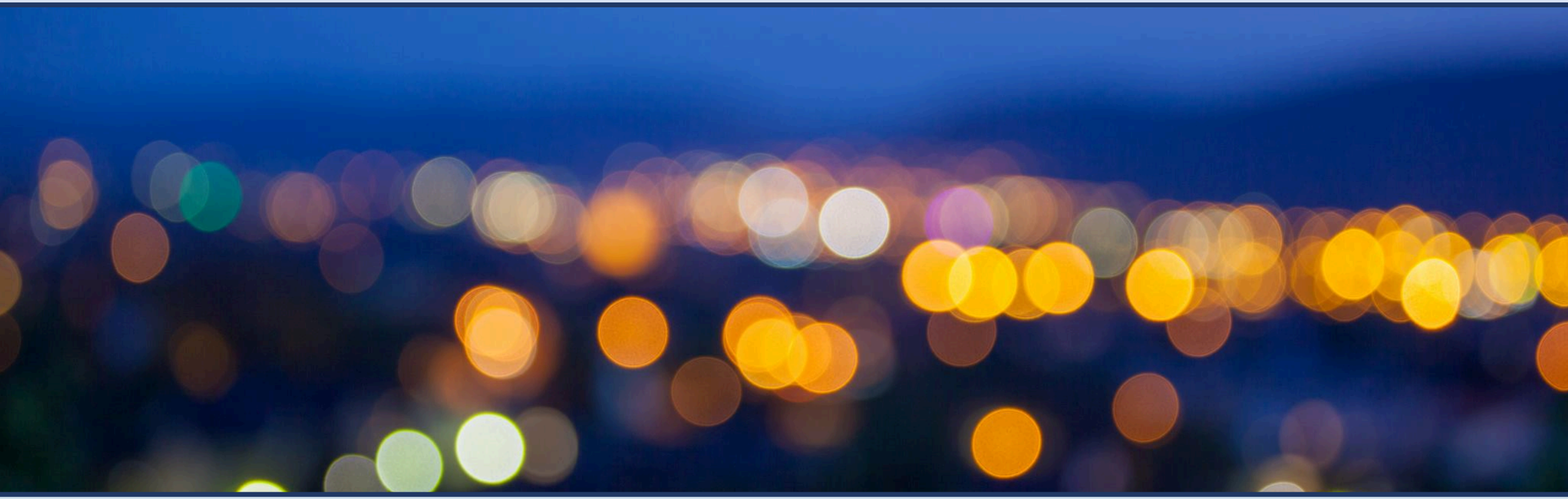
Software Developer Work Habits

Communication, Code Plagiarism, Expertise Development



Sebastian Baltes

@s_baltes

Universität Trier

Visit @ Universität Stuttgart

Questions are appreciated

Course of my Ph.D. studies

2013

Sketches and Diagrams in Practice

SketchLink

Communication

How Developers Locate Performance Bugs

LivelySketches

Issues In Sampling Software Developers

Methodology

Code Plagiarism

Stack Overflow Code Snippets in GitHub Projects

Towards a Theory of Software Development Expertise

Expertise Development

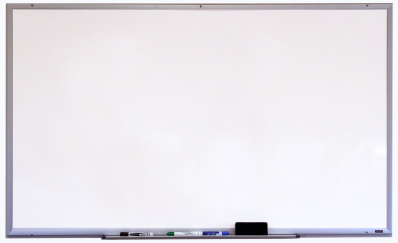
2018 (?)

Communication

How do software developers use sketches and diagams?
How could we provide better tool support?

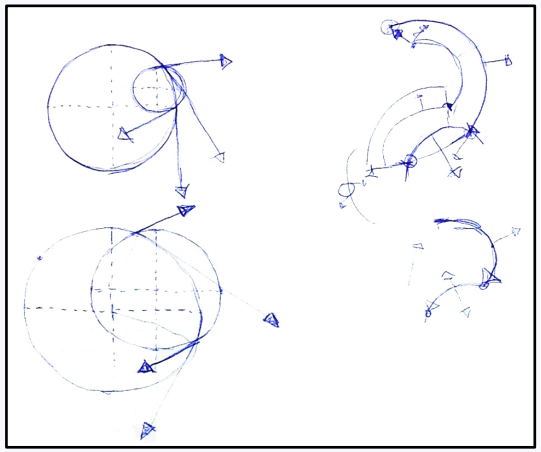
How do developers locate performance bugs?
How do they use sketches for communication?

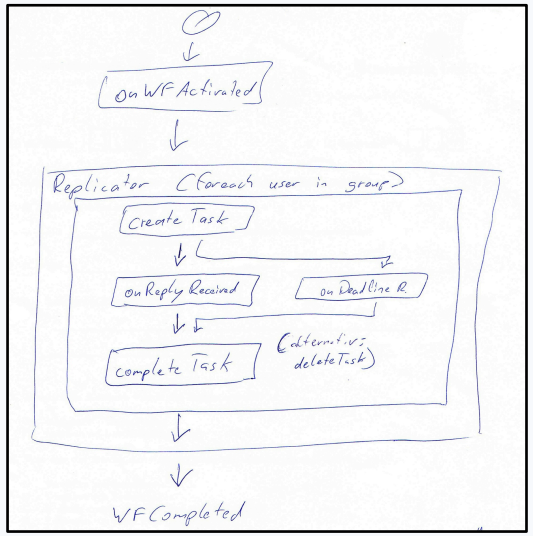
Medium

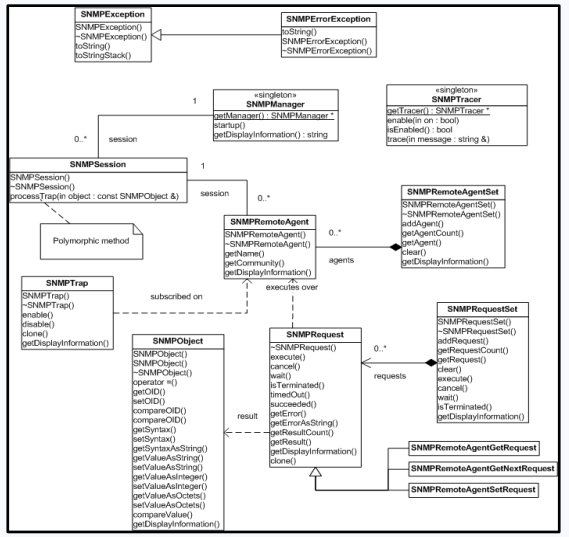


Application Area

Sketches vs. Diagrams







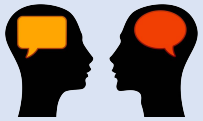
informal — **formal**

Sketch

Diagram

Related Work

Past studies: Sketches and diagrams important in daily work of software developers

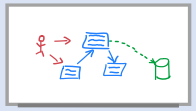


Purpose: Understanding, designing, communicating

[Cherubini07]

Depict **mental model** of software

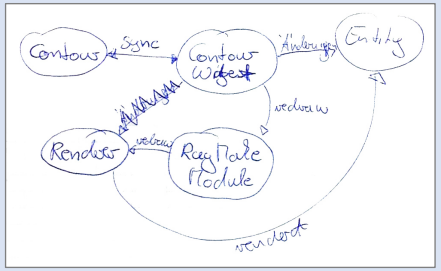
[LaToza06]

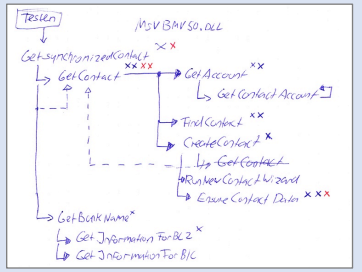


Medium: Whiteboard, paper, computer [Cherubini07, Walny11]

Psychology: Sketching augments **information processing**, sketches are sources of **creativity**

[Goldschmidt03, Tversky03]





Teams **improvise** representations, sketches/diagrams often **informal**

[Dekel07, Petre13]

Why another study?

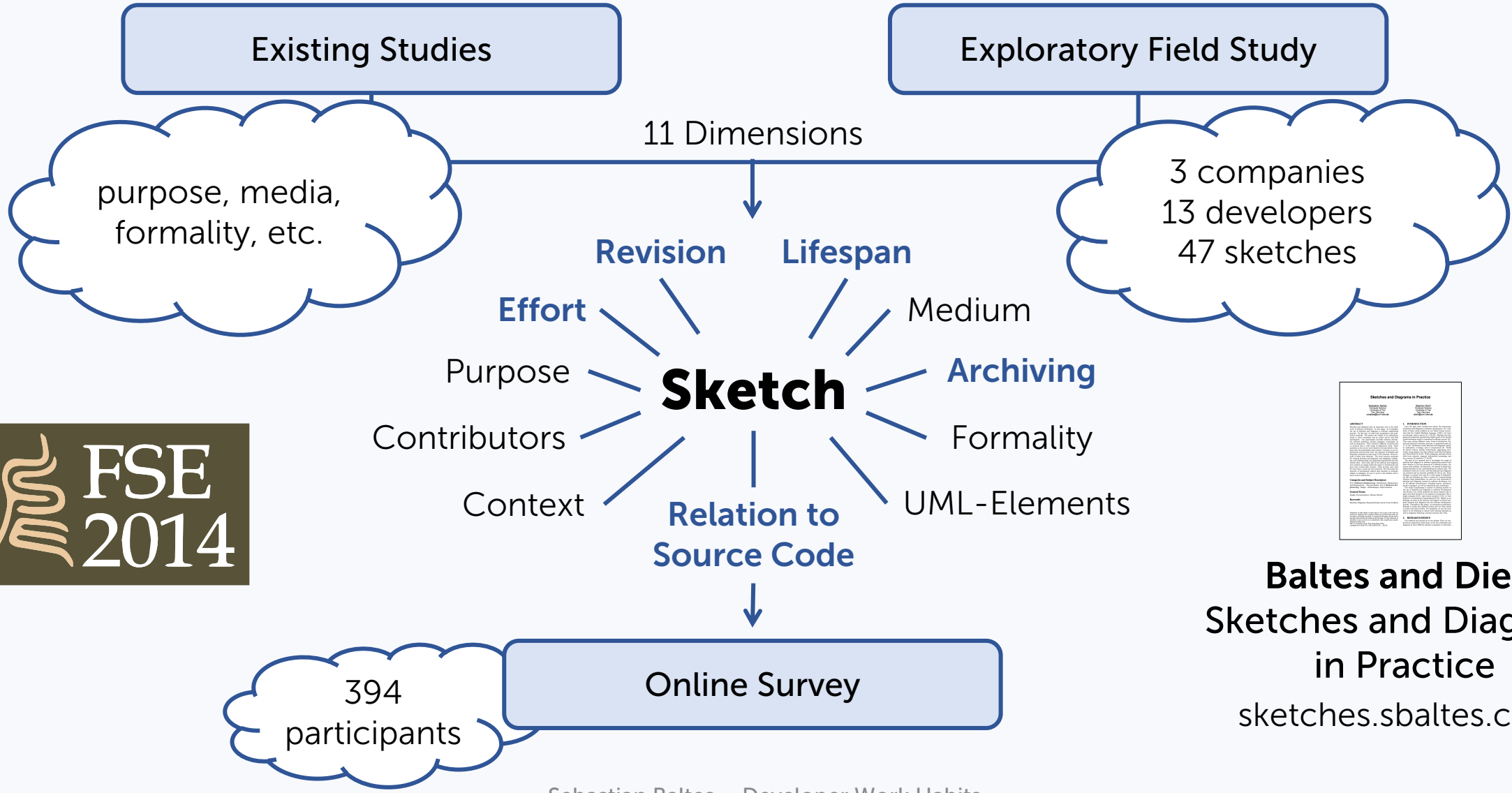
Existing studies:

- Concentrated on certain aspects
- Single companies or academic environment
- Some had small number of participants

Our goal: Thorough description of how sketches and diagrams are used in software engineering practice

Better tool support for integrating sketches and diagrams into software development process

Research Design



FSE 2014

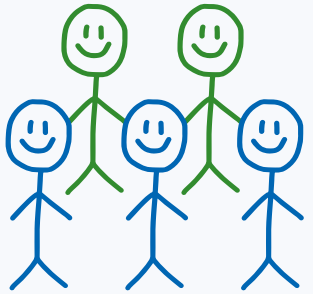
Baltes and Diehl
Sketches and Diagrams in Practice
sketches.sbaltes.com

Online Survey: Design

- **Target population**: "software practitioners"
- **Concise**:
  - ~10 minutes to complete
  - 28 questions, 15 about last sketch
- **Recruiting**:
  - Network of colleagues and contacts
  - Social networks
  - IRC channels and online communities
  - Directly contacted software companies
  - Article on major German IT news website

heise online

Online Survey: Participants



- n=394
- 32 countries
  - 54% Germany  15% North America 
- 52% software developers, 22% software architects
- Time spent developing software: **80%** (median)
- Professional work experience: **10 years** (median)
- Software projects from various **application areas**

Is it ethical to “spam” software developers for research purposes?

Worse than

Issues In Sampling Software Developers

Motivation

- **Reaching out** to professional software developers is crucial part of empirical software engineering research
- **Survey research** is important method to investigate state of practice
- When sampling developers for surveys, several **practical and ethical issues** arise

We report on:

1. The problem of **convenience samples**
2. **Own experience** with different sampling strategies
3. **Ethical** implications of these strategies
4. Assessment strategy for **external validity**

Sampling: Ideal Scenario

Theoretical Population

Who do you want to generalize to?

Study Population

What populations can you get access to?

Sampling Frame

How can you get access to them?

Random Selection

Sample

Who is in your study?

Respondents

Who participated?

Sampling: Common Scenario

Main problem: Availability of suitable sampling frames, reachability of participants.

➔ Reliance on available subjects:
convenience sampling, snowball sampling

➔ Likely leads to **biased samples**:

- Self-selection bias
- Researchers contact people from their own social and cultural group
- **Limited generalizability**

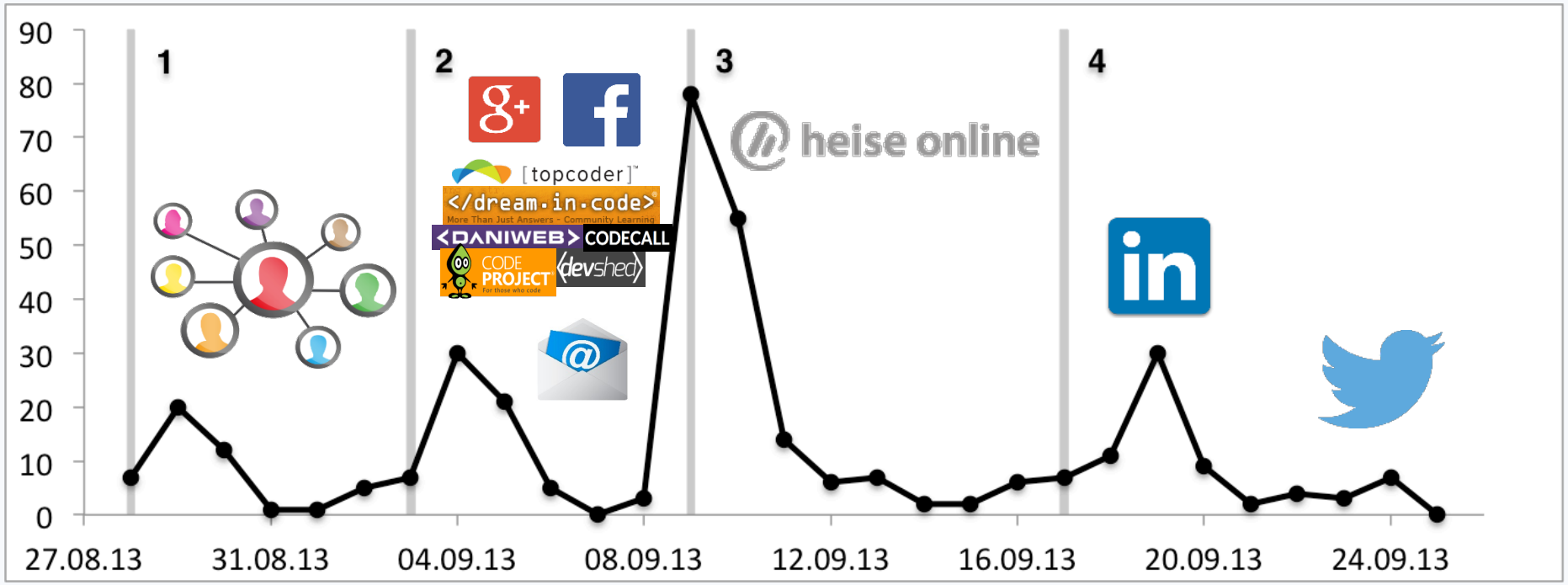
Strategies:

- (Try to) select **broad cross-section of the target population**
- **Clear description** of sampling approach and participants
- Take care not to overgeneralize
- Alert readers to the **limitations**

Sampling Strategies

Sampling Strategies: Experience Report

- **Survey** on the usage of sketches and diagrams in software development with **394 participants**
- **Four** recruitment phases



Sampling Strategies: Experience Report

- **Personal network:**
  - Not very effective
  - May dependent on quality and quantity of network
  - Better suited for other study designs (interviews, controlled experiments)

- **Online networks and communities:**
  - Not very effective
  - Mostly positive feedback in online forums
  - Some criticism in IRC channels

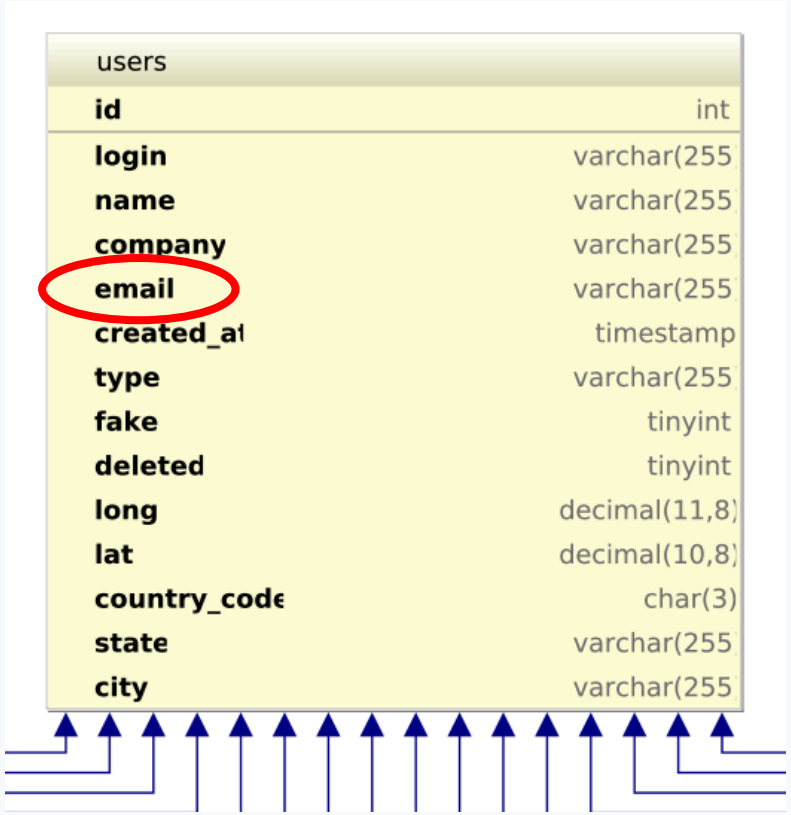
- **Directly contacting companies:**
  - Difficult to cross company borders without a *gatekeeper*

- **Public media:**
  - Most effective and efficient strategy (about 40% of responses)
  - Again gatekeeper in editorial team helpful

- **“Testimonials” (Twitter):**
  - Rather efficient
  - Again problem of biased sample

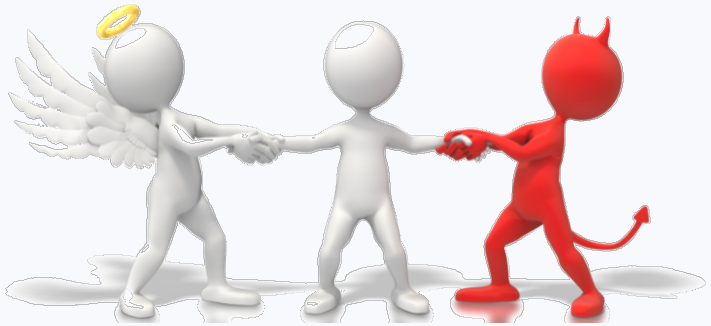
Sampling Strategies: GHTorrent

GHTorrent



- GHTorrent:
  - Project collecting **data about public GitHub projects**
  - Available online and as data dump
- Possibility to filter users according to their activity on GitHub
- **Random sampling**
- Email addresses removed in March 2016 after heated discussion on GitHub
- **Alternative:** Collect email addresses from user profiles or commits
- Convenient, but raises **ethical questions**

Ethical Considerations



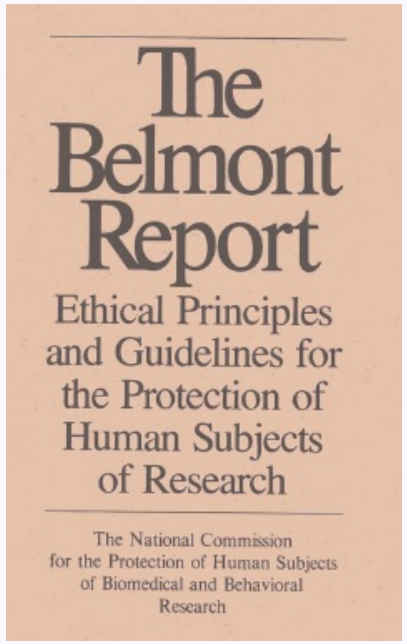
Ethical Considerations

- **Ethics:** "Rules of behavior based on ideas about what is morally good and bad" [Merriam-Webster]
- Legal aspects out of scope for this talk

*"I get emails like this **every week**. You might not realize this but it's majorly annoying and I consider this problem now **worse than spam**, since Google at least filters out spam for me. [...] **[Y]ou send one, I get one per week - or more.** I was playing along for the first 30 or so, and by now (after several hundred emails) **I'm quite annoyed**."*

- Sending mails to users on a large scale causes **costs**, even if they don't participate
- Active users get annoyed and do not answer → **selection bias**

The Belmont Report



- USA: *Belmont Report* (1979) and subsequent legislation of *Common Rule* (1981)

- Three guiding ethical principles:
  - **Respect** for research participants
    - Must enter research voluntarily and with adequate information (informed consent)
  - **Beneficence**
    - (1) Do not harm
    - (2) Maximize possible benefits and minimize possible harms
  - **Justice** in participant selection
    - Fairly distribute benefits and burdens of research

Belmont Report vs. GHTorrent

Sampling using GHTorrent:

- Users may change their behavior due to “survey spam” (e.g., remove email address from profile)
- Active users are likely to get contacted more often
- Frequently contacted users may refuse to answer → biased samples

Beneficence?

