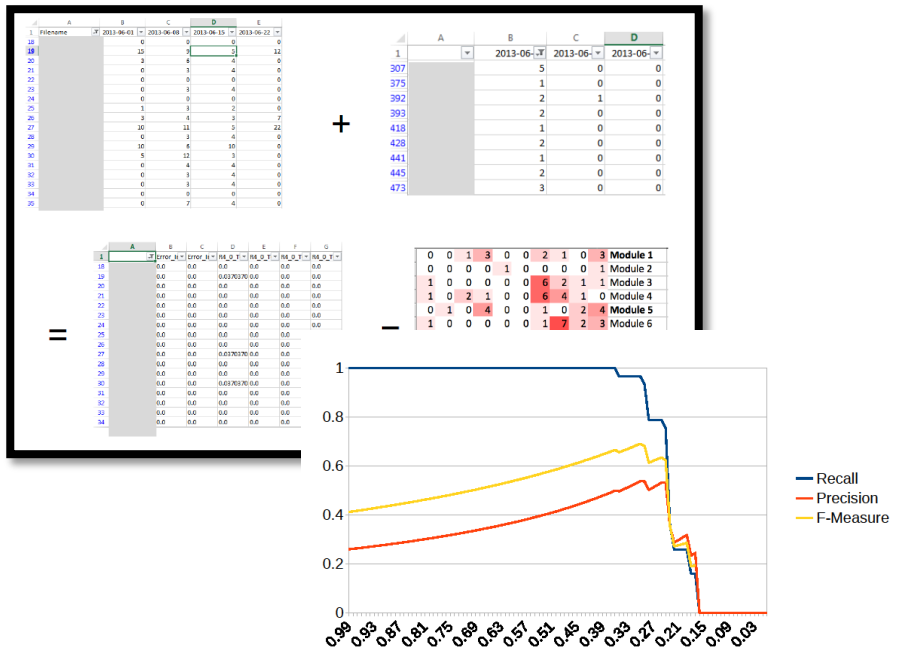
Testing Challenges in Continuous Software Engineering

TEST CASE SELECTION

CCTS

Software Center

The Code-Churn Test Selection model identifies the most optimal test suite based on the changes in the source code



- Reduction of test suite by 73% without any loss of effectiveness
- Can speed up continuous integration and reduce cycle times
- Can be applied at all test levels

- E. Knauss, M. Staron, W. Meding, O. Söder, A. Nilsson, M. Castell, “Supporting Continuous Integration by Code-Churn Based Test Selection”, Proceedings of the 2nd International Workshop on Rapid and Continuous Software Engineering (RCoSE), ICSE 2015, Italy.

For more information please contact eric.knauss@cse.gu.se, agneta.nilsson@cse.gu.se and/or miroslaw.staron@cse.gu.se

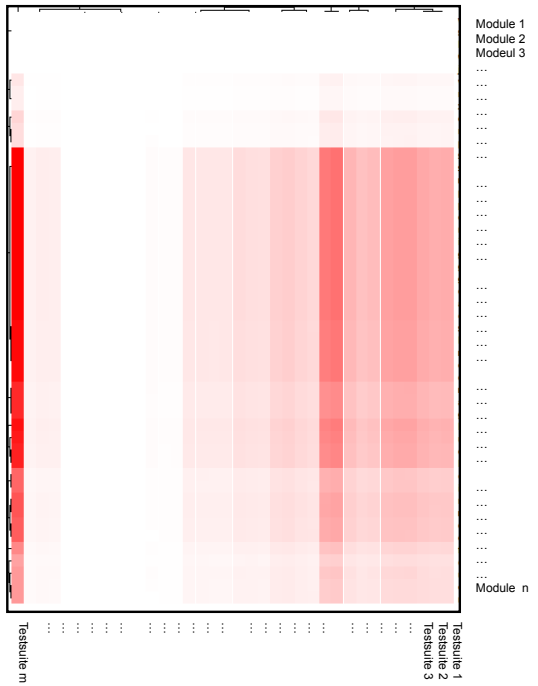
Why test prioritization & selection?

