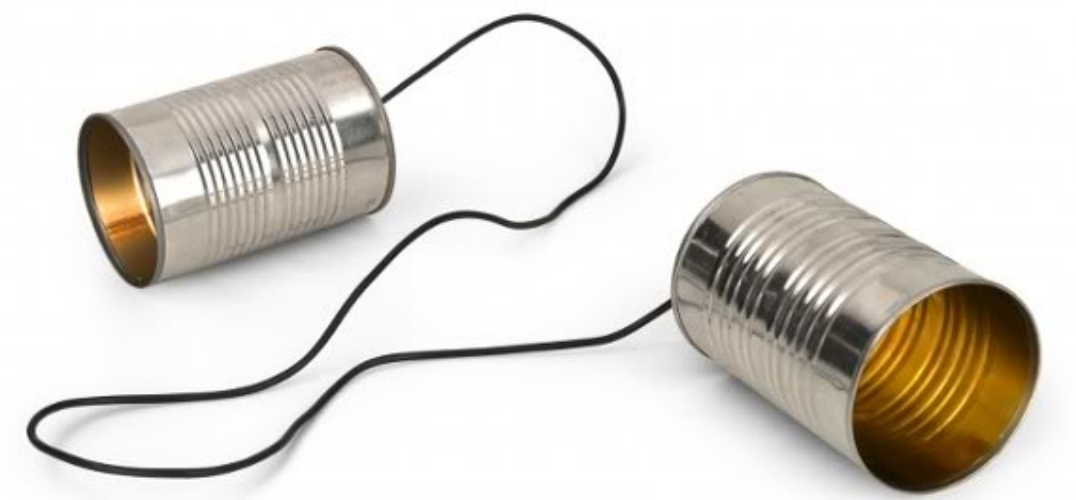
CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Agile Information Flows

Agile Development Processes

Eric Knauss



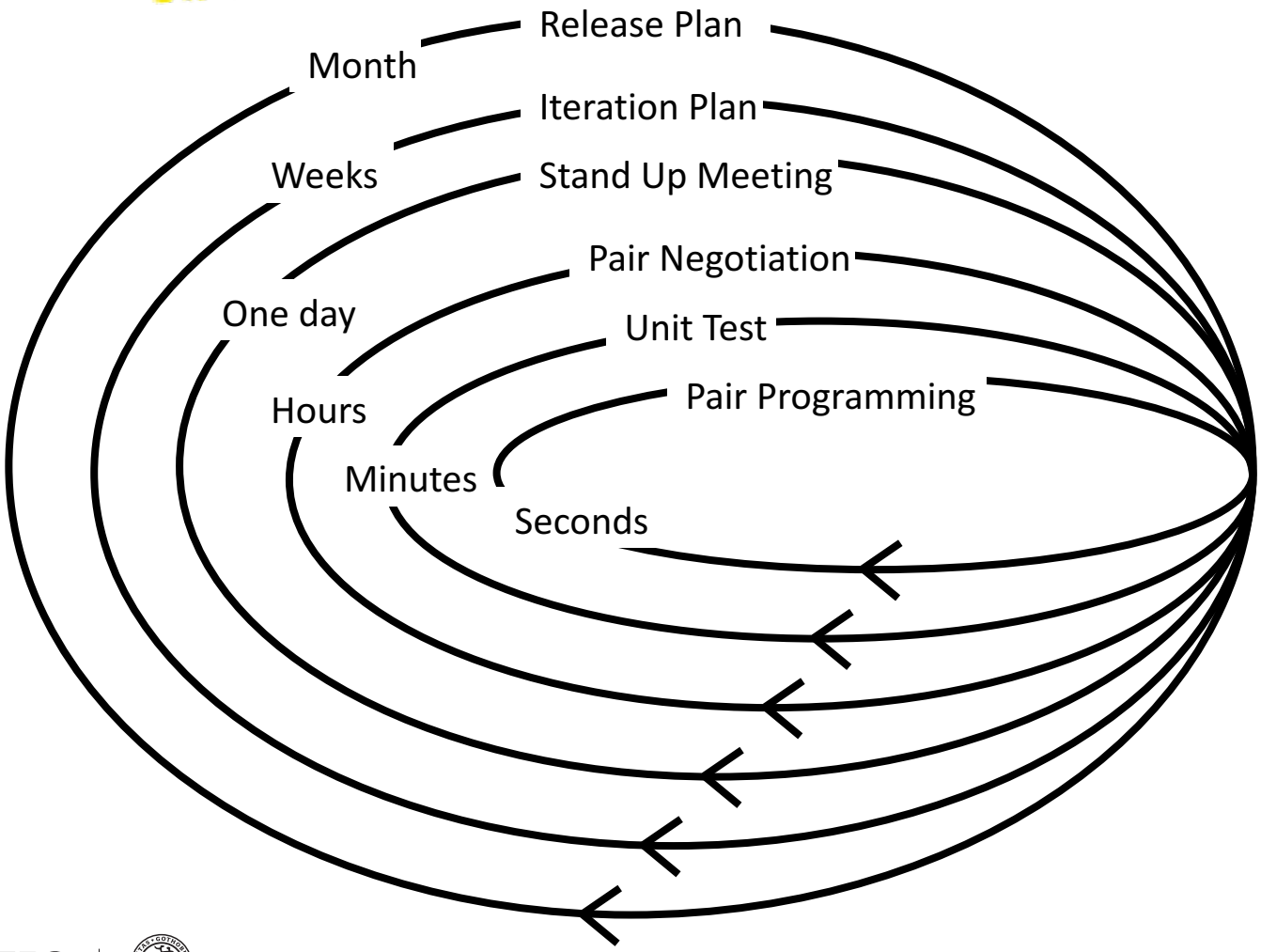
http://3badbullies.files.wordpress.com/2013/10/tin-can-telephone.jpg

Course Objectives

	Knowledge and understanding	Skills and ability	Judgement and approach
Sprint 1	Compare agile and traditional softw. dev,	Forming a team organically	Explain: people/commun. centric dev.
	Relate lean and agile development	Collaborate in small software dev. teams	Apply fact: people drive project success
	Contrast different agile methodologies	Interact and show progress continuously	Describe: No single methodology fits all
	Use the agile manifest and its accompanying principles	Develop SW using small and frequent iterations	Discuss: methodology needs to adopt to culture
	Discuss what is different when leading an agile team	Use test-driven dev. and automated tests	
Sprint 2		Refactor a program/design	
		Be member of agile team	
		Incremental planning using user stories	
Sprint 3			

Legend
Addressed
Open
Mainly in project
Focus today

Feedback in XP



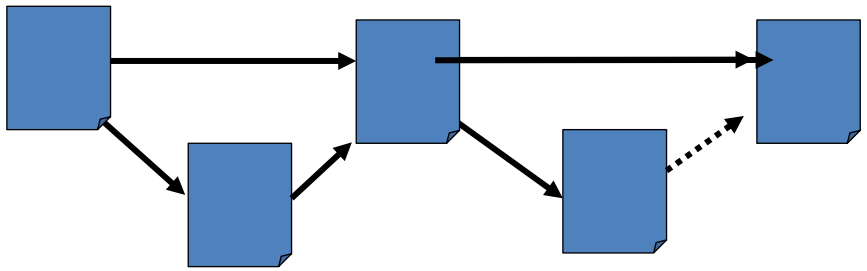
How to make this project more agile?

Consider a project with problems

- Large specification
- Frequent changes – best designers manage those

Quotes

- “It‘s just too many documents. […] Sure we need both user requts spec. and system reqts spec. But often, I change code and then go back to adjust the requirements.“
- “Why is the customer not working on the user reqts spec? Are they confused by the many changes themselves?“
- “System requirements specification? I know it is supposed to be useful. But currently I just try to keep it in sync with the unit tests we are writing.“
- “We probably should adjust the design document. It is outdated, but so far we seem to be all on the same page. It would be such a pain to bring it up to date!“

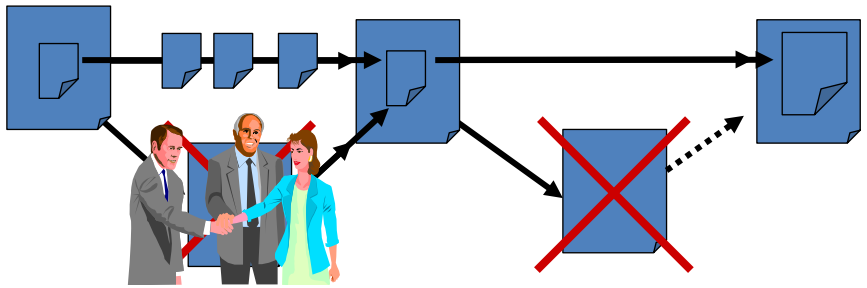


Task (10min):
How to make this project more agile?