Justice in partici-pant selection?

Beneficence?

Exemplary Guidelines

CASRO
The Voice and Values of Research

CASRO code of ethics has section about "internet research"

Criteria for email recruitment:

(1) substantive **pre-existing relationship**
(2) based on relationship "reasonable expectation" to be contacted
(3) not opted out
(4) **no recruitment via unsolicited emails**

Problematic strategies: Contacting companies and using GHTorrent

- No substantive pre-existing relationship
- Unsolicited emails
- GitHub users did not share email to be contacted for research

Assessing External Validity of Online Surveys

Dealing with Convenience Samples

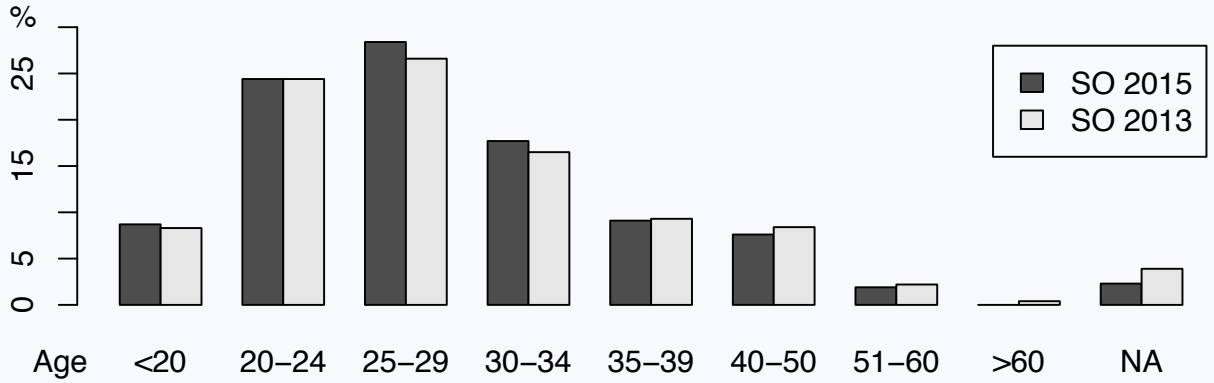
What do we know about the target population of software developers?

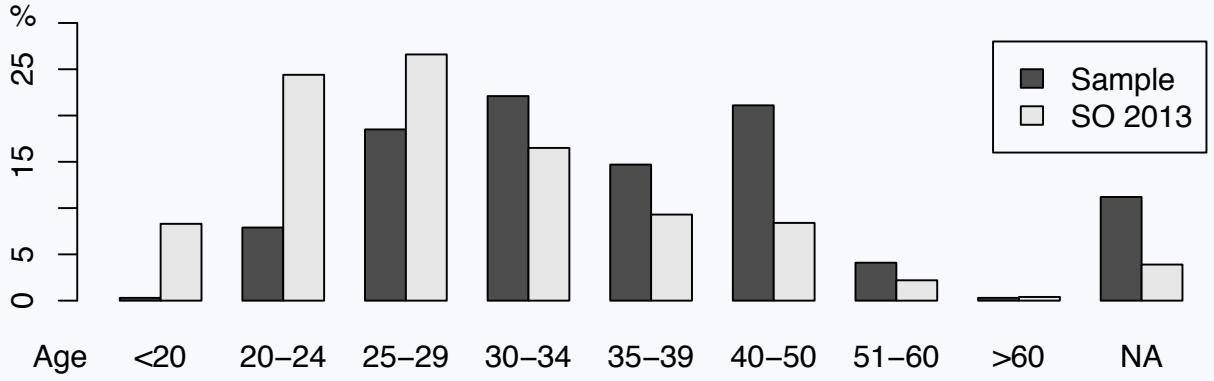
Strategy for dealing with convenience samples:

> **"Carefully select broad cross-section of the target population"**

- No structured and systematic database with demographics of software developers available
- Yearly Stack Overflow developer survey (2010-2017)

Assessing External Validity

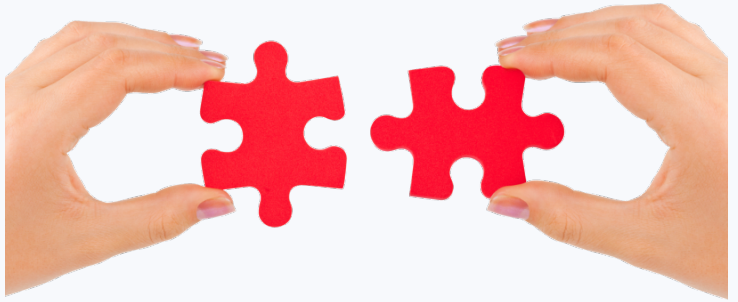




- No major differences between 2013 (n=7,644) and 2015 (n=26,086) data set

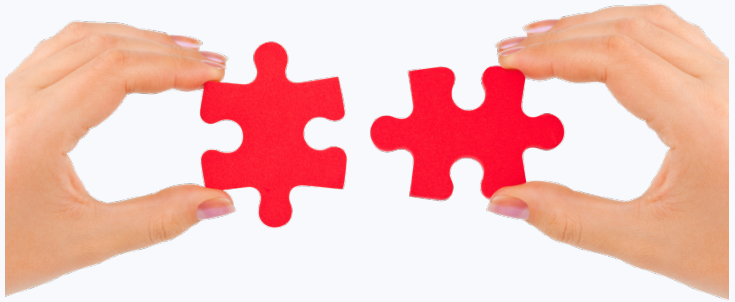
- Our sample **biased towards older and more experienced developers**
- More participants refused to provide their age (5.6% vs. 1.8%)
- Fewer female respondents (2.8% vs. 4.8%)

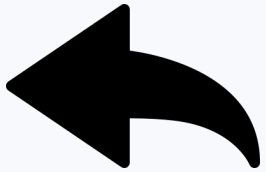
Conclusion



Conclusion

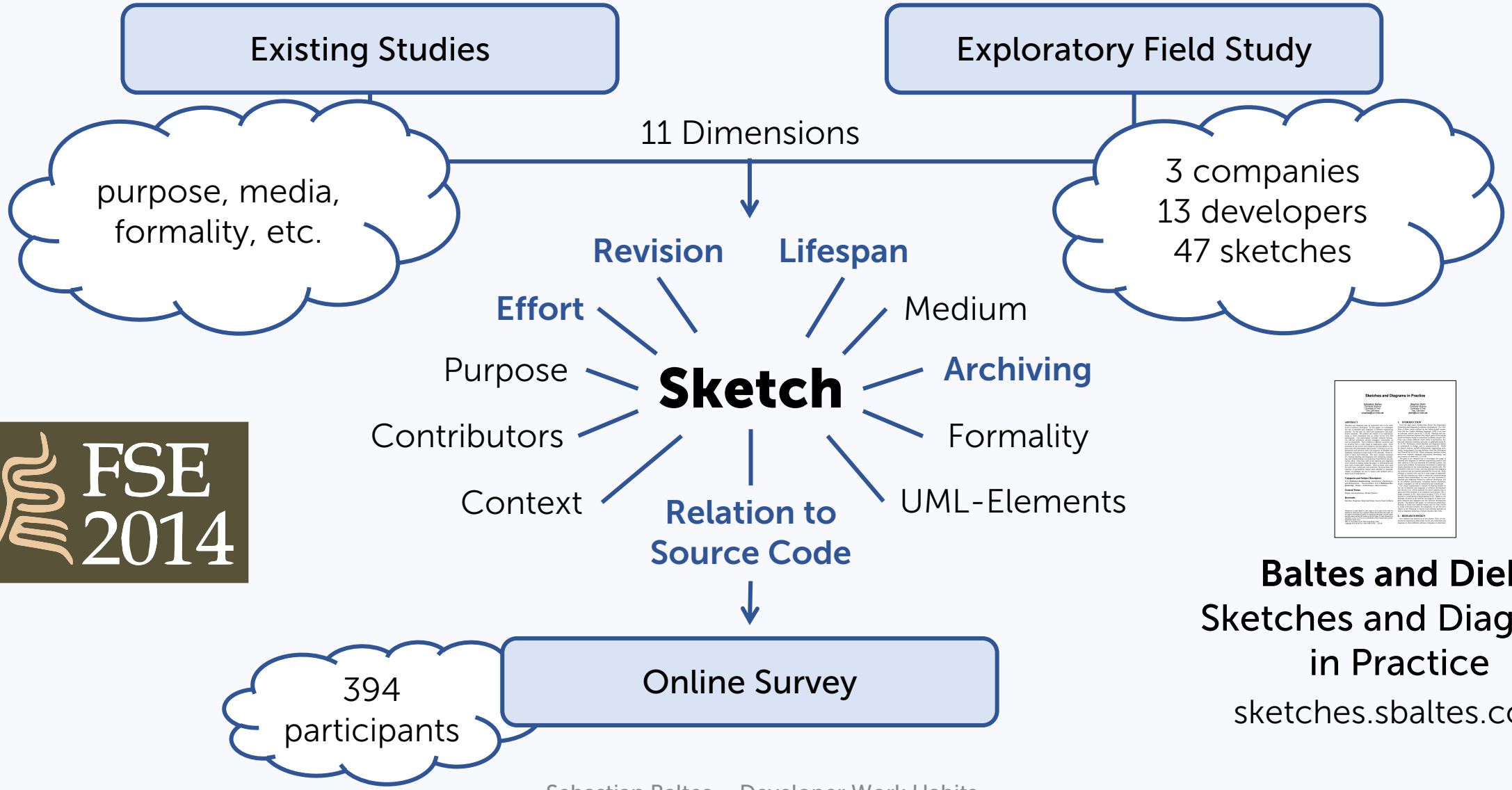
- **Gatekeepers** are important to crosscompany borders
- "Testimonials" on Twitter and an article on a IT news website worked best for us
- Using **GHTorrent** for sampling is compelling, but raises ethical issues
  - We should **discuss ethical implications** of our research at workshops and conferences (see, e.g., CHI and CSCW).
  - Survey with SE researchers about their notion of ethics
- To assess external validity of a survey, a **collection of developer demographics** is needed
  - Systematic literature review (e.g., age, experience, typical sample sizes)
  - Contacting authors of surveys conducted over the past years





Back to our study

Research Design



FSE 2014

Baltes and Diehl
Sketches and Diagrams in Practice
sketches.sbaltes.com

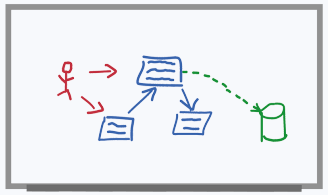
Survey Results

Purpose

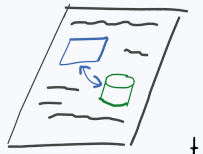
Designing (75 %) 1
Explaining (60 %) 2
Understanding (56 %) 3
Analyzing Requirements (45 %)

Revision

54 %



whiteboard (40 %)



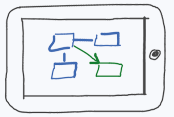
paper (18 %)

analog (58 %)

Media

transitions between media are common

computer (39 %)



tablet (<1 %)

digital (40 %)

Revision

77 %

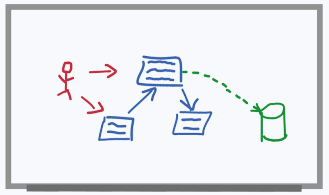
Survey Results

Purpose

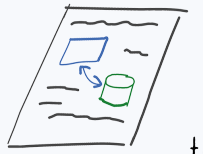
Designing (75 %) 1
Explaining (60 %) 2
Understanding (56 %) 3
Analyzing Requirements (45 %)

Revision

54 %



whiteboard (40 %)



paper (18 %)

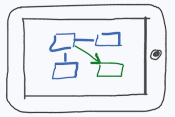
analog (58 %)

Media

transitions between media are common

Revision

computer (39 %)

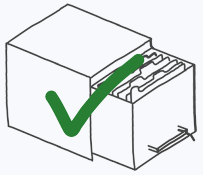


tablet (<1 %)

77 %

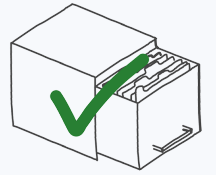
digital (40 %)

Archiving



archived (38 %)

not archived (62 %)

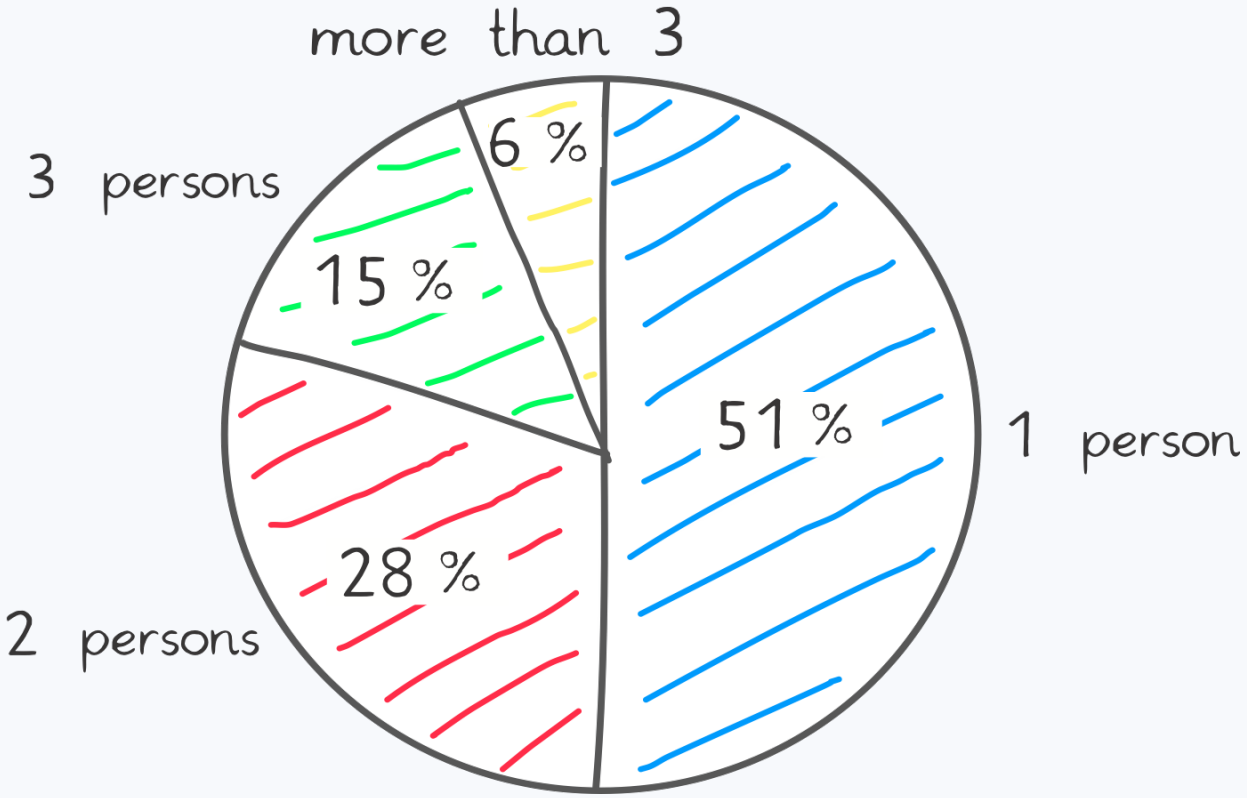


archived (94 %)

not archived (6 %)

Survey Results

Contributors:

