Human Aspects of Software Engineering

Thomas Fritz
University of Zurich

Objectives for today

- Introduction Me & Research Overview
- Introduction You
- Course Overview
- Research Projects
- Next Steps
- Exercise

Introduction

About me…

- Originally from Munich

- PhD + a few years in Vancouver

- …

Boosting (Developer) Productivity

Understanding developer productivity

- Examine productivity perceptions of individuals & teams
- Identify productive behavior & impediments to productivity

Sensing developers’ productivity

- Identify measures of productivity, focus, and task difficulty
- Examine use of biometric and computer interaction sensors

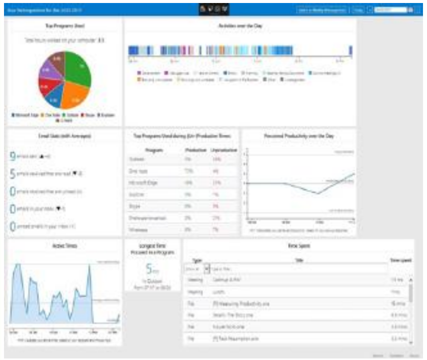
Supporting productive behavior

- Provide awareness & actionable insights
- Reduce costly interruptions & support focused work
- Prevent bugs / defects
- (Semi-)Automate Workflows

Developer Productivity

Unde

Developer Analytics



Sensi

Sensing code difficulty

Supporting

FlowLight reducing interruptions

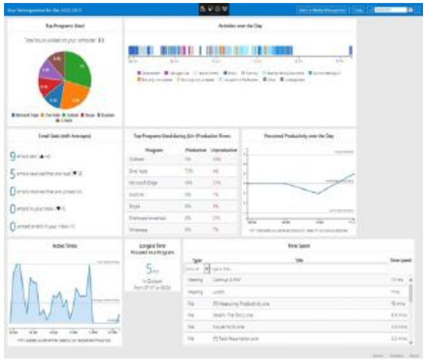
Developer Productivity

Unde

Sensi

Supporting

Developer Analytics



Sensing code difficulty

FlowLight reducing interruptions

[FSE'14]

Developer Productivity & Analytics

What does it mean for developers to be productive?

Survey

379 developers

28 questions

→

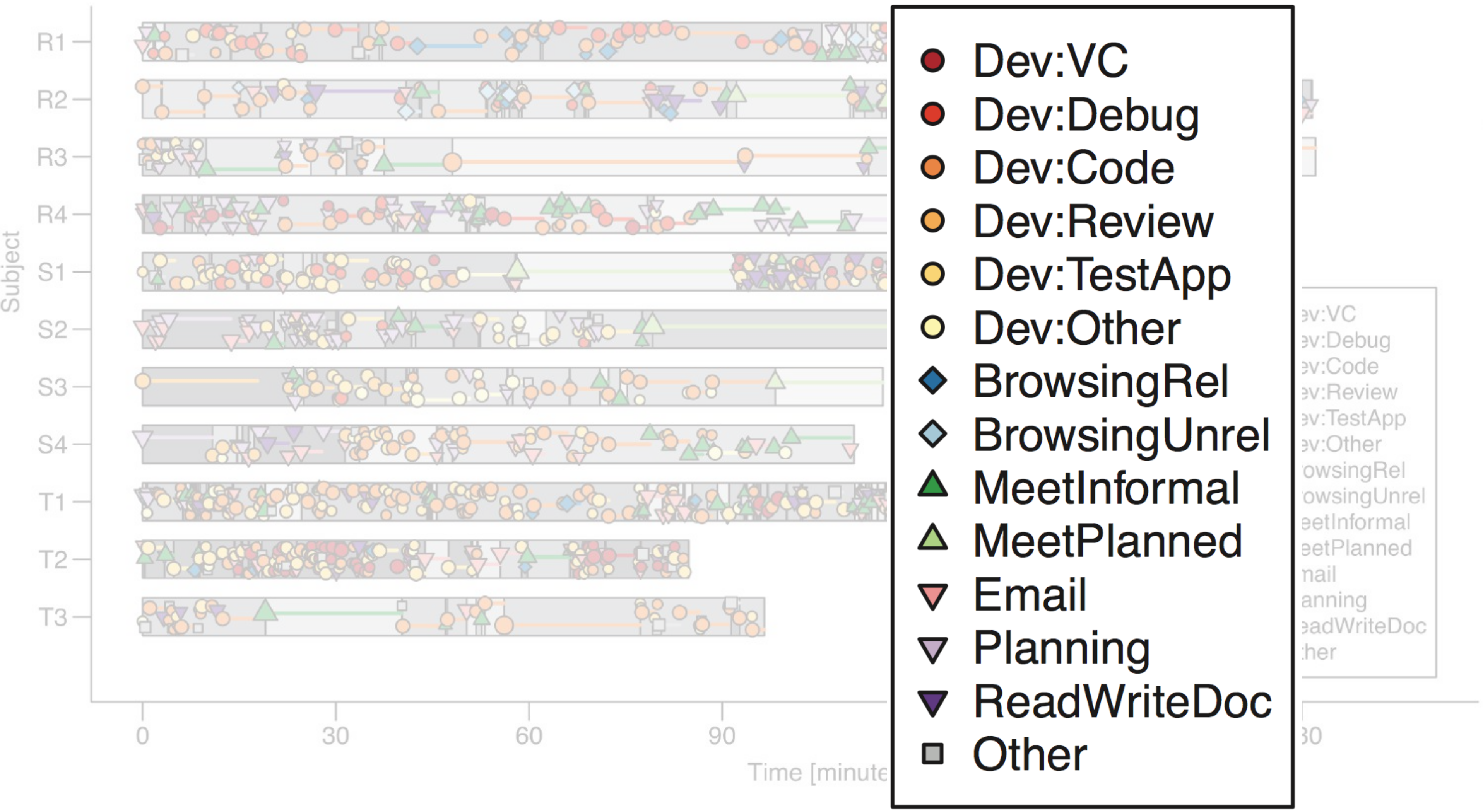
Observations

11 developers

2x2 hours, 2650 events

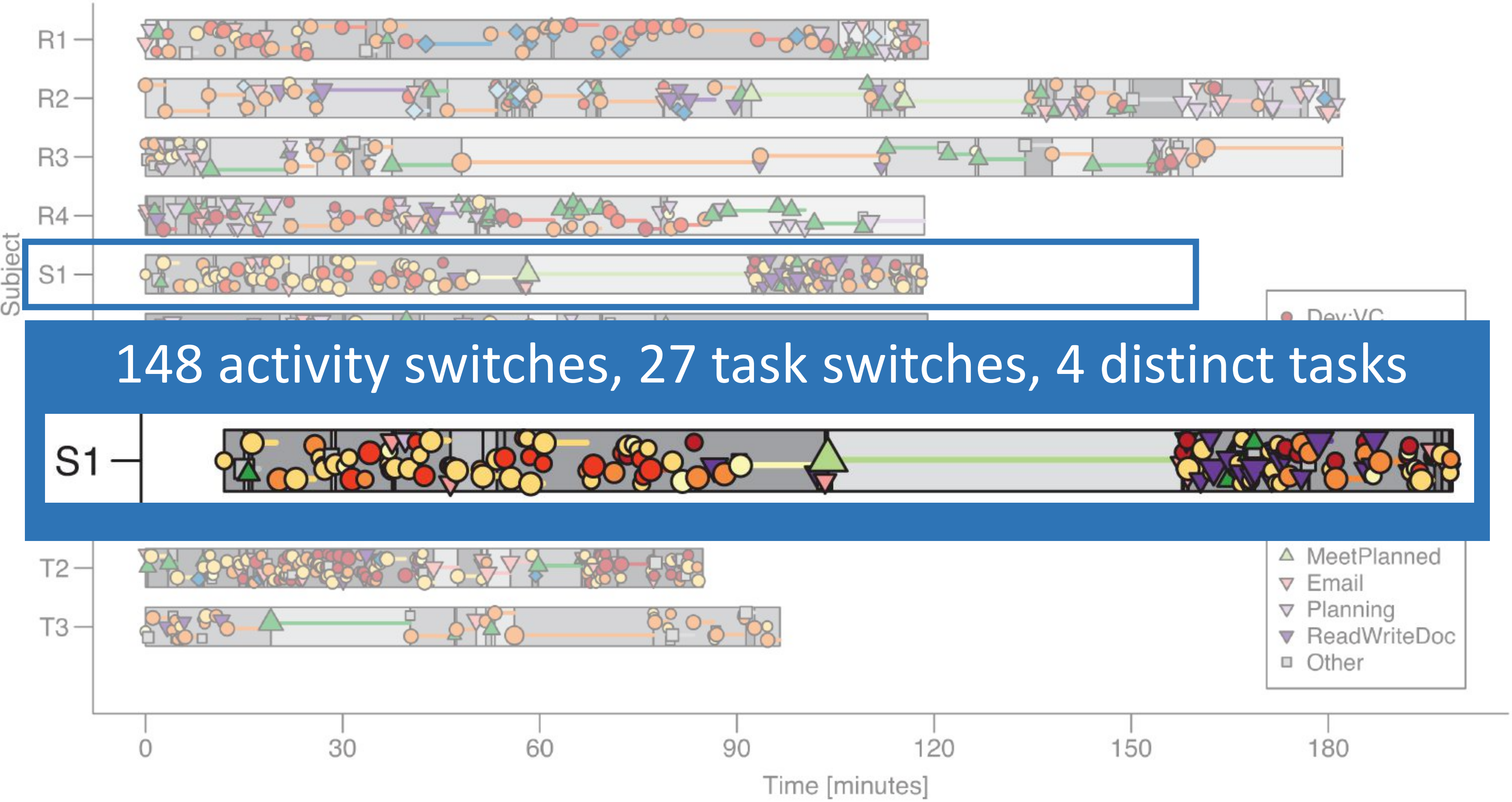
Developers feel productive when they make **progress on tasks** with **few context switches / interruptions**

