Alberto Bacchelli
Delft University of Technology
The Netherlands

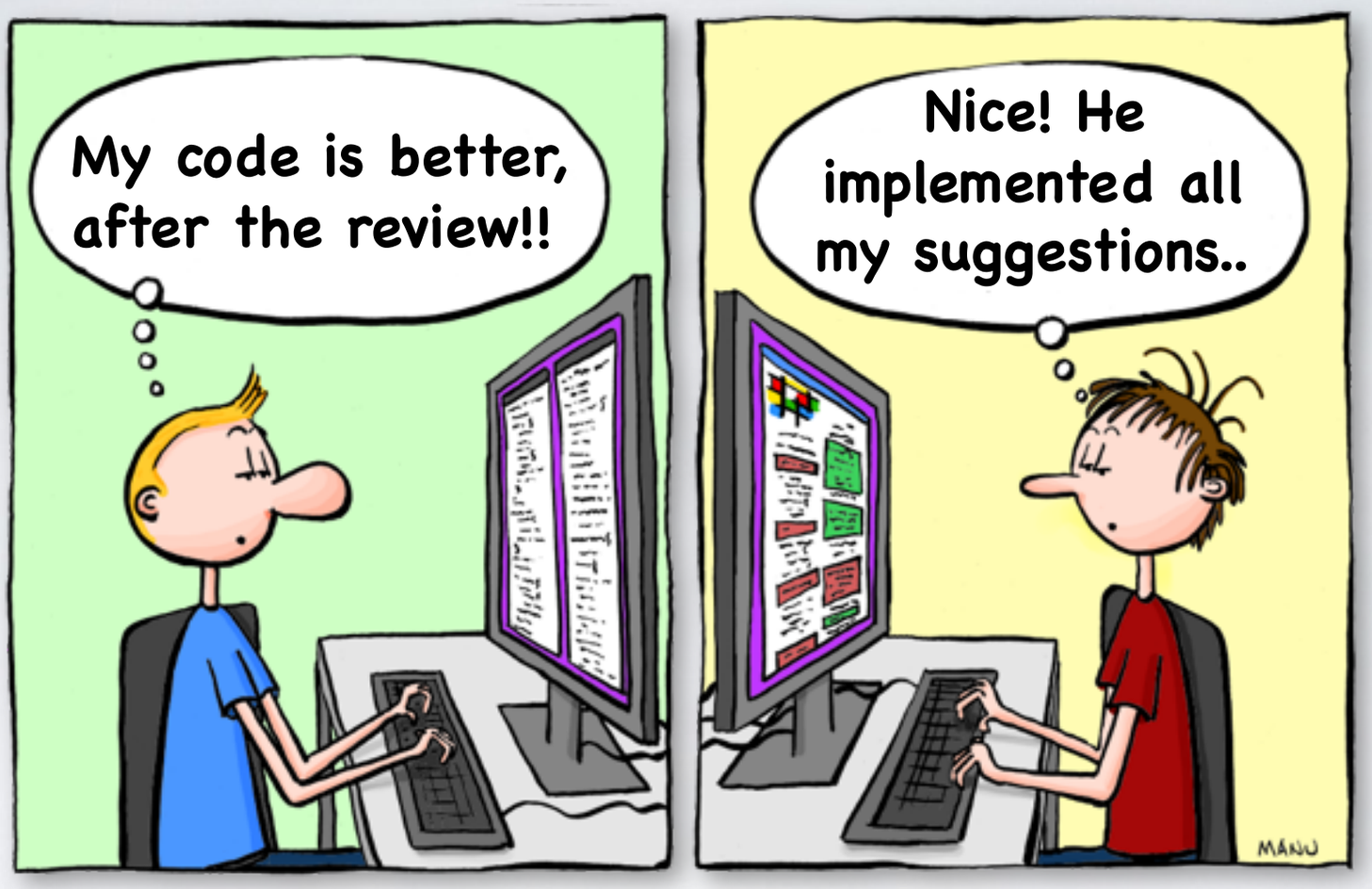
What Do Code Reviews at Microsoft and in Open Source Projects Have in Common?