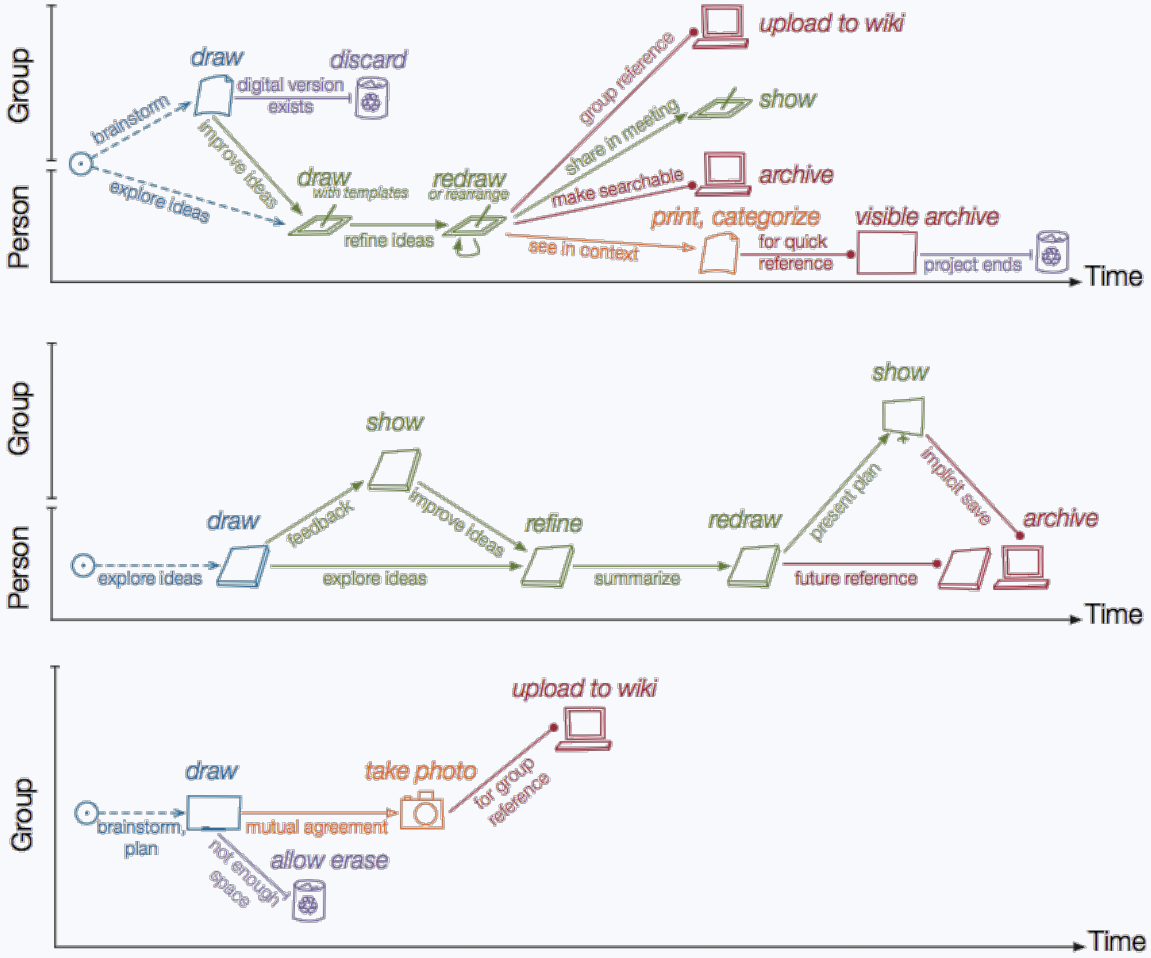


Related Work


Follow that Sketch: Lifecycles of Diagrams and Sketches in Software Development

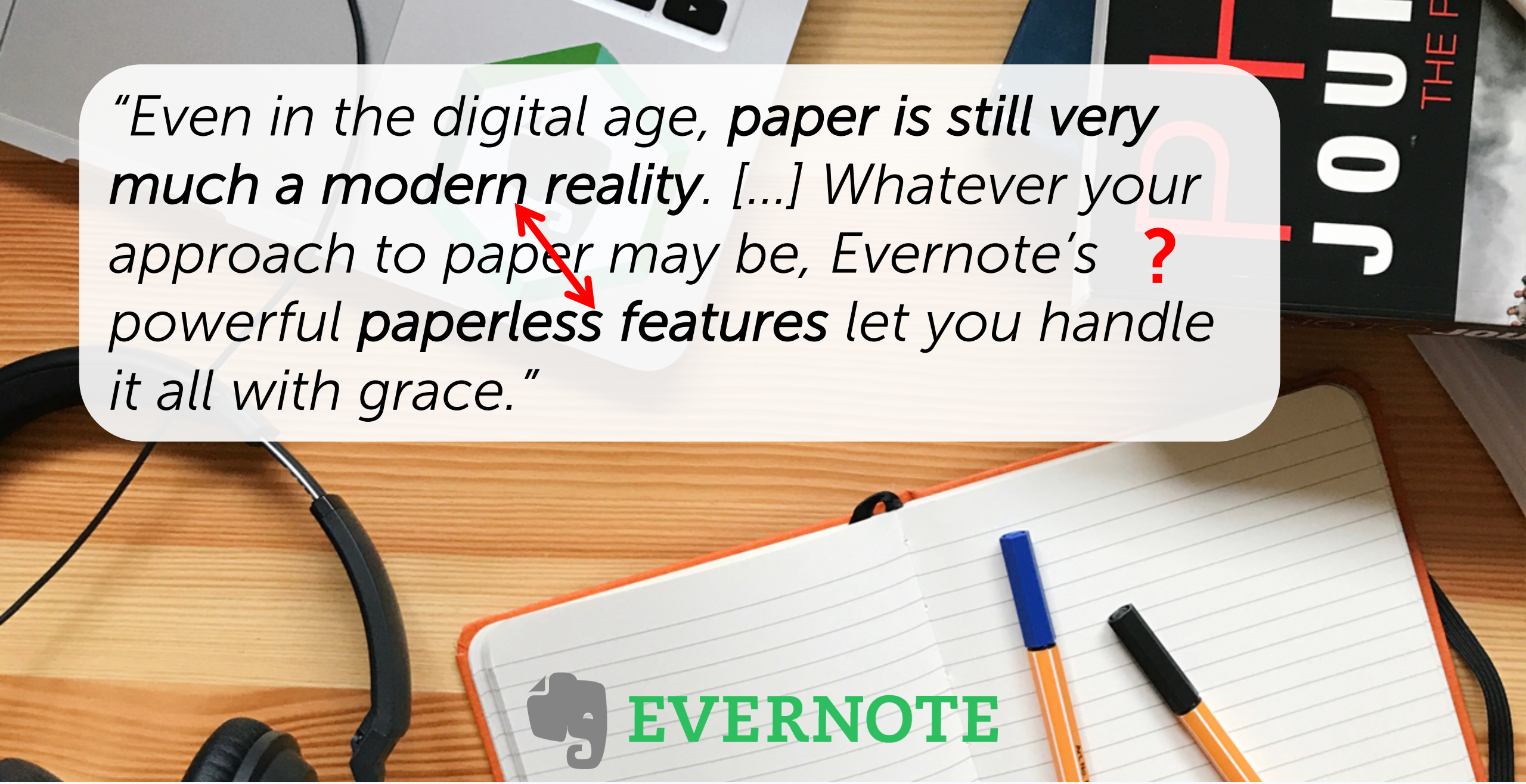
Walny et al.
Follow that sketch: Lifecycles of diagrams and sketches in software development

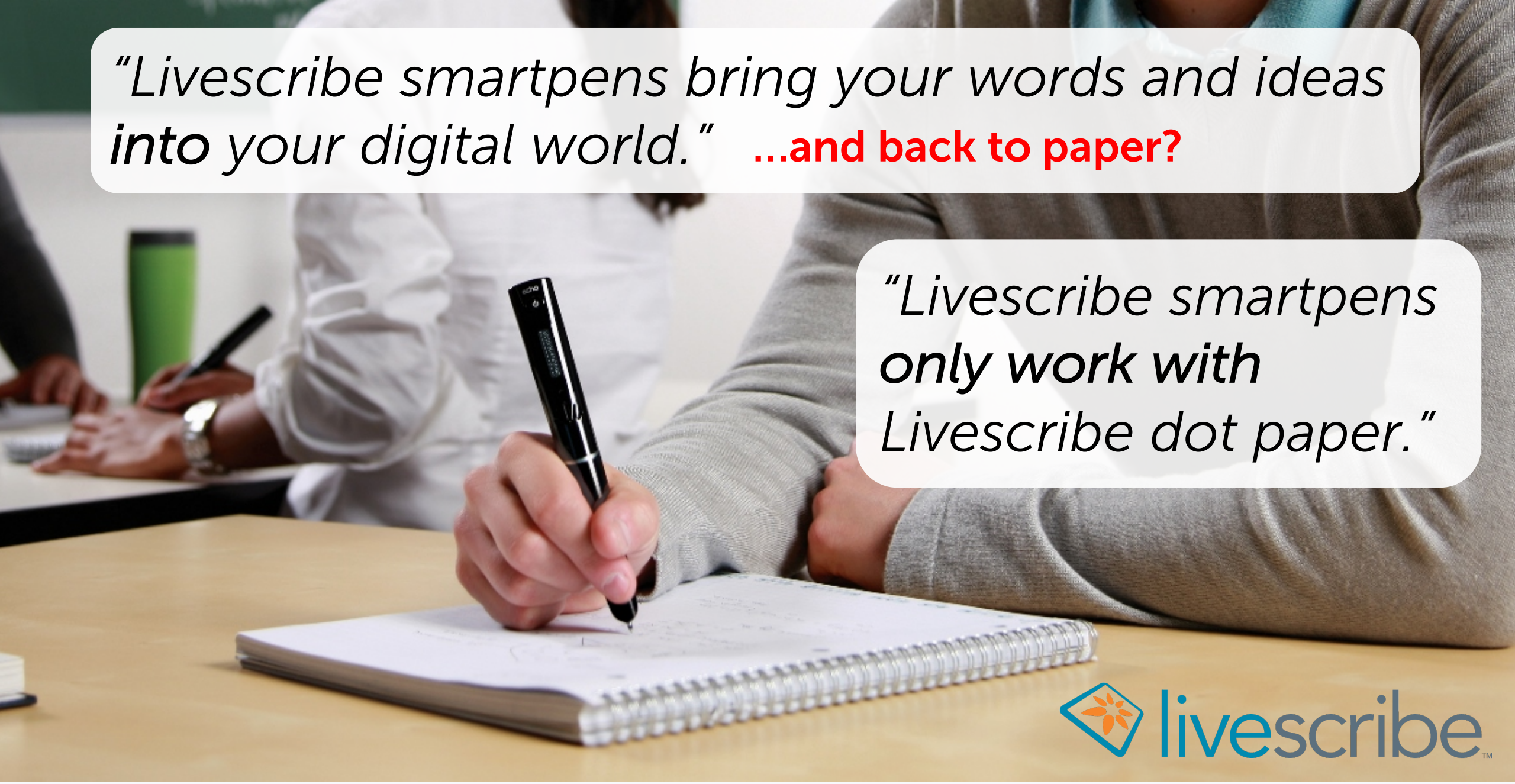
VISSOFT 2011



Tool support?


?
"Create your way"
OneNote


"Even in the digital age, paper is still very much a modern reality. [...] Whatever your approach to paper may be, Evernote's powerful paperless features let you handle it all with grace."
?
EVERNOTE

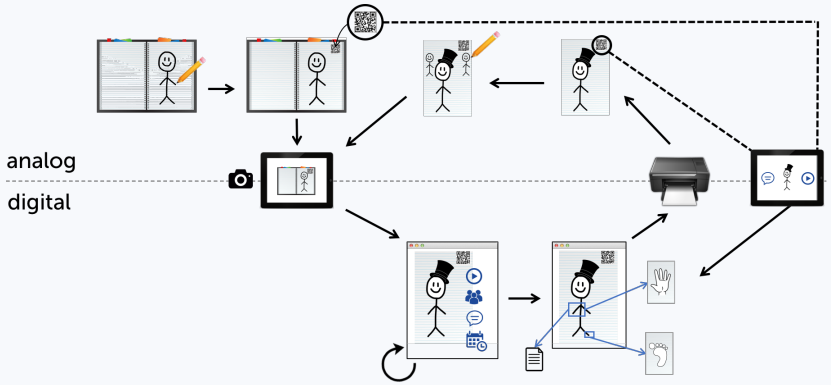

"Livescribe smartpens bring your words and ideas **into** your digital world." ...and back to paper?
"Livescribe smartpens **only work with** Livescribe dot paper."
livescribe

Our Motivation

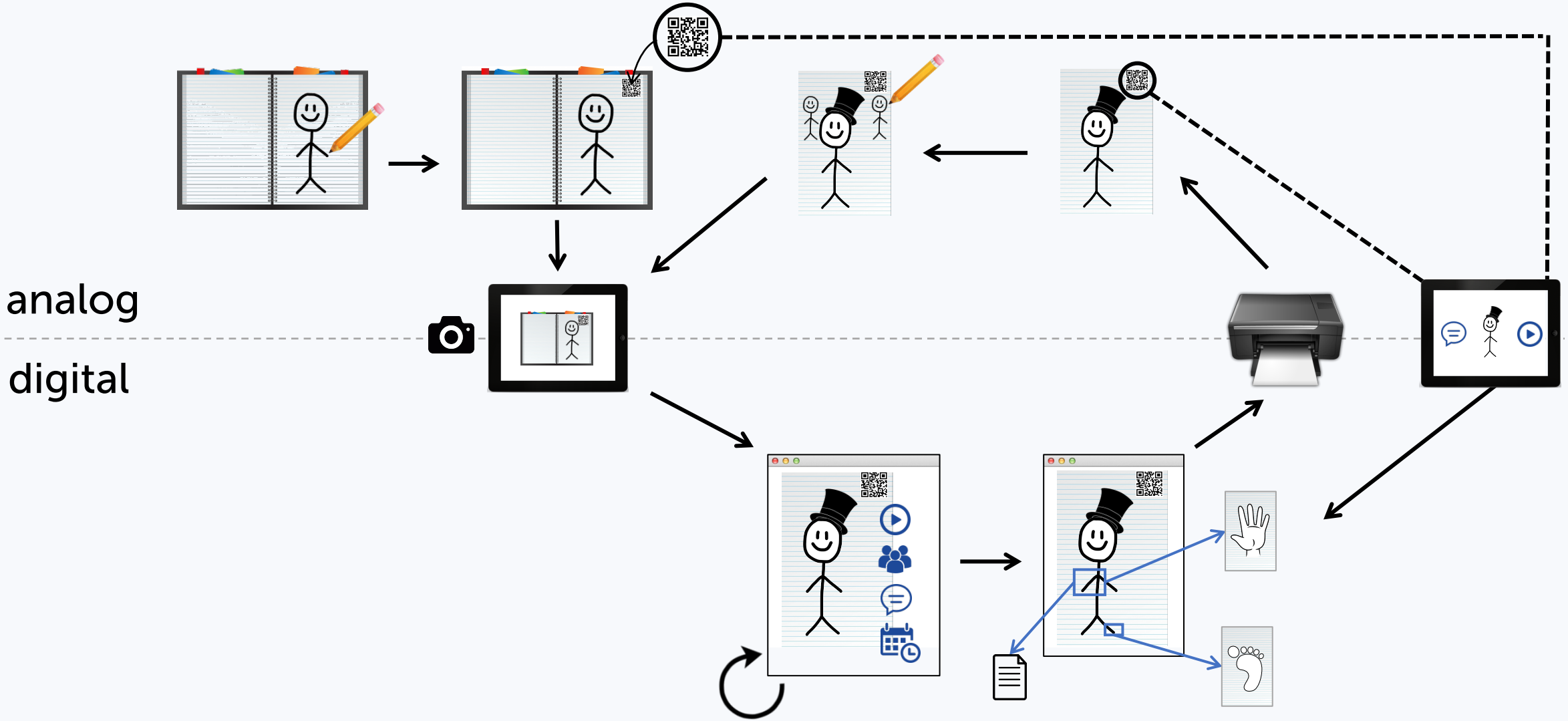
- Many people still use (and prefer?) analog media
- Why force them to use a different (paperless) workflow?

Our goal: Do not treat analog sketches as subordinate artifacts but support different analog-digital workflows

Round-Trip Sketching

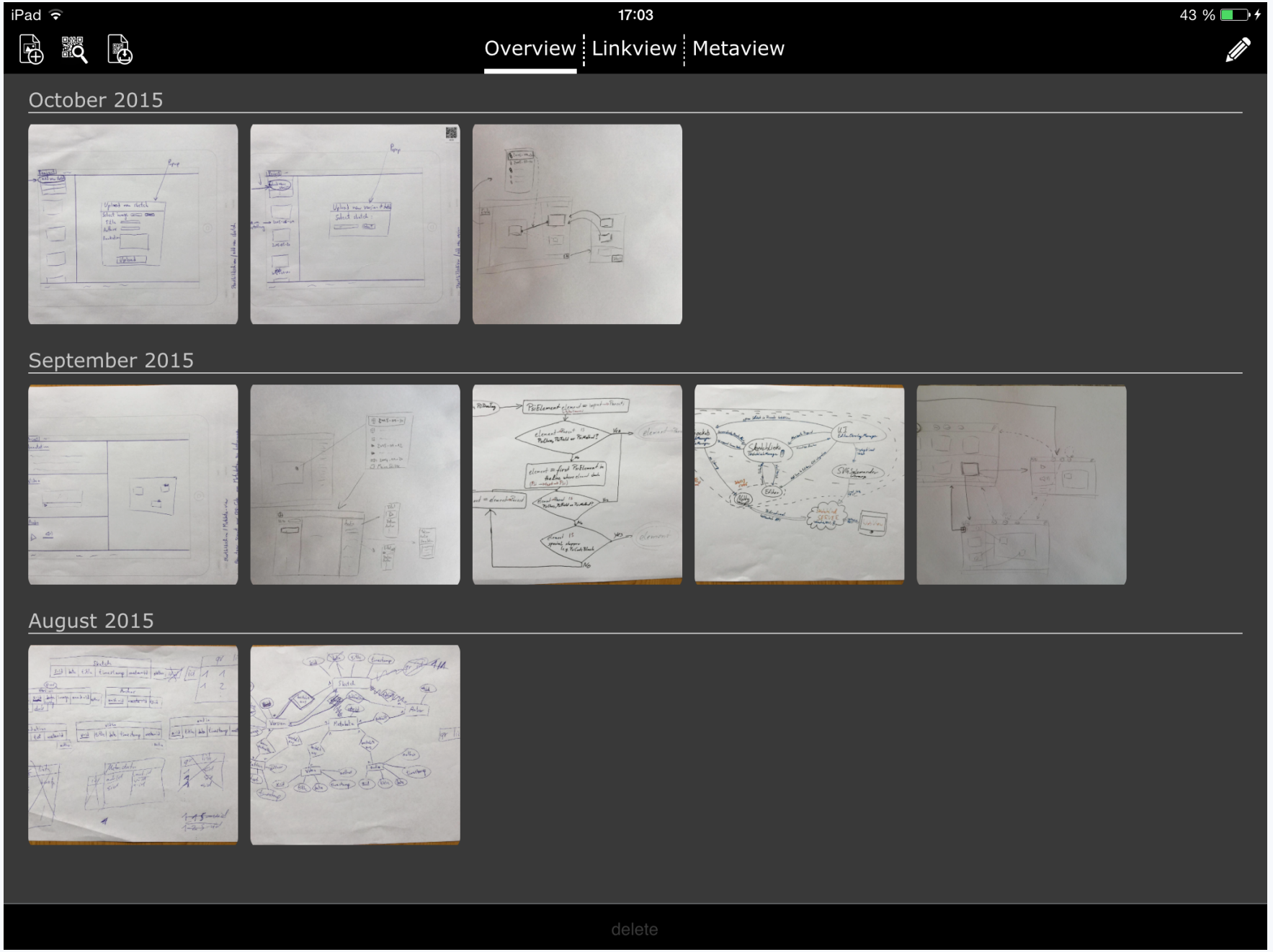


Concept

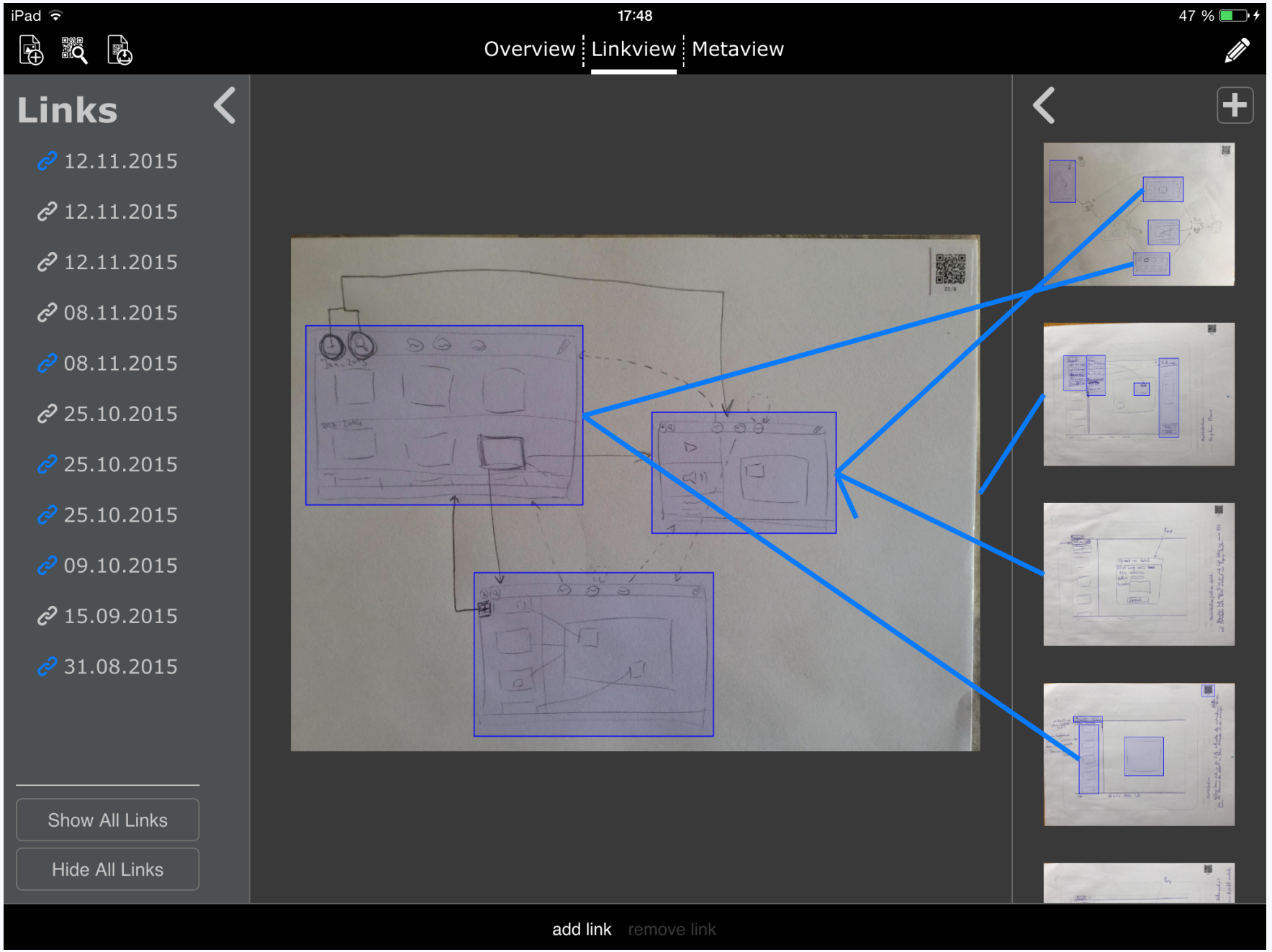


Prototype: LivelySketches

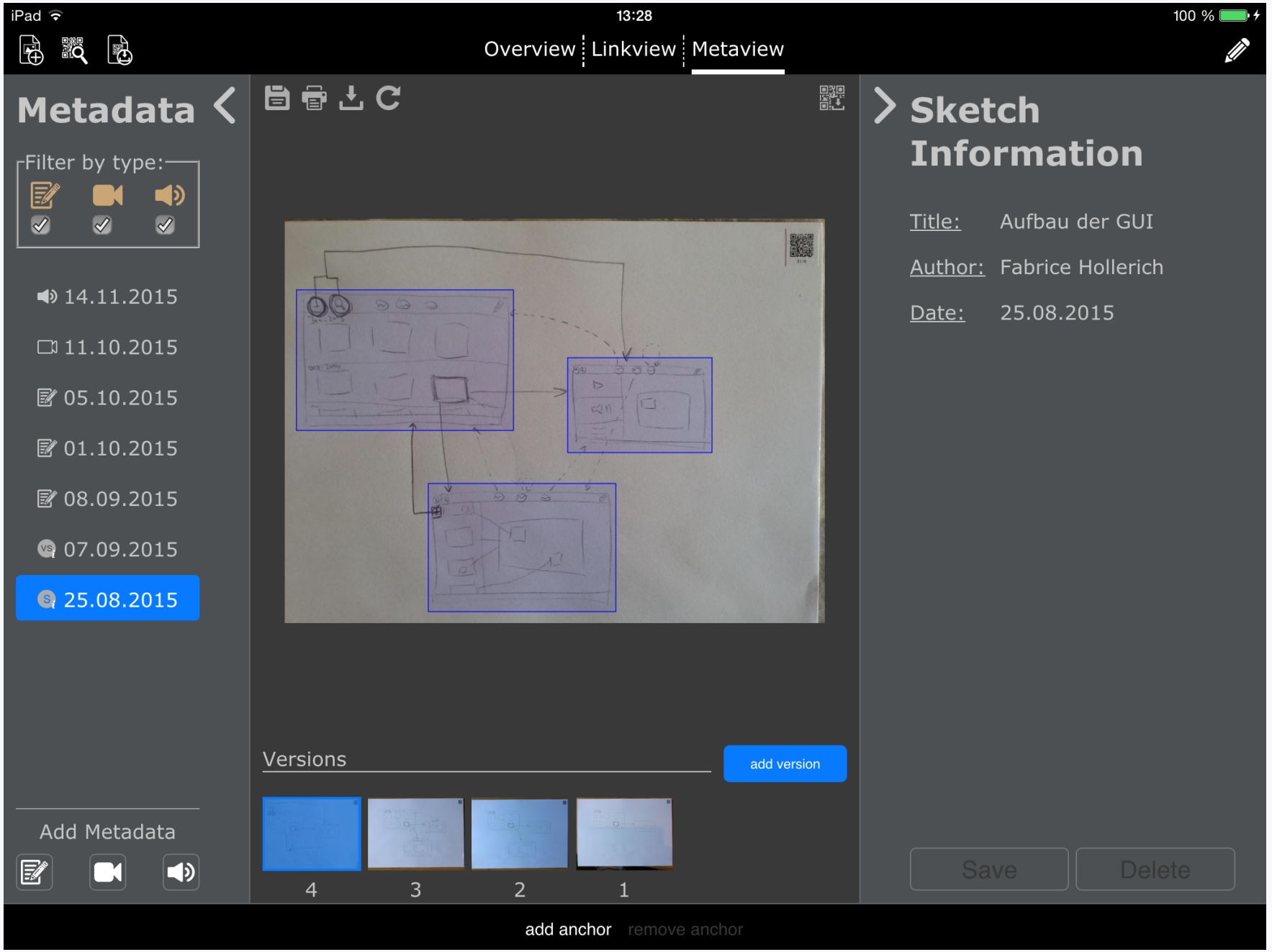
Overview


iPad
17:03
43 %
Overview
Linkview
Metaview
October 2015
September 2015
August 2015
delete