Observed work flow



- ● Dev:VC
- ● Dev:Debug
- ● Dev:Code
- ● Dev:Review
- ● Dev:TestApp
- ● Dev:Other
- ◆ BrowsingRel
- ◇ BrowsingUnrel
- ▲ MeetInformal
- △ MeetPlanned
- ▽ Email
- ▽ Planning
- ▼ ReadWriteDoc
- ■ Other

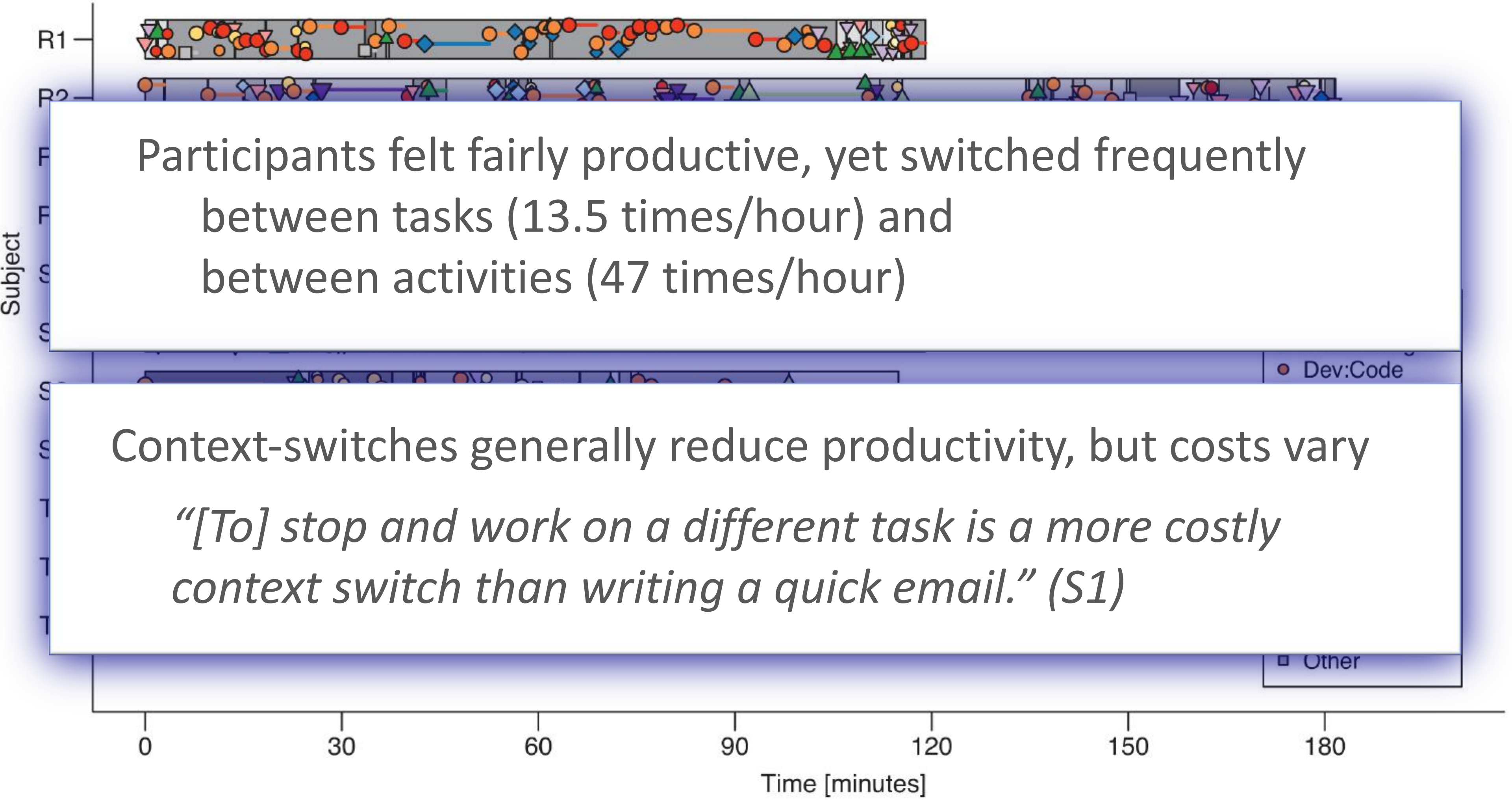
Observed work flow



148 activity switches, 27 task switches, 4 distinct tasks

[FSE’14]

Observed work flow



Participants felt fairly productive, yet switched frequently
between tasks (13.5 times/hour) and
between activities (47 times/hour)

Context-switches generally reduce productivity, but costs vary

“[To] stop and work on a different task is a more costly context switch than writing a quick email.” (S1)

Developer Analytics & Retrospection

What does it mean for developers to be productive?

Survey

379 developers

28 questions

Observations

11 developers

4 hours, 2650 events

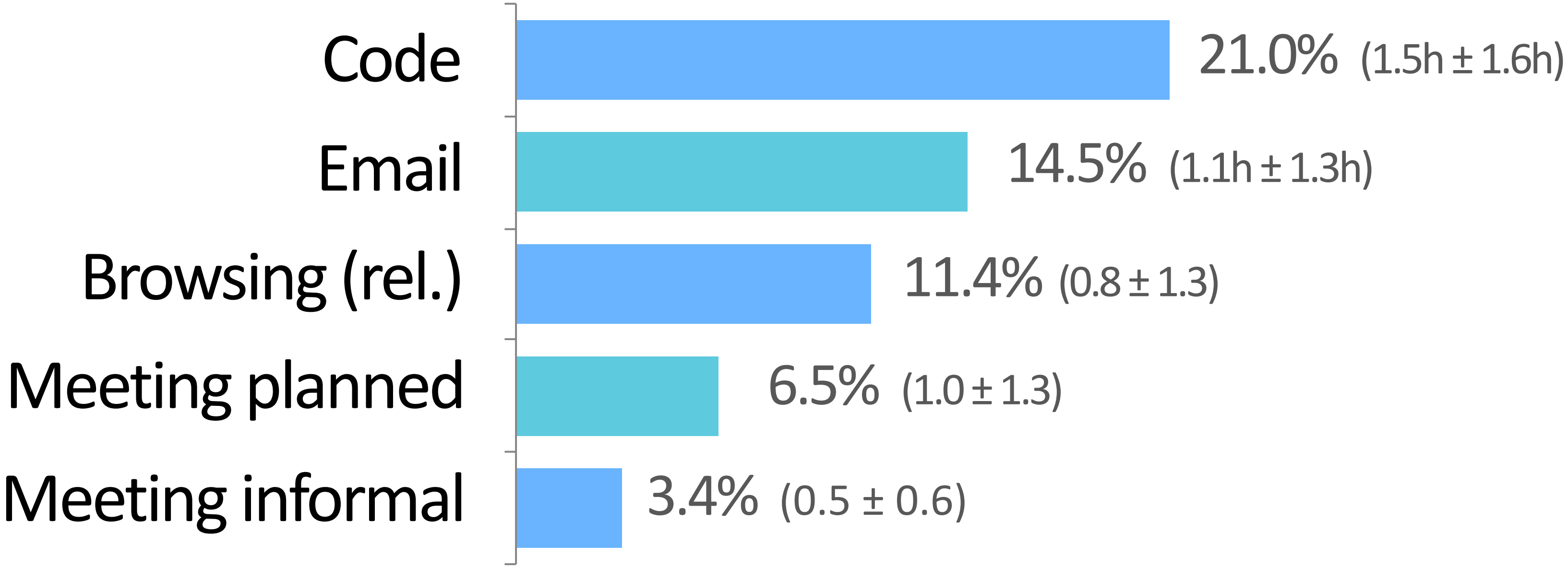
Monitoring

20 developers

220 days, 1350 ratings

Developers feel productive when they make **progress on tasks** with **few expensive context switches** / interruptions