- Large product, many integration tests
  - Days or even weeks to run them all

- Continuous X = quick feedback
  - "We have 2h. Run the most important tests."
  - "As a cross-functional team, I want to run important integration tests regularly so that integration will be smooth."
  - "As an integration tester, I want to give feedback to developers as quickly as possibe."

Recommending tests (idea)

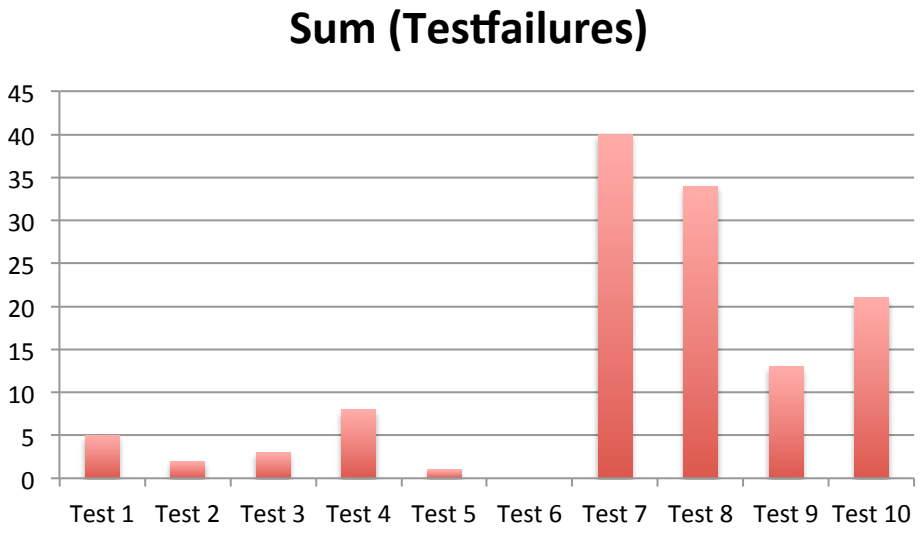
- Prioritize tests based on heatmap
  - Collect which modules/ components where changed
  - Sort tests by frequency that the failed in connection with collected changes
  - Cut off list by some criteria



Principle 1

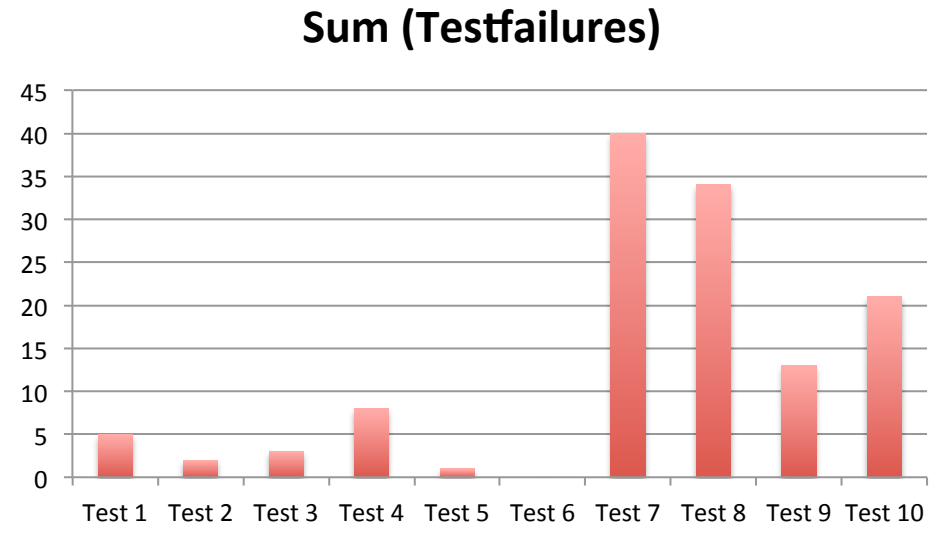
Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	
0	0	1	3	0	0	2	1	0	3	**Module 1**
0	0	0	0	1	0	0	0	0	1	Module 2
1	0	0	0	0	0	6	2	1	1	Module 3
1	0	2	1	0	0	6	4	1	0	Module 4
0	1	0	4	0	0	1	0	2	4	**Module 5**
1	0	0	0	0	0	1	7	2	3	Module 6
0	1	0	0	0	0	3	1	0	4	**Module 7**
1	0	0	0	0	0	8	5	1	1	Module 8
0	0	0	0	0	0	10	7	3	1	Module 9
1	0	0	0	0	0	3	7	3	3	Module 10