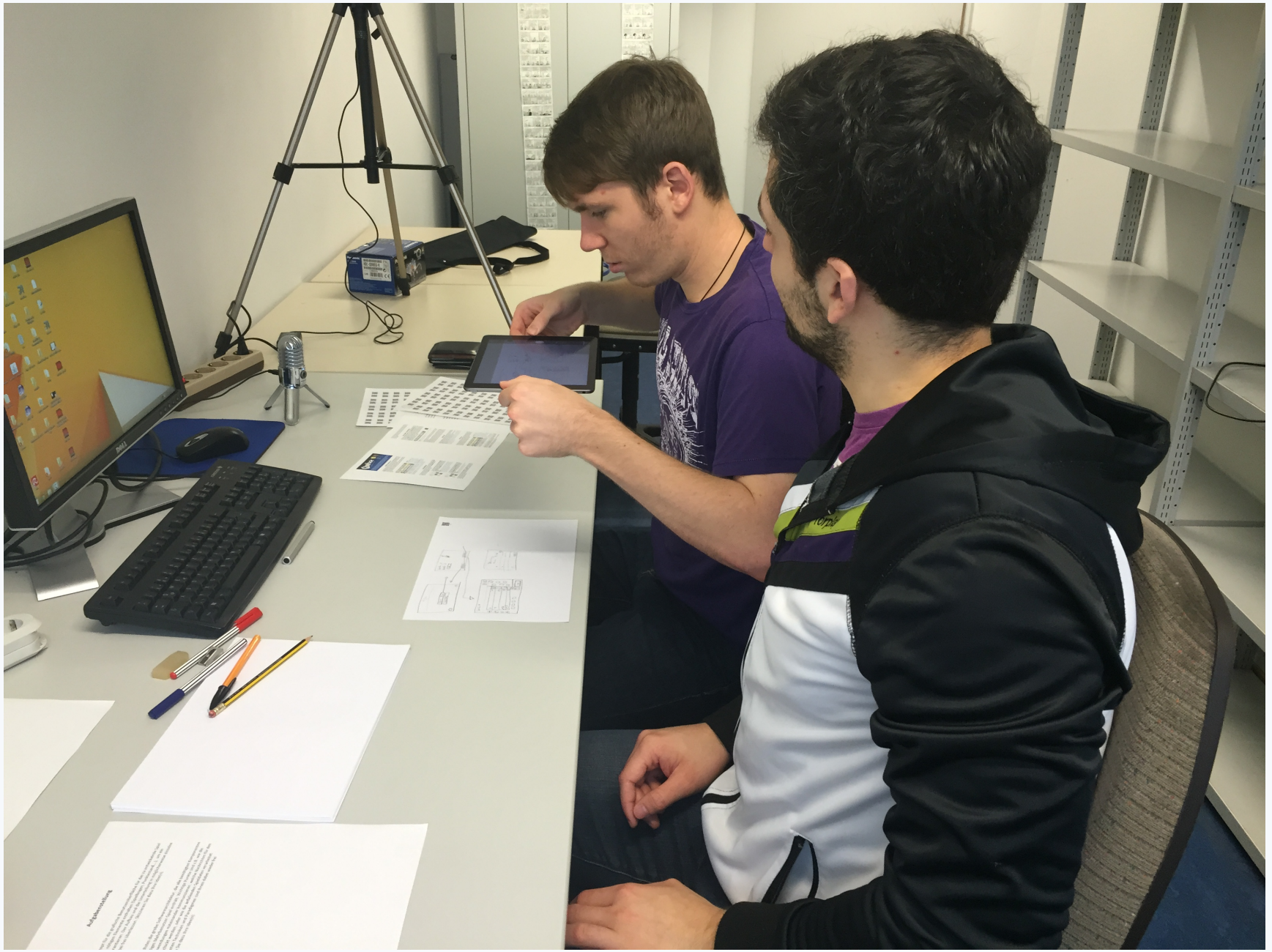
Linkview

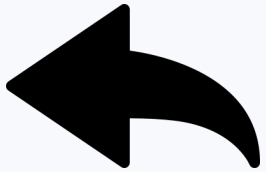

iPad
17:48
47 %
Overview
Linkview
Metaview
Links
12.11.2015
12.11.2015
12.11.2015
08.11.2015
08.11.2015
25.10.2015
25.10.2015
25.10.2015
09.10.2015
15.09.2015
31.08.2015
Show All Links
Hide All Links
add link
remove link

Metaview


iPad
13:28
100 %
Overview
Linkview
Metaview
Metadata
Filter by type:
14.11.2015
11.10.2015
05.10.2015
01.10.2015
08.09.2015
07.09.2015
25.08.2015
Add Metadata
Versions
add version
4
3
2
1
Sketch Information
Title: Aufbau der GUI
Author: Fabrice Hollerich
Date: 25.08.2015
Save
Delete
add anchor
remove anchor

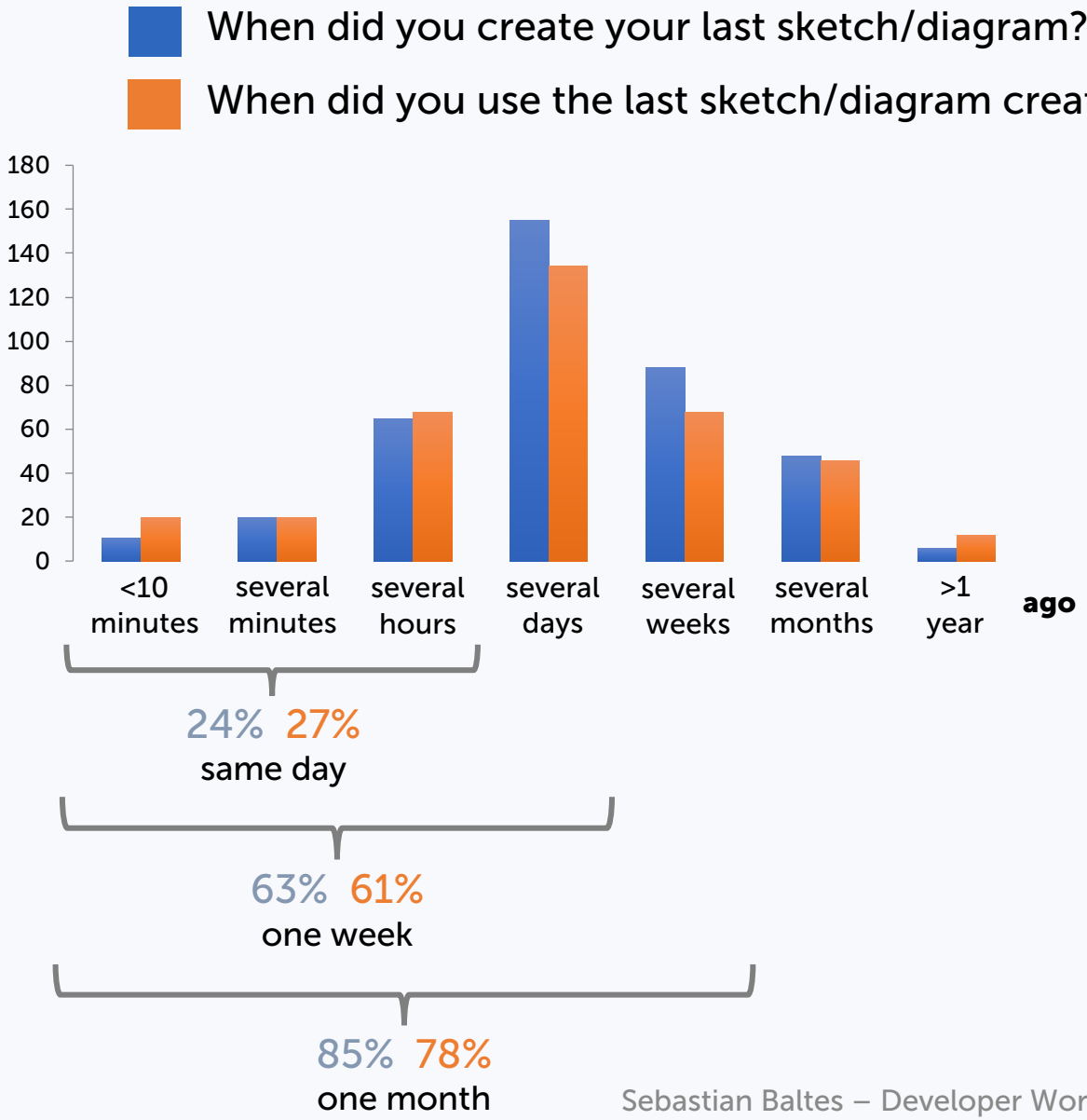
User Study





Back to our study (again)

Creation and Usage

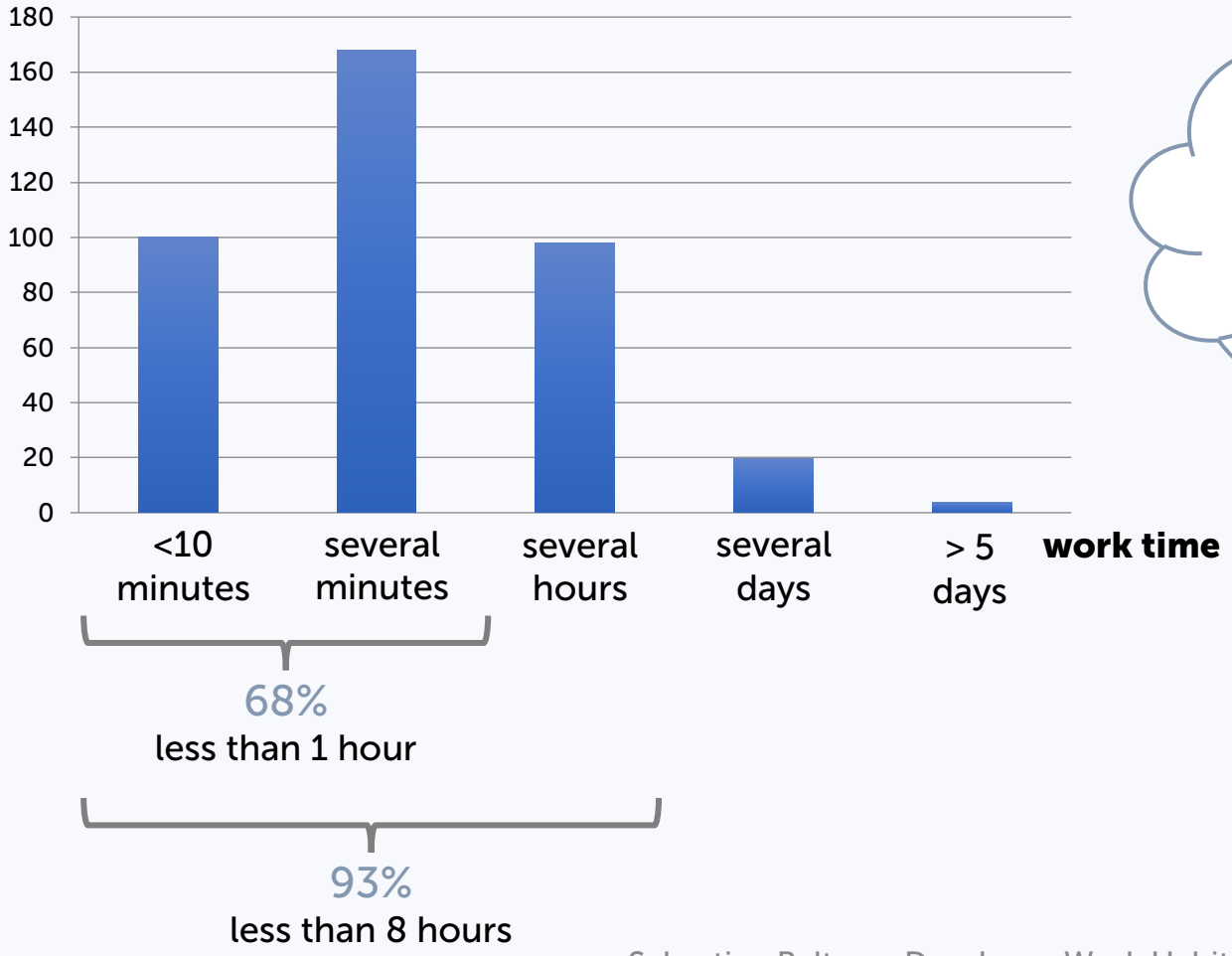


Takeaway 1:

Creating own sketches **and using** sketches created by others are frequent tasks among software practitioners.

Effort and Revision

How much effective work time went into the creation and revision of the sketch/diagram up to now?



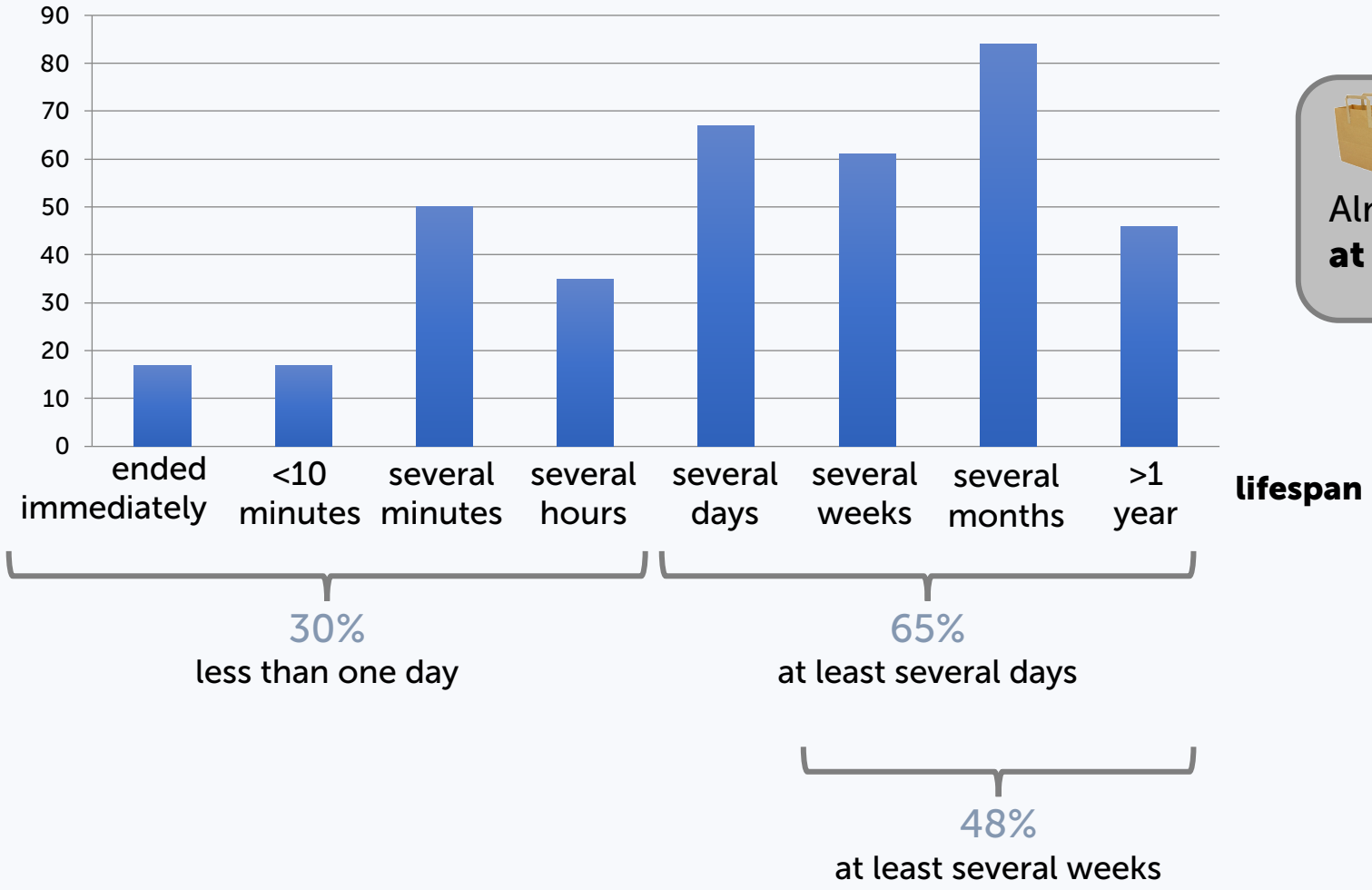
Revision:
15% revised once,
74% multiple times

Takeaway 2:

Most sketches are created in **less than one hour** and are **revised** at least once.

Lifespan

Please try to estimate the lifespan of the sketch/diagram (how long did/will you use it)?



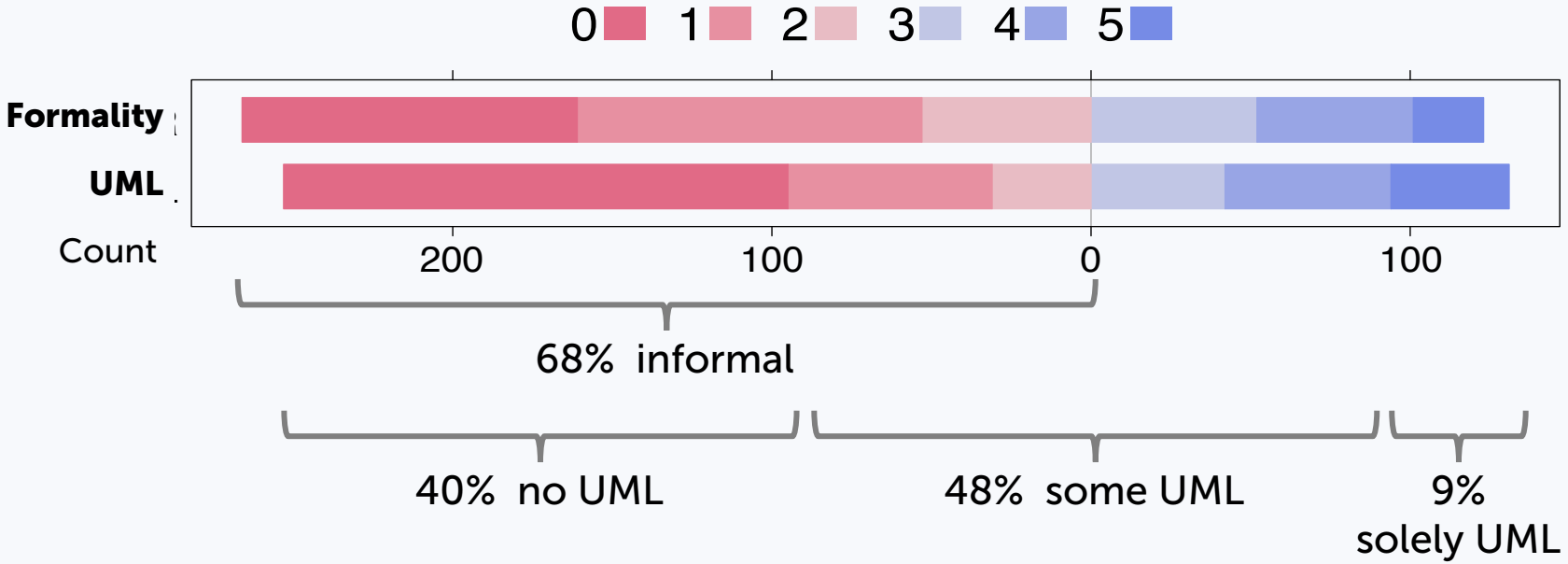
Takeaway 3:

Almost half of the sketches are used for **at least several weeks**.