- Remove pressure through lightweight approachs
  - Discard unnecessary documents
  - Minimize process-requirements and templates

- Provide for vague requirements and changes
  - Quickly to the core system, then incremental evolution

- Better feedback
  - Organizational and technical change
  - Closer collaboration with customer

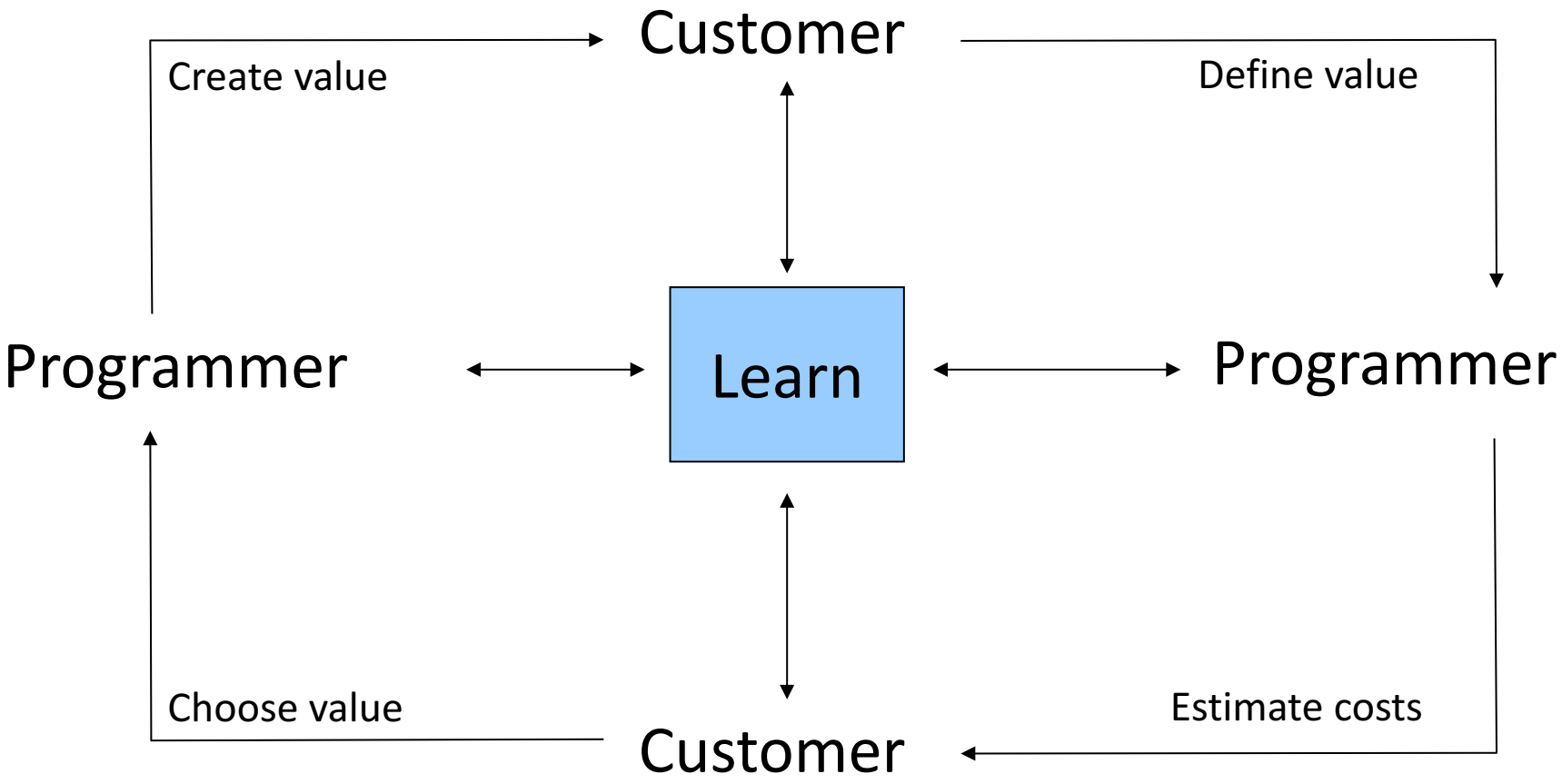


Thought experiment

Ideal transfer of information: *not* via Documents!

- Starting point: face-to-face
  - Spatial proximity: Gestures, expressions etc.
  - "Osmotic communication"
- Remove co-location: Video-Conference
  - Synchronous seeing and hearing
- Remove visual channel: Telephone
  - Synchronous listening, questions, and feedback
- Remove audio channel: email
  - Questions and feedback possible, but written and with delay
- Remove questions and feedback
  - Read documents (e.g. on paper): So much is missing here!

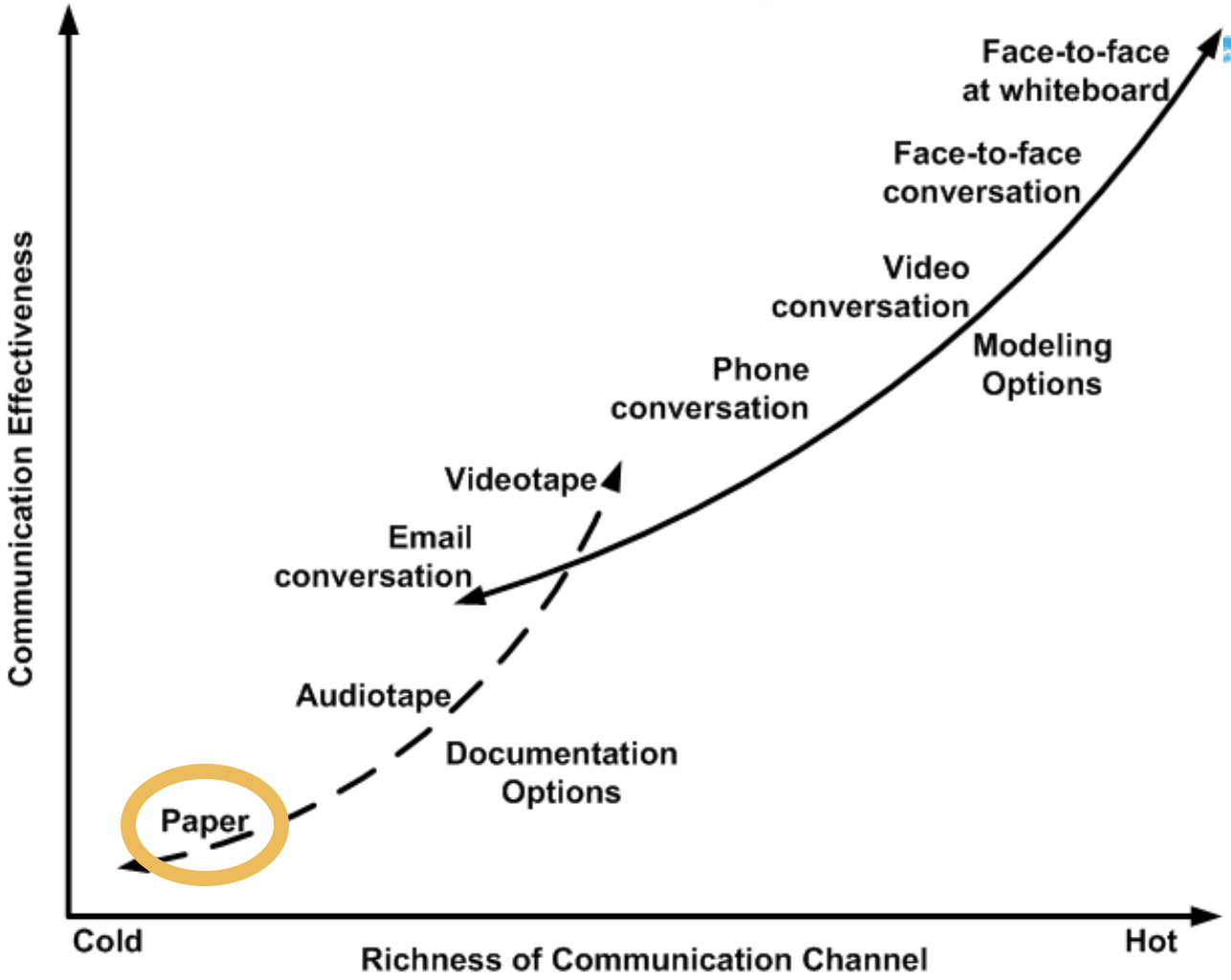
Lifecycle of an XP project



Ron Jeffries et.al. XP installed

What do project members learn from each other?

Modes of Communication



http://www.agilemodeling.com/essays/communication.htm

Task (15min)

- In small groups: Choose either XP or Scrum
- Assume you are agile coaches for a team of 8 developers
  - BUT: 5 work here, 2 in Helsinki, 1 in New York
- How do you make this work?
  - Which reoccurring, scheduled information flows are needed?
  - Which ad hoc information flows are needed?
  - Which continuous information flows are needed?
- What communication technology do you use? When?

FLOW Mapping

One approach to the previous task

Kai Stapel et al.: FLOW Mapping: Planning and Managing Communication in Distributed Teams. In Proceedings of 6th IEEE International Conference on Global Software Engineering (ICGSE ’11), pages 190–199, Helsinki, Finland, 2011.