Monitored activities

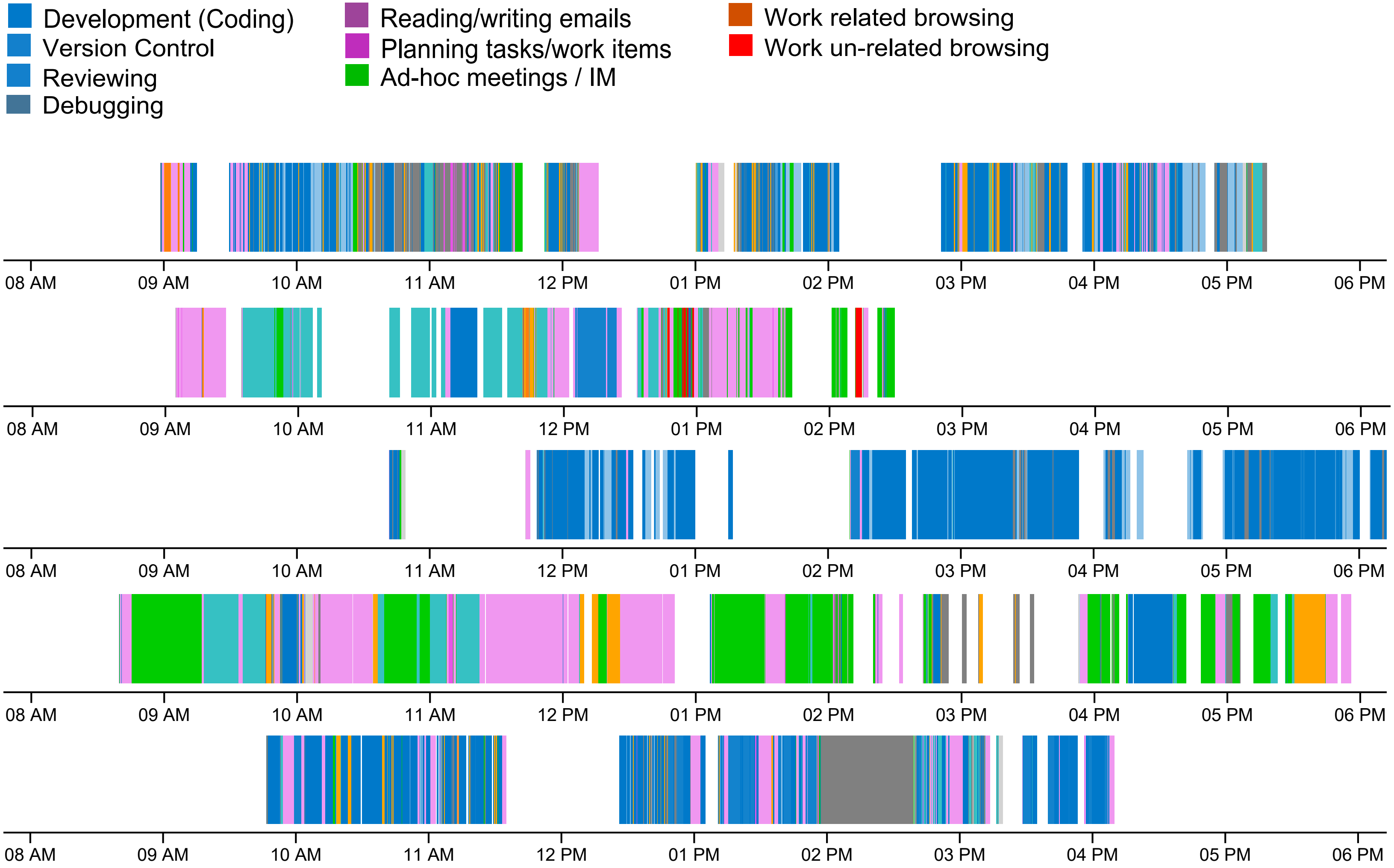


Range and time spend on activities varies greatly

Development work is **highly fragmented**

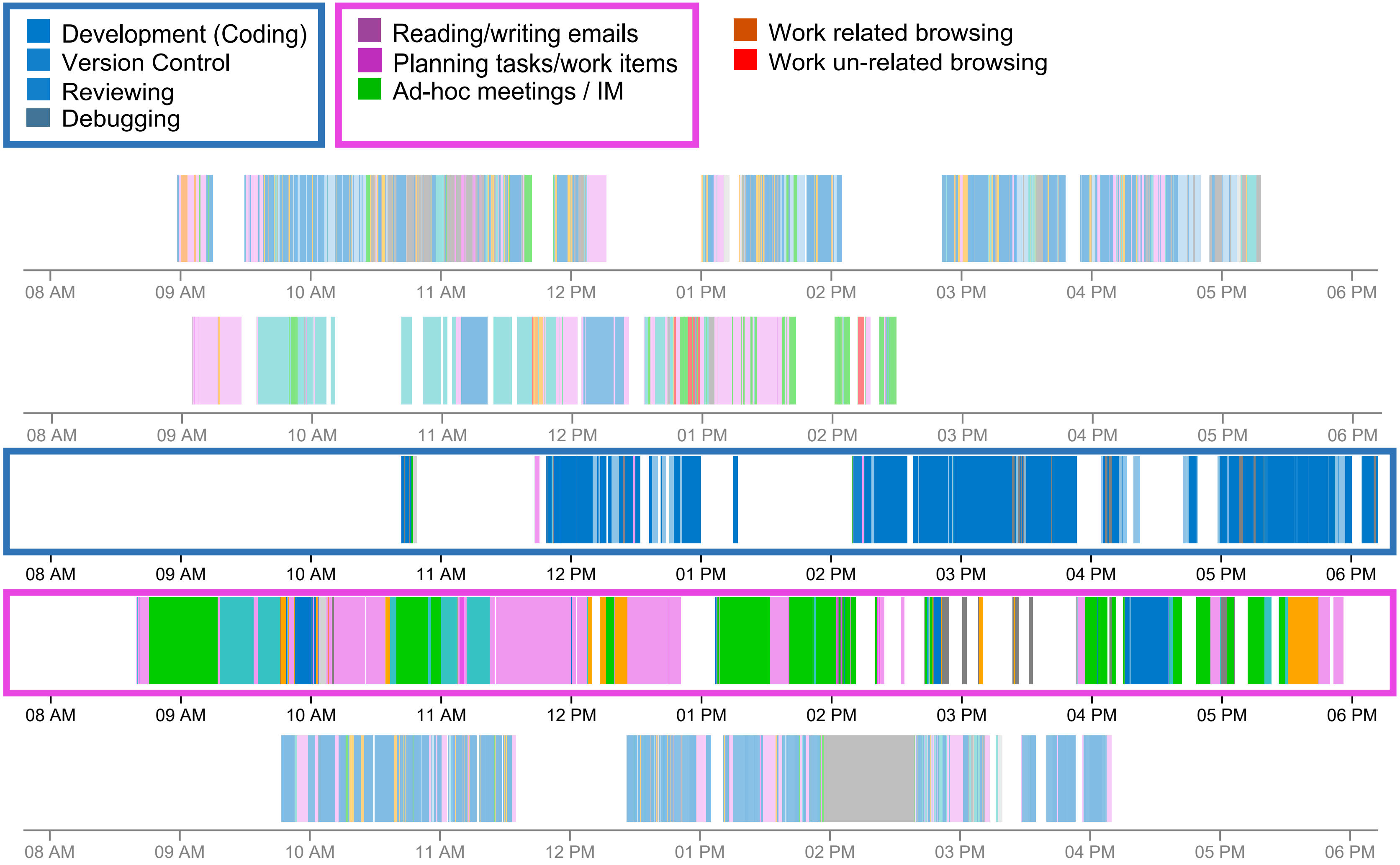
(0.3 to 2.0 min per activity)

Monitored activities



Development work is **highly fragmented**

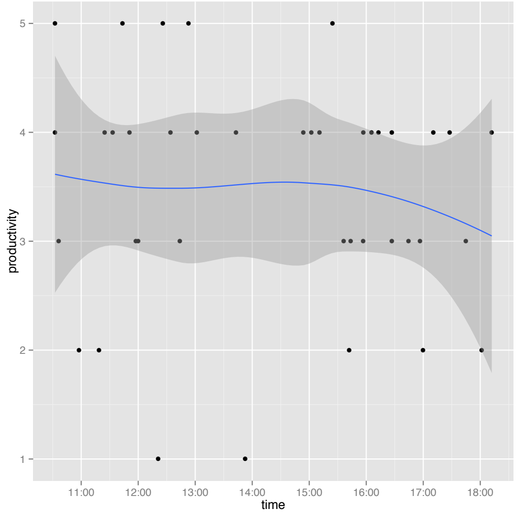
Monitored activities



Development work is highly fragmented and **follows themes**

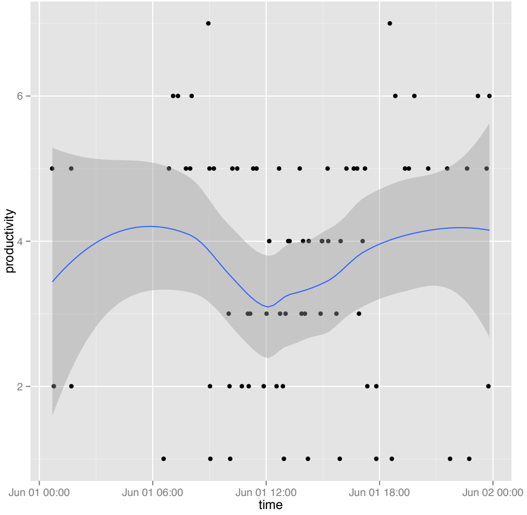
[TSE’17]