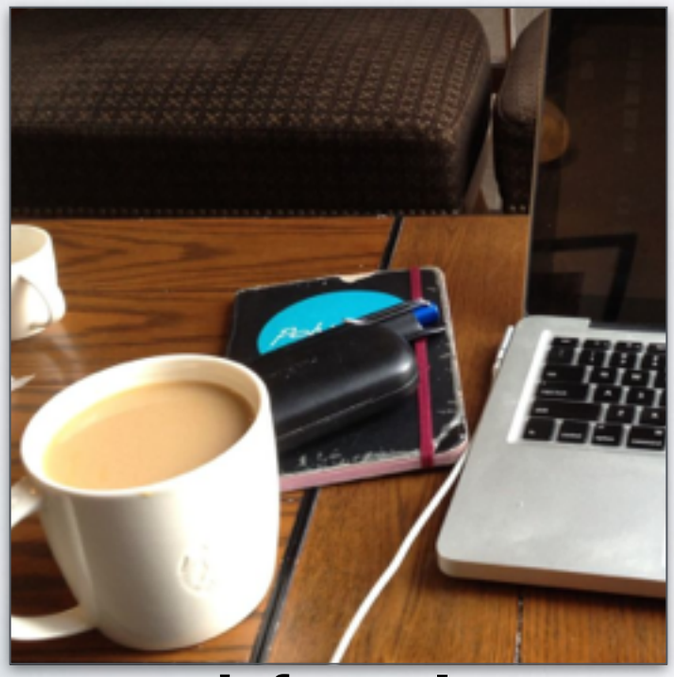

ULB
UNIVERSITÉ LIBRE DE BRUXELLES
ULB
FOSCB


CINECA

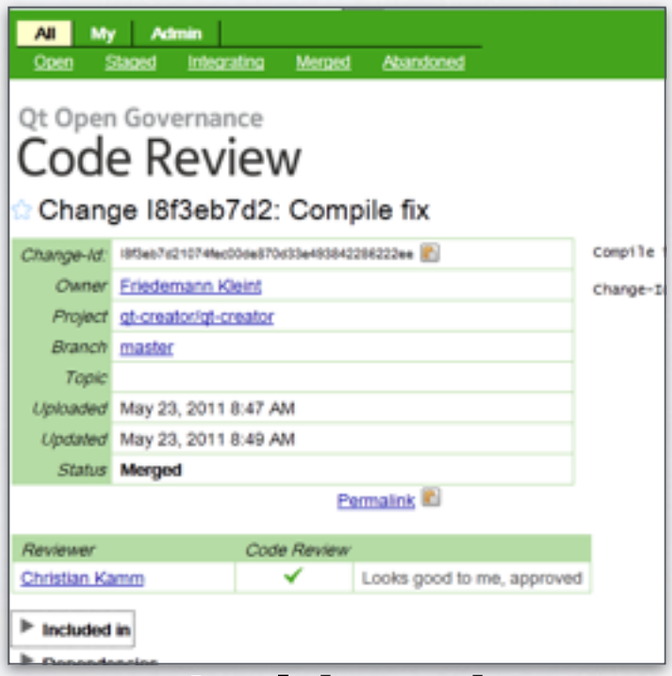

Microsoft
Research

Modern code review





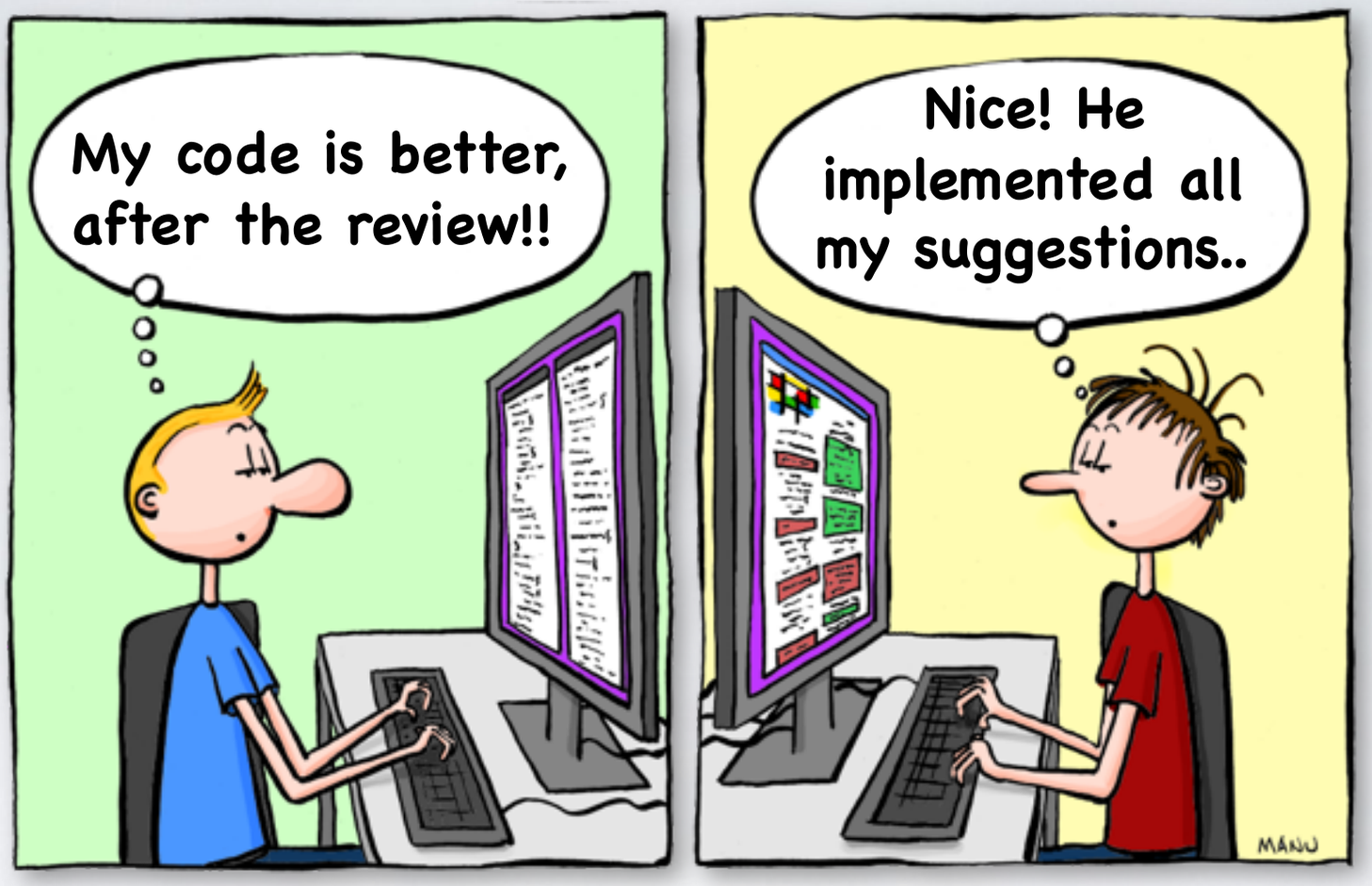
informal

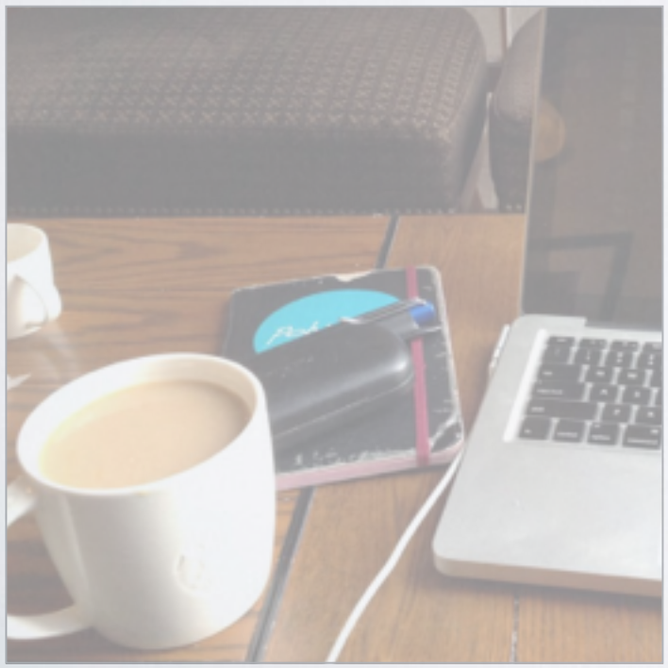


tool-based

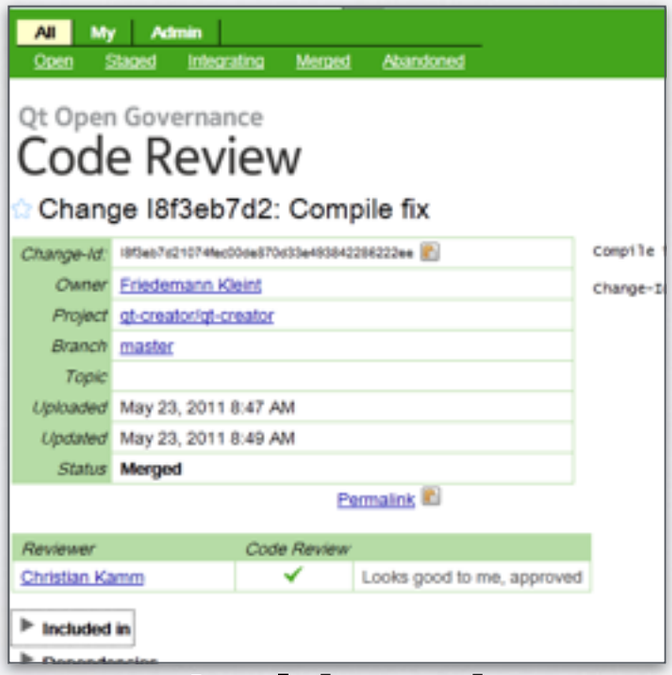
asynchronous

Modern code review

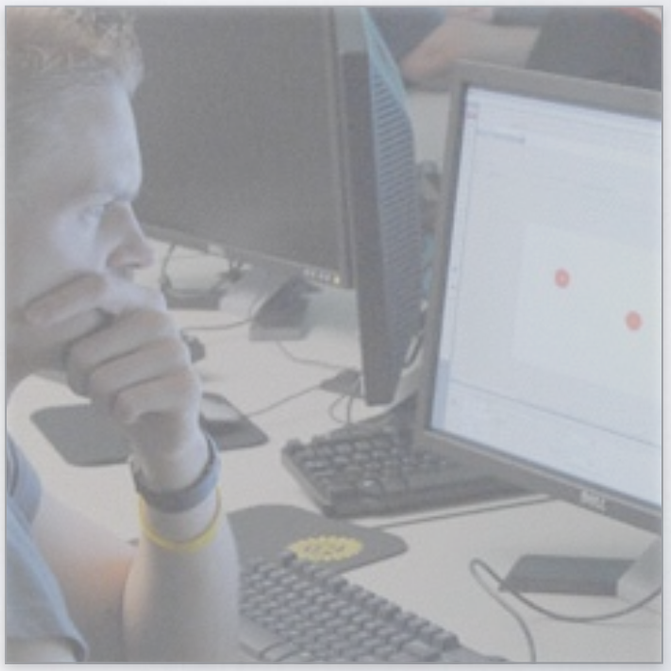




informal



tool-based



asynchronous

Code review tools: Gerrit

All Projects Documentation

Open Merged Abandoned

Change #, SHA-1, tr:id or owner:email Search Sign In

android open source project

Browse Repositories

Change-Id:	I26cd0da83acc9df2162b6ec6174fc25712f68de2
Owner	jp abgrall
Project	kernel/common
Branch	android-3.4
Topic	
Uploaded	Feb 25, 2013 10:05 PM
Updated	Feb 25, 2013 10:12 PM
Status	Merged

Commit Message *Permalink*

```
sock_diag: Fix out-of-bounds access to sock_diag_handlers[]

Userland can send a netlink message requesting SOCK_DIAG_BY_FAMILY
with a family greater or equal then AF_MAX -- the array size of
sock_diag_handlers[]. The current code does not test for this
condition therefore is vulnerable to an out-of-bound access opening
doors for a privilege escalation.

Signed-off-by: Mathias Krause <minipli <at> googlemail.com>
```