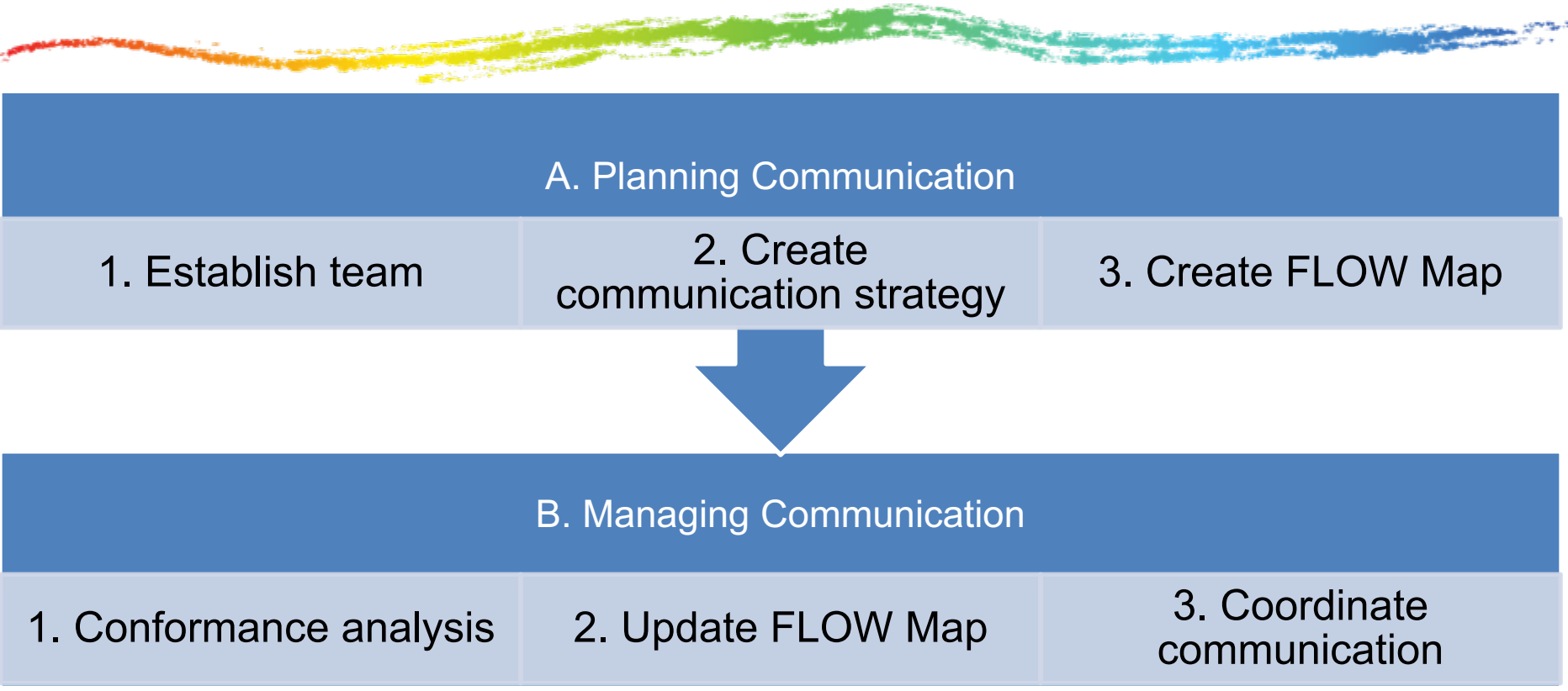
Problem and Proposed Solution

- Communication in a distributed setting is more difficult
  - Unfamiliarity with each other
  - Limited communication media
  - Informal communication does not happen as naturally

➢ FLOW Mapping, a systematic approach for planning and managing communication in distributed projects
  - 2 phase process
  - Support for process steps

Stapel et al. (2011): Flow Mapping

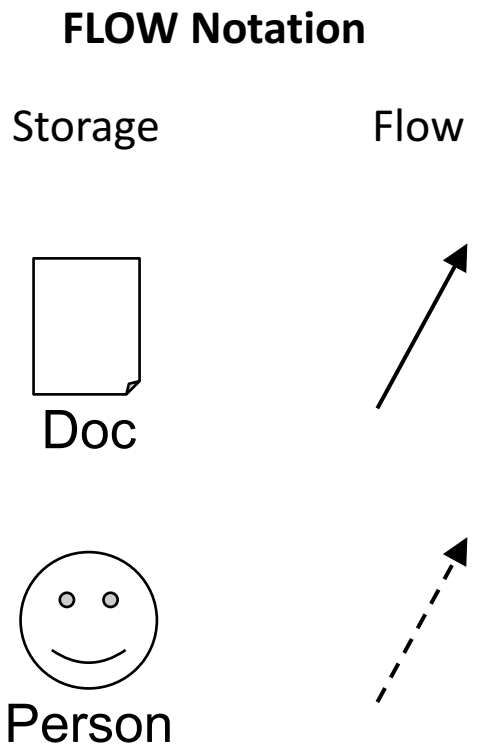
FLOW Mapping Process



Stapel et al. (2011): Flow Mapping

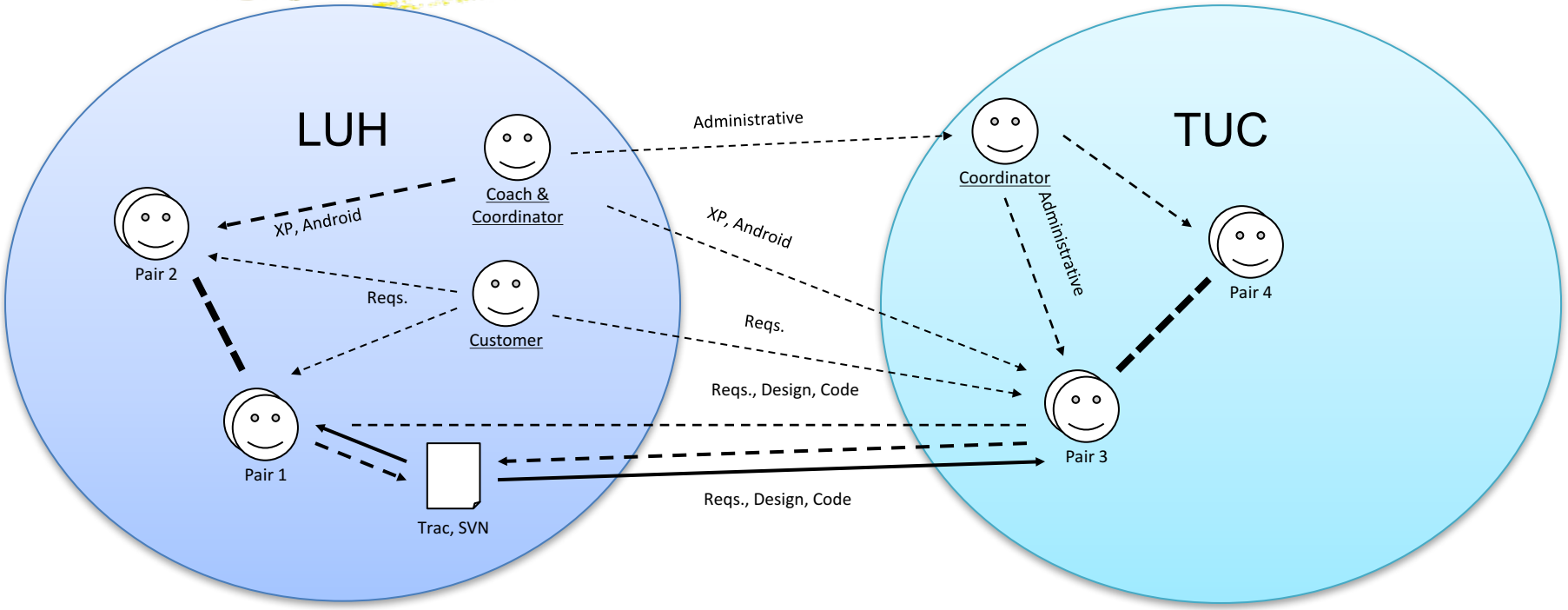
FLOW

- FLOW Mapping is based on FLOW
  - Information flow perspective on software development
  - Informal communication incorporated
  - Metaphor of state of information
- Solid information is
  - Long term accessible
  - Repeatable accessible
  - Understandable by third parties
- Fluid information is **not** solid, i. e. one of the above criteria is not met
- Notation to visualize information flows