Formality and UML

Formality: Please try to specify the formality of your sketch/diagram.
(6-point Likert scale (0-5) from "very informal" to "very formal")

UML: To which degree does the sketch/diagram contain UML elements?
(6-point Likert scale (0-5) from "no UML elements" to "only UML elements")



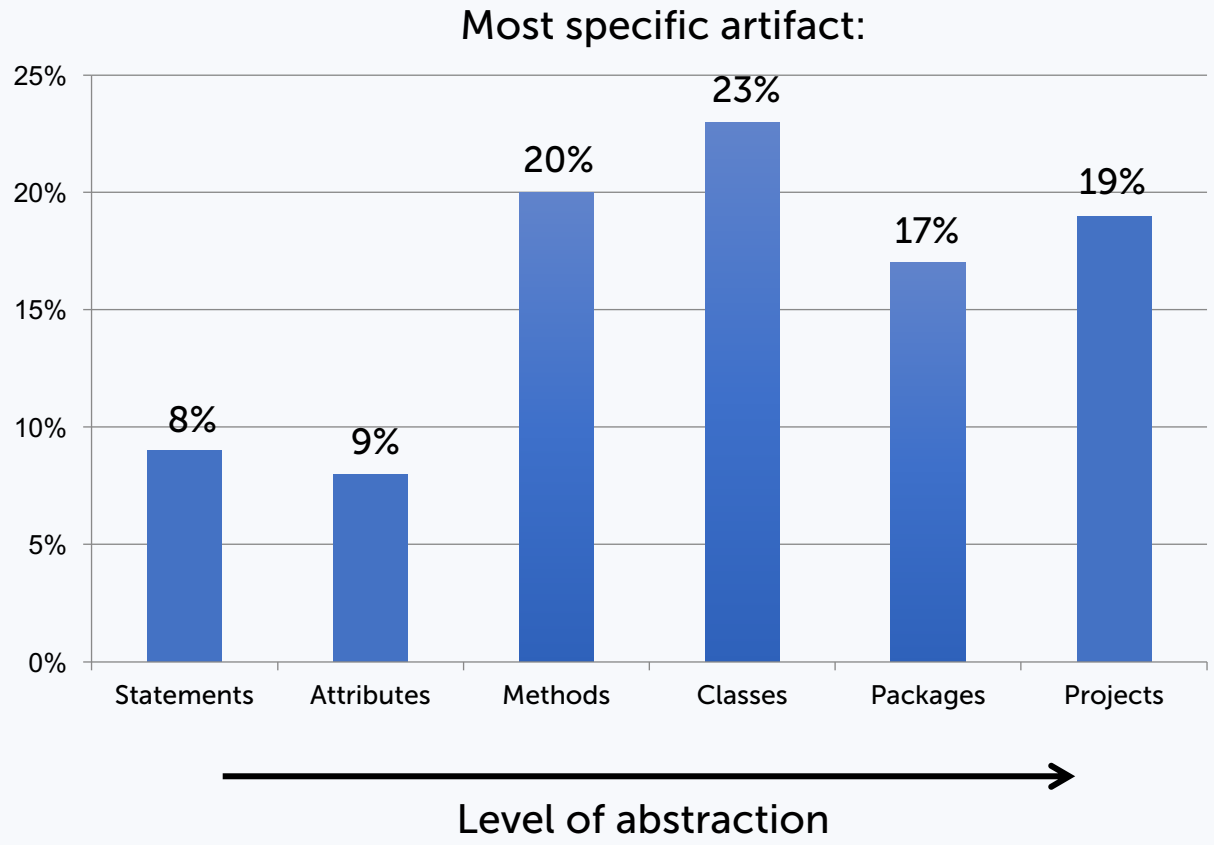
Takeaway 4:

The majority of sketches and diagrams are **informal**.
If UML is used, it is often mixed with other notations.

Relation to Source Code

Please select the software artifact(s) to which the content of the sketch/diagram is related?

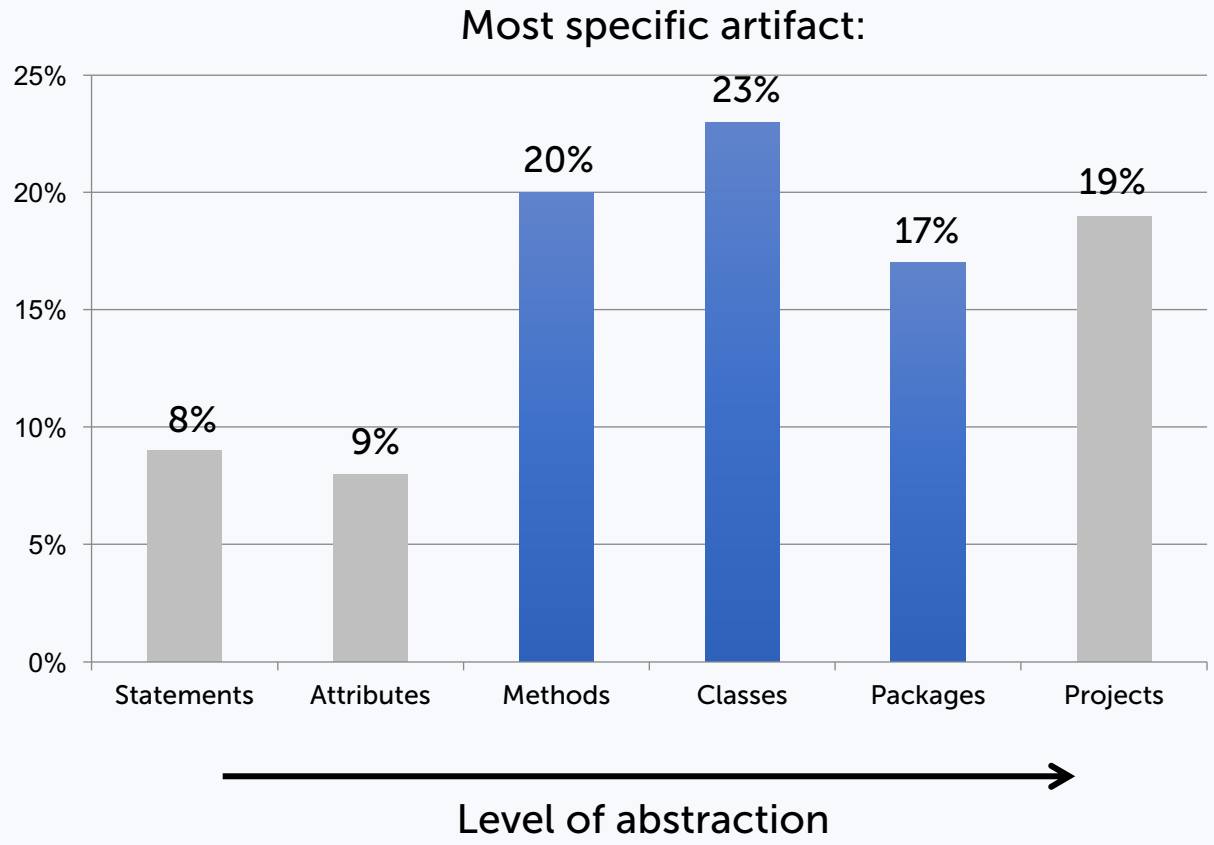
(multiple answers or no answer possible)



Relation to Source Code

Please select the software artifact(s) to which the content of the sketch/diagram is related?

(multiple answers or no answer possible)

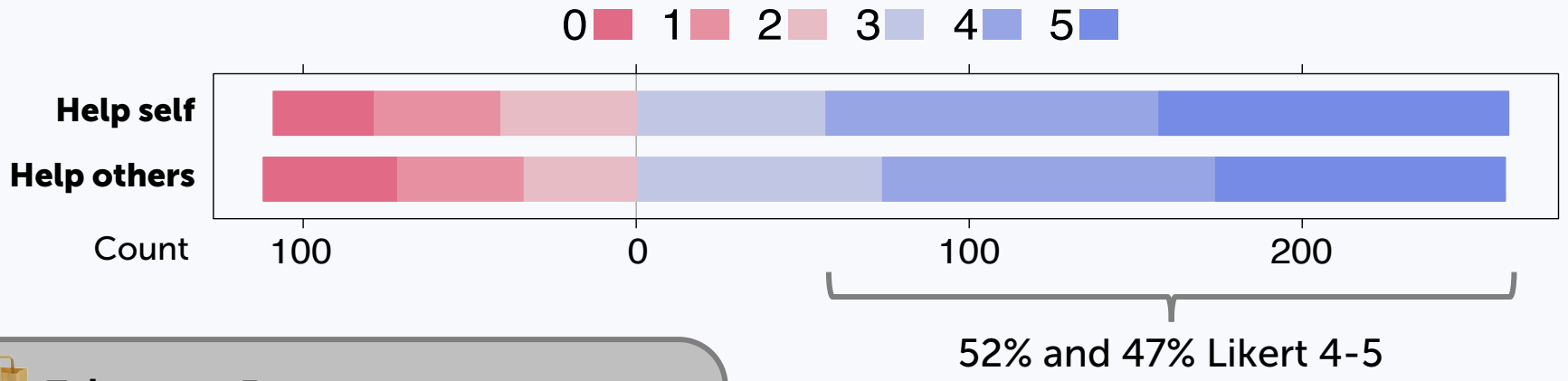


Relation to Source Code

Help self: Do you think that the sketch/diagram could help you in the future to understand the related source code artifact(s)?

Help others: ... help other software developers ...

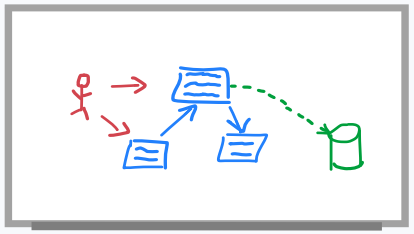
(6-point Likert scale (0-5) from "It will definitely not help " to "It will definitely help")



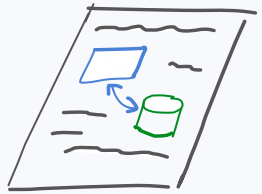
Takeaway 5:

About **half of the sketches are rated as helpful** to understand the related source code artifact(s) in the future.

Archiving

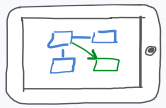


Whiteboard (40%)

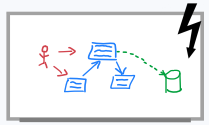


Paper (18%)

Computer (39%)

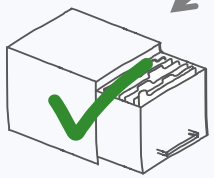


Tablet (0.8%)



E-Whiteboard (1.5%)

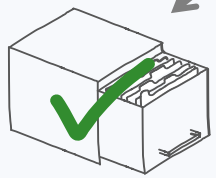
Analog (58%)



Archived (38%)

Not archived (62%)

Digital (42%)



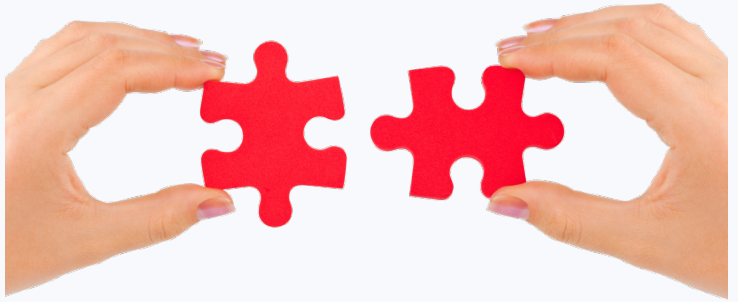
Archived (94%)

Not archived (6%)

Takeaway 6:

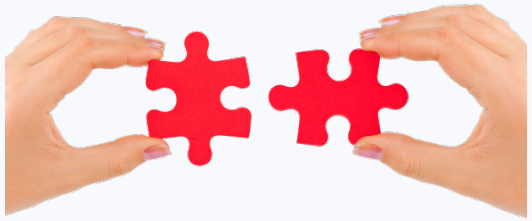
Most digital sketches, but also more than one third of the analog sketches, **are archived**.

Conclusion



Conclusion

- **Software documentation** is frequently **poorly written** and out of date
  [Forward02, Lethbridge03]
- Sketches and diagrams could serve as a **supplement** to conventional documentation
- Software practitioners are **willing to keep** their sketches and diagrams
- **Better tool support needed** for archiving and retrieving sketches/diagrams related to source code artifacts



- Tools should support **evolution** of sketches/diagrams (and software)

More Tool Support

Linking Sketches and Diagrams to Source Code Artifacts

SketchLink

- Video with voice-over:
  https://www.youtube.com/watch?v=3IuLKZx7Wbs

- Video with subtitles:
  https://www.youtube.com/watch?v=mG6xCiQpS80

Navigate, Understand, Communicate

How Developers Locate Performance Bugs

Definitions

"A bug that affects speed or responsiveness."

(Bugzilla@Mozilla)

"Defects where relatively simple source code changes can significantly speed up software, while preserving functionality."

(Jin et al. - *Understanding and Detecting Real-World Performance Bugs,* PLDI'12)

Research Gap

Most existing debugging studies focused on how developers fix functional bugs.

But:

Performance

- is a non-functional requirement
- is difficult to measure (benchmarks?)

Performance bugs

- may corrupt user experience
- may waste resources (time, energy)
- can be difficult to reproduce and locate
- require knowledge of program state and runtime consumption

No study focusing on how developers locate (and fix) performance bugs.

Research Questions

RQ1:

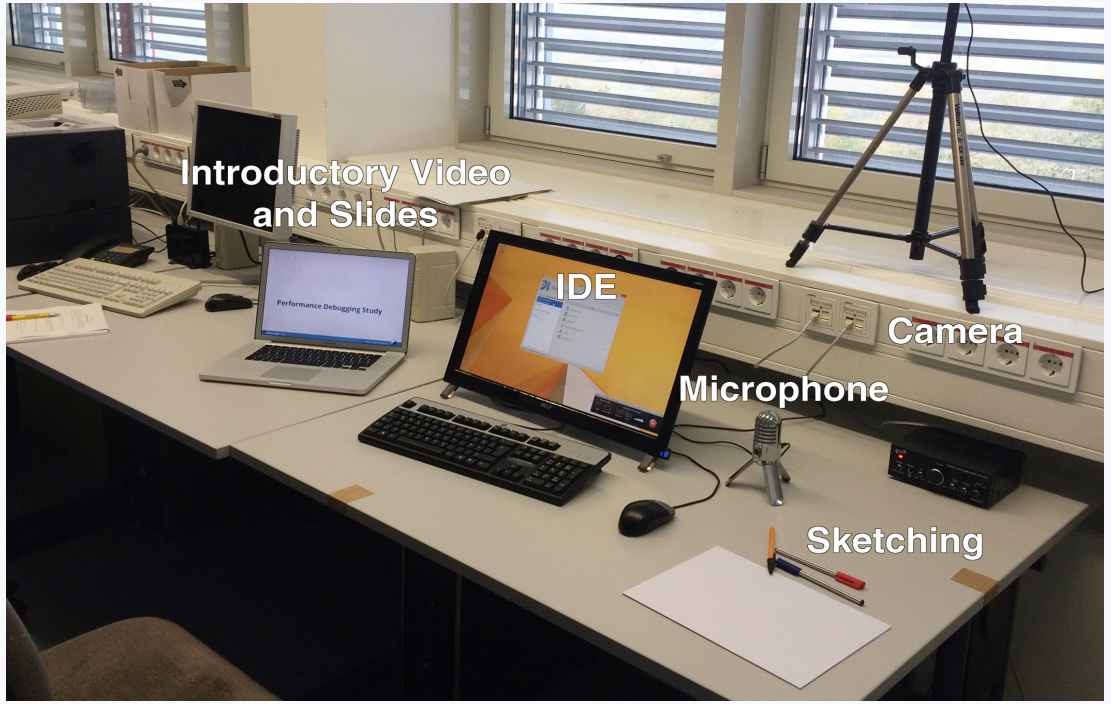
How do developers **navigate** the source code and what **information and representation** is supportive for **locating** a performance bug?

RQ2:

How do developers try to **understand** and **explain** the causes of performance bugs?

Study Design

- Qualitative observation study
- Controlled setting
- 12 developers, pair programming
- Locate and fix four performance bugs in collection libraries
  (Apache Commons Collections and Google Guava Libraries)