Reviewer	*Code-Review*	*Verified*
jp abgrall	✓	✓
Nick Kralevich	+1	

▶ Included in

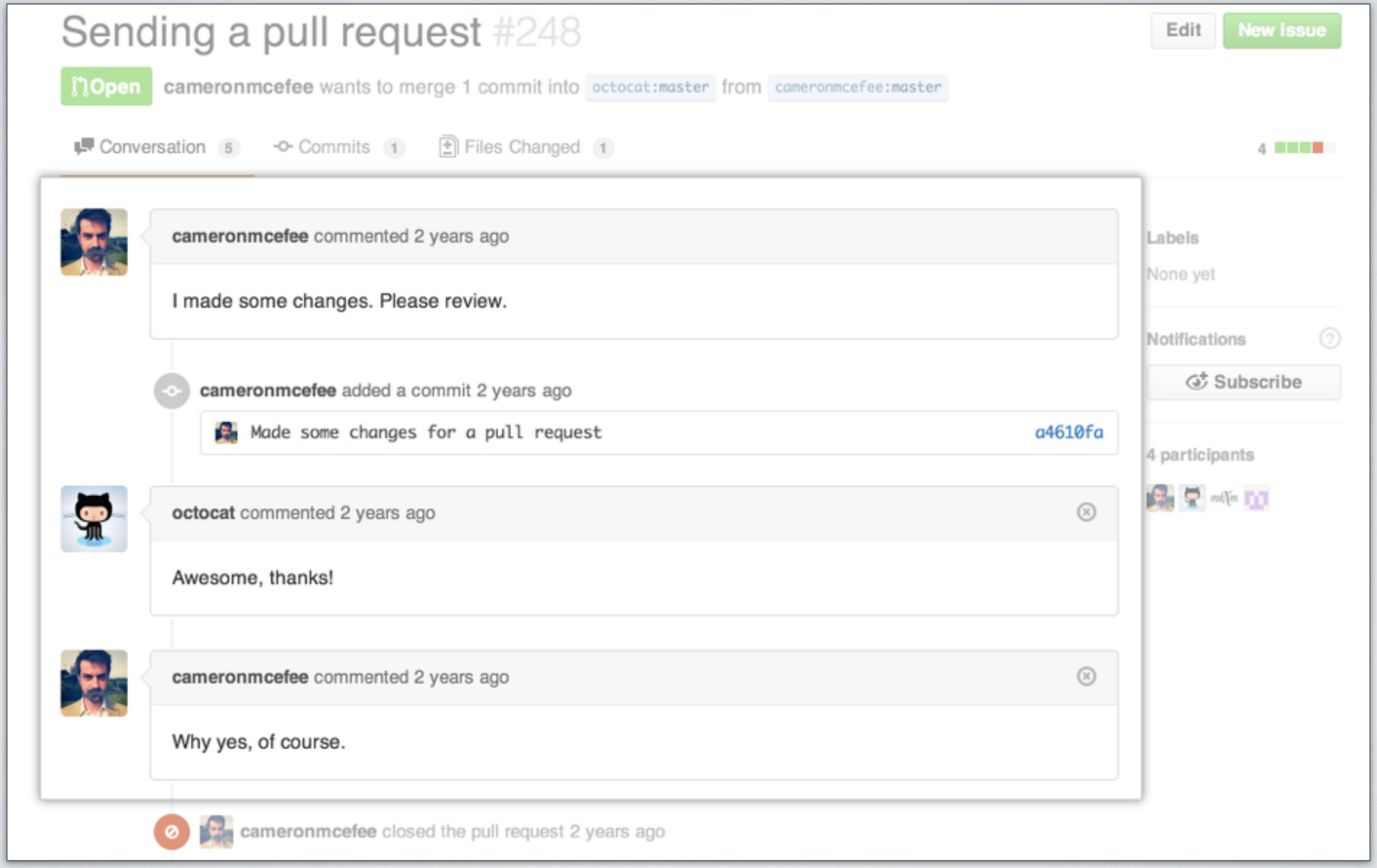
▶ Dependencies

Old Version History: Base

▼ Patch Set 1 26cd0da83acc9df2162b6ec6174fc25712f68de2 (gitiles)

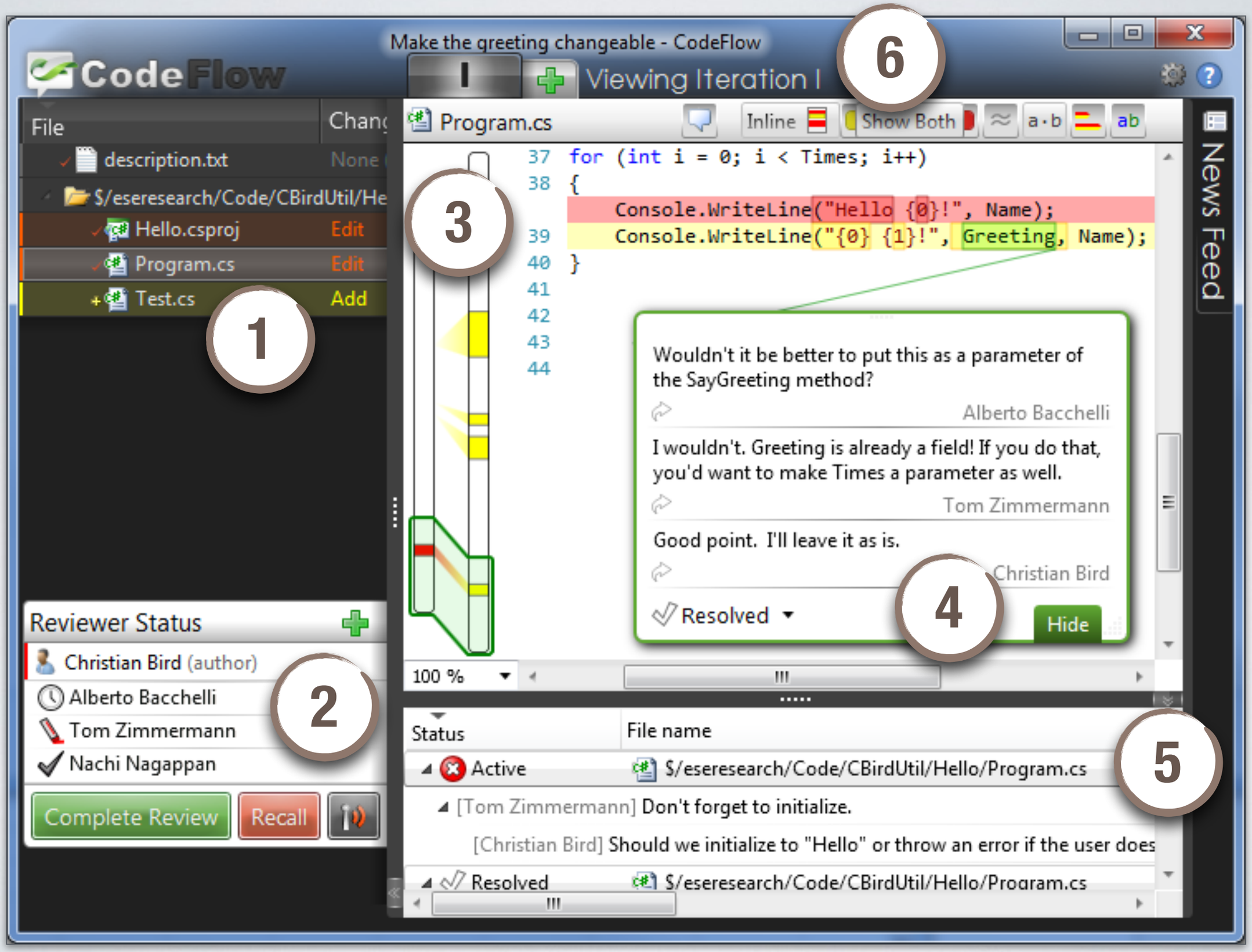
Author	Mathias Krause <minipli at> Feb 23, 2013 12:13 PM
Committer	JP Abgrall <jpa@google.com> Feb 25, 2013 9:57 PM
Parent(s)	1a6c74e0d58f6ca028f36c3d794fecff8543bbfb gpu: ion: Add support for sharing buffers with dma buf kernel handles
Download	checkout | checkout | pull | cherry-pick | patch |

Code review tools: GitHub pull requests



Code review tools: Atlassian Crucible

Code review tools: Microsoft CodeFlow

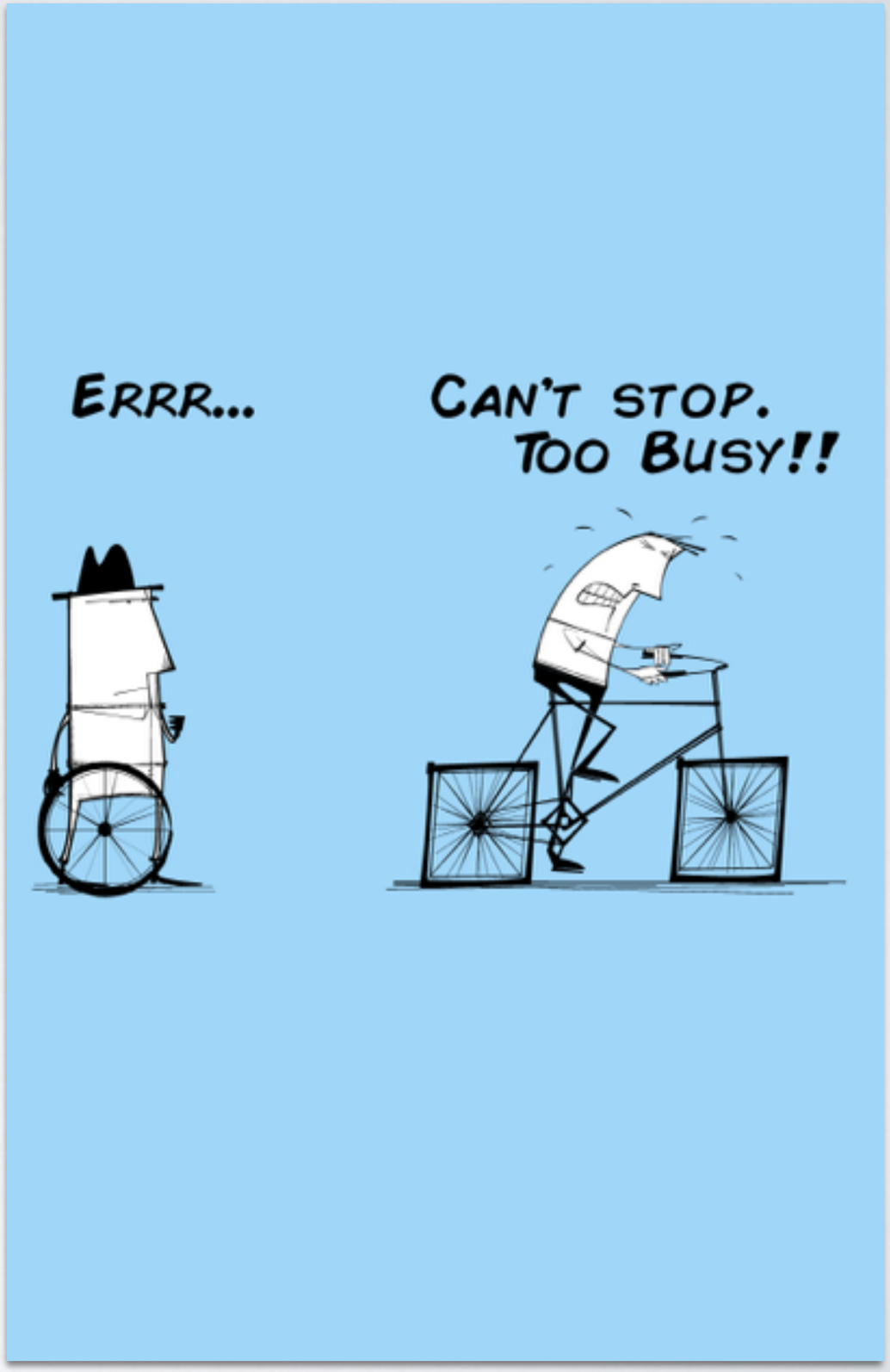


Code review, an interactive online survey!

http://sback.it/fosdem.html

http://goo.gl/forms/hknZvi1YUo

Why research on modern code review?