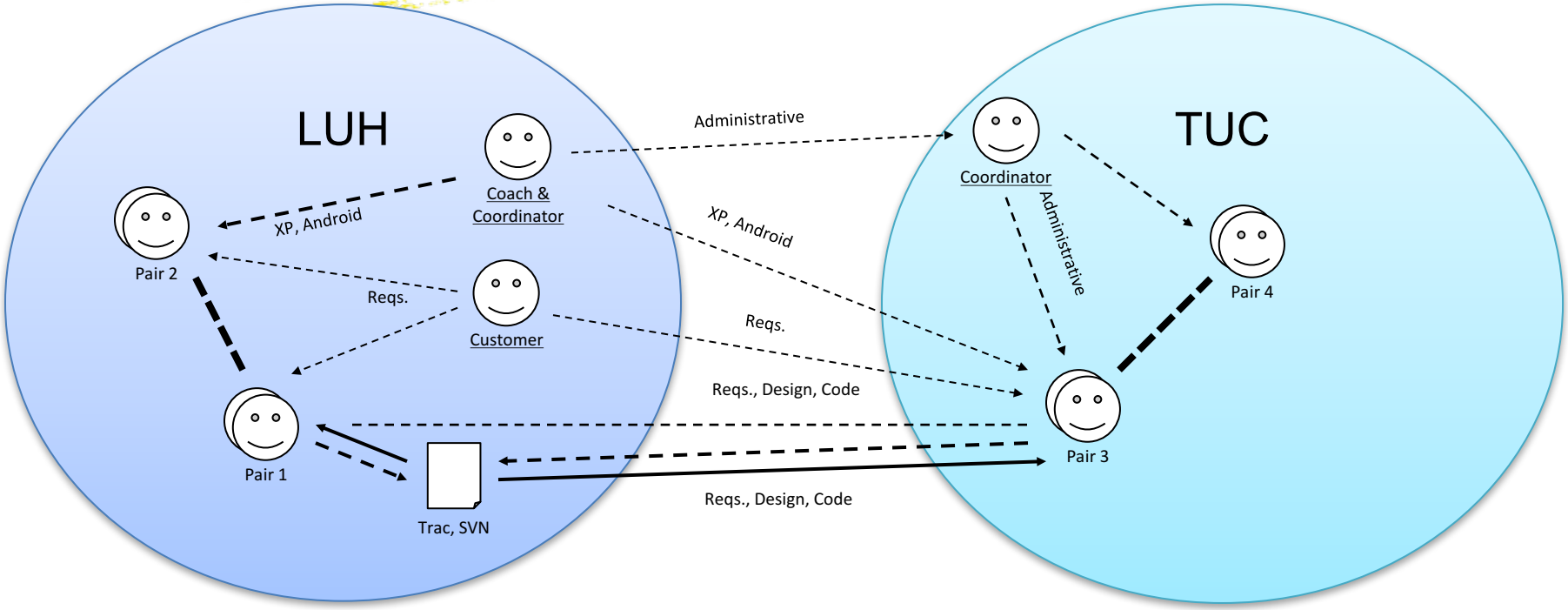
Stapel et al. (2011): Flow Mapping

FLOW Map – Example



Stapel et al. (2011): Flow Mapping

FLOW Map – Example



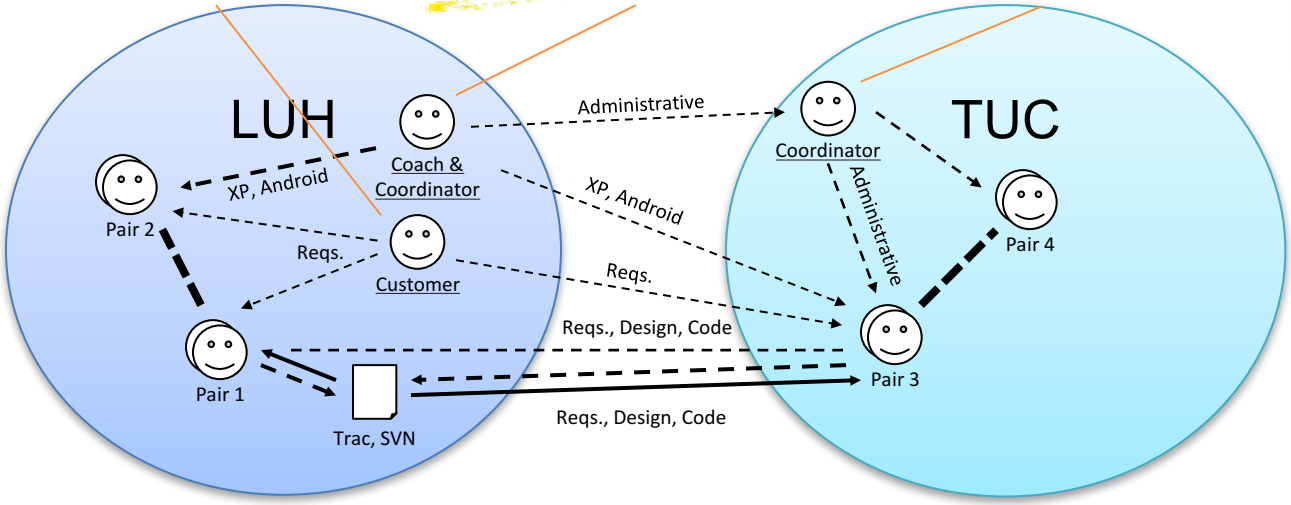
Stapel et al. (2011): Flow Mapping

******** *********
Customer
Skype: **********

******** *********
Coach
Skype: ****.*******

******** *********
Coordinator
Skype: ******.********

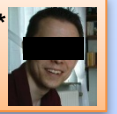
Links
WebConf:
http://webconf.*******/********/
Trac:
https://trac.se.uni-hannover.de/********/



Skype ID: pair1-H

Story Card #B9
Naming conventions
Estimate: -

******** *********
Developer (H)
MySQL

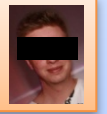


******** *********
Developer (H)
GUI, Swing

Skype ID: pair2-H

Story Card #T3
Adjust fonts
Estimate: 1

******** *********
Developer (H)
Android, GWT



******** *********
Developer (H)
UML, Patterns

Skype ID: pair3-C

Story Card #32
Core questions
Estimate: 3

******** *********
Developer (C)
GUI, SWT

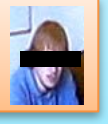
******** *********
Developer (C)
Hibernate

Skype ID: pair4-C

Story Card #T2
Data persistece
Estimate: 2

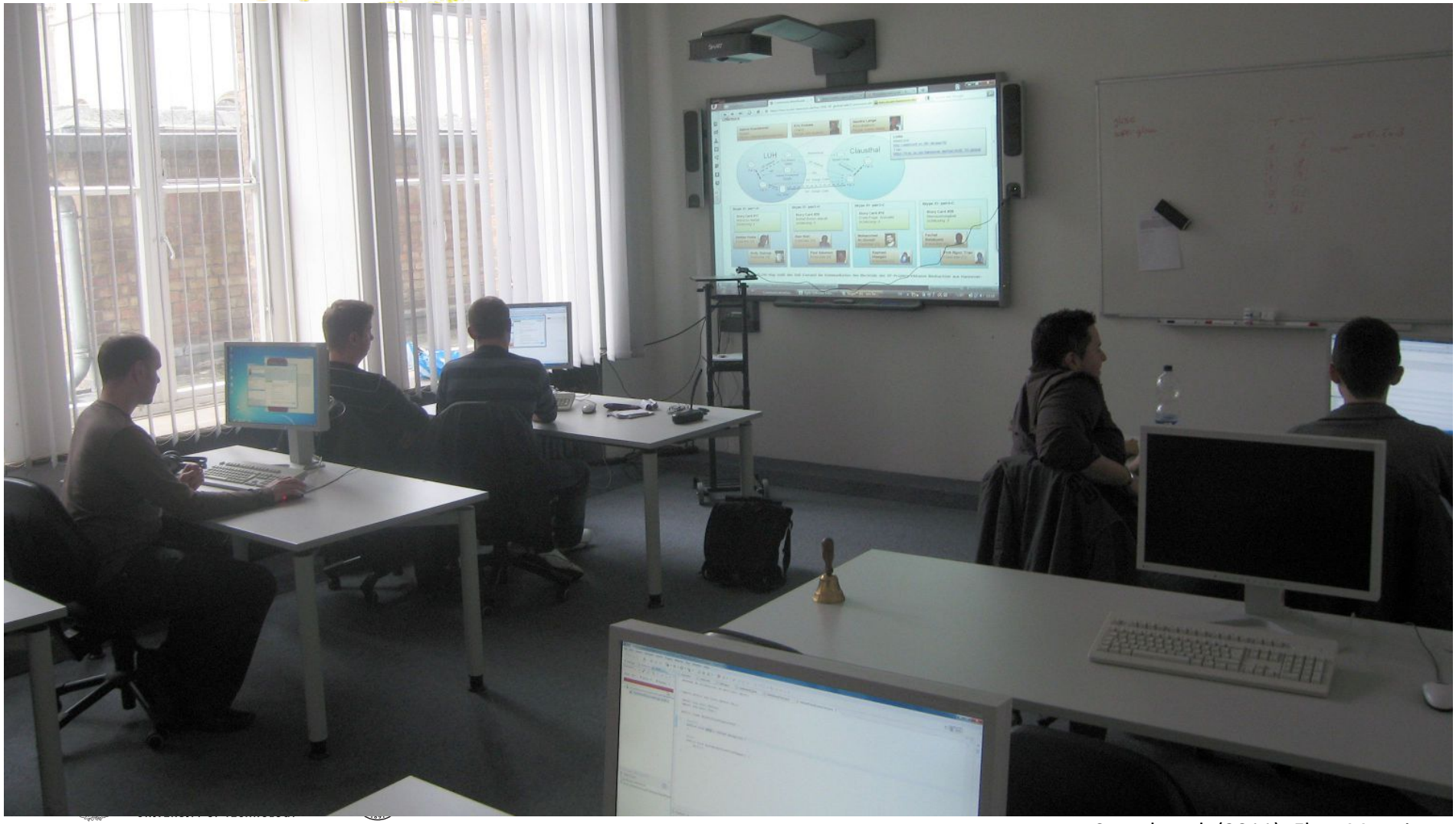
******** *********
Developer (C)
Trac, Bugzilla