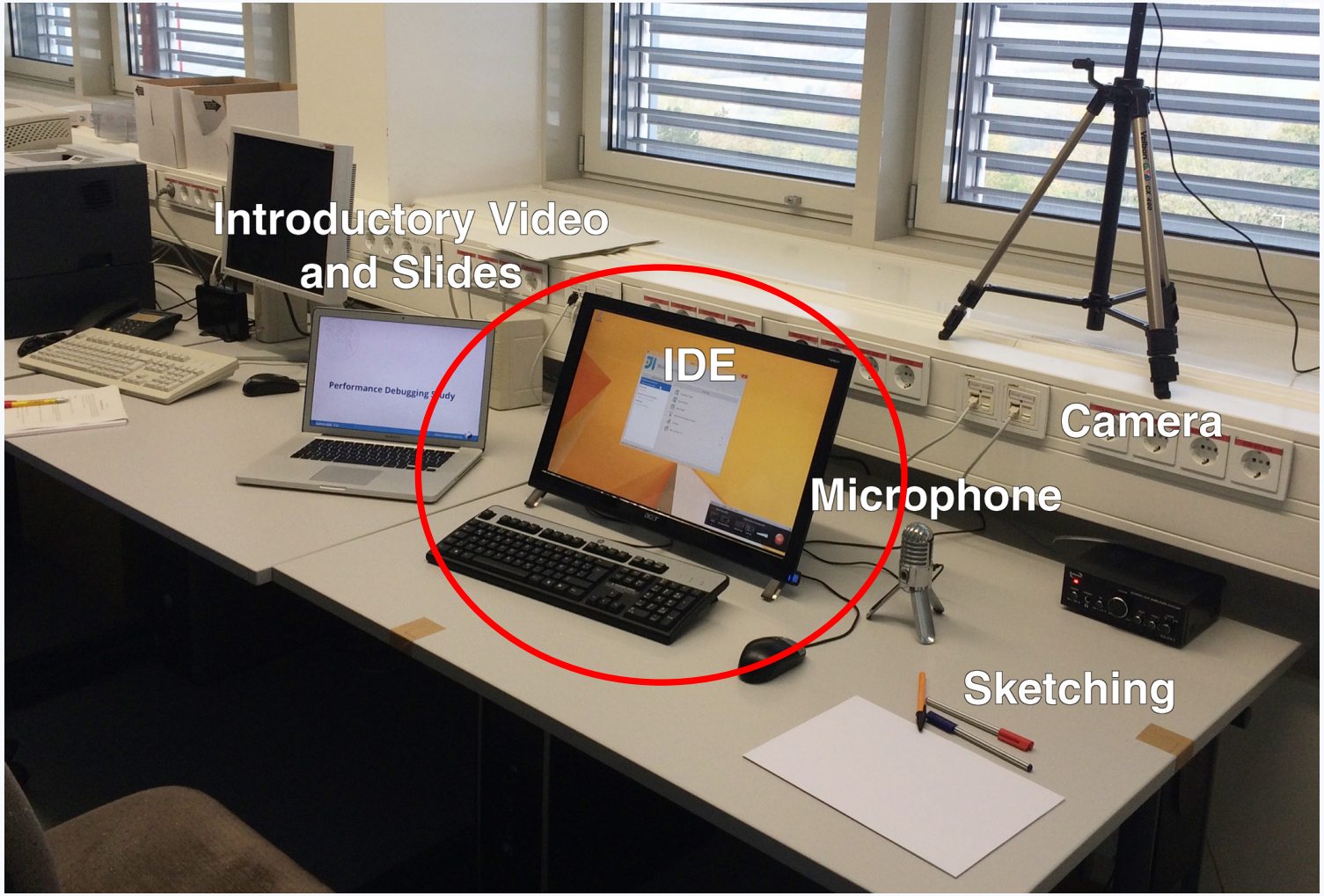


Participants

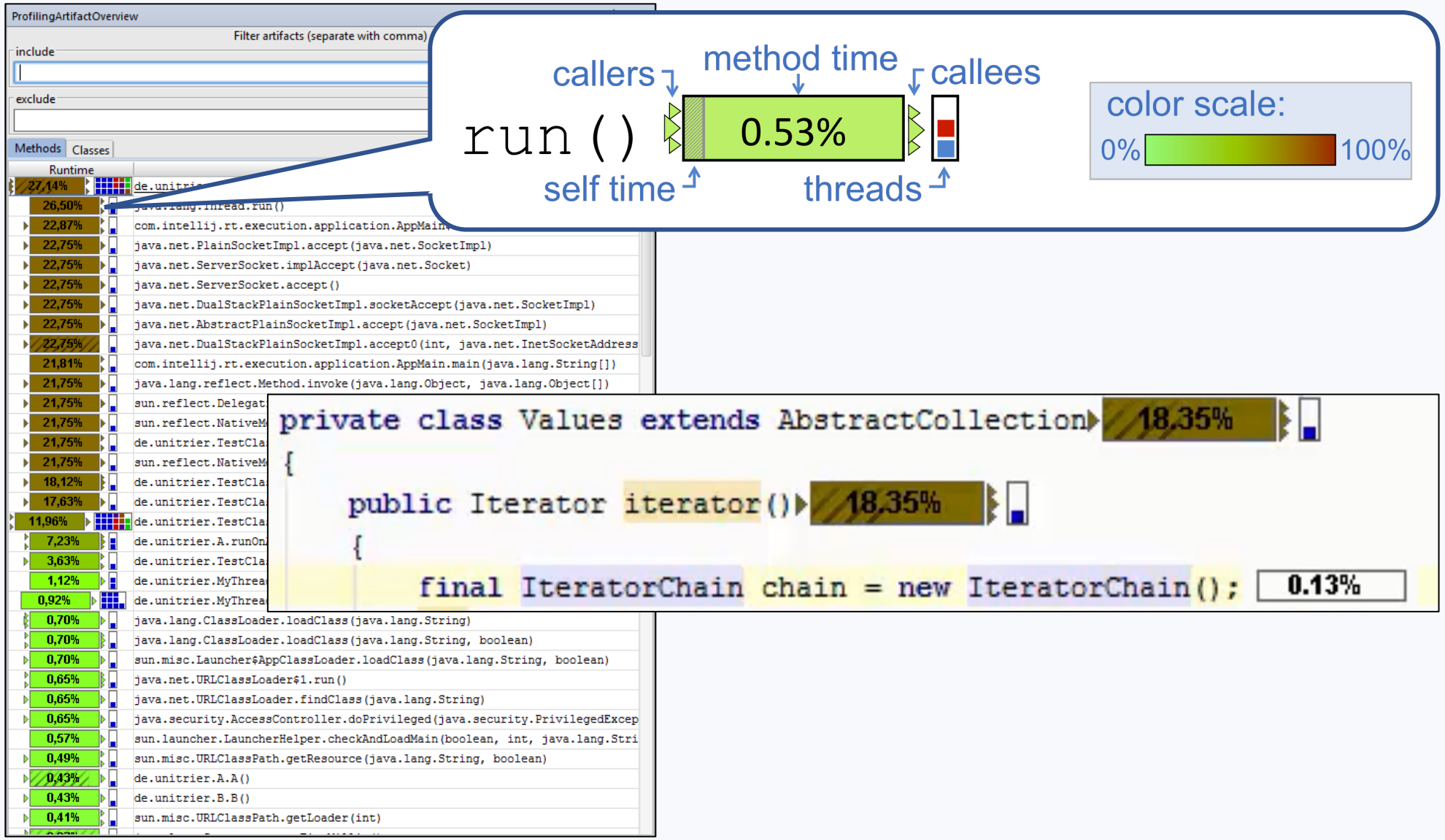
Team	Participant	Current Occupation	Work Exp. (years)	Experience (no exp. = 0 to 4 = expert) OOP	Java	Collec.	IntelliJ	IDEs	Perf.Bugs	Our Tool	Profiling
T1	P1	Research assistant	5	4	4	3	3	3	1	1	0
	P2	Research assistant	5	4	4	4	1	4	2	1	1
T2	P3	MSc student, industry exp.	1	3	3	2	0	3	1	0	2
	P4	MSc student, industry exp.	3	3	3	3	1	2	1	0	1
T3	P5	Software developer	3	4	3	4	1	3	3	1	2
	P6	Diploma student	4	3	3	3	4	2	1	1	0
T4	P7	MSc student	0	3	2	3	1	2	1	0	0
	P8	MSc student	0	1	1	0	0	1	1	0	1
T5	P9	Research assistant, industry exp.	10	3	2	3	0	4	4	0	3
	P10	Research assistant, industry exp.	6	2	2	2	3	1	3	0	2
T6	P11	Software developer	15	3	1	3	0	3	2	1	2
	P12	Software developer	1	3	3	2	2	2	1	0	1
		mean values:	4.4	3.0	2.6	2.7	1.3	2.5	1.8	0.4	1.3

- All male
- Between 22 and 43 years old
- All except one team had industry experience
- Good level of expertise in OOP, Java, and data structures
- Lack of experience with IntelliJ IDE
- Not much experience fixing performance bugs (rare event)

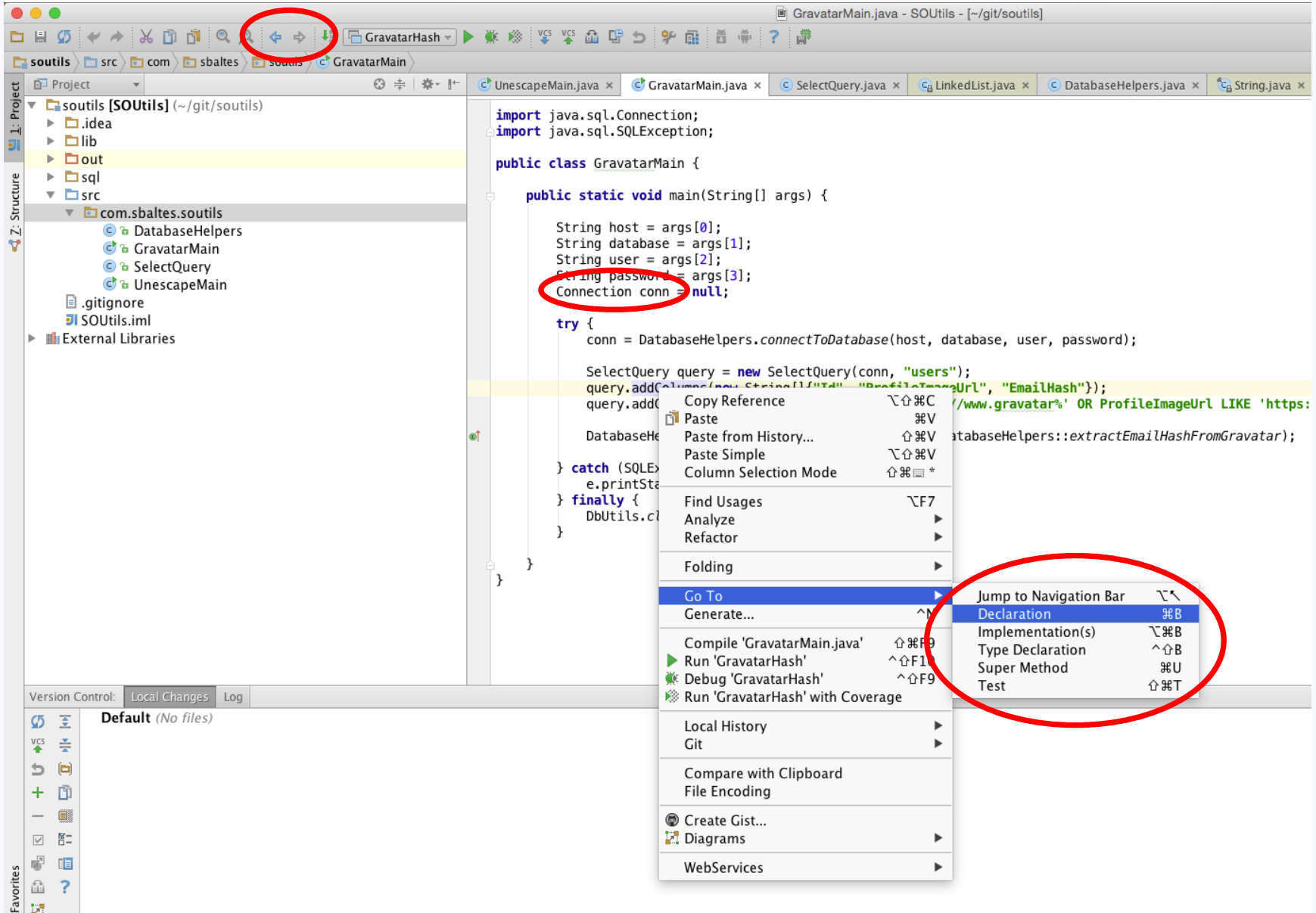
Setup


Introductory Video
and Slides
IDE
Camera
Microphone
Sketching

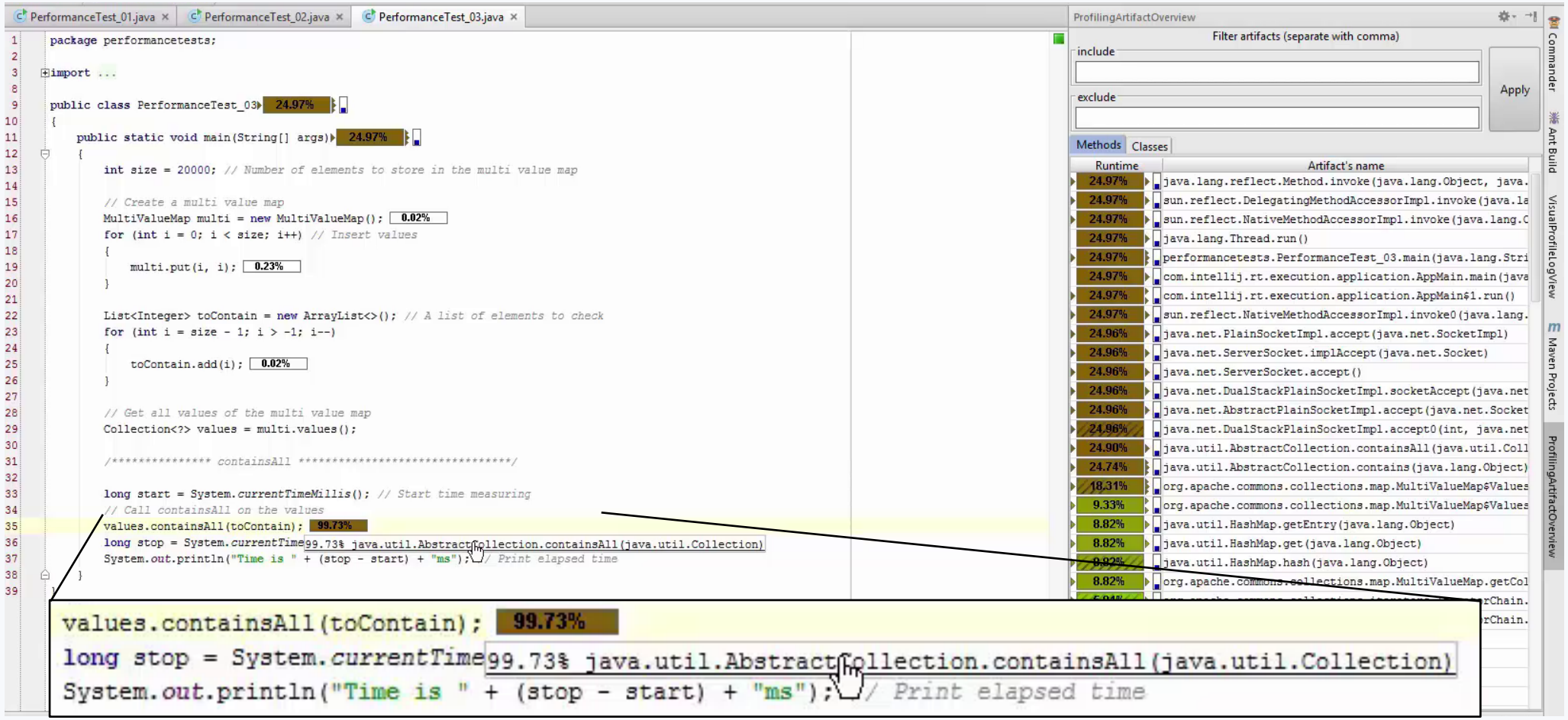
Visualization of Profiling Data



Navigation: IDE

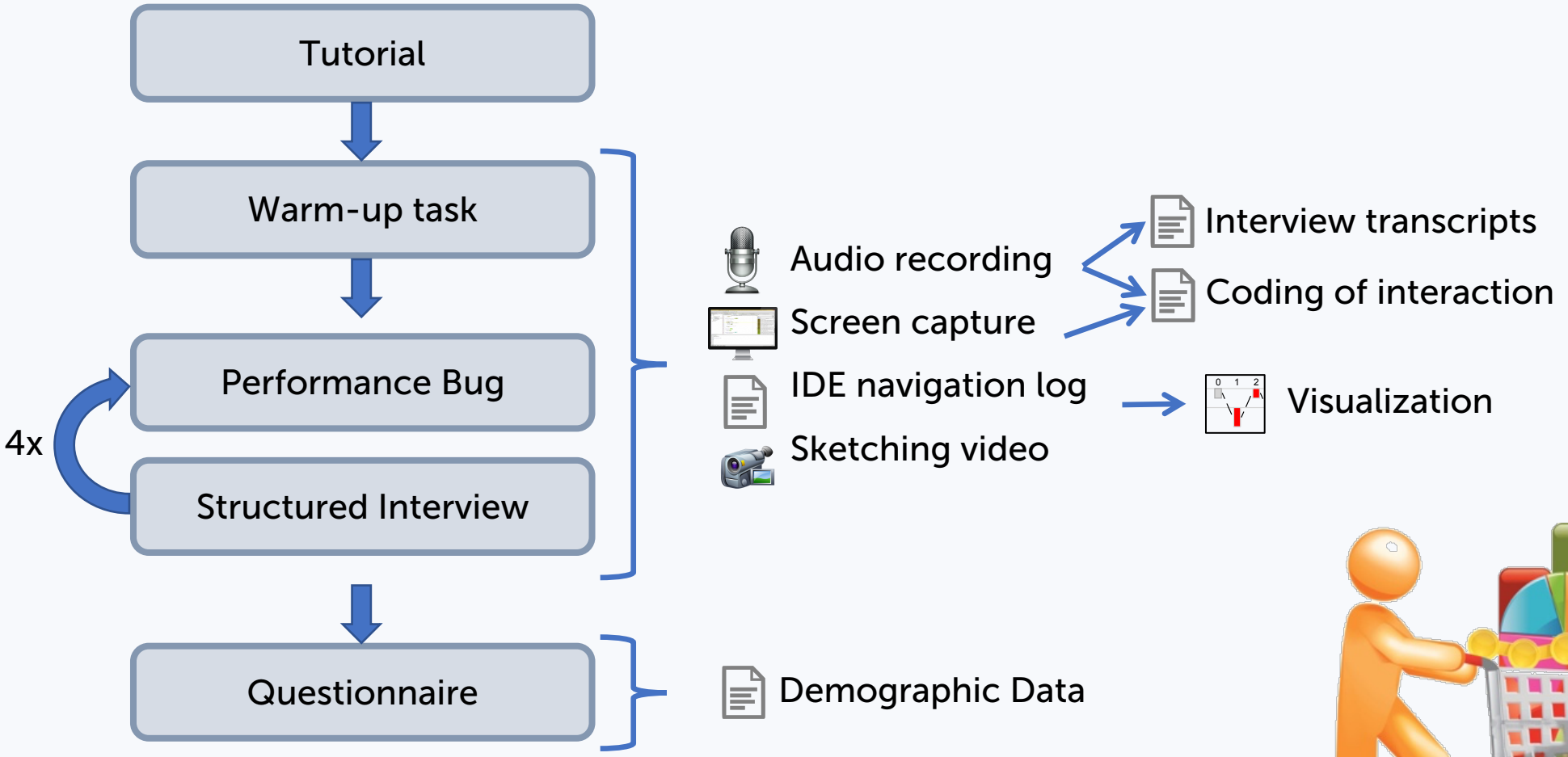


Navigation: Profiling Tool



Data Collection

Course of a study session:



Methods (Selection)

RQ1:

How do developers **navigate** the source code and what **information and representation** is supportive for **locating** a performance bug?

Interview transcripts (bug 1-4)

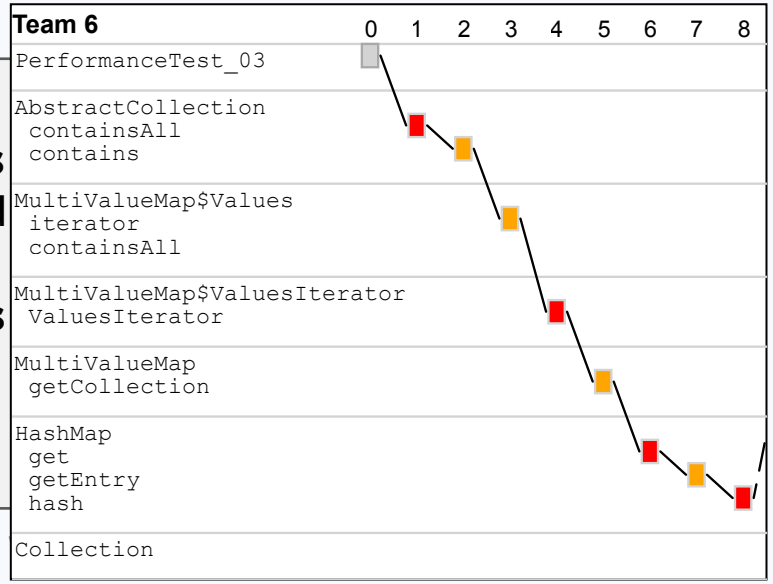
→ Cross-case analysis [Seaman99]

Navigation visualization (bug 3)

→ Pattern search

TABLE II. PROPOSITIONS BASED ON CROSS-CASE ANALYSIS OF INTERVIEW ANSWERS RELATED TO RQ1.1 (TOP) AND RQ1.2 (BOTTOM).

No.	Proposition	Teams
1.1	The dynamic instance of a method call and connected runtime information are important for navigation.	T1, T3, T4, T5
1.2	Following high quantities of runtime in the dynamic method call graph is helpful as a navigation strategy.	T1, T2, T3, T6
1.3	The more complex the performance bug is, the less helpful the provided tool support and information becomes.	T1, T3, T5, T6
2.1	The integration into the code view provides additional context for the profiling visualization.	T1, T2, T4, T6
2.2	The overview (list view) was not needed in this setting.	T1, T4, T5
2.3	The overview (list view) could be used as a starting point for further analyses.	T1, T2, T4



Results (Selection)

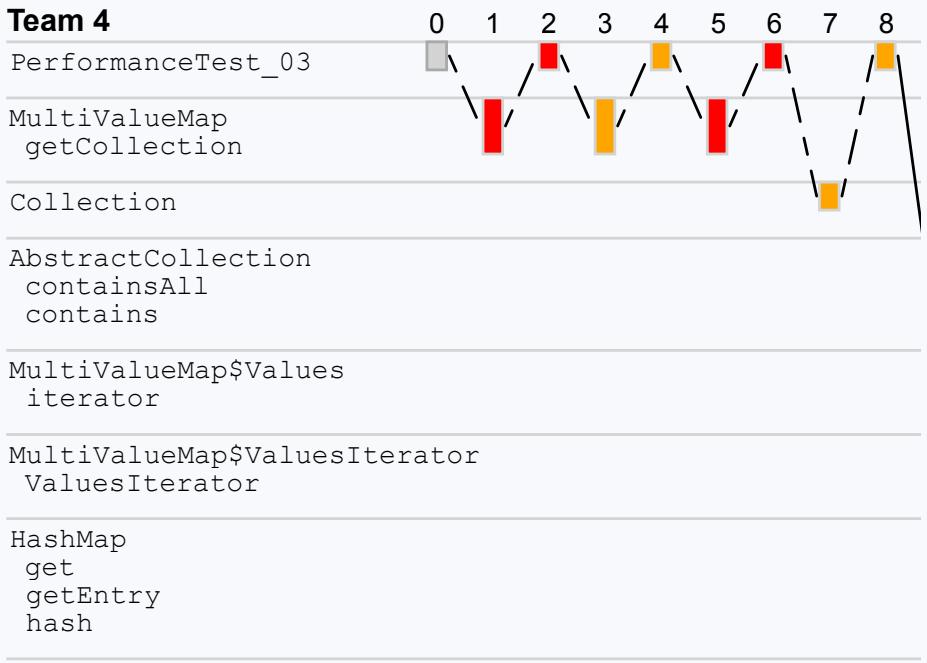
RQ1.3: What **navigation strategies** do developers pursue to locate a performance bug?