Code inspections

Code "inspection is a method of static testing to verify that software meets its requirements."

"It engages developers and others in a formal process [...] that usually detects more defects in the product than does machine testing."

Advances in Software Inspections
M. E. Fagan
[IEEE TSE 1986]

744 IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. SE-12, NO. 7,

Advances in Software Inspections

MICHAEL E. FAGAN, MEMBER, IEEE

Abstract—This paper presents new studies and experiences that enhance the use of the inspection process and improve its contribution to development of defect-free software on time and at lower costs. Examples of benefits are cited followed by descriptions of the process and some methods of obtaining the enhanced results.

Software inspection is a method of static testing to verify that software meets its requirements. It engages the developers and others in a formal process of investigation that usually detects more defects in the product—and at lower cost—than does machine testing. Users of the method report very significant improvements in quality that are accompanied by lower development costs and greatly reduced maintenance efforts. Excellent results have been obtained by small and large organizations in all aspects of new development as well as in maintenance. There is some evidence that developers who participate in the inspection of their own product actually create fewer defects in future work. Because inspections formalize the development process, productivity and quality enhancing tools can be adopted more easily and rapidly.

Index Terms—Defect detection, inspection, project management, quality assurance, software development, software engineering, software quality, testing, walkthru.

INTRODUCTION

THE software inspection process was created in 1972, in IBM Kingston, NY, for the dual purposes of improving software quality and increasing programmer productivity. Its accelerating rate of adoption throughout the software development and maintenance industry is an acknowledgment of its effectiveness in meeting its goals. Outlined in this paper are some enhancements to the inspection process, and the experiences of some of the many companies and organizations that have contributed to its evolution. The author is indebted to and thanks the many people who have given their help so liberally.

Because of the clear structure the inspection process has brought to the development process, it has enabled study of both itself and the conduct of development. The latter has enabled process control to be applied from the point at which the requirements are inspected—a much earlier point in the process than ever before—and throughout development. Inspections provide data on the perform... of individual development operations, thus p... unique opportunity to evaluate new tools... At the same time, studies of inspecti... fostered improvement of its ... very high defect detection... be conducted routi...

...velopment and design and code inspections prompted the adaptation of the principles of the inspection process to inspections of requirements, user information, and documentation, and test plans and test cases. In each instance, the new uses of inspection were found to improve product quality and to be cost effective, i.e., it saved more than it cost. Thus, as the effectiveness of inspections are improving, they are being applied in many new and different ways to improve software quality and reduce costs.

BENEFITS: DEFECT REDUCTION, DEFECT PREVENTION, AND COST IMPROVEMENT

In March 1984, while addressing the IBM SHARE User Group on software service, L. H. Fenton, IBM Director of VM Programming Systems, made an important statement on quality improvement due to inspections [1]:

"Our goal is to provide defect free products and product information, and we believe the best way to do this is by refining and enhancing our existing software development process.

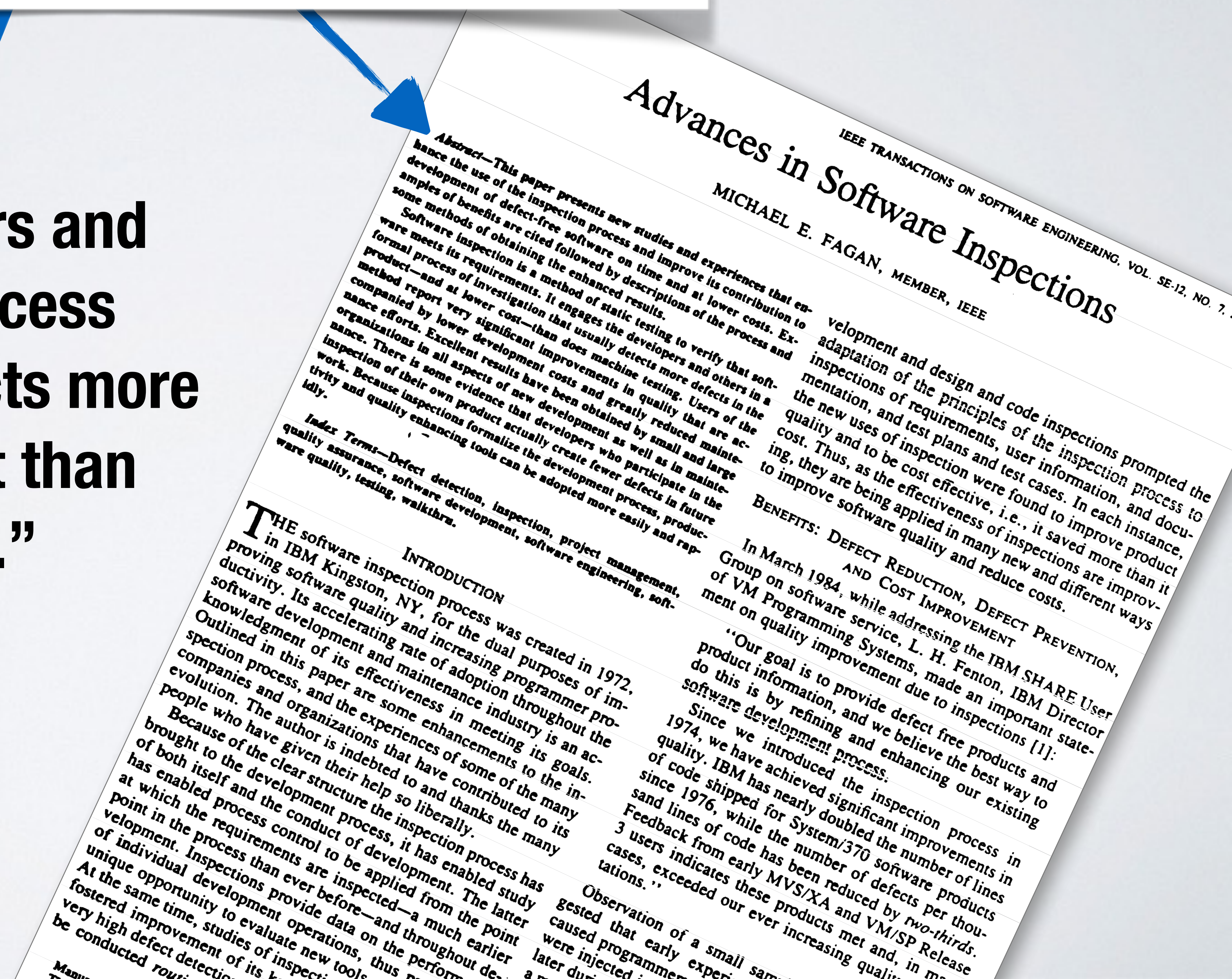
Since we introduced the inspection process in 1974, we have achieved significant improvements in quality. IBM has nearly doubled the number of lines of code shipped for System/370 software products since 1976, while the number of defects per thousand lines of code has been reduced by *two-thirds*. Feedback from early MVS/XA and VM/SP Release 3 users indicates these products met and, in ma... cases, exceeded our ever increasing quality... tations."

Observation of a small sam... gested that early experi... caused programmer... were injected i... later du...

Code inspections

Can you trust these results apply to modern code review?

“It engages developers and others in a formal process […] that usually detects more defects in the product than does machine testing.”



IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. SE-12, NO. 7,

Advances in Software Inspections

MICHAEL E. FAGAN, MEMBER, IEEE

Abstract—This paper presents new studies and experiences that enhance the use of the inspection process and improve its contribution to development of defect-free software on time and at lower costs. Examples of benefits are cited followed by descriptions of the process and some methods of obtaining the enhanced results.

Software inspection is a method of static testing to verify that software meets its requirements. It engages the developers and others in a formal process of investigation that usually detects more defects in the product—and at lower cost—than does machine testing. Users of the method report very significant improvements in quality that are accompanied by lower development costs and greatly reduced maintenance efforts. Excellent results have been obtained by small and large organizations in all aspects of new development as well as in maintenance. There is some evidence that developers who participate in the inspection of their own product actually create fewer defects in future work. Because inspections formalize the development process, productivity and quality enhancing tools can be adopted more easily and rapidly.

Index Terms—Defect detection, inspection, project management, quality assurance, software development, software engineering, software quality, testing, walkthru.

INTRODUCTION

THE software inspection process was created in 1972, in IBM Kingston, NY, for the dual purposes of improving software quality and increasing programmer productivity. Its accelerating rate of adoption throughout the software development and maintenance industry is an acknowledgment of its effectiveness in meeting its goals. Outlined in this paper are some enhancements to the inspection process, and the experiences of some of the many companies and organizations that have contributed to its evolution. The author is indebted to and thanks the many people who have given their help so liberally.

Because of the clear structure the inspection process has brought to the development process, it has enabled study of both itself and the conduct of development. The latter has enabled process control to be applied from the point at which the requirements are inspected—a much earlier point in the process than ever before—and throughout development. Inspections provide data on the perform... of individual development operations, thus ... unique opportunity to evaluate new tools ... At the same time, studies of inspecti... fostered improvement of its ... very high defect detectio... be conducted routi...