******** *********
Developer (C)
Eclipse, GWT



Stapel et al. (2011): Flow Mapping

FLOW Map in Action

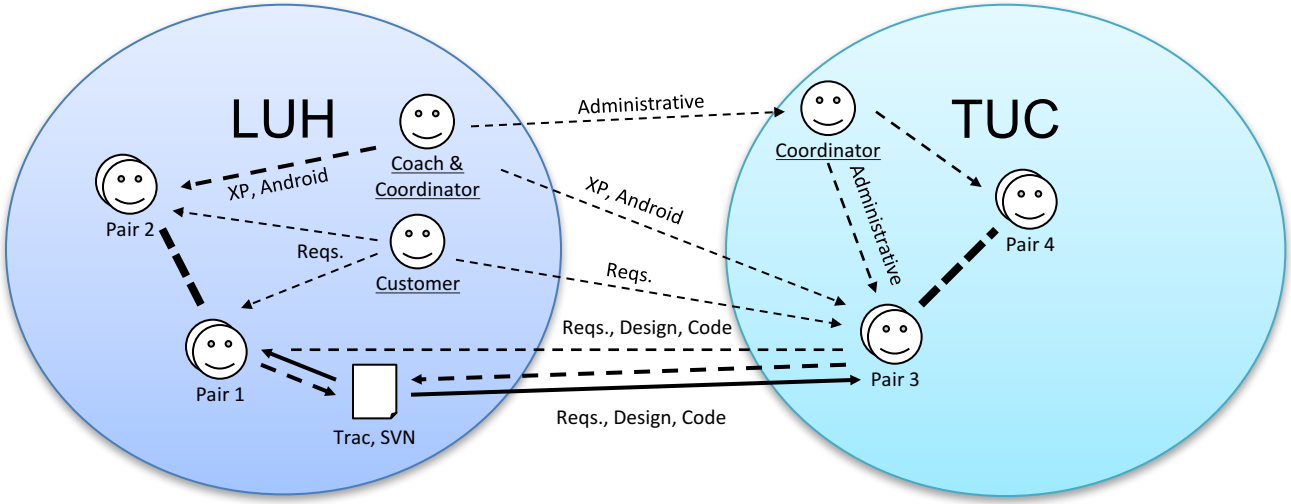


Stapel et al. (2011): Flow Mapping

Plan Communication – Establish Team

A. Planning Communication

1. Establish team | 2. Create communication strategy | 3. Create FLOW Map



******** *********
Developer (H)
MySQL

******** *********
Developer (H)
Android, GWT

******** *********
Developer (C)
GUI, SWT

******** *********
Developer (C)
Trac, Bugzilla

******** *********
Developer (H)
GUI, Swing

******** *********
Developer (H)
UML, Patterns

******** *********
Developer (C)
Hibernate

******** *********
Developer (C)
Eclipse, GWT

Stapel et al. (2011): Flow Mapping

Plan Communication – Communication Strategy

A. Planning Communication

1. Establish team | 2. Create communication strategy | 3. Create FLOW Map

Communication activity	Schedule / event	Communication media
Stand-up[a] / Wrap-up[a]	Every morning / evening	HQ video conference
Planning game[a]	Start of iteration (~ every 2. day)	HQ video conference with shared mind map
Acceptance test of iteration	Iteration completed	HQ video conference with shared desktop
Acceptance test of user stories[a]	User story completed	Skype call with shared desktop
Informal collaboration	Ad-hoc	Skype call/chat and desktop sharing
Informal coordination	Ad-hoc	Skype call / chat
Status update[a]	Status change	Skype status

prepare conformance analysis

Plan Communication – Communication Strategy

- Status update conformance template

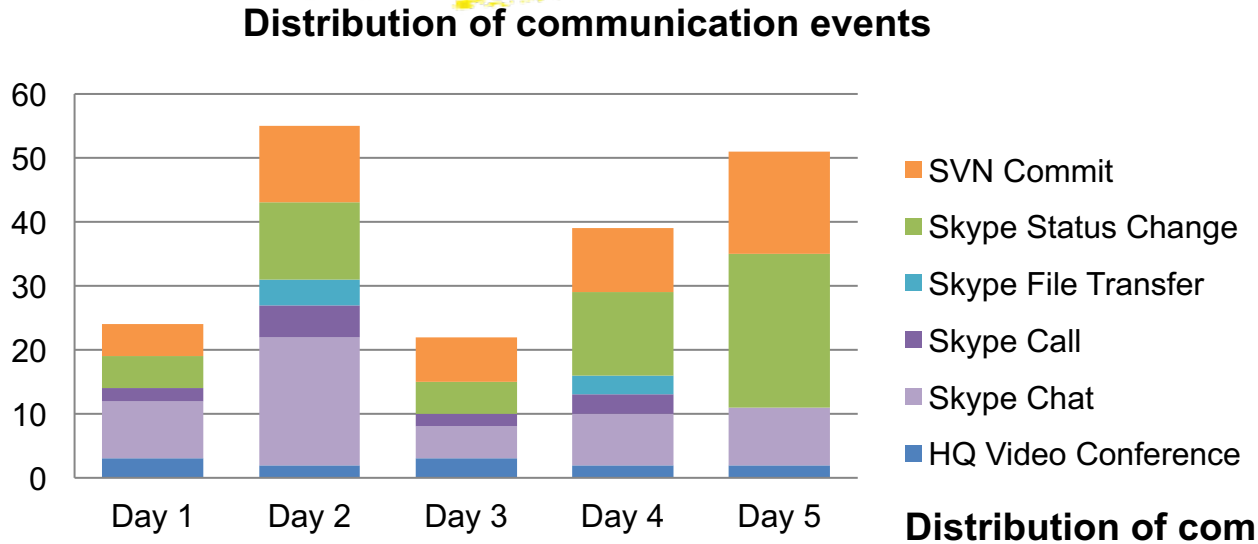
Communication Activity	Status update
Goal	
Definition	
Collected Data	
Violations	

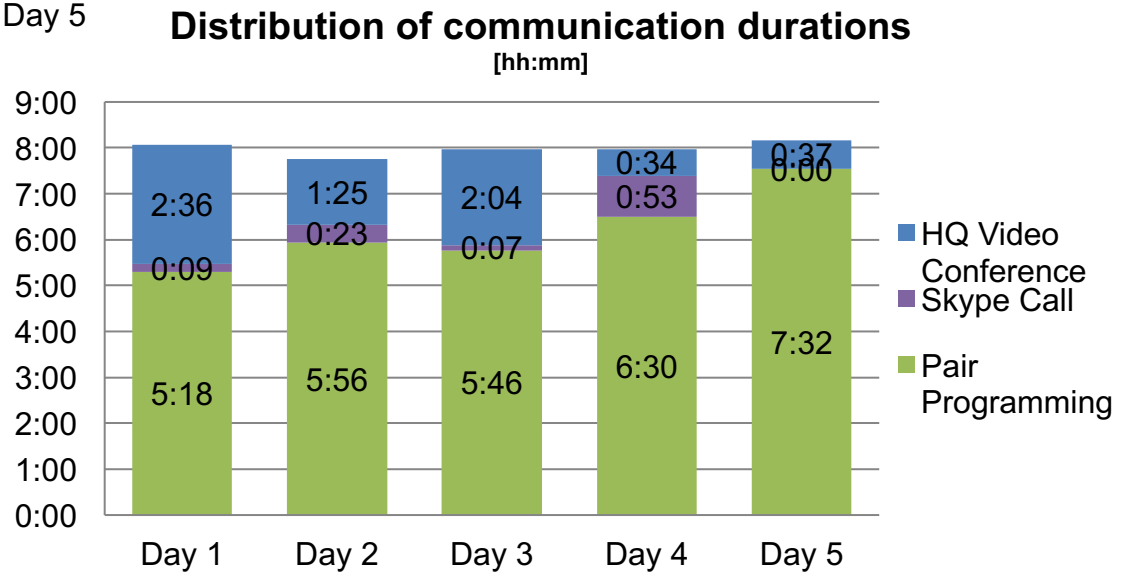
Plan Communication – Communication Strategy