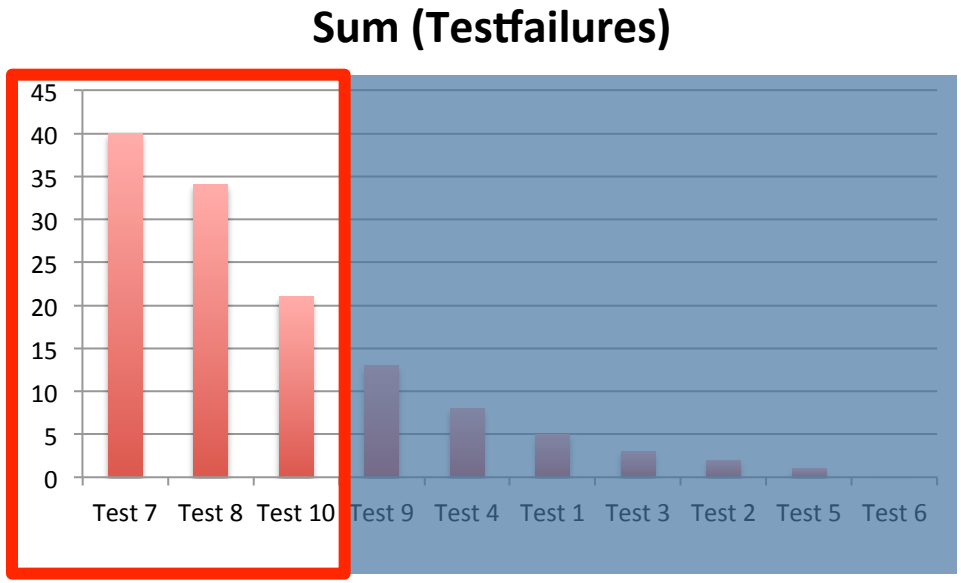
Starting from a heatmap…



…we can understand test efficiency

Principle



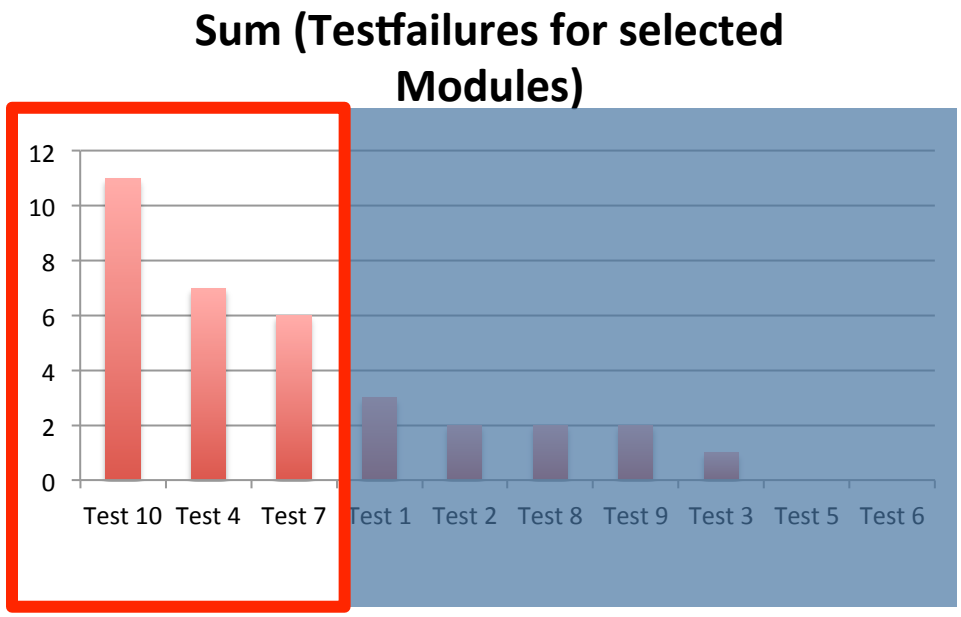


We can prioritize these tests (=sorting)

…and select tests.