- About 70% of navigation through IDE, 30% with our tool
- Navigation with method call visualization dominant (in-situ)
- List view never used for bug 3
- Identified two navigation strategies:

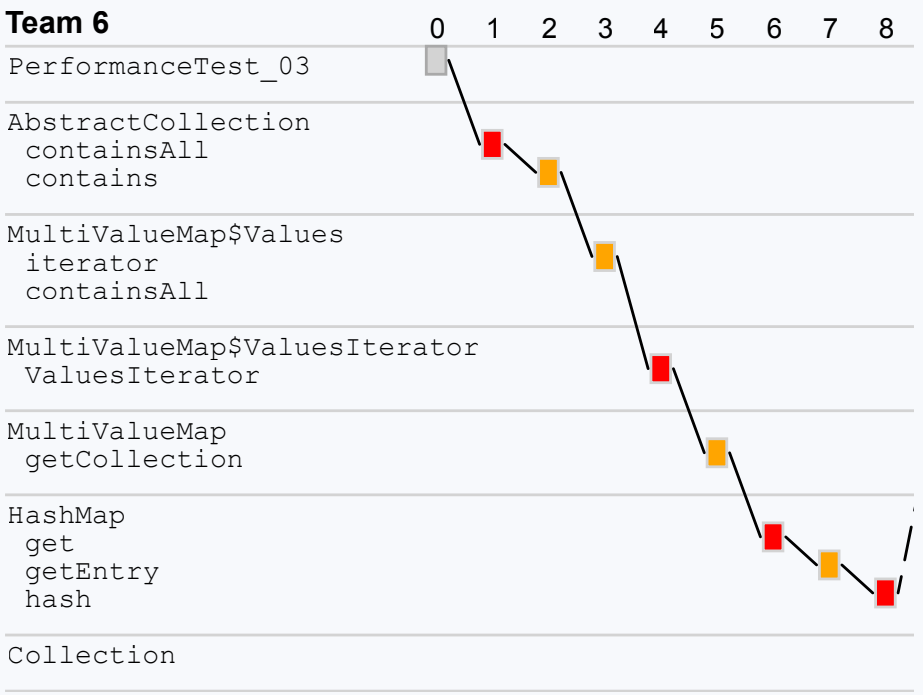
Strategy 1 (Toggle): Frequent switching between test class and and other classes related to bug (IDE navigation).

Strategy 2 (Path Following): Follow dynamic method calls with high runtime consumption (In-situ visualization).

Results (Selection)



Strategy 1 (Toggle)



Strategy 2 (Path Following)

Methods (Selection)

RQ2:

How do developers try to **understand** and **explain** the causes of performance bugs?

- Interview transcripts (bug 1-4)
  - Cross-case analysis
- Coding of interaction (bug 3)
  - Descriptive statistics

TABLE VI. PROPOSITION
INTERVIEW ANS

No.	Proposition
3.1	Sketches are a useful tool mance bug, but context in understand them afterwards.
3.2	Sketches are a suitable d "polished" enough).
3.3	If and how much sketching sketching experience of the
3.4	A common sketch vocabular
3.5	More complex problems or likely to be sketched.
3.6	Sketches can be used to exp a program.

TABLE IV. INTERACTIONS WHILE LOCATING PERFORMANCE BUG 3
(D: DURING, A: AFTER LOCATING BUG, *: NAVIGATOR TOOK OVER ROLE OF DRIVER, CODES: SEE TABLE V)

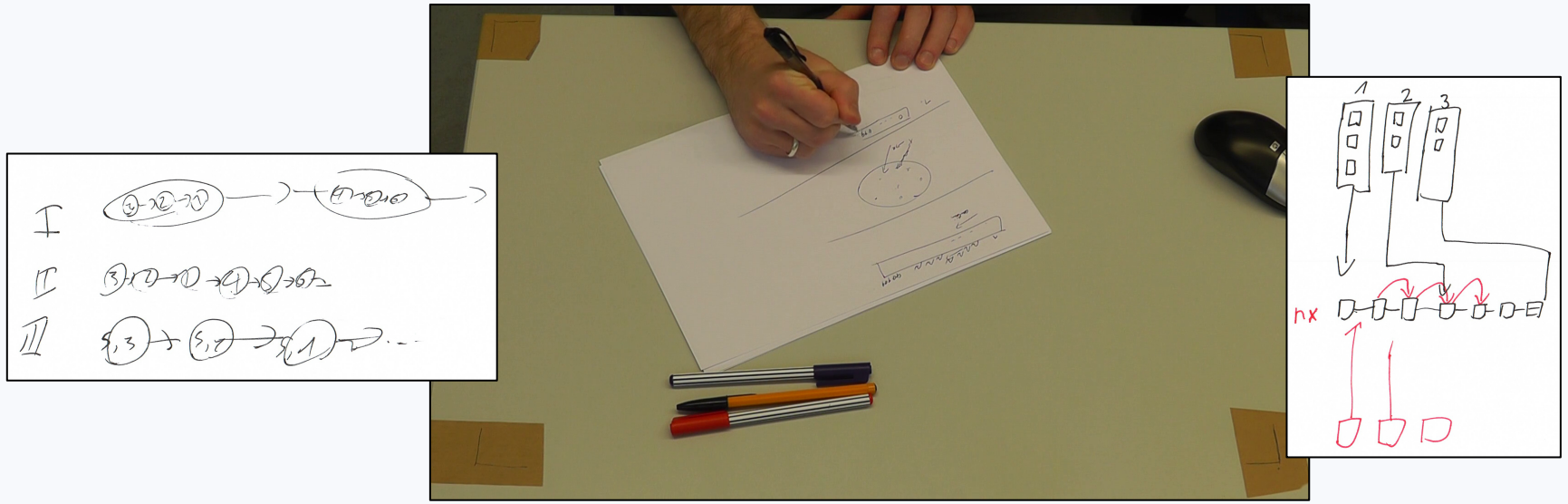
Team	Time (min.)	Success	Driver	Navigator	Codes								First Strategy	Sketch
					Total	DC+HC	DR+HR	QC+QR	PN+PI	CO	RD+RC+RE	Other		
T1	30	✓	P2	P1	165 45% 55%	46 57% 43%	11 55% 45%	28 21% 79%	5 0% 100%	10 20% 80%	11 55% 45%	54 54% 46%	1	D
T2	30	✓	P4	P3	112 57% 43%	21 67% 33%	19 58% 42%	24 54% 46%	9 11% 89%	6 33% 67%	9 56% 44%	24 75% 25%	1	A
T3	24	✓	P5	P6	78 63% 37%	18 83% 17%	13 85% 15%	10 90% 10%	6 0% 100%	7 0% 100%	3 100% 0%	21 52% 48%	2	A
T4	35	✓	P7	P8	136 46% 54%	24 58% 42%	22 68% 32%	20 20% 80%	15 0% 100%	7 29% 71%	10 20% 80%	38 68% 32%	1	D
T5	20	○	P10*	P9*	48 35% 65%	14 21% 79%	9 44% 56%	10 30% 70%	2 0% 100%	0 0% 0%	2 100% 0%	11 45% 55%	-	D
T6	24	×	P12	P11	40 63% 38%	15 73% 27%	13 77% 23%	1 0% 100%	2 0% 100%	3 0% 100%	0 0% 0%	6 67% 33%	2	D

Methods (Selection)

RQ2:

How do developers try to **understand** and **explain** the causes of performance bugs?

📹 Sketching video (bug 3)



Results (Selection)

RQ2.1: How do developers communicate with each other when locating a performance bug?

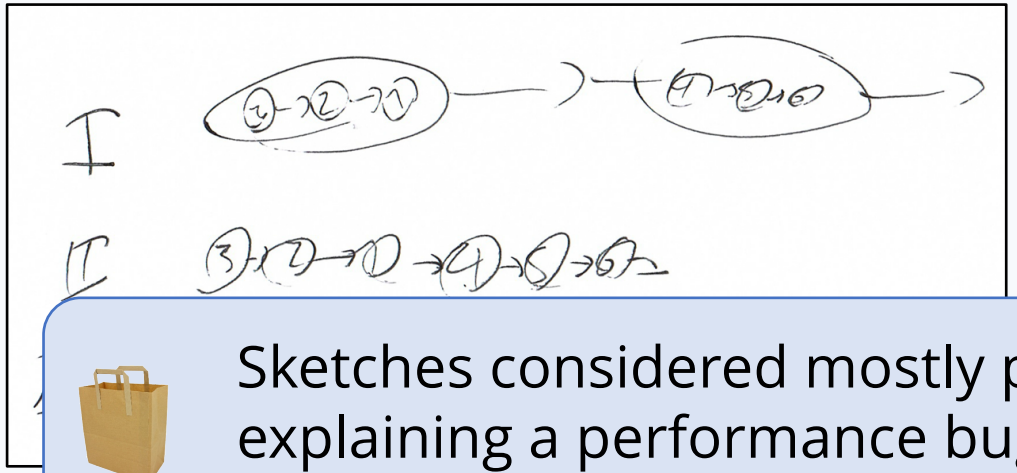
- 4 of 6 teams expressed first hypothesis about cause of bug in the first half of session
- Driver and navigator mostly worked on **same level of abstraction**
- 3 teams had very **active navigator** (e.g. asking questions about code, prompting driver to navigate to certain methods)
- 2 teams had very **passive navigator** (mostly observed)
- Different levels of **expertise** can be reason for active/passive role

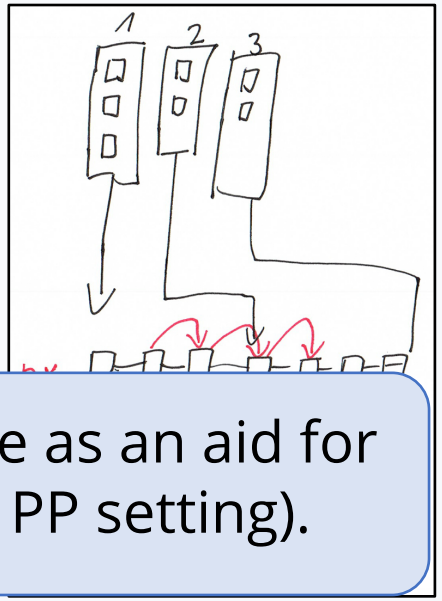
Driver and navigator work on **same level of abstraction**; interaction could be affected by different levels of expertise.

Results (Selection)

RQ2.2: Could sketches help to understand and communicate a performance bug?

- Four teams spontaneously created a sketch while locating bug 3
- All sketches created by **navigator**
- Sketching **static structure** (e.g. `MultiValueMap`)
- Sketching **dynamic aspects** (execution of method `contains(...)`)
- Keeping track of **alternative hypotheses**





Sketches considered mostly positive as an aid for explaining a performance bug (in a PP setting).

Expertise Development

Research Questions

5.1.1 Characteristics of experts:

RQ1.1: What characteristics do developers assign to experts?
RQ1.2: What character traits or behaviors are supportive for becoming a software development expert?
RQ1.3: What tasks should a software development expert be good at?

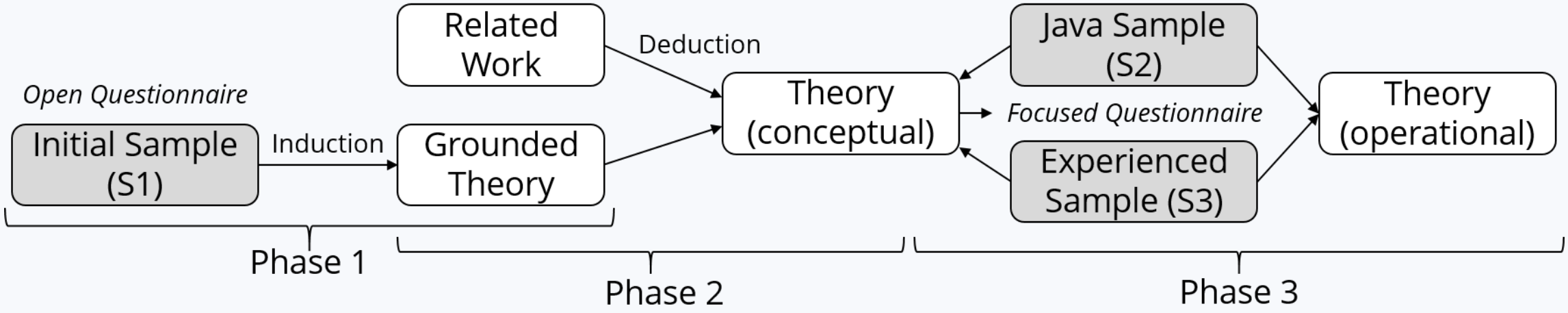
5.1.2 Motivation and task context:

RQ2.1: What challenges do developers face in their daily work?
RQ2.2: What motivates developers?
RQ2.3: Whom do developers consider a "mentor" in becoming a better software developer?
RQ2.4: What are typical reasons for a decline of programming performance over time?
RQ2.5: What can employers do to facilitate a continuous development of their employees' software development skills?

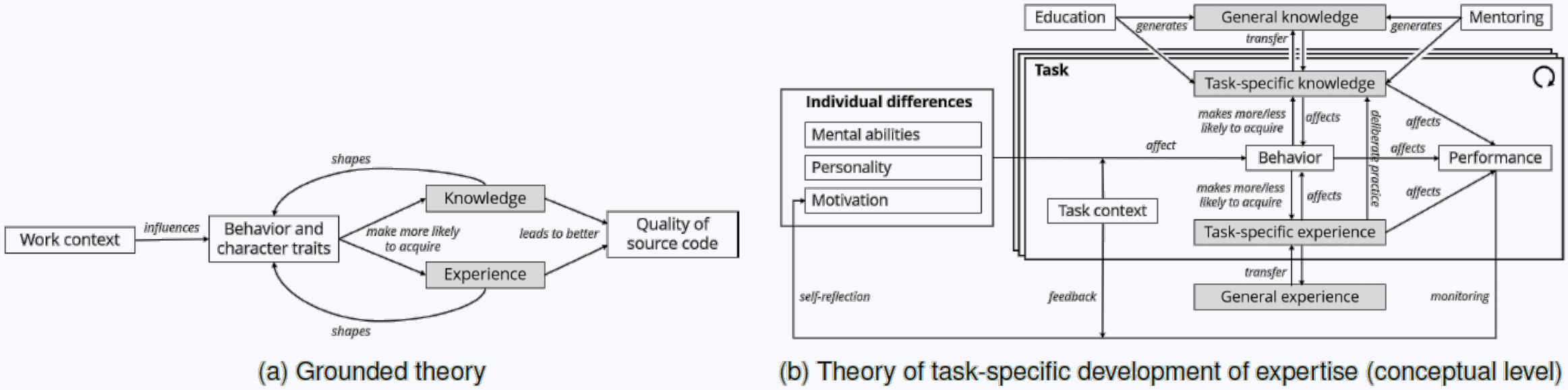
5.1.3 Monitoring and self-reflection:

RQ3.1: What criteria do developers consider when estimating their own programming expertise?
RQ3.2: How do developers monitor their software development activities?
RQ3.3: How does the context of self-assessment questions influence developers' ratings?

Research


Related Work
Deduction
Java Sample (S2)
Theory (operational)
Open Questionnaire
Initial Sample (S1)
Induction
Grounded Theory
Theory (conceptual)
Focused Questionnaire
Experienced Sample (S3)
Phase 1
Phase 2
Phase 3

GT and Conceptual Theory



(a) Grounded theory

(b) Theory of task-specific development of expertise (conceptual level)

Fig. 2. Theories we developed in Phase 1 (left) and Phase 2 (right)

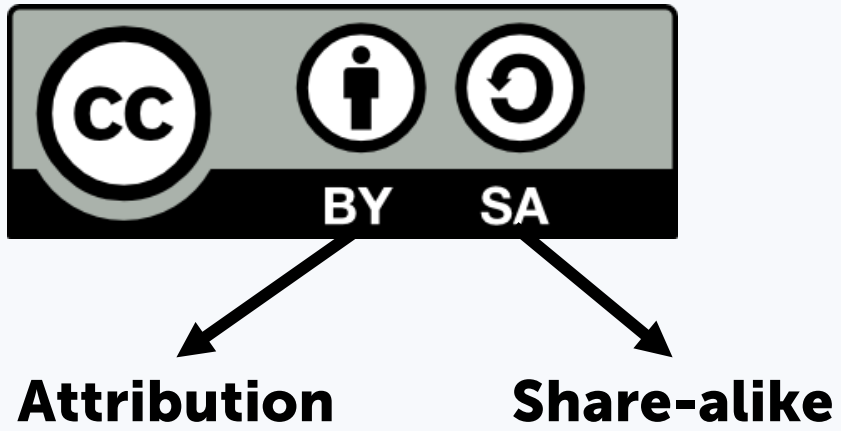
Code Plagiarism

Motivation

Who of you admits copying non-trivial code snippets from Stack Overflow without attribution?

Motivation

Who of you knew that all content on Stack Overflow is licensed under CC BY-SA 3.0?



Current Research

Usage and Attribution of

stackoverflow Code Snippets

in GitHub Projects

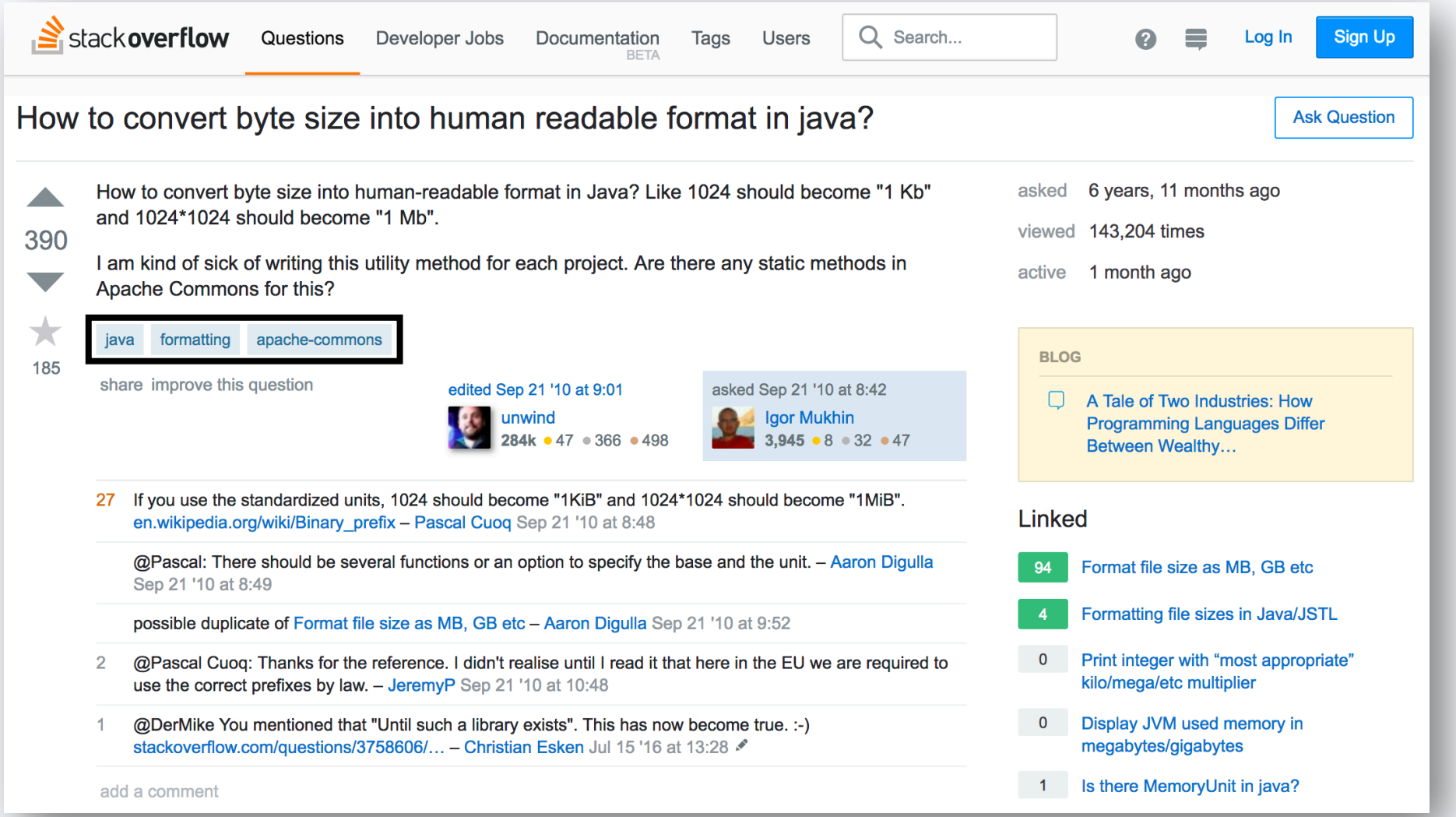
Open Question

How many **outdated** and possibly **buggy** clones of Stack Overflow snippets exist in GitHub projects?