- Status update conformance template

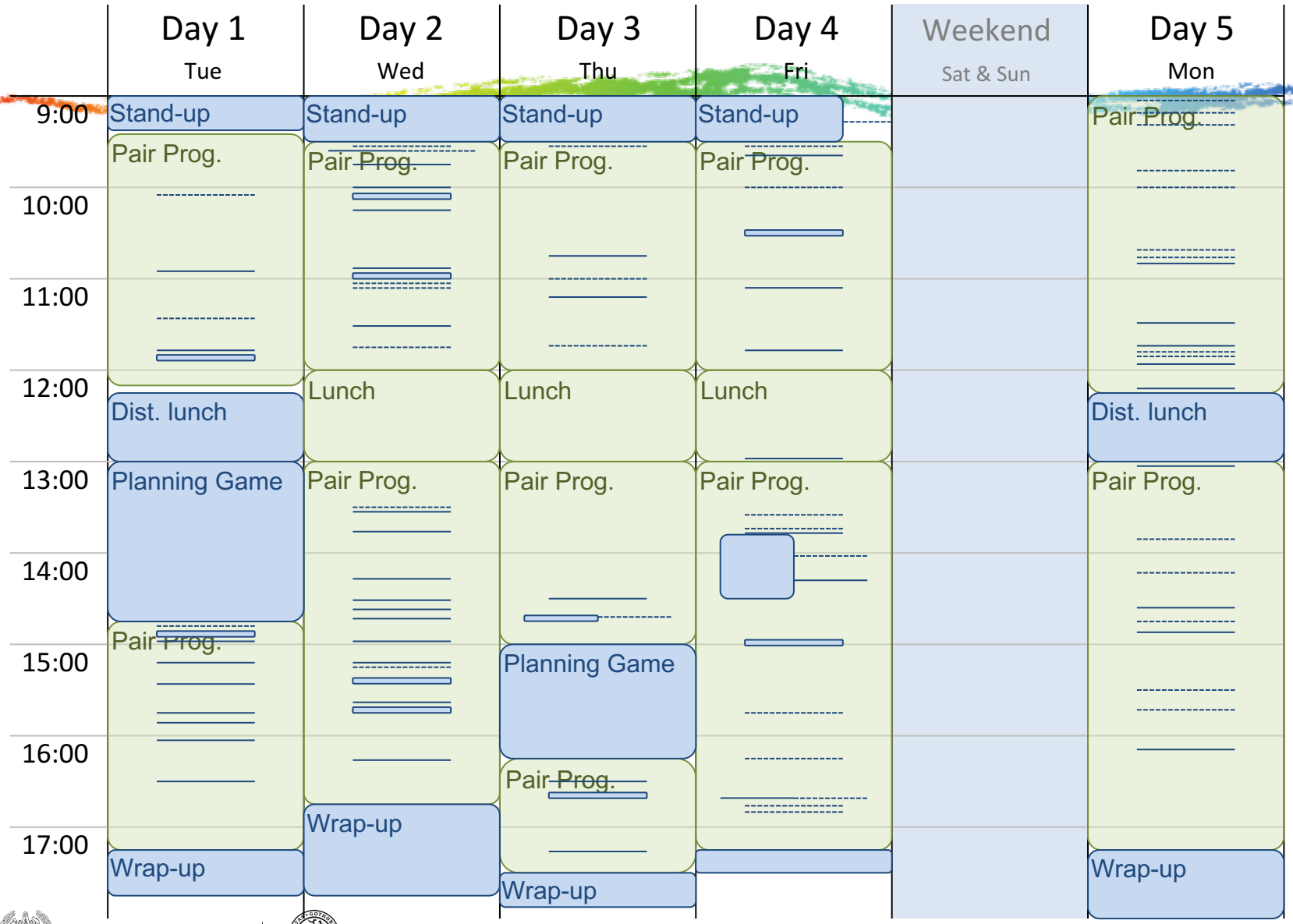
Communication Activity	Status update
Goal	Increase awareness on who is working with whom on what task
Definition	Developers should use Skype status messages to broadcast who is working with whom on which User Story in a timely manner. The status message should contain User Story ID and the names of the pair programmers.
Collected Data	Skype status log for each workstation containing: timestamp and status message and status change events (pair switches, assignment of new User Stories)
Violations	**Temporal:** (1) Status message not updated for more than one hour (2) Status message suggests that a developer is working in two pairs concurrently **Qualitative:** (1) Incomplete information, e.g. User Story ID missing.

Case Study – Communication Overview



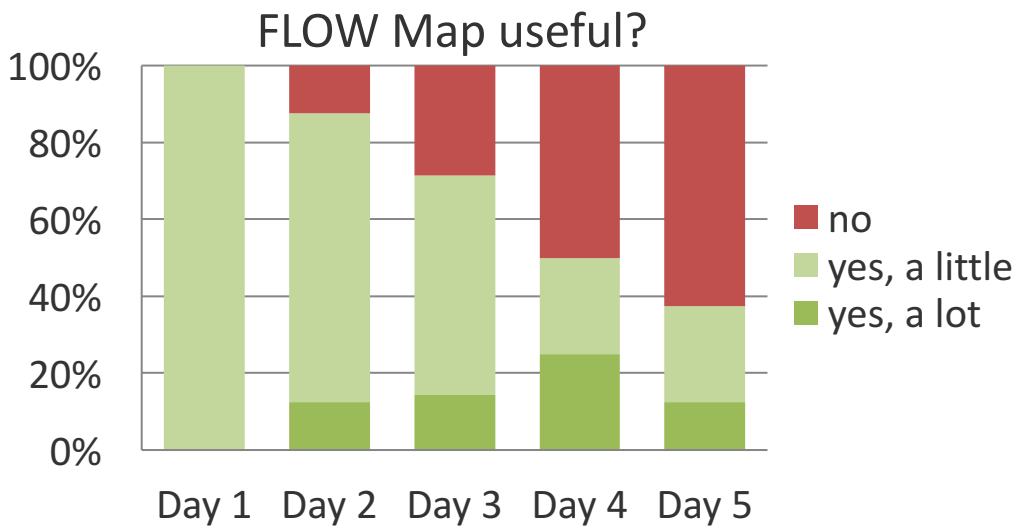


CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF


Day 1
Tue
Day 2
Wed
Day 3
Thu
Day 4
Fri
Weekend
Sat & Sun
Day 5
Mon
9:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00
17:00
Stand-up
Pair Prog.
Dist. lunch
Planning Game
Wrap-up
Lunch

Discussion

- Impact
  - FLOW Map perceived to be useful
  - Especially at project start (team grows together)
  - Problem with manual update process → tool support