Productive times



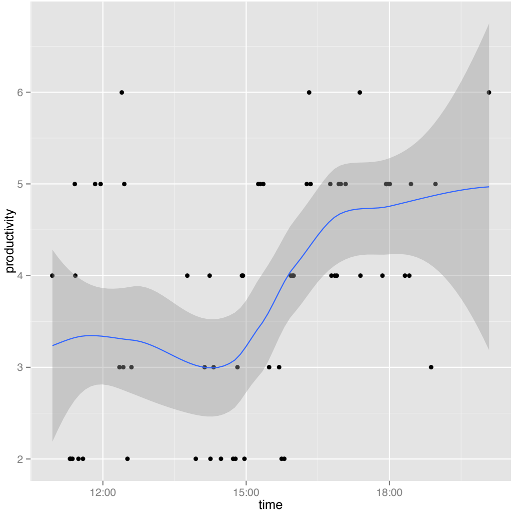
Morning person

(20%)



Low at lunch

(35%)



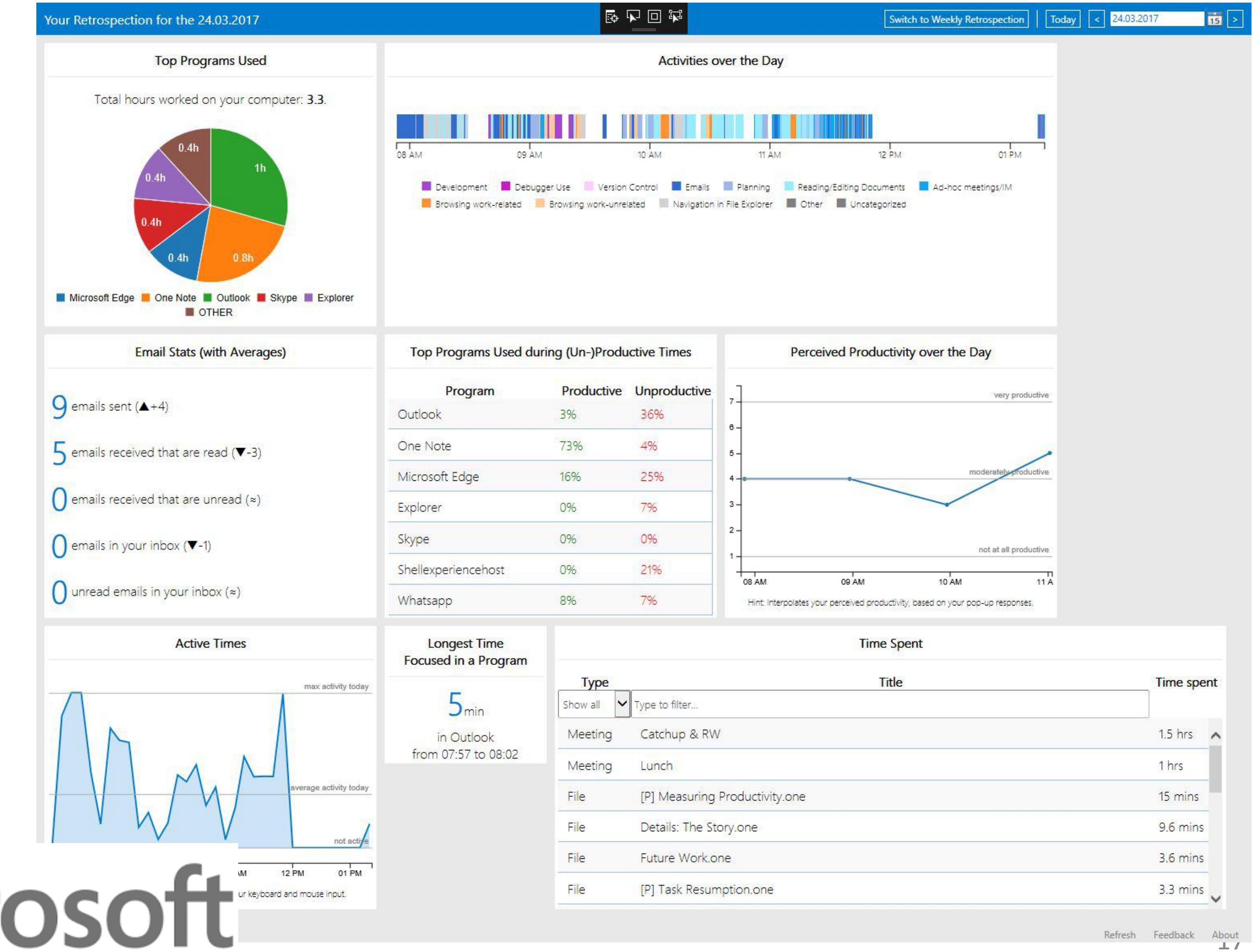
Afternoon person

(40%)

Developers’ perceived productivity follows **habitual patterns**

Developer Retrospection

(Fitbit for Developers)



Your Retrospection for the 24.03.2017

Top Programs Used

Total hours worked on your computer: 3.3.

Activities over the Day

Email Stats (with Averages)

9 emails sent (▲+4)

5 emails received that are read (▼-3)

0 emails received that are unread (≈)

0 emails in your inbox (▼-1)

0 unread emails in your inbox (≈)

Top Programs Used during (Un-)Productive Times

Program	Productive	Unproductive
Outlook	3%	36%
One Note	73%	4%
Microsoft Edge	16%	25%
Explorer	0%	7%
Skype	0%	0%
Shellexperiencehost	0%	21%
Whatsapp	8%	7%

Perceived Productivity over the Day

Hint: Interpolates your perceived productivity, based on your pop-up responses.

Active Times

ur keyboard and mouse input.

Longest Time Focused in a Program

5min

in Outlook
from 07:57 to 08:02

Time Spent

Type	Title	Time spent
Show all	Type to filter...	
Meeting	Catchup & RW	1.5 hrs
Meeting	Lunch	1 hrs
File	[P] Measuring Productivity.one	15 mins
File	Details: The Story.one	9.6 mins
File	Future Work.one	3.6 mins
File	[P] Task Resumption.one	3.3 mins

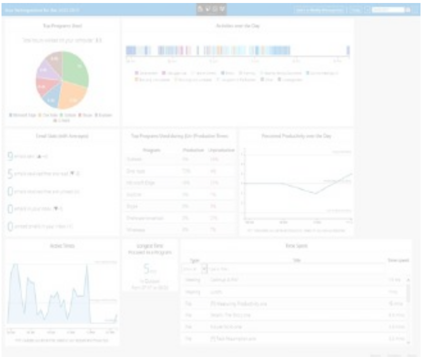
Developer Productivity

Unde

Sensi

Supporting

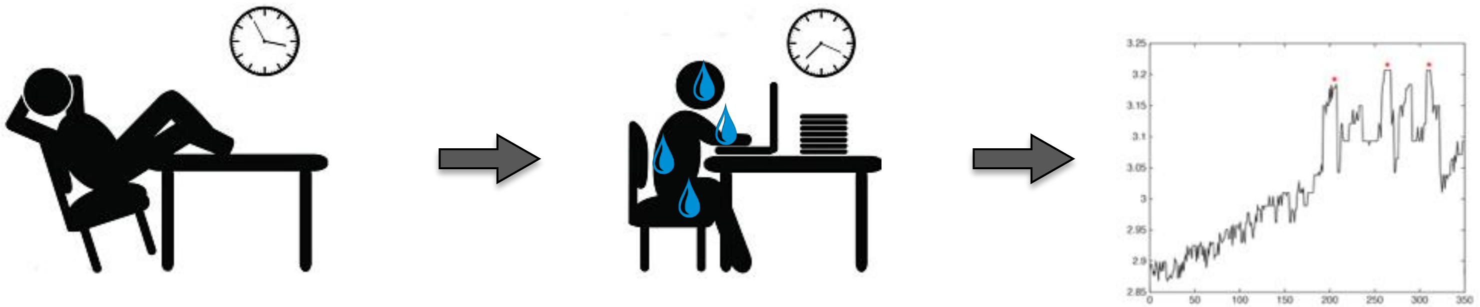
Developer Analytics



Sensing code difficulty

FlowLight reducing interruptions

Psycho-physiology (biometrics)



Psychological *(mind)*

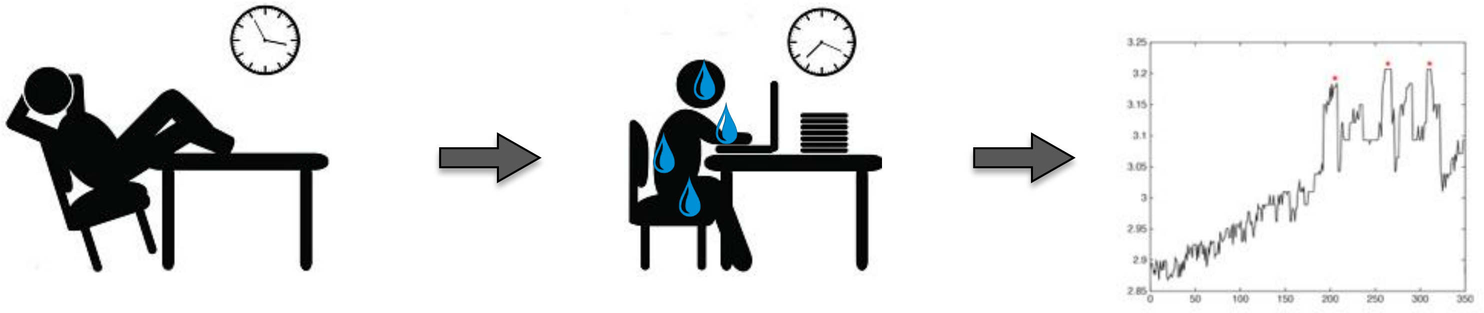
- Mental load
- Task engagement
- Excitement
- Emotions
- …