Principle 2

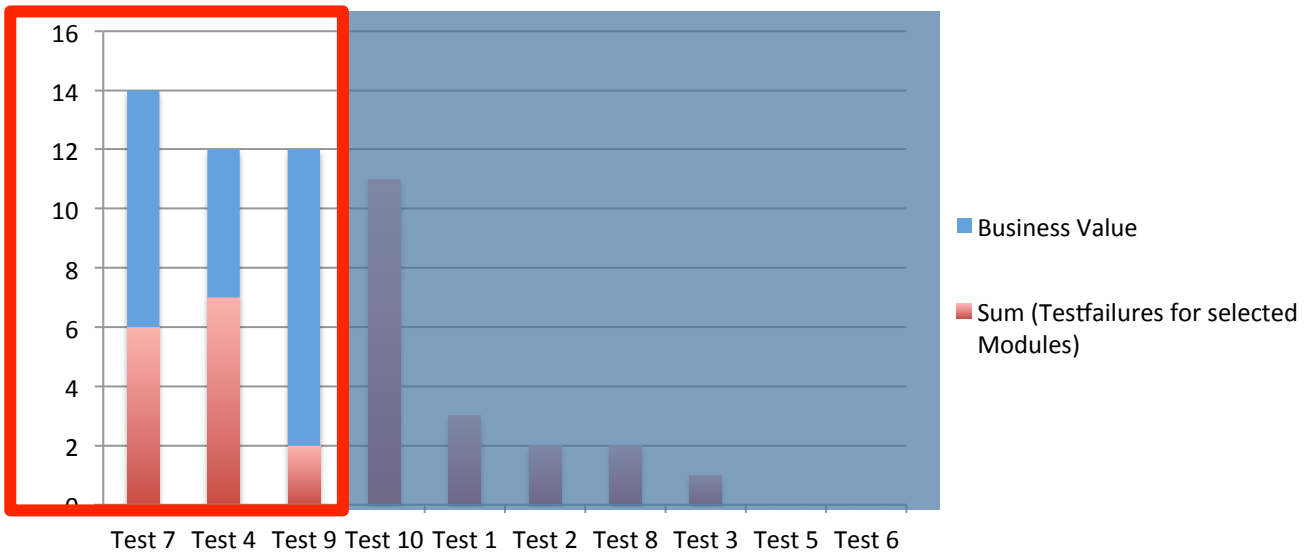
Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	
0	0	1	3	0	0	2	1	0	3	**Module 1**
0	0	0	0	1	0	0	0	0	1	Module 2
1	0	0	0	0	0	6	2	1	1	Module 3
1	0	2	1	0	0	6	4	1	0	Module 4
0	1	0	4	0	0	1	0	2	4	**Module 5**
1	0	0	0	0	0	1	7	2	3	Module 6
0	1	0	0	0	0	3	1	0	4	**Module 7**
1	0	0	0	0	0	8	5	1	1	Module 8
0	0	0	0	0	0	10	7	3	1	Module 9
1	0	0	0	0	0	3	7	3	3	Module 10
5	2	3	8	1	0	40	34	13	21	Sum
0	2	1	7	0	0	6	2	2	11	Selected Modules