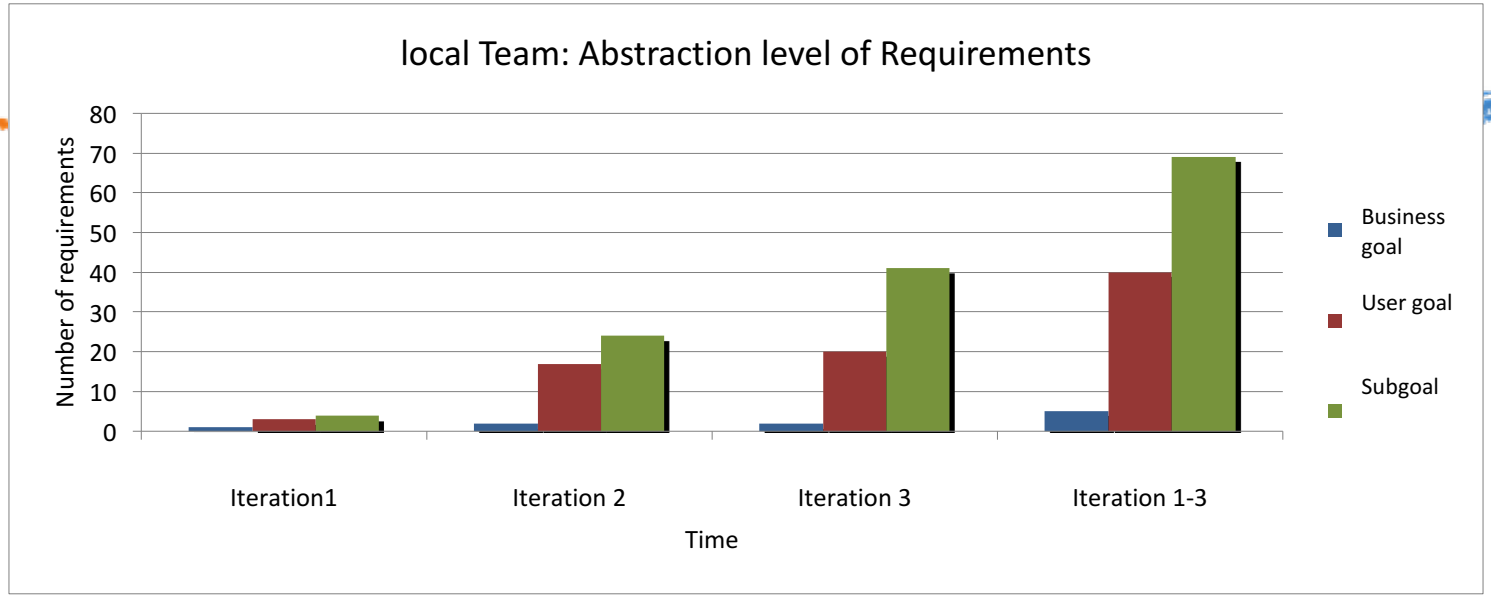


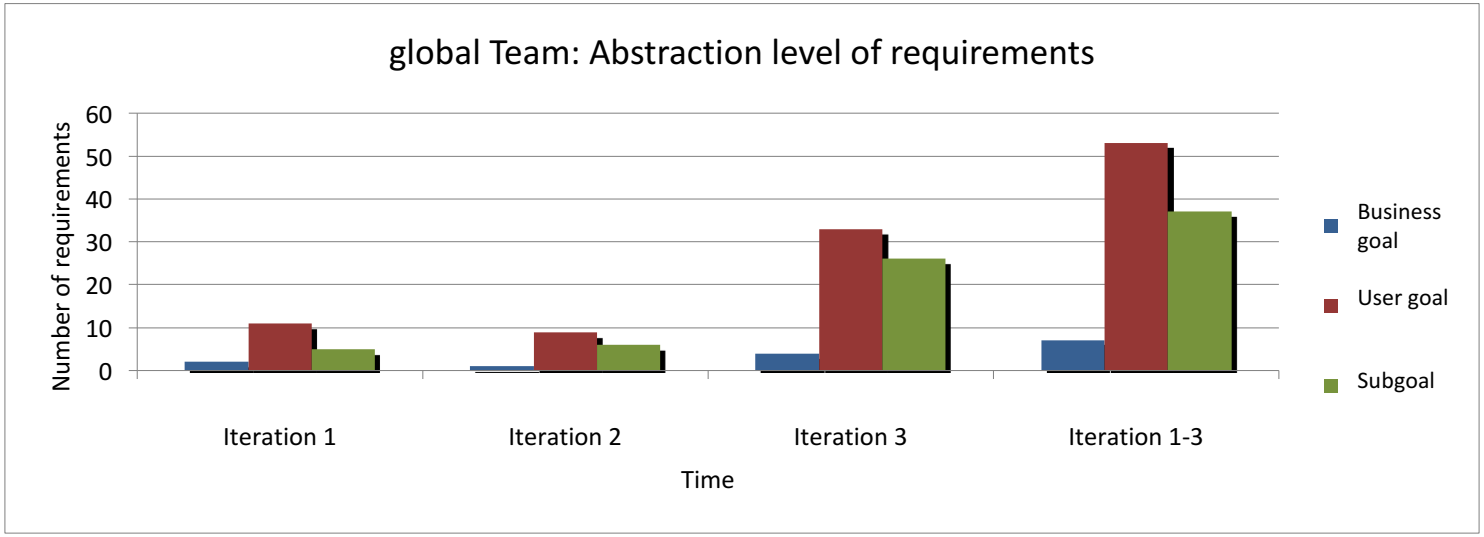
Discussion

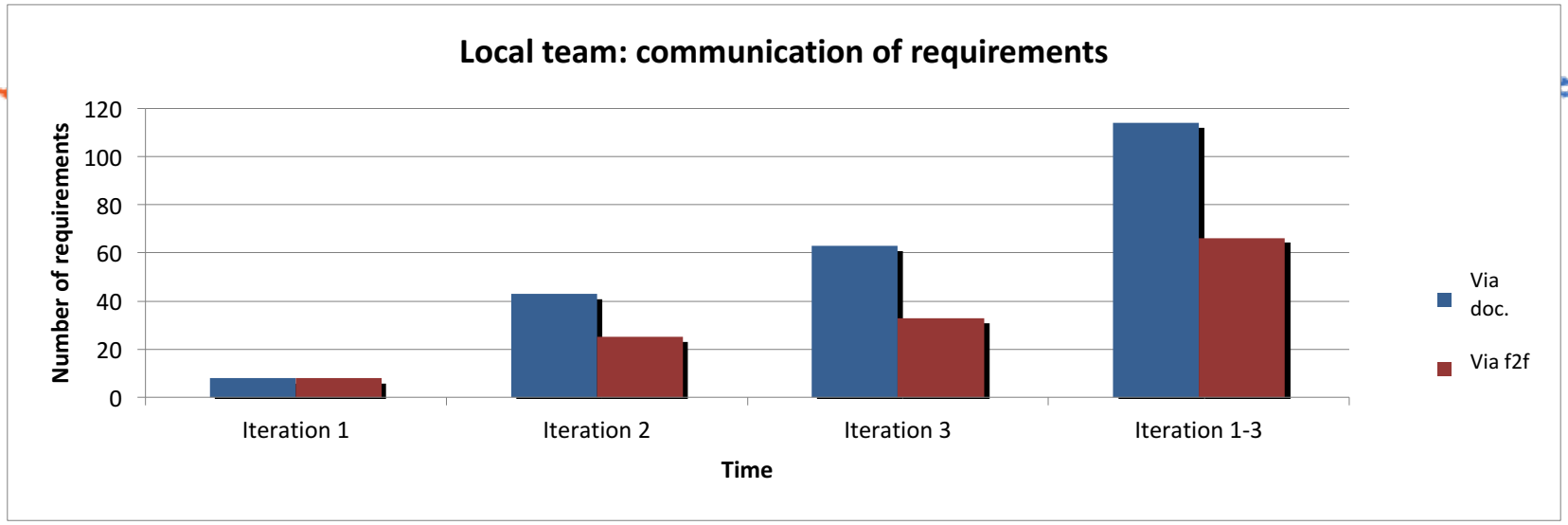
- Impact
- Cost
  - Plan: 1d strategy + 0.5d conformance + 2d prepare data collection
  - Execute: observer + 1h/activity for conformance analysis + 10 min./change to update FLOW Map
- Management feasibility
  - Violations can be detected during project
  - Monitoring electronic media helps (see costs)
- Planning feasibility
  - Communication was planned
  - Strategy was followed (79% - 88%)