~

Physiological *(body)*

- Brainwaves
- Sweat
- Heart rate variability
- Pupil size
- Eye blink rate
- …

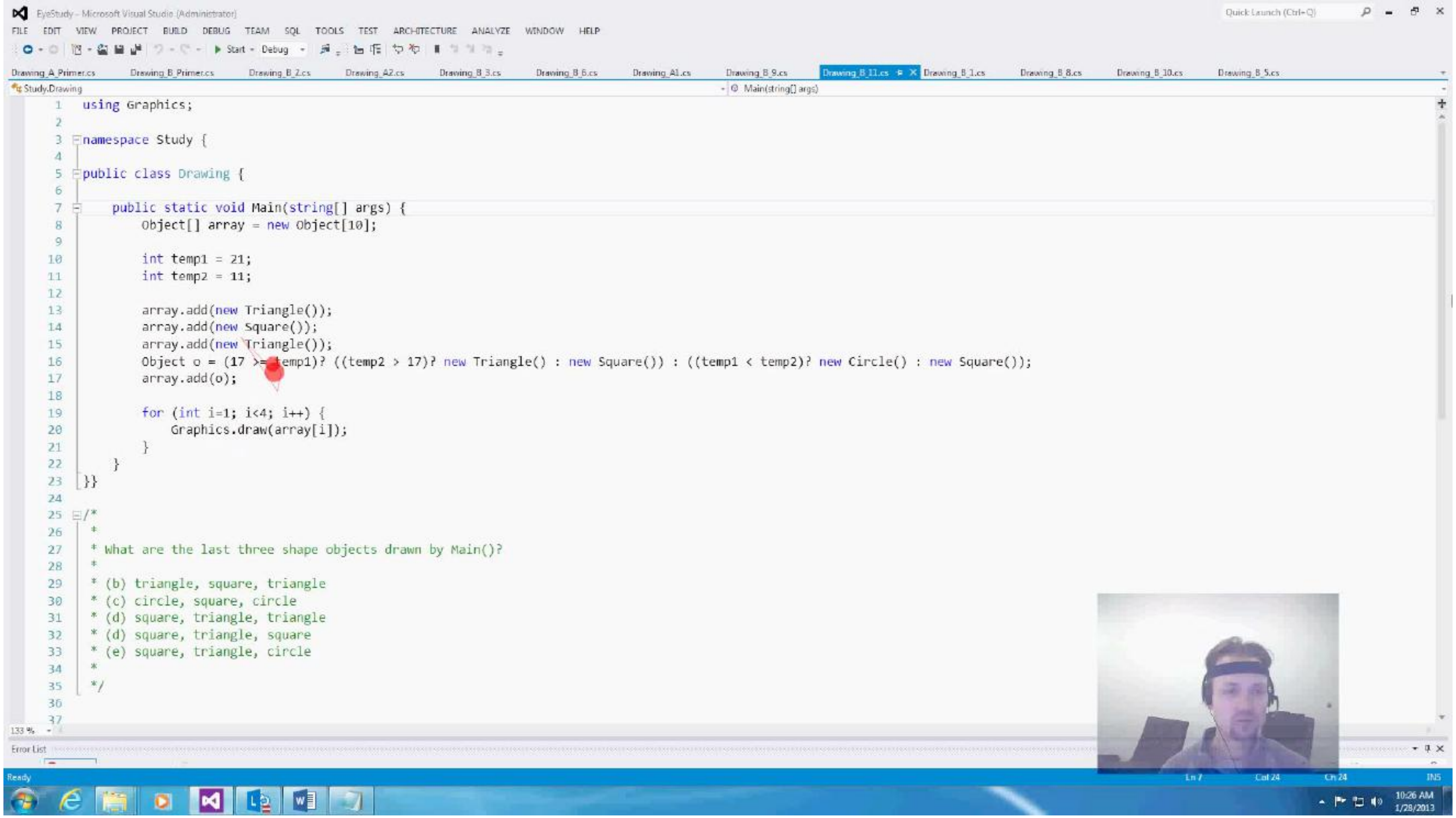
Biometric Sensing of Code Difficulty



Lab & field studies

Biometric sensors can be used to predict code interruptibility, difficulty and quality concerns
→ **Prevent bugs & costly interruptions**

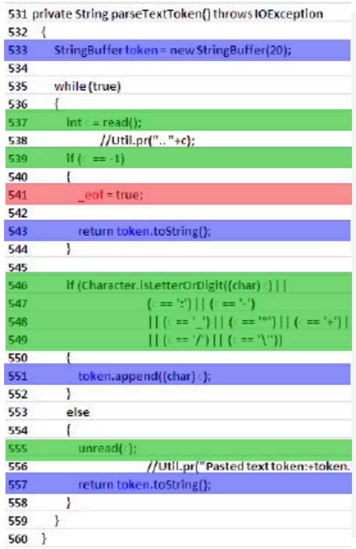
Watch a developer at work!



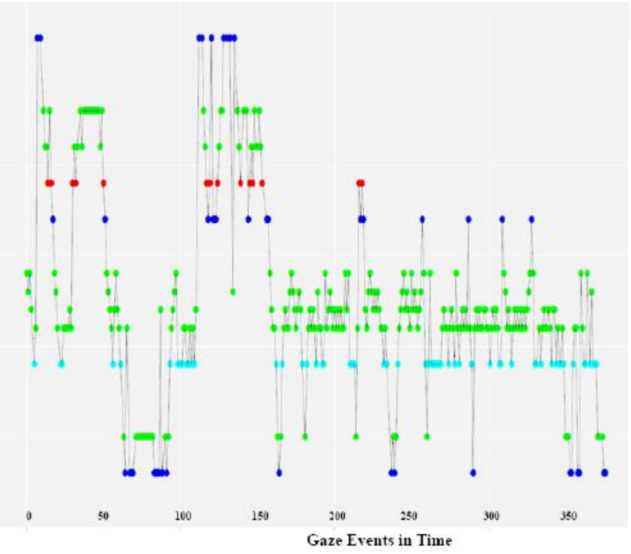
```
using Graphics;

namespace Study {

public class Drawing {

    public static void Main(string[] args) {
        Object[] array = new Object[10];

        int temp1 = 21;
        int temp2 = 11;

        array.add(new Triangle());
        array.add(new Square());
        array.add(new Triangle());
        Object o = (17 >= temp1)? ((temp2 > 17)? new Triangle() : new Square()) : ((temp1 < temp2)? new Circle() : new Square());
        array.add(o);

        for (int i=1; i<4; i++) {
            Graphics.draw(array[i]);
        }
    }
}}

/*
 *
 * What are the last three shape objects drawn by Main()?
 *
 * (b) triangle, square, triangle
 * (c) circle, square, circle
 * (d) square, triangle, triangle
 * (d) square, triangle, square
 * (e) square, triangle, circle
 *
 */
```

Tracing Software Developers’ Eyes for Change Tasks

Understanding developers’ code interactions for better tool support



```
private String parseTextToken() throws IOException
{
    StringBuffer token = new StringBuffer(20);

    while (true)
    {
      int c = read();
              //Util.pr(".."+c);
      if (c == -1)
      {
        _eof = true;

        return token.toString();
      }

      if (Character.isLetterOrDigit((char) c) ||
                        (c == ':') || (c == '-')
                        || (c == '_') || (c == '*') || (c == '+') |
                        || (c == '/') || (c == '\'))
      {
        token.append((char) c);
      }
      else
      {
        unread(c);
                        //Util.pr("Pasted text token:+token.
        return token.toString();
      }
    }
}
```



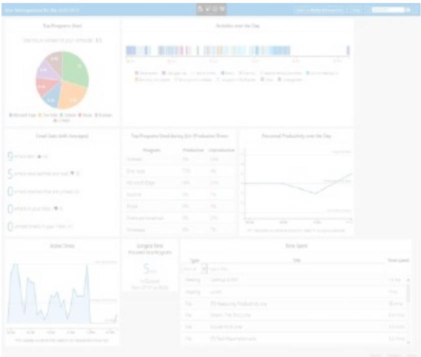
Study with 12 professional developers and 10 students

- Developers only look at small fragments of code elements and often follow data flow within a method