Stack Overflow

Question

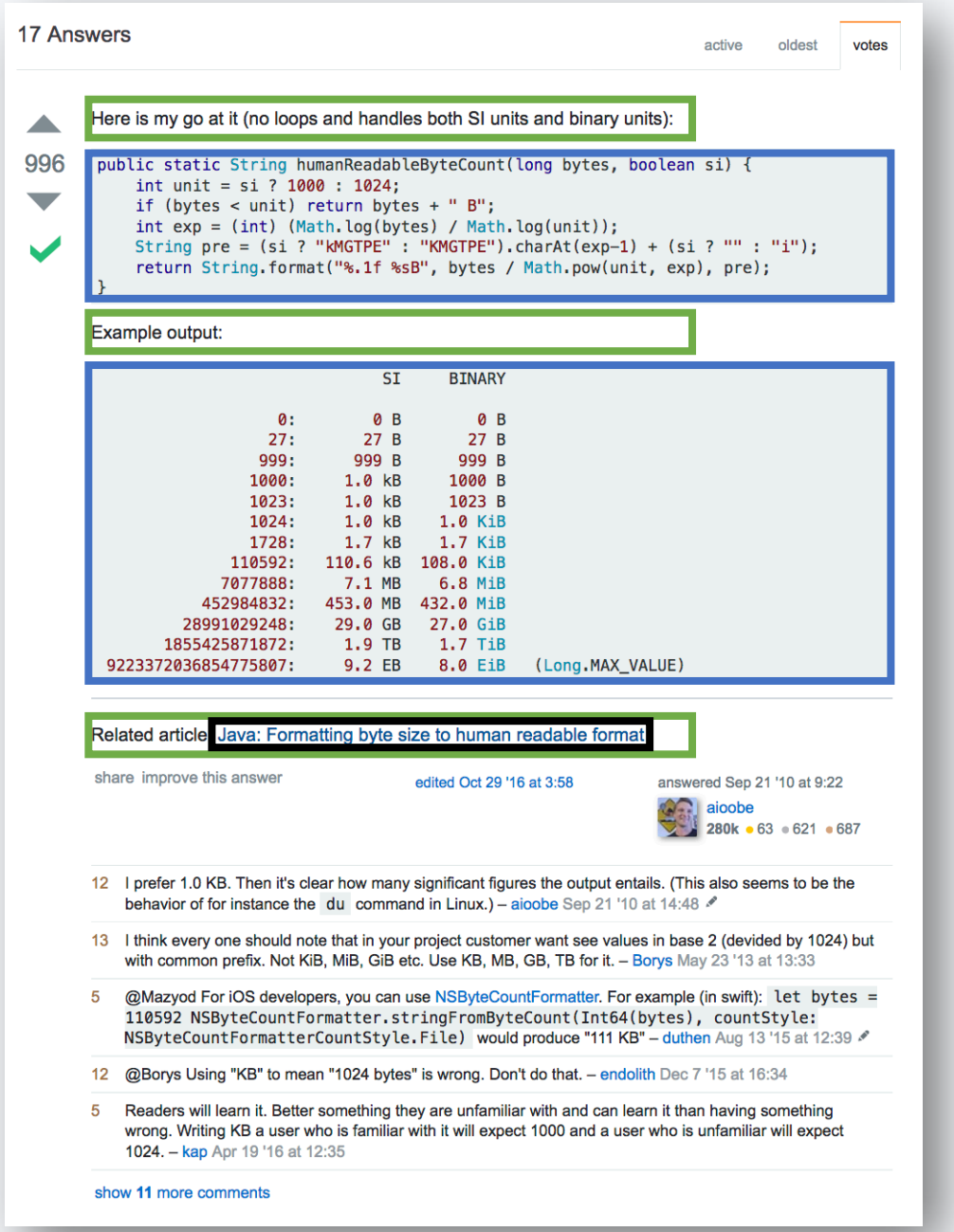
Tags



Answer(s)

All SO content available as XML dump, content in Markdown format.

Link



17 Answers — active | oldest | votes

996

Here is my go at it (no loops and handles both SI units and binary units):

```
public static String humanReadableByteCount(long bytes, boolean si) {
    int unit = si ? 1000 : 1024;
    if (bytes < unit) return bytes + " B";
    int exp = (int) (Math.log(bytes) / Math.log(unit));
    String pre = (si ? "kMGTPE" : "KMGTPE").charAt(exp-1) + (si ? "" : "i");
    return String.format("%.1f %sB", bytes / Math.pow(unit, exp), pre);
}
```

Example output:

```
                              SI     BINARY

                   0:        0 B        0 B
                  27:       27 B       27 B
                 999:      999 B      999 B
                1000:     1.0 kB     1000 B
                1023:     1.0 kB     1023 B
                1024:     1.0 kB    1.0 KiB
                1728:     1.7 kB    1.7 KiB
              110592:   110.6 kB  108.0 KiB
             7077888:     7.1 MB    6.8 MiB
           452984832:   453.0 MB  432.0 MiB
         28991029248:    29.0 GB   27.0 GiB
       1855425871872:     1.9 TB    1.7 TiB
 9223372036854775807:     9.2 EB    8.0 EiB   (Long.MAX_VALUE)
```

Related article: Java: Formatting byte size to human readable format

share improve this answer — edited Oct 29 '16 at 3:58 — answered Sep 21 '10 at 9:22 aioobe 280k ● 63 ● 621 ● 687

12 I prefer 1.0 KB. Then it's clear how many significant figures the output entails. (This also seems to be the behavior of for instance the `du` command in Linux.) – aioobe Sep 21 '10 at 14:48

13 I think every one should note that in your project customer want see values in base 2 (devided by 1024) but with common prefix. Not KiB, MiB, GiB etc. Use KB, MB, GB, TB for it. – Borys May 23 '13 at 13:33

5 @Mazyod For iOS developers, you can use NSByteCountFormatter. For example (in swift): `let bytes = 110592 NSByteCountFormatter.stringFromByteCount(Int64(bytes), countStyle: NSByteCountFormatterCountStyle.File)` would produce "111 KB" – duthen Aug 13 '15 at 12:39

12 @Borys Using "KB" to mean "1024 bytes" is wrong. Don't do that. – endolith Dec 7 '15 at 16:34

5 Readers will learn it. Better something they are unfamiliar with and can learn it than having something wrong. Writing KB a user who is familiar with it will expect 1000 and a user who is unfamiliar will expect 1024. – kap Apr 19 '16 at 12:35

show 11 more comments

Text

Code

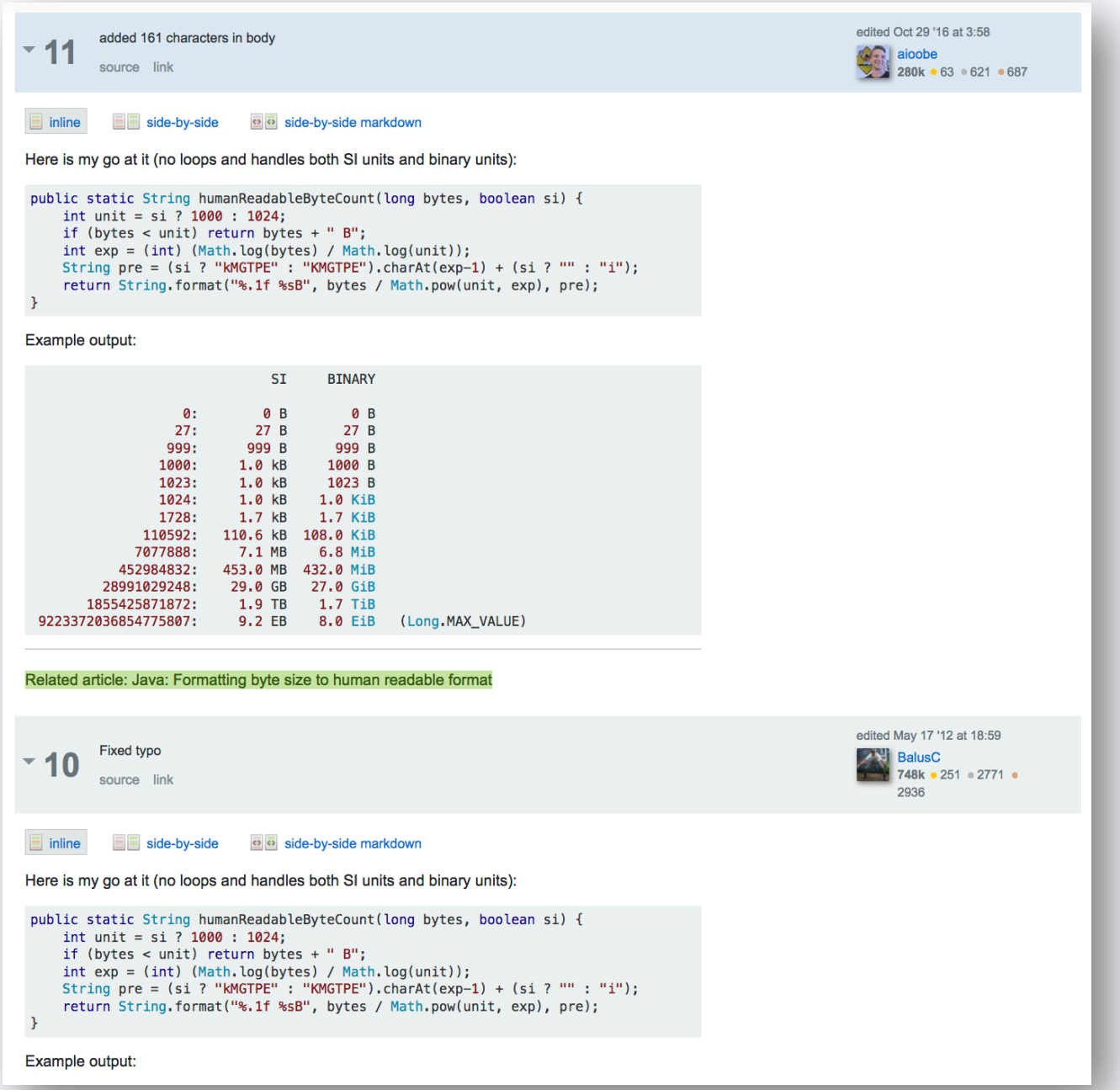
Text

Code

Text

Revisions

Main Issue:
No separation between text and code.



11 added 161 characters in body
source link

edited Oct 29 '16 at 3:58
aioobe
280k • 63 • 621 • 687

inline side-by-side side-by-side markdown

Here is my go at it (no loops and handles both SI units and binary units):

```
public static String humanReadableByteCount(long bytes, boolean si) {
    int unit = si ? 1000 : 1024;
    if (bytes < unit) return bytes + " B";
    int exp = (int) (Math.log(bytes) / Math.log(unit));
    String pre = (si ? "kMGTPE" : "KMGTPE").charAt(exp-1) + (si ? "" : "i");
    return String.format("%.1f %sB", bytes / Math.pow(unit, exp), pre);
}
```

Example output:

```
                         SI     BINARY

                  0:        0 B        0 B
                 27:       27 B       27 B
                999:      999 B      999 B
               1000:     1.0 kB     1000 B
               1023:     1.0 kB     1023 B
               1024:     1.0 kB    1.0 KiB
               1728:     1.7 kB    1.7 KiB
             110592:   110.6 kB  108.0 KiB
            7077888:     7.1 MB    6.8 MiB
          452984832:   453.0 MB  432.0 MiB
        28991029248:    29.0 GB   27.0 GiB
      1855425871872:     1.9 TB    1.7 TiB
 9223372036854775807:     9.2 EB    8.0 EiB   (Long.MAX_VALUE)
```

Related article: Java: Formatting byte size to human readable format

10 Fixed typo
source link

edited May 17 '12 at 18:59
BalusC
748k • 251 • 2771 • 2936

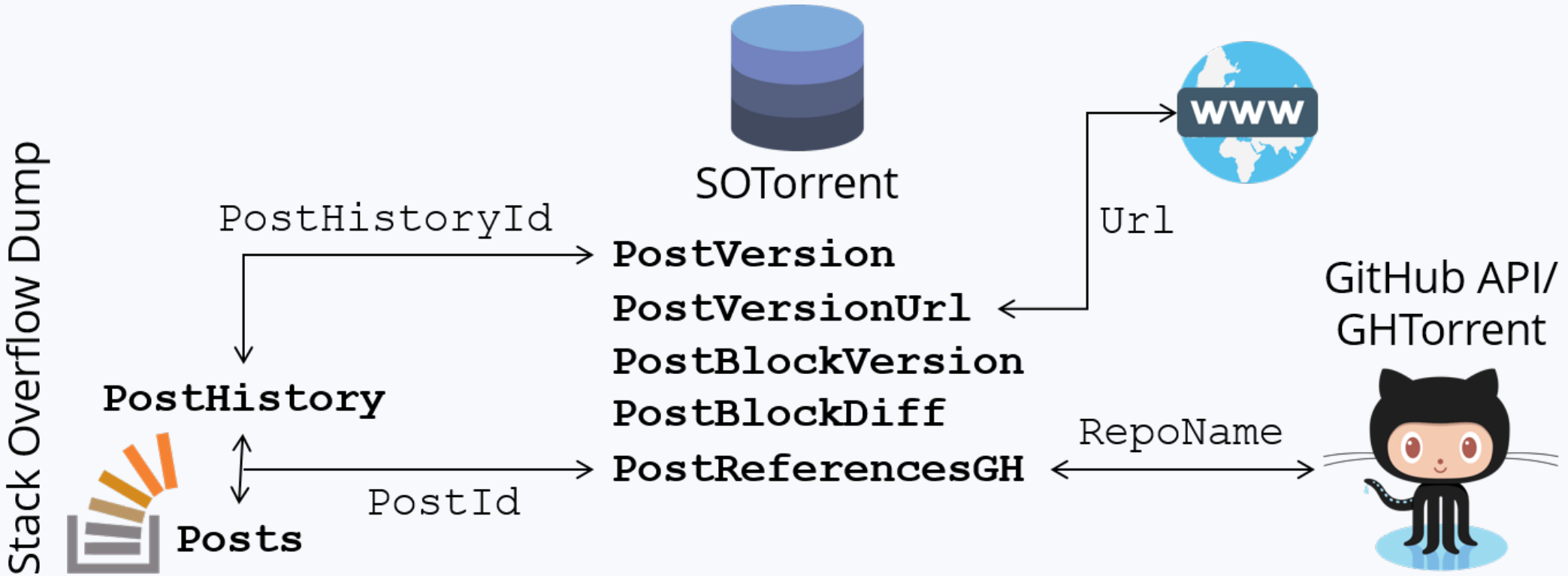
inline side-by-side side-by-side markdown

Here is my go at it (no loops and handles both SI units and binary units):

```
public static String humanReadableByteCount(long bytes, boolean si) {
    int unit = si ? 1000 : 1024;
    if (bytes < unit) return bytes + " B";
    int exp = (int) (Math.log(bytes) / Math.log(unit));
    String pre = (si ? "kMGTPE" : "KMGTPE").charAt(exp-1) + (si ? "" : "i");
    return String.format("%.1f %sB", bytes / Math.pow(unit, exp), pre);
}
```

Example output:

SOTorrent


Stack Overflow Dump
PostHistoryId
PostHistory
Posts
PostId
SOTorrent
PostVersion
PostVersionUrl
PostBlockVersion
PostBlockDiff
PostReferencesGH
Url
www
GitHub API/
GHTorrent
RepoName

SOTorrent

PostBlockVersion

Id	PostId	PostHistoryId	PostBlockTypeId	LocalId	Content	Length	LineCount	PredPostBlockId	PredEqual	PredSimilarity	PredCount	SuccCount
2472490	3758880	7875126	2	4	SI BINARY ...	690	14	2472484	1	1	1	1
2472492	3758880	15577610	1	1	Here is my go at it (no loops and han...	74	1	2472485	1	1	1	1
2472494	3758880	15577610	2	2	public static String humanReadable...	380	7	2472487	1	1	1	1
2472497	3758880	15577610	1	3	Example output:	15	1	2472488	1	1	1	1
2472499	3758880	15577610	2	4	SI BINARY ...	690	14	2472490	0	0.98901098...	1	1
2472501	3758880	24534909	1	1	Here is my go at it (no loops and han...	74	1	2472492	1	1	1	1
2472502	3758880	24534909	2	2	public static String humanReadable...	380	7	2472494	1	1	1	1
2472504	3758880	24534909	1	3	Example output:	15	1	2472497	1	1	1	1
2472506	3758880	24534909	2	4	SI BINARY ...	690	14	2472499	0	1	1	1
2472507	3758880	130380462	1	1	Here is my go at it (no loops and han...	74	1	2472501	1	1	1	0
2472508	3758880	130380462	2	2	public static String humanReadable...	380	7	2472502	1	1	1	0
2472509	3758880	130380462	1	3	Example output:	15	1	2472504	1	1	1	0
2472510	3758880	130380462	2	4	SI BINARY ...	690	14	2472506	1	1	1	0
2472511	3758880	130380462	1	5	---------- Related article: [Java: Forma...	163	2	NULL	NULL	NULL	0	0

...

Version 24534909

Version 130380462

Text block

Code block

Blocks not changed

Block added

Block changed, but similarity metric was 1.0

PostVersion

First version

PostId	PostHistoryId	PostTypeId	PredPostHistoryId	SuccPostHistoryId
3758880	7873162	2	NULL	7873319
3758880	7873319	2	7873162	7873489
3758880	7873489	2	7873319	7873581
3758880	7873581	2	7873489	7874248
3758880	7874248	2	7873581	7874570
3758880	7874570	2	7874248	7874704
3758880	7874704	2	7874570	7875126
3758880	7875126	2	7874704	15577610
3758880	15577610	2	7875126	24534909
3758880	24534909	2	15577610	130380462
3758880	130380462	2	24534909	NULL

Most recent version

PostBlockDiff

Id	PostId	PostHistoryId	PredPostBlockVersionId	PostBlockVersionId	PostBlockDiffOperationId	Text
1771391	3758880	24534909	2472499	2472506	0	SI BINARY ...
1771395	3758880	24534909	2472499	2472506	1	7077888: 7.1 kB 6.8 MiB
1771393	3758880	24534909	2472499	2472506	-1	7077888: 7.1 MB 6.8 MiB
1771397	3758880	24534909	2472499	2472506	0	452984832: 453.0 MB 432.0 MiB ...

Equal

Deleted

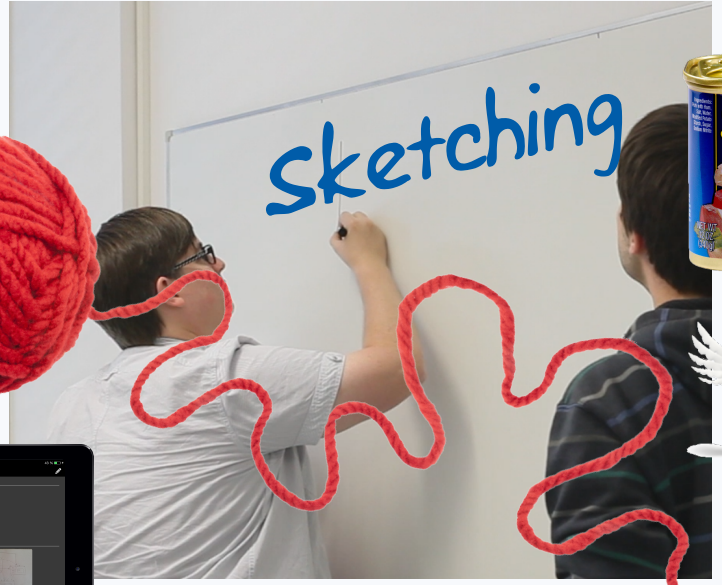
Added

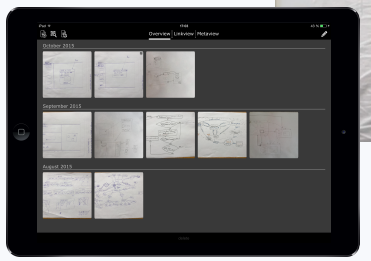
Change

Research Questions

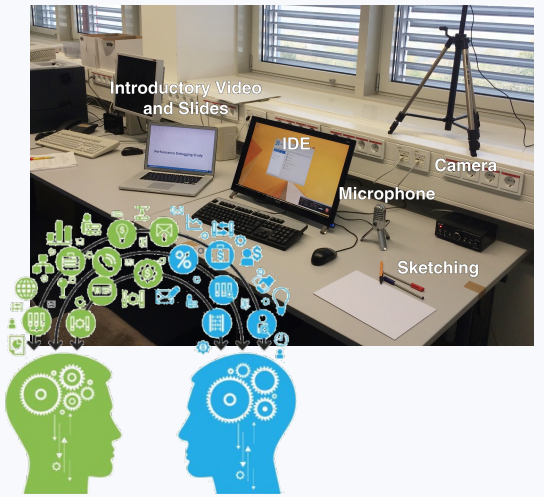
- How does code posted on SO evolve?
- How are issues with the code reported?
- How soon are issues fixed?
- How easily can possible issues in older version of the snippets be identified?
- How can the SO UI be improved to included (hints about) evolution?

→ **SOTorrent** data set helps to answer those questions





SketchLink



stackoverflow

Questions?

sbaltes.com

@s_baltes

research@sbaltes.com