...velopment and design and code inspections prompted the adaptation of the principles of the inspection process to inspections of requirements, user information, and documentation, and test plans and test cases. In each instance, the new uses of inspection were found to improve product quality and to be cost effective, i.e., it saved more than it cost. Thus, as the effectiveness of inspections are improving, they are being applied in many new and different ways to improve software quality and reduce costs.

BENEFITS: DEFECT REDUCTION, DEFECT PREVENTION, AND COST IMPROVEMENT

In March 1984, while addressing the IBM SHARE User Group on software service, L. H. Fenton, IBM Director of VM Programming Systems, made an important statement on quality improvement due to inspections [1]:

“Our goal is to provide defect free products and product information, and we believe the best way to do this is by refining and enhancing our existing software development process.

Since we introduced the inspection process in 1974, we have achieved significant improvements in quality. IBM has nearly doubled the number of lines of code shipped for System/370 software products since 1976, while the number of defects per thousand lines of code has been reduced by *two-thirds*. Feedback from early MVS/XA and VM/SP Release 3 users indicates these products met and, in many cases, exceeded our ever increasing quality expectations.”

Observation of a small sam... gested that early experi... caused programmer... were injected ... later du...

Modern Code Review @ Microsoft

Dr. Christian Bird

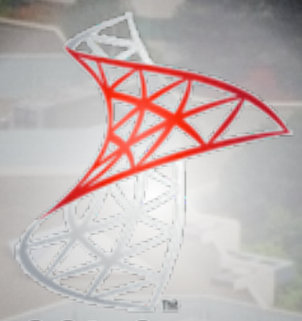
Modern Code Review @ Microsoft

Modern Code Review @ Microsoft

Excel

Windows Phone

XBox



SQL Server

...

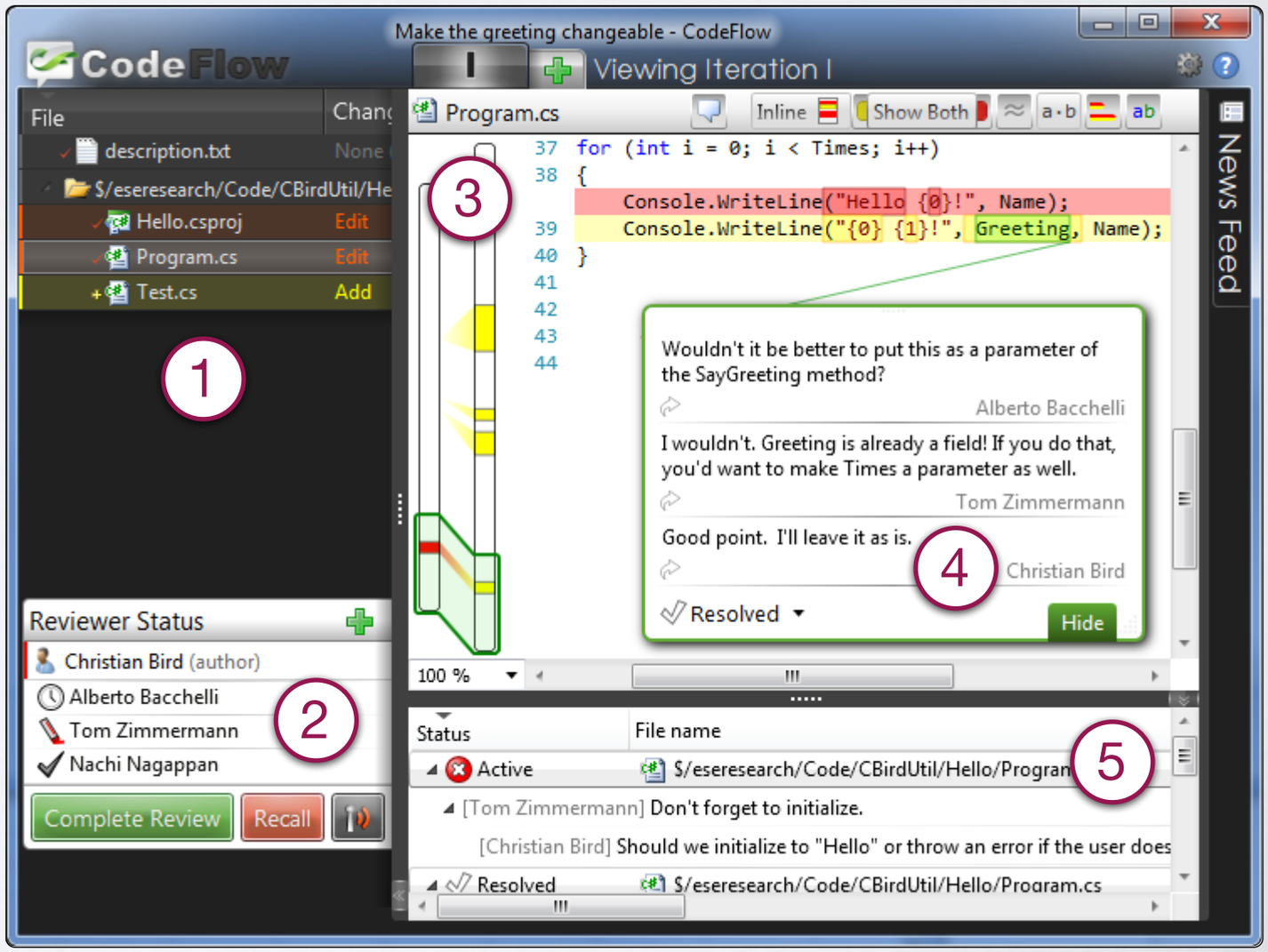
The CodeFlow review tool

Used across all Microsoft product teams by more than 70,000 developers, so far.

Excel

XBox

SQL Server



observations

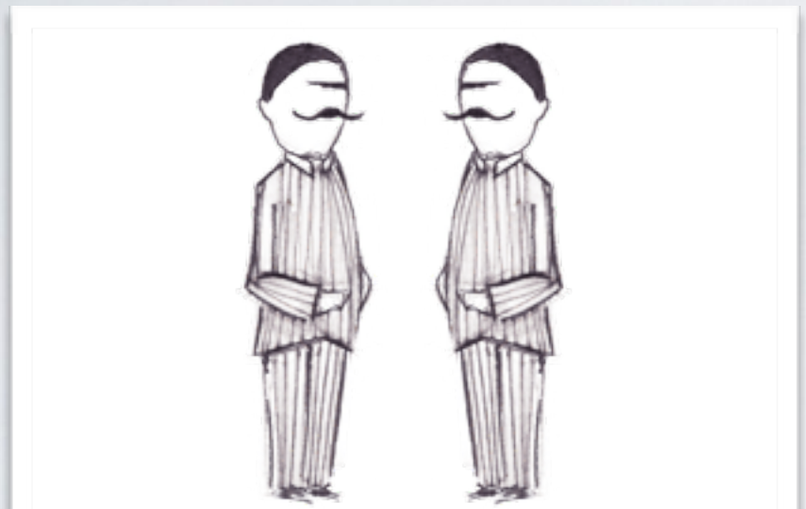
interviews

18 interviews with observations
~40 minutes long
developers, testers
different roles
signed off at least 50 reviews


Shuttle Connect
SIENNA LE AWD
Microsoft
363
Shuttle
Connect
363

observations

interviews



survey to 165 managers

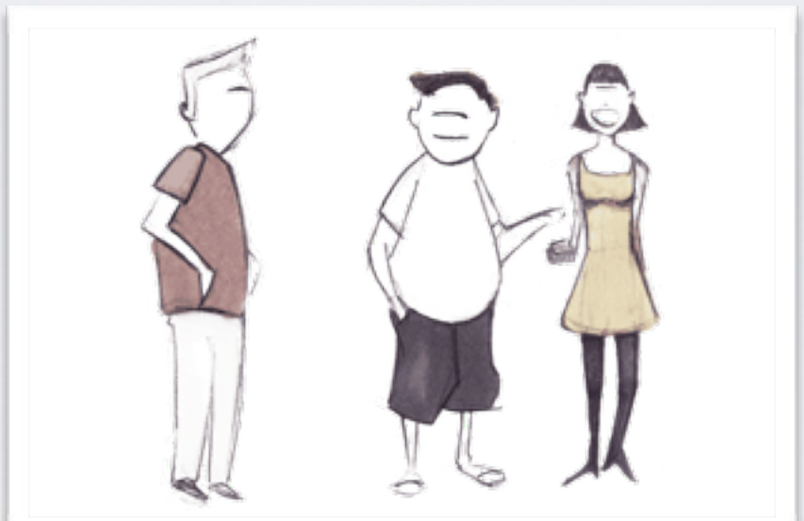
List of motivations for doing code review