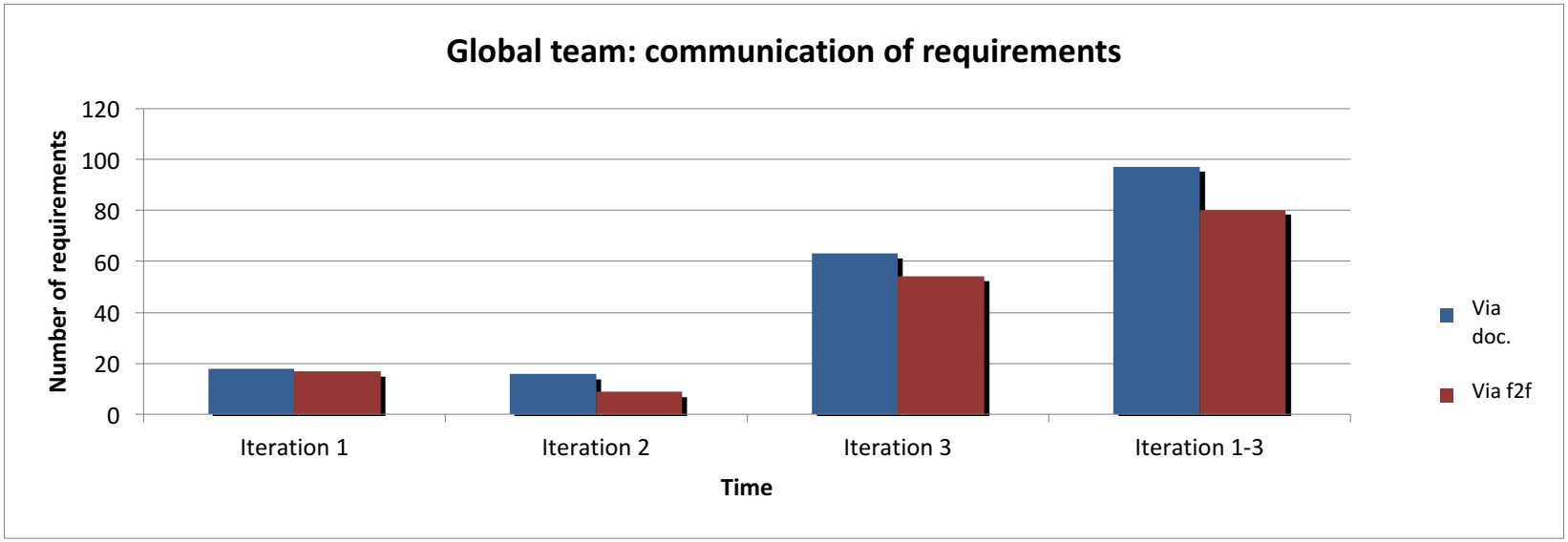
Distributed vs. Not distributed

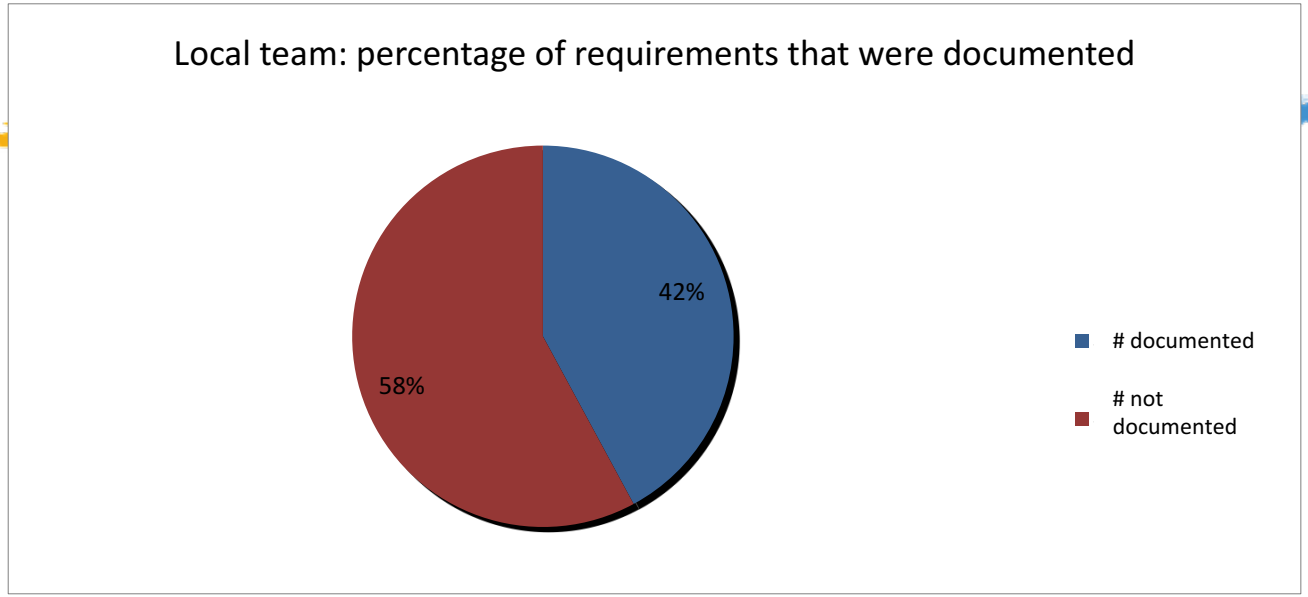
Some personal experience

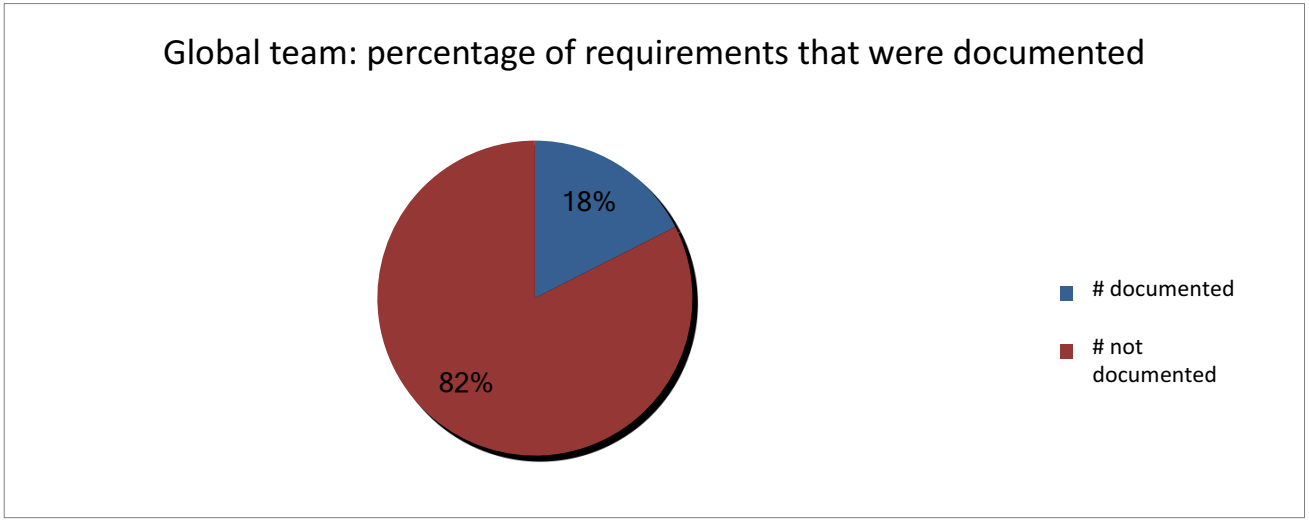

local Team: Abstraction level of Requirements
Number of requirements
80
70
60
50
40
30
20
10
0
Iteration1
Iteration 2
Iteration 3
Iteration 1-3
Time
Business goal
User goal
Subgoal


global Team: Abstraction level of requirements
Number of requirements
60
50
40
30
20
10
0
Iteration 1
Iteration 2
Iteration 3
Iteration 1-3
Time
Business goal
User goal
Subgoal


Local team: communication of requirements
Number of requirements
120
100
80
60
40
20
0
Iteration 1
Iteration 2
Iteration 3
Iteration 1-3
Time
Via doc.
Via f2f

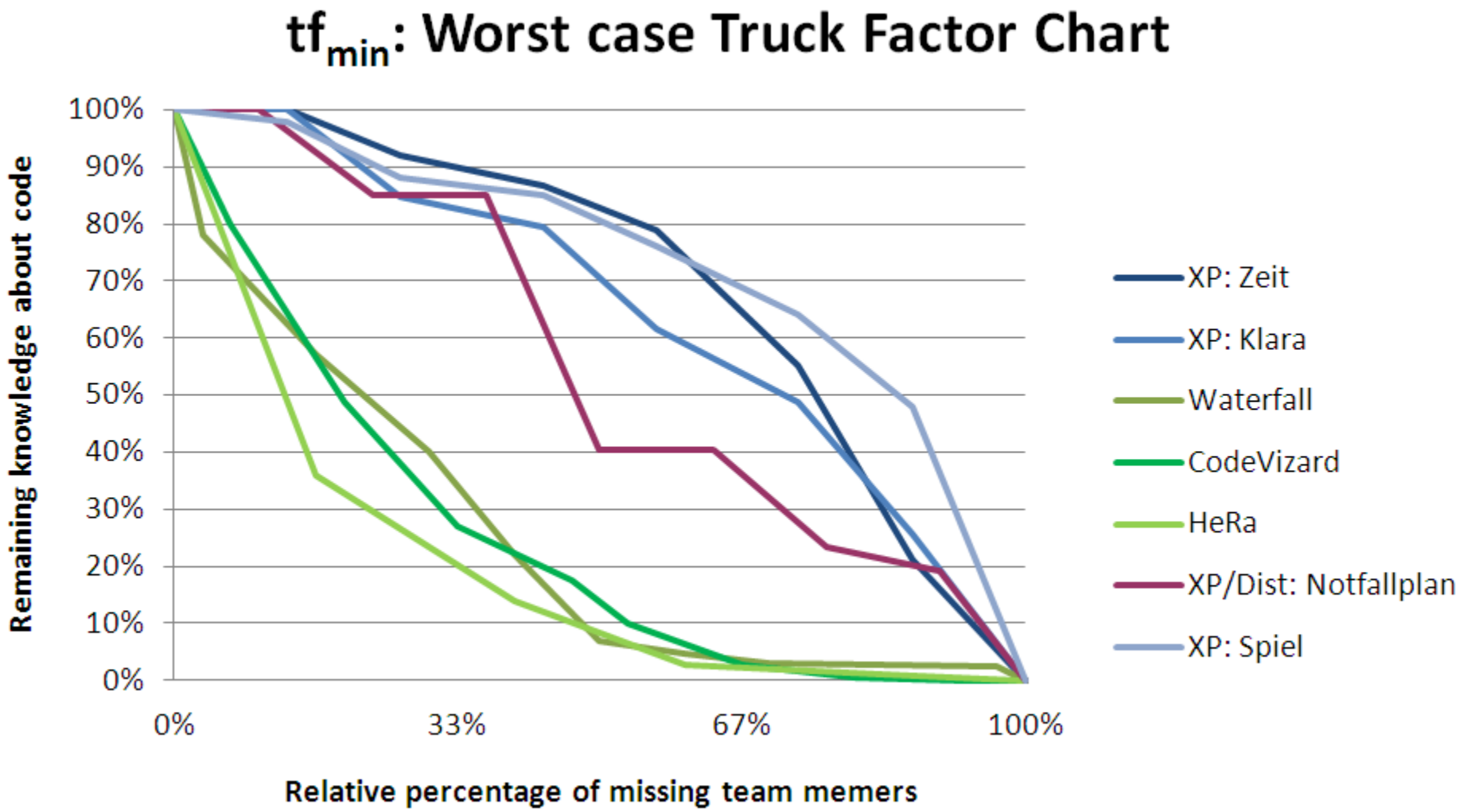

Global team: communication of requirements
Number of requirements
120
100
80
60
40
20
0
Iteration 1
Iteration 2
Iteration 3
Iteration 1-3
Time
Via doc.
Via f2f


Local team: percentage of requirements that were documented
42%
58%
documented
not documented


Global team: percentage of requirements that were documented
18%
82%
documented
not documented

Nico Zazworka, Kai Stapel, Eric Knauss, Forrest Shull, Victor R. Basili, Kurt Schneider. Are developers complying with the process: an XP study. In Proc. of the 4th Int. Symp. on Empirical Software Engineering and Measurement (ESEM '10), Bolzano-Bozen, Italy, 2010.

Truck Factor



Summary

- We did our best to make distributed agile work
- Not a surprise:
  - Truck factor analysis shows that we are not as agile in the distributed project as in the co-located one
- Interesting:
  - Distributed team discussed requirements in less detail
- Surprise:
  - Distributed team documented less

Why? Because it was @#%!$ difficult!