Developer Productivity

Unde

Developer Analytics



Sensi

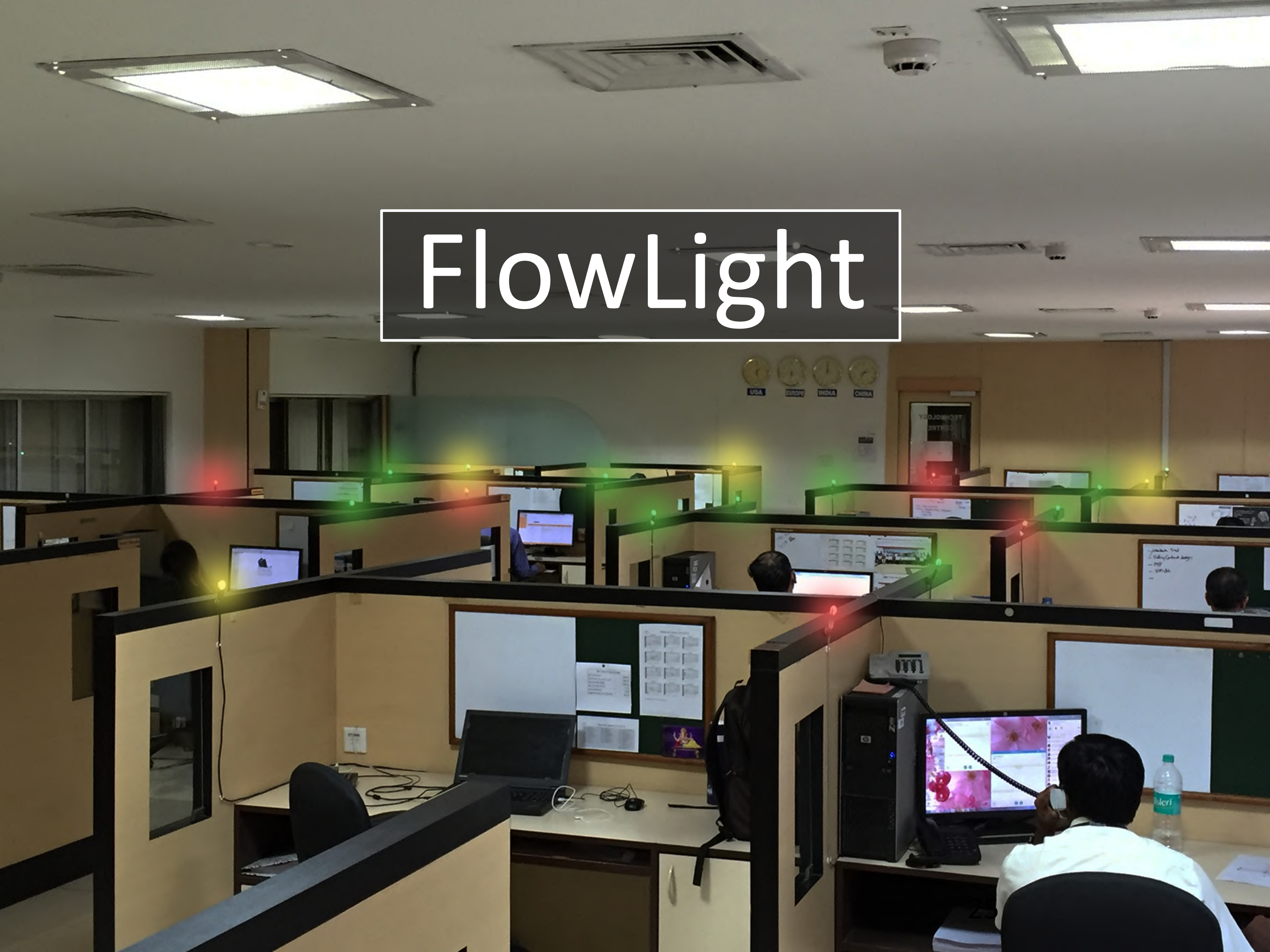
Sensing code difficulty

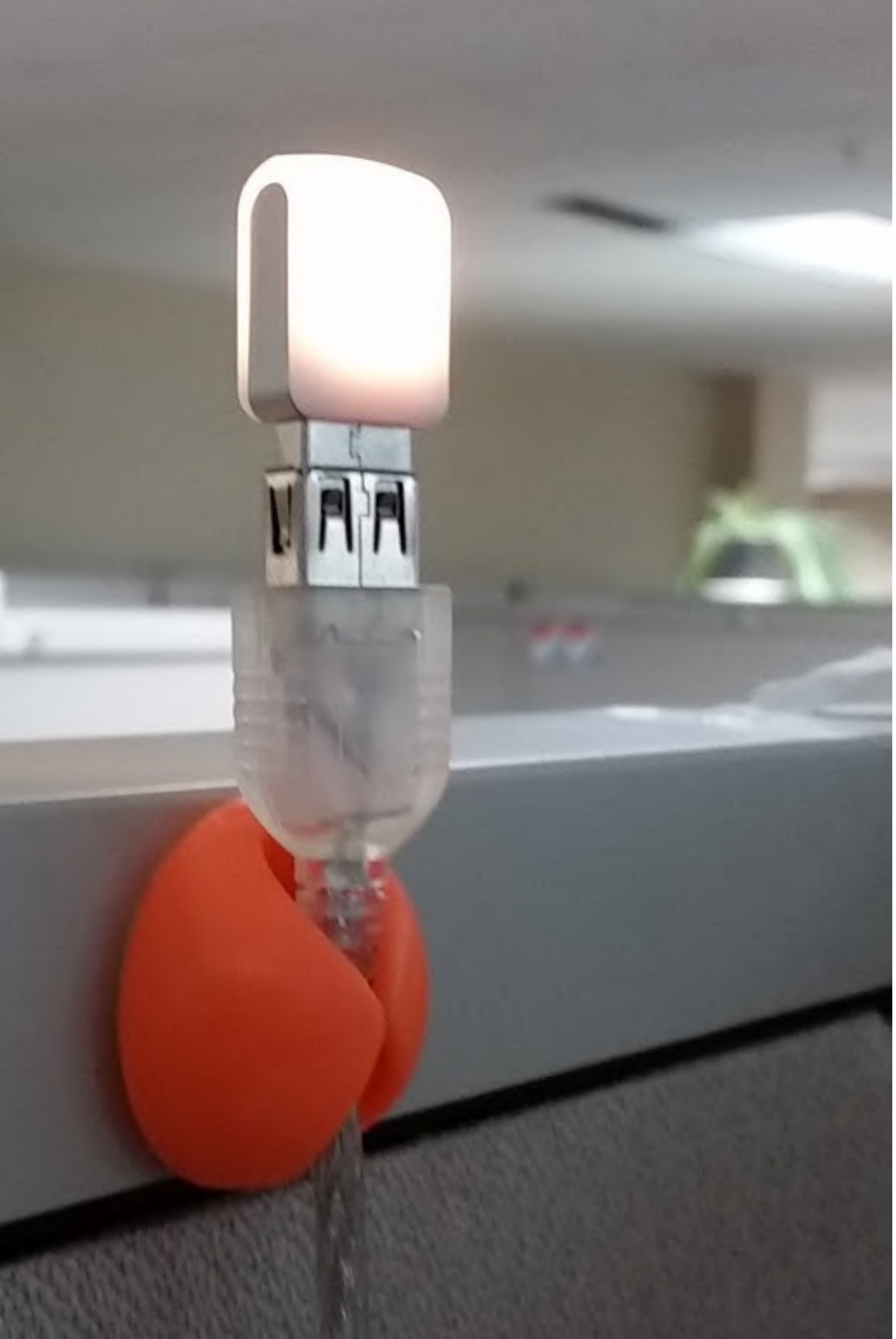
Supporting

FlowLight reducing interruptions

Interruptions at work

FlowLight





FlowLight

PULSATING RED

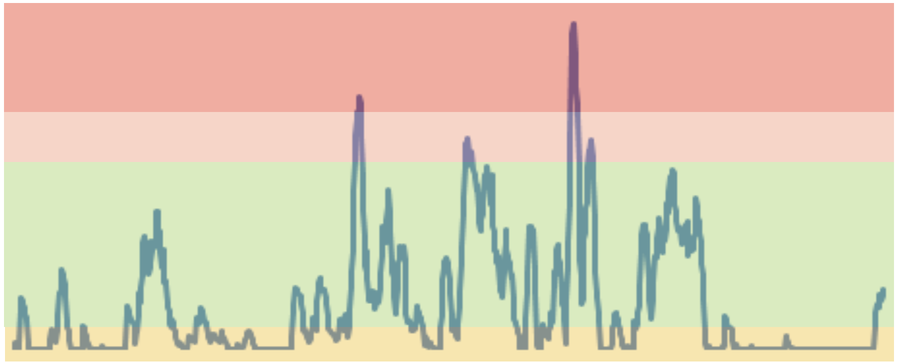
RED

GREEN

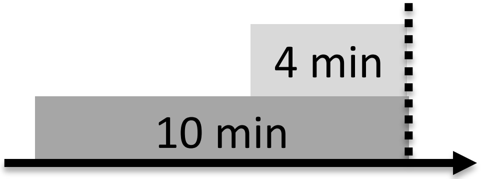
YELLOW

FlowLight – automatic algorithm

Tracking



Individualized Thresholds

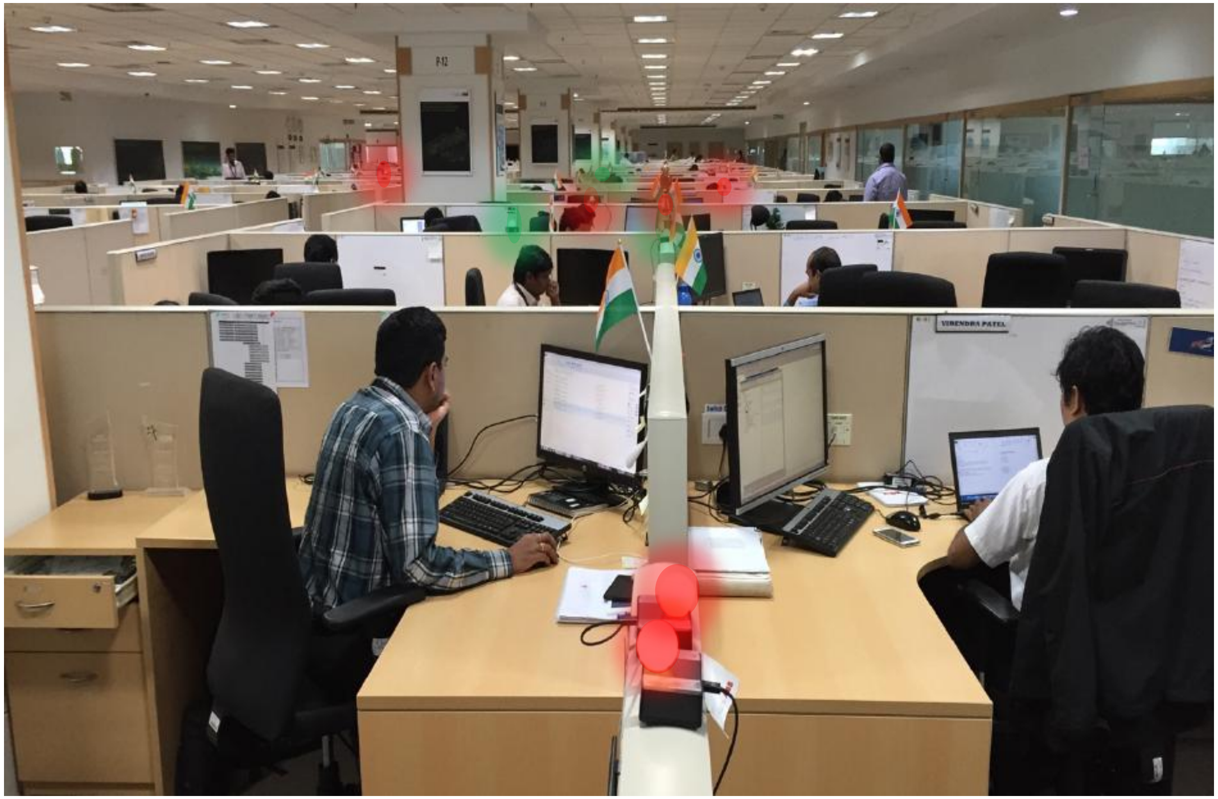


Smoothing

FlowLight – reducing costly interruptions

449 professional knowledge workers

12 countries, 15 locations



FlowLight – reducing costly interruptions

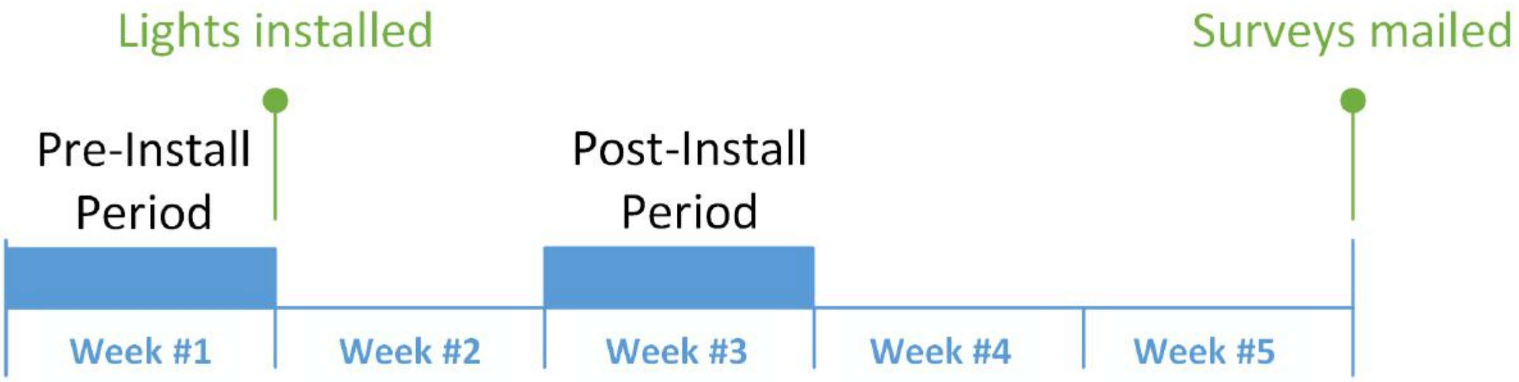
449 professional knowledge workers

12 countries, 15 locations

Work as usual (initially 5 week period)

183 survey responses, 23 in-depth interviews

36 self-reported interruption logs



Lights installed

Surveys mailed

Pre-Install Period

Post-Install Period

Week #1

Week #2

Week #3

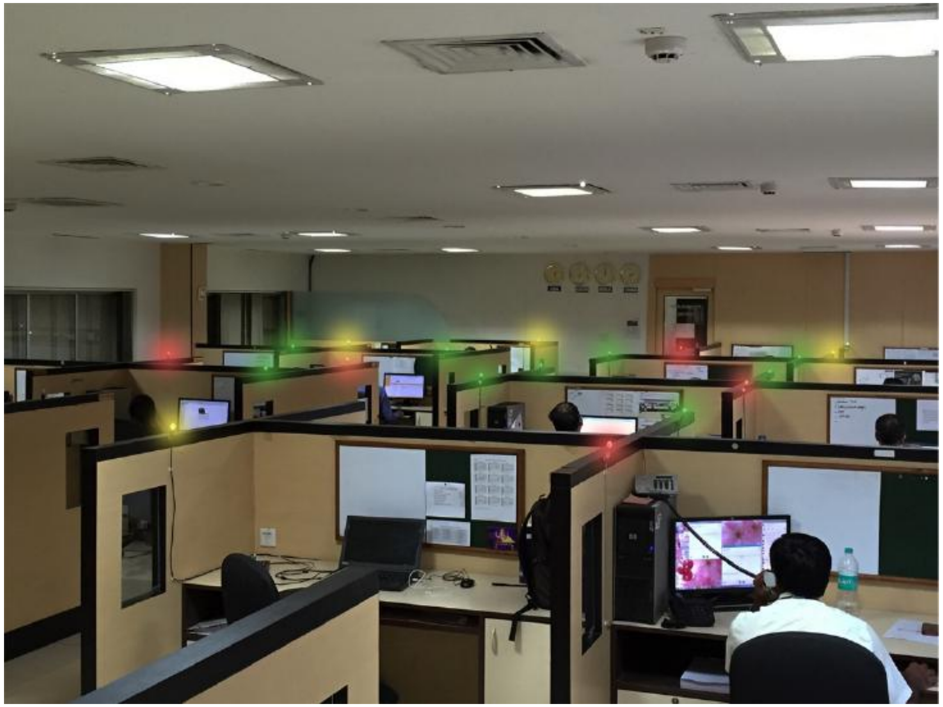
Week #4

Week #5

FlowLight – Reducing Costly Interruptions