- Alternative Solutions
- Avoid Build Breaks
- Code Improvement
- Team Assessment
- Share Code Ownership
- Team Awareness
- Knowledge Transfer
- Improve Dev. Process
- Track Rationale
- Finding Defects

observations

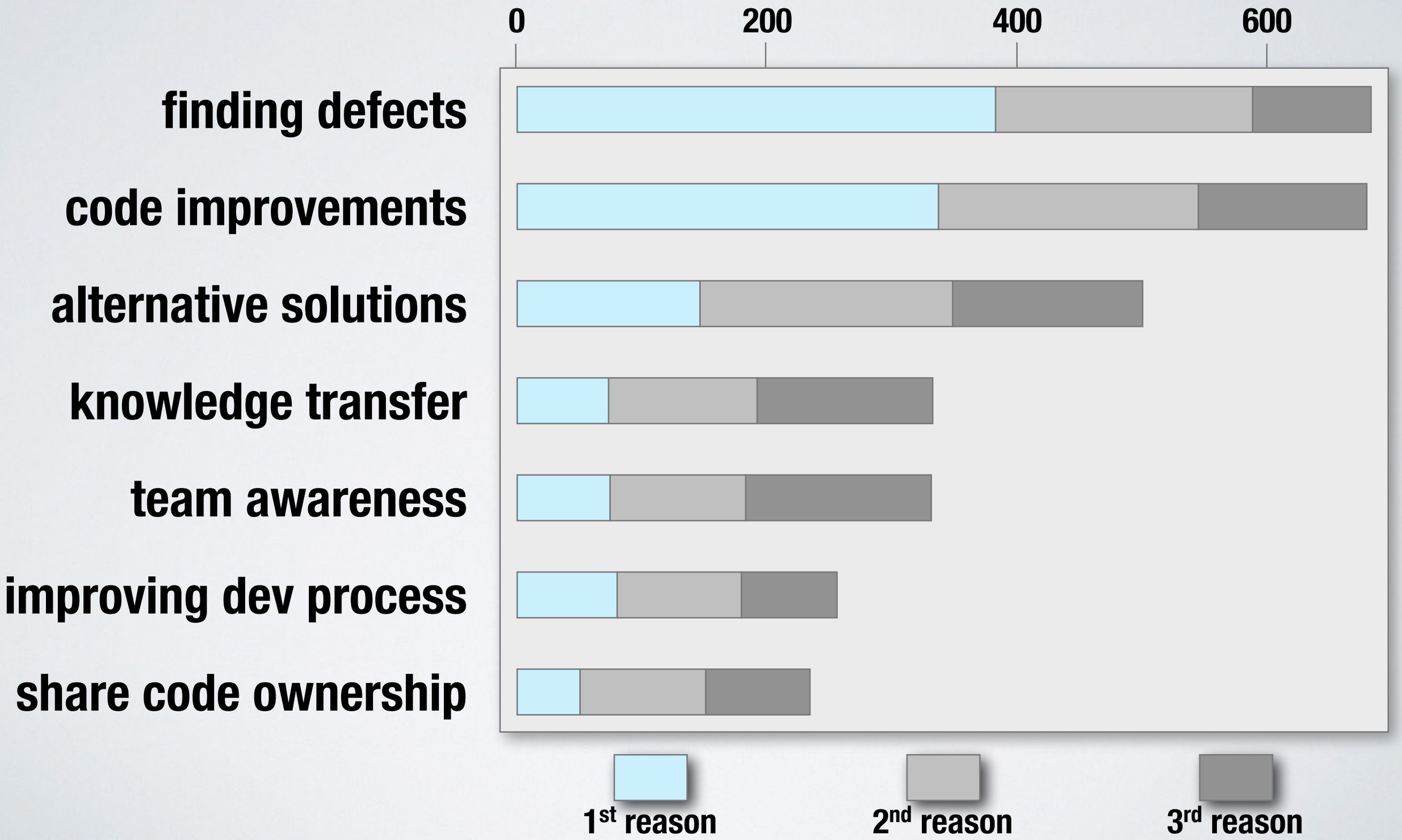
interviews

survey to 165 managers



survey to 873 developers

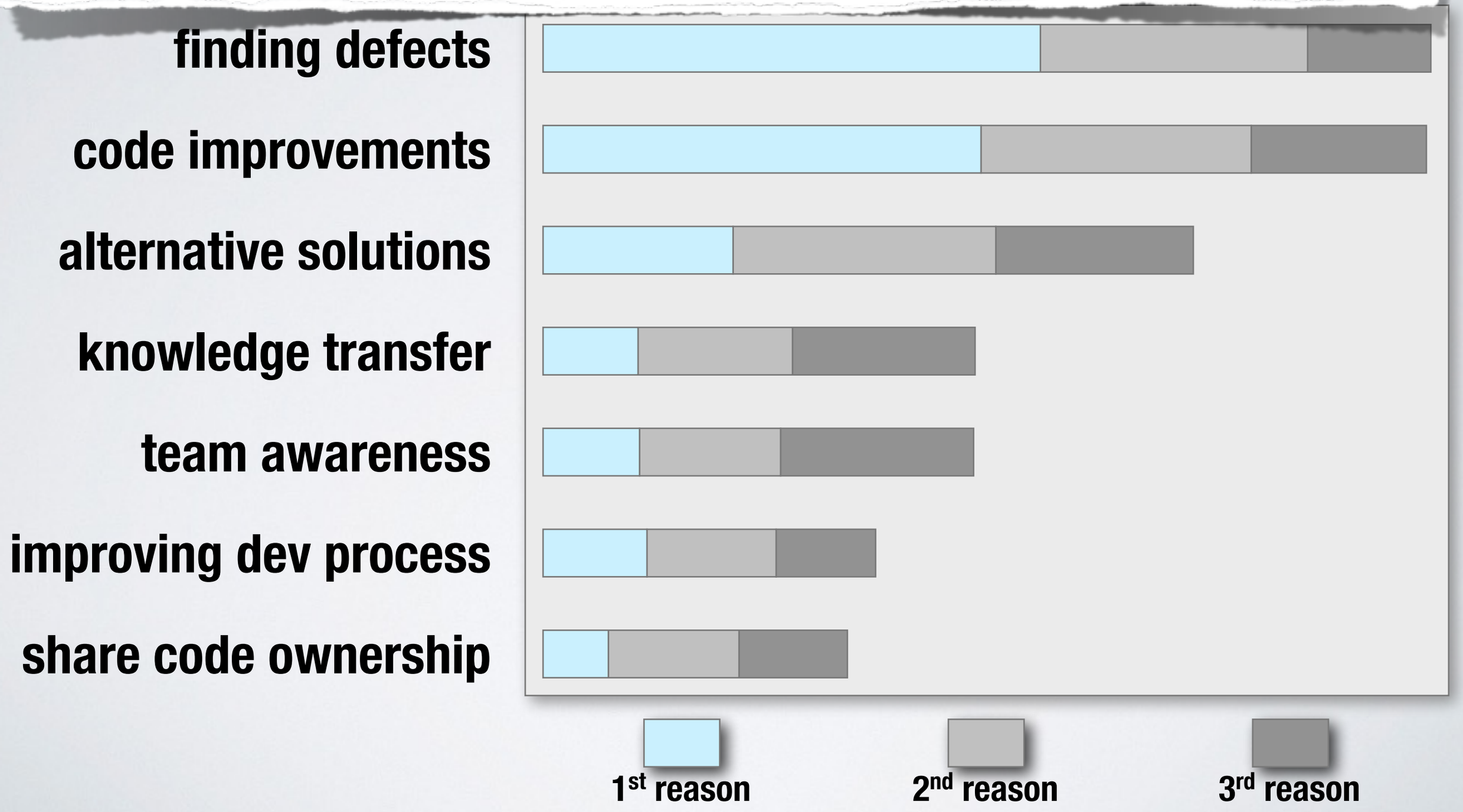
Why do Microsoft developers do code reviews?



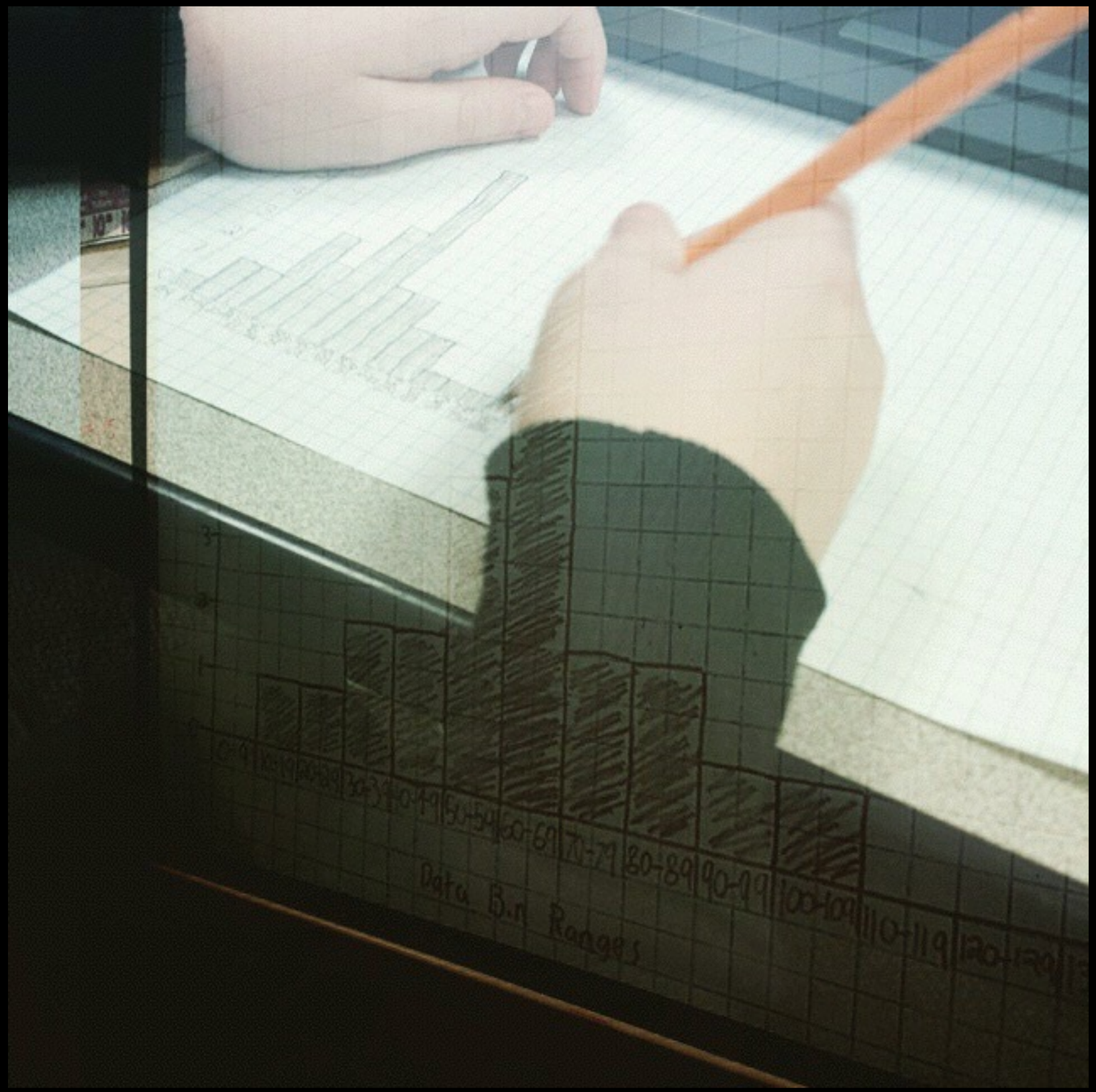
Why do Microsoft developers do code reviews?