If we know the modules that have recently changed…

…we can characterize test efficiency specifically for these modules

Principle 3



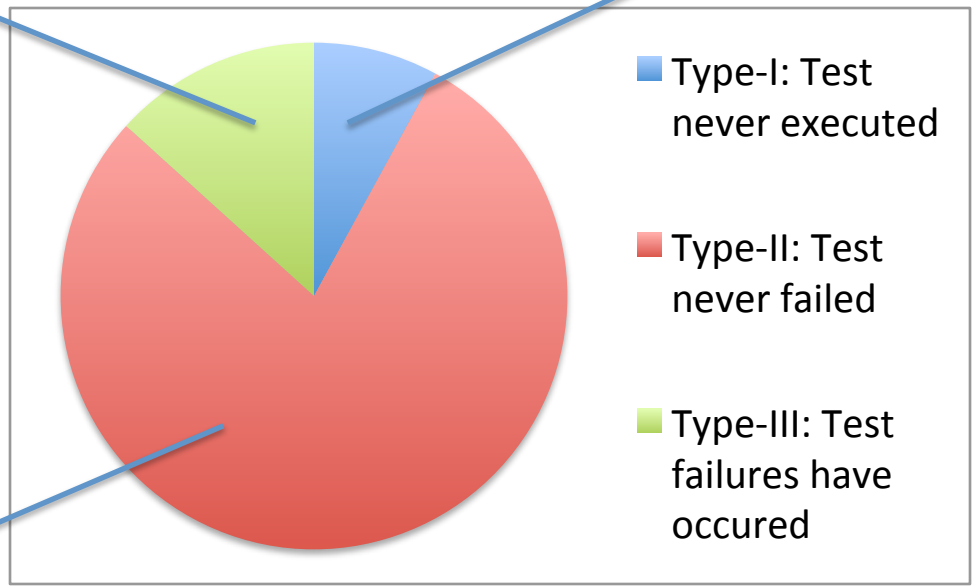
Perhaps a test is not likely to fail, but it would be strategically problematic. Therefore, we can also take into account business value

Applied to realistic data

Principle 2: Use historical data on code change

Optimize the last 13%? But might be larger in your case!

Not a selection problem



Linlin Wang: Implementation and Evaluation of an Automatic Recommender for Integration Test Cases. Master thesis at Chalmers, Gothenburg, Sweden, supervisor: Eric Knauss, examiner: Miroslaw Staron. 2015

Principle 1: Use historical data on test execution

918 Tests, history over one year (255 test executions)

Did not fail last year → don't run daily

Applied to realistic data (ongoing)

Principle 1: Use historical data on test execution

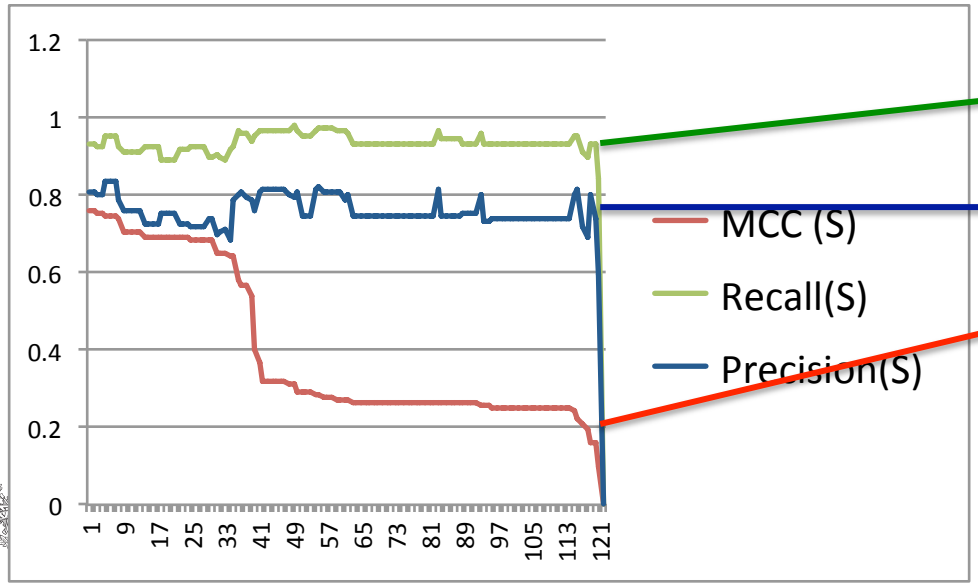
Did not fail last year → don't run daily

"You can do that tomorrow!"

Principle 2: Use historical data on code change

Optimize the last 13%? But might be larger in your case!

Still good potential for speedup.



If you apply Principle 2:

- *Only 6% of the tests that fail were not recommended*
- *Only 23% of the tests we did recommend don't fail*
- *Overall: **Moderate** strength recommendations (MCC > 0.2*
- *On average/for on third of the tests: **Strong** (MCC > 0.4) recommendations*