Field study with 449 participants, 12 countries

ABB



46% less interruptions
85% continued using it
on a daily basis

BBC WORLD NEWS

THE NEW YORKER

New Scientist

THE TIMES

WSJ

cnBeta.COM 中文业界资讯站

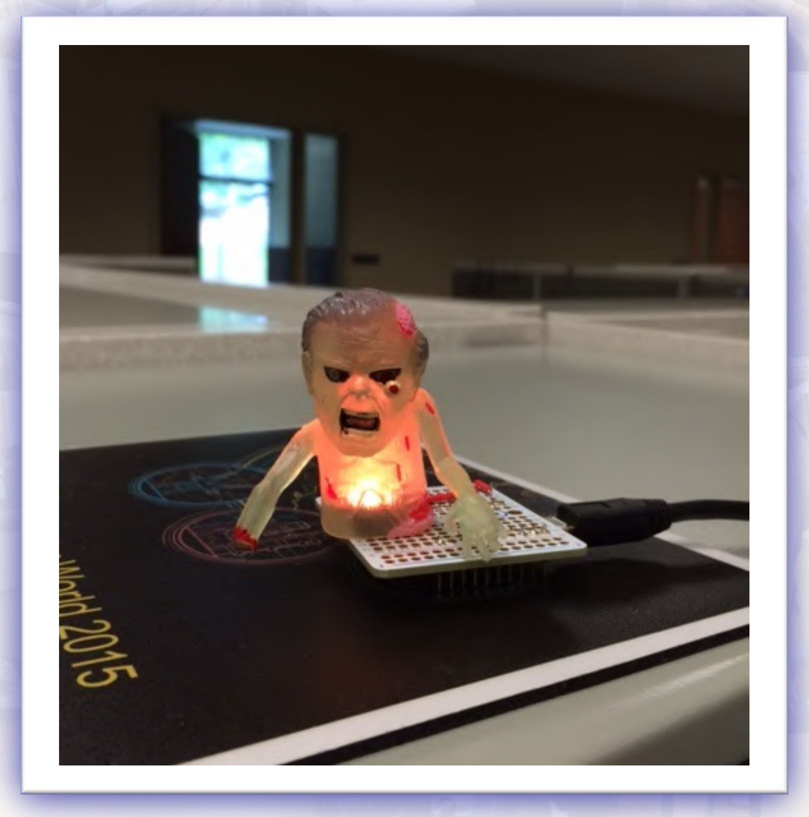
NRJ

GeekWire

DIGITAL TRENDS

The Telegraph

NBC NEWS

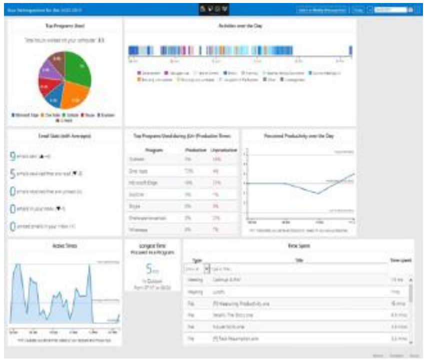


World 2015

Pig Chart
Go for Chat
Busy
On the Phone
DnD
World is ending

Boosting Productivity & Well-being

Understanding developer productivity



Sensing developers' productivity

Supporting productive behavior

- Provide awareness & actionable insights
- Reduce costly interruptions & support focused work
- Prevent bugs / defects
- Provide easy access to relevant information

Our Research

Boosting (developer) Productivity

Developer Productivity & Analytics

Sensing Developers

Monitoring & Empowering Productivity

Mental Health & Safety

Increasing Focus

Workflow Automation

ABB quickmobile Microsoft iQmetrix MIO TRAIN WITH HEART® IBM SAP® Schindler InteraXon THOUGHT CONTROLLED COMPUTING BISON adesso VENTYX™ AN ABB COMPANY Tasktop

Please, introduce Yourself!