“Finding defects is the main reason for doing code review.”

72 managers and 384 developers @ Microsoft

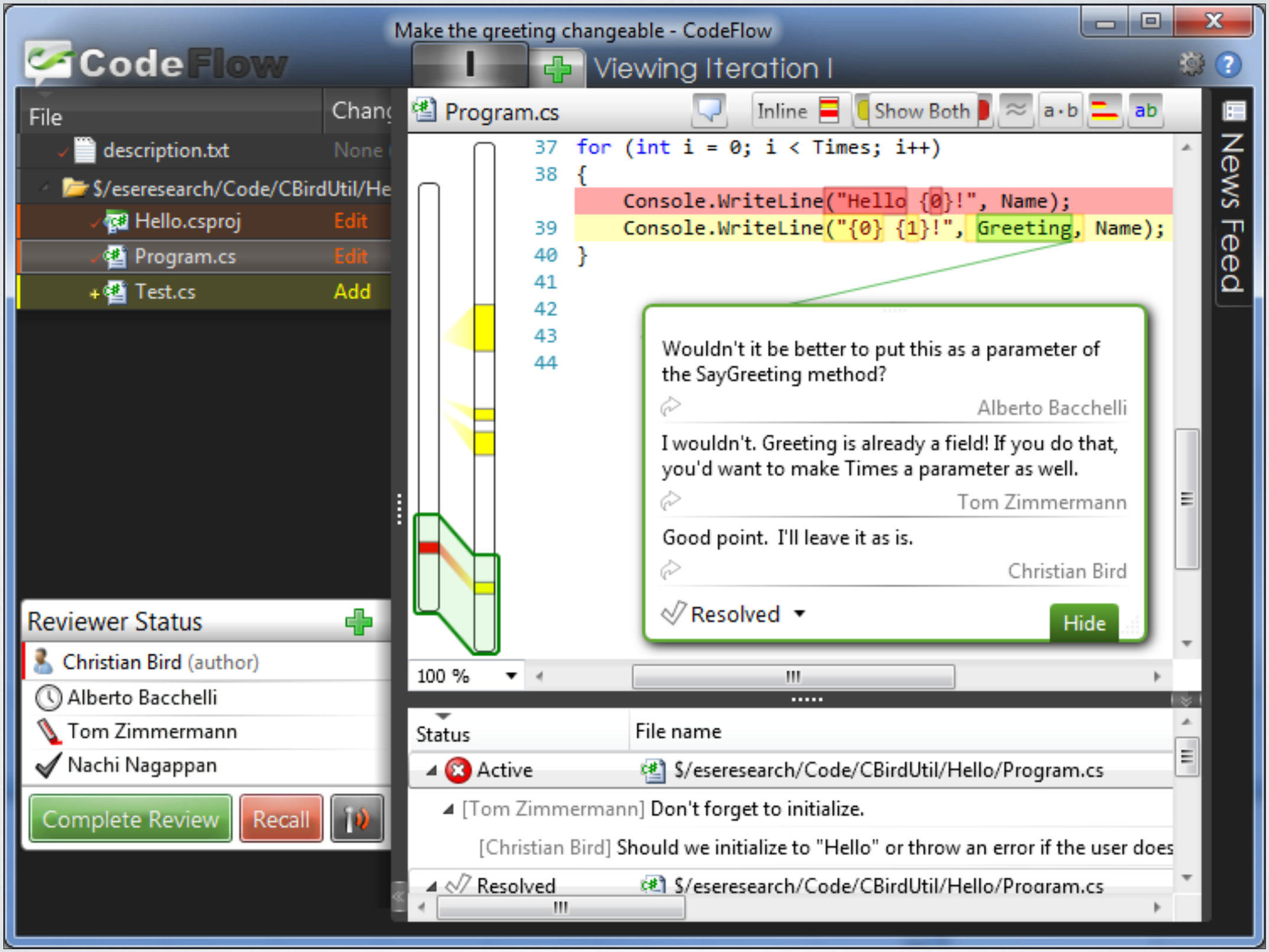


Let’s look at the survey’s answers!

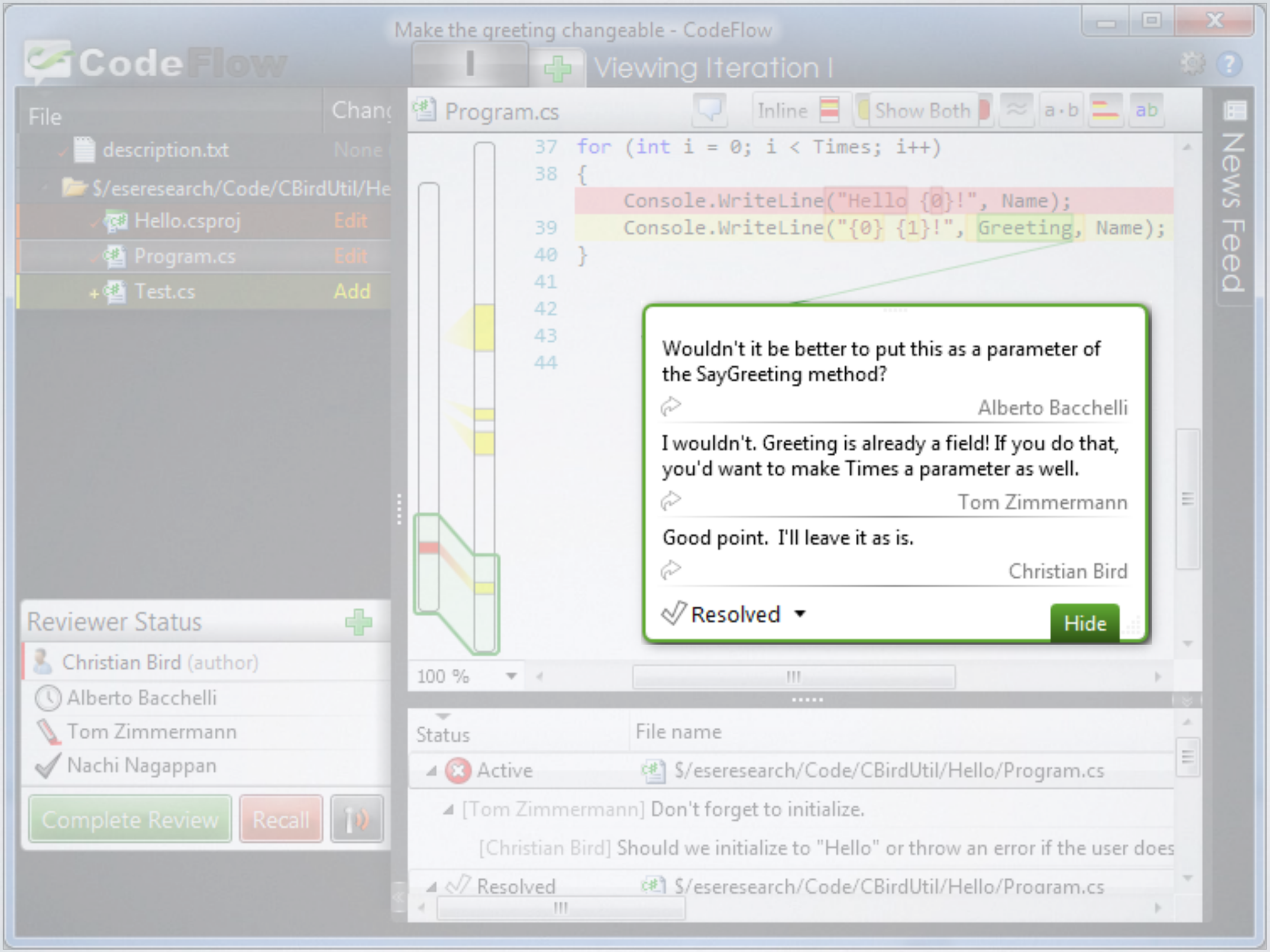


What is the outcome of code review at Microsoft?