Who are you?

Do you have and SE experience and if so what?

[optional] Is there anything that you are particularly interested in or would you like to learn about?

Course Overview

Course Objectives

- Deepen and broaden your knowledge of Software Engineering research (and some HCI) by ***reading, reflecting and discussing*** current and classic literature

- Learn to ***define and study research questions***

- Experience a glimpse of Software Engineering research through a research ***project*** work

Focus on Research and the Process

The ***process*** is important

- Identifying interesting research questions (RQs)
- determining how to investigate them and running analysis
- presenting and writing up results

Research is mostly an ***iterative*** process

- Identifying relevant RQs is difficult and discussing and then revising them is important and part of research

Non-Traditional Course Format

Outside of class

- 2 to 3 papers per week (more papers in the beginning, to get into domain, less papers & more meetings/activities later on)

  read papers, think about them, write a short response paper

- Research Project

  find a research question, write and present a proposal, do it, write it up and present it (you can team up for it)

In class

- Discussion and moderation of research papers

  One student introduces paper (5min), everyone participates in discussion

- Other times: small activities, weekly meetings on project progress, presentation

No exams, however, projects will take time. [6 ECTS]

Tentative Schedule

Sept 16 Course overview & Introduction

Sep 23 (Empirical) Research in Software Engineering (Papers & Discussion)

- What makes good research in software engineering?
- A practical guide to controlled experiments of software engineering tools...
- + Think about two to three problems/questions in SE

Sep 30 Developer Productivity & Support (P&D)

- The Work Life of Developers: Activities, Switches and Perceived Productivity
- What predicts software developers' productivity?
- Code Bubbles: Rethinking the User Interface Paradigm of IDEs

Oct 7 Sensing and Supporting Code Difficulty (P&D)

- Learning a metric for code readability.
- Measuring Neural Efficiency of Program Comprehension
- Helping Developers Help Themselves: Automatic Decomposition of Code …

Tentative Schedule (2)

Oct 14/15 Proposal discussions (One-on-one meetings; writeup due before)

→ sign up early by email to fritz@ifi.uzh.ch with 3 preferences of 30mins slots for Mon / Tue

proposals due the day before, i.e. October 13th

Oct 21 Proposal Presentations (presentation)

Oct 28 SCRUM in Research & Eye-Tracking in SE & ML

Improving Automated Source Code Summarization via an Eye-Tracking Study of Programmers

Detecting Personality Traits Using Eye-Tracking Data

A brief introduction to Machine Learning

Nov 4 Scrum & Developer Support

Context-Aware Conversational Developer Assistants

Augmenting Code with In Situ Visualizations to Aid Program Understanding

Tentative Schedule (3)

Nov 11	Meetings & Progress Report
Nov 18	no class
Nov 25	Quick Project Update in Class & weekly meeting
Dec 2	Weekly Meetings
Dec 8	Project Report due
Dec 12	Peer-reviews due
Dec 16	Project Presentations in class
	final report due + presentations

Focus & Topics of Course

- (Developer) Productivity
- Biometric sensing
- Developer workflows, activities, work fragmentation, interruptions
- Data on developers: interaction logs, biometric data, observation logs, activity logs …
- Developer support
- Self-monitoring and goal setting
- Program comprehension, software evolution, …
- Empirical Research and studies of software developers (quantitative and qualitative)

Grading

- 65% Project (including proposal, report, presentation)
- 25% Readings (including response papers, class participation & leading discussion)

  Class attendance expected

  (let me know ahead of time if you’re not able to come)
- 10% Peer reviews of project reports

Response Papers

- Encouragement to read and reflect
  - Class discussions work better if everyone has read and thought about the paper
- At most one page per class (300 to 500 words)
- **NOT a summary**. Think of it this way
  - If I asked you what you thought about a movie you recently went to, you wouldn’t just summarize it
- **Sometimes,** short question to be addressed.
- Grading based on “thoughtfulness”
- Due by **8pm** on day before class
  - Submit by email

Response Papers

- Questions of interest
  - What did you think about it and what did you find important or interesting?
  - What are main contributions of the paper?
  - What are strengths or weaknesses of the paper/research?
  - What are five questions you have about it?
  - What could be improved?
  - How could you imagine extending the work?
  - Do you agree or disagree with the findings?
  - How does the research relate to other papers for this lecture?
  - …
- Express your perspective, ***address all readings*** and ***draw connections between readings*** when possible
- Example provided on web site!

Discussions

- Discuss the research:

  which problem are they trying to address, how are they tackling the problem, how do they evaluate their approach, …

- Share your opinions, ideas and thoughts
- Ask questions about the work
- See what others thought
- Listen and speak actively
- Look for contributions not just flaws in reading
- **If it’s your turn**: introduce the paper for ~3mins & come up with questions, moderate the discussion
- SIGN UP NOW

Research Projects

Research Project – Empirical Analysis

- Identify a real problem developers face / investigate specific aspect of SE
- Read related work and determine your niche
- Identify relevant/interesting research question
- Determine how to address the research question
- Run analysis
- Write up results in a scientific manner

Some Possible Projects

- Hands-on project with biometric sensors

- Examine developer activity and productivity
- Analyze biometric / eye-tracking or interaction data
- Examine software repository histories and metrics
- Develop and evaluate tool support
- …

Research Project – Empirical Analysis

- Each project accompanied by a paper (max. 5 or 10 pages)
- Individual or in groups (up to 2 people, depending on class size)
- One page project proposal *draft* due on **October 13**
- Project proposal presentation
- Final one- to two-page proposal due on **October 22**
- Written report due on **December 8**
- Project presentation

Research Project – Empirical Analysis (2)

- Project report: ACM paper format

- One-on-one meetings shortly before and after project proposal is due

- SCRUM: class meetings, each one has 2mins to state what they have done last week and what they will do next week

- Continuous short progress meetings (depending on class size)
  - discuss progress, next steps, open questions, keep on track …
  - take advantage of them, i.e. prepare and ask!
  - ~10 minutes

Peer Review of two Project Reports

- Research communities rely on peer reviews of results, if you want to be a researcher you need to learn about critiquing research papers

- Paper review will also help you to learn what is important when you write up your own work

- Templates will be provided

  Summary, what are its strengths, what are the weaknesses

  (realize that the authors would usually read it, so be constructive)

Peer Review (2)

- Assess projects like a program committee
  - Everyone will read and review two project reports
  - Reviews are organized via OLAT
- Hand in review (will also be sent to authors)

Some More & Next Steps

Me

- I’m here to help
- Talk to me if you want feedback or need help
- Talk to me if you do not find a topic or want to discuss your idea
- In class, I’m here to discuss

To Dos

- Choose papers that you would like to introduce/present **by end of today**

- Start thinking about projects as soon as possible: what are you interested in?

Next week Monday

- Two papers on Empirical Research
  - **What makes good research in software engineering?** Shaw, Int. Journal of Software Tools for Technology 2002.
  - **A practical guide to controlled experiments of software engineering tools with human participants,** Ko et al., Empirical Software Engineering 2013.

- Read and write ONE short response paper that discusses both papers

  Also and only this time: Include two to three questions/problems/ideas in software development (a bullet point for each is enough)

- Submit by email

Discussion Starter / 3 mins intro

Who is up for the first ones?

1) What makes good research in software engineering?
2) A practical guide to controlled experiments of software engineering tools with human participants

More Information

- See website:

  http://www.ifi.uzh.ch/en/seal/teaching/courses/hase.html

- Contact:

  Thomas Fritz fritz@ifi.uzh.ch

Exercise

In groups, discuss…

All together:

- Think about a problem you have/had programming / developing software

In teams of 2 or 4

- How could you study or improve it?
- How could you show the improvement?

Some problems/questions/ideas mentioned…

- Fostering equal contributions in teams
- Staying motivated for your work
- Merge conflicts
- Team awareness
- Where to get relevant information (people, web,…) and what are the costs of getting and providing it
- Traceability, matching code and documentation
- Remote vs office work
- Good work practices