What is the outcome of code review at Microsoft?



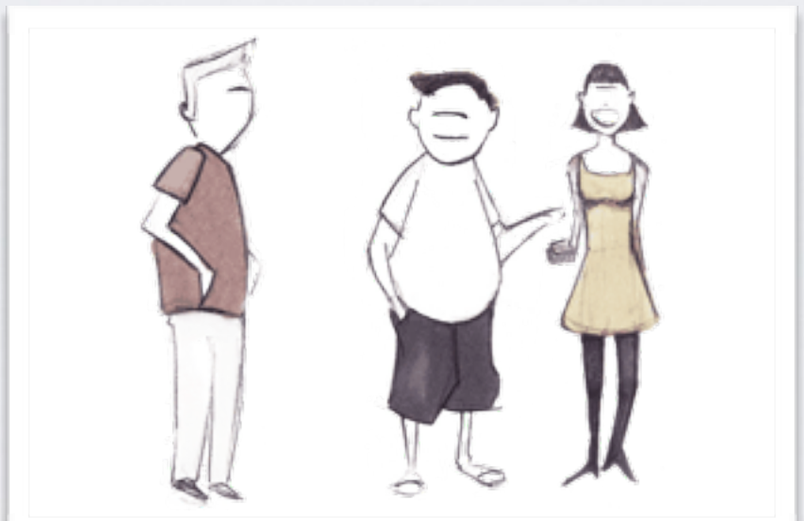
Recorded code review comments



observations

interviews

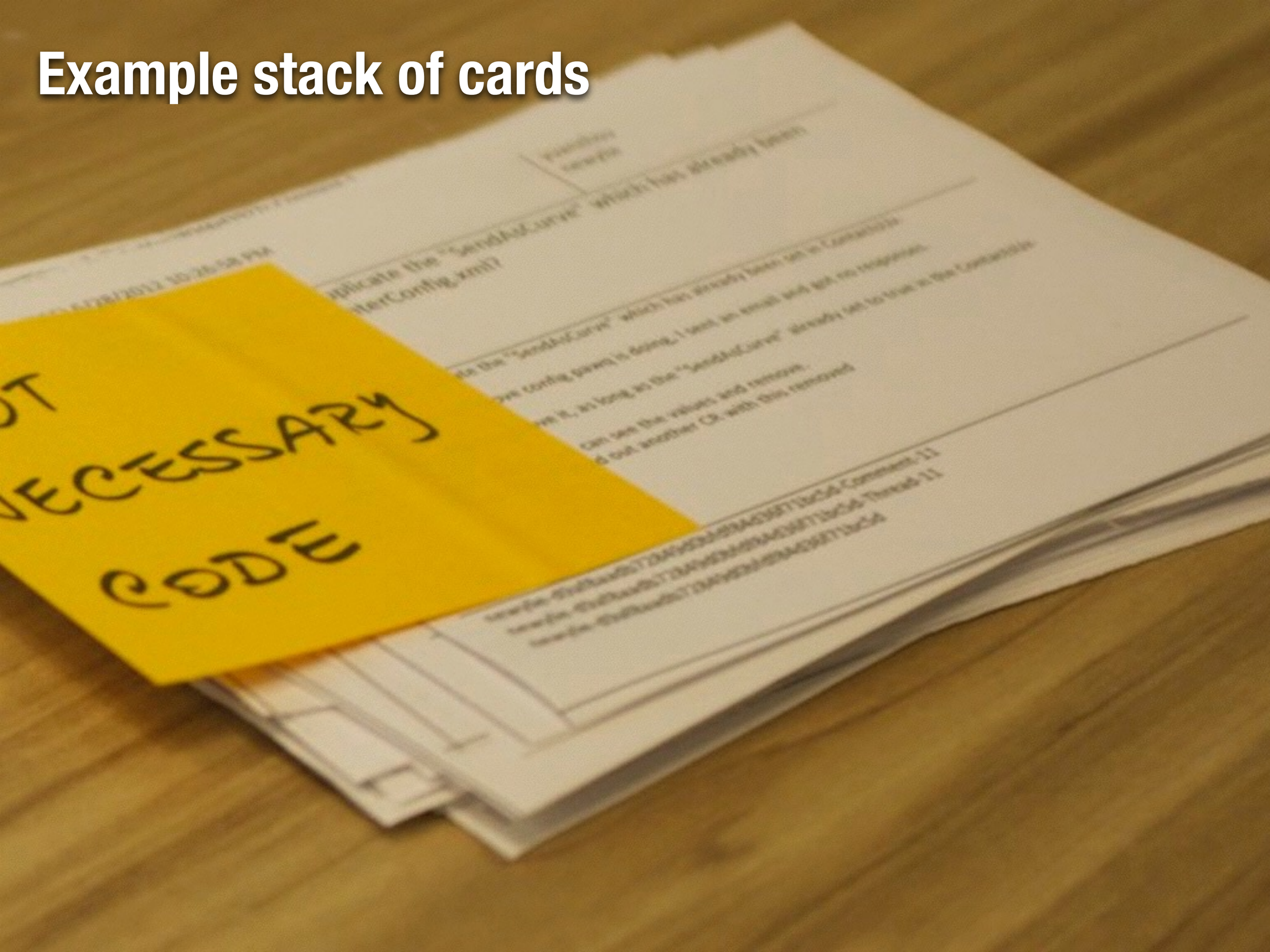
survey to 165 managers



survey to 873 developers

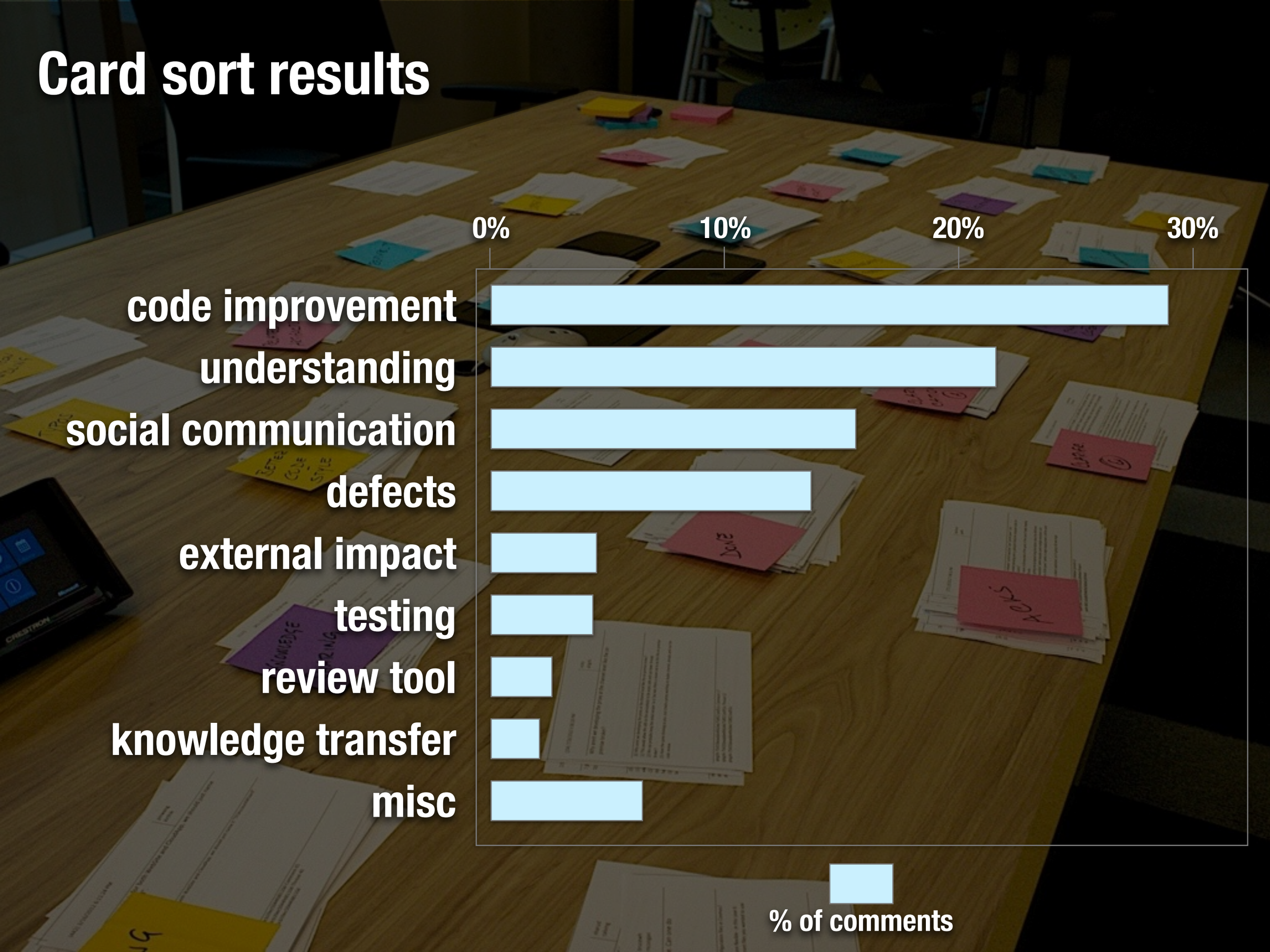
classification of
570 review comments

Example stack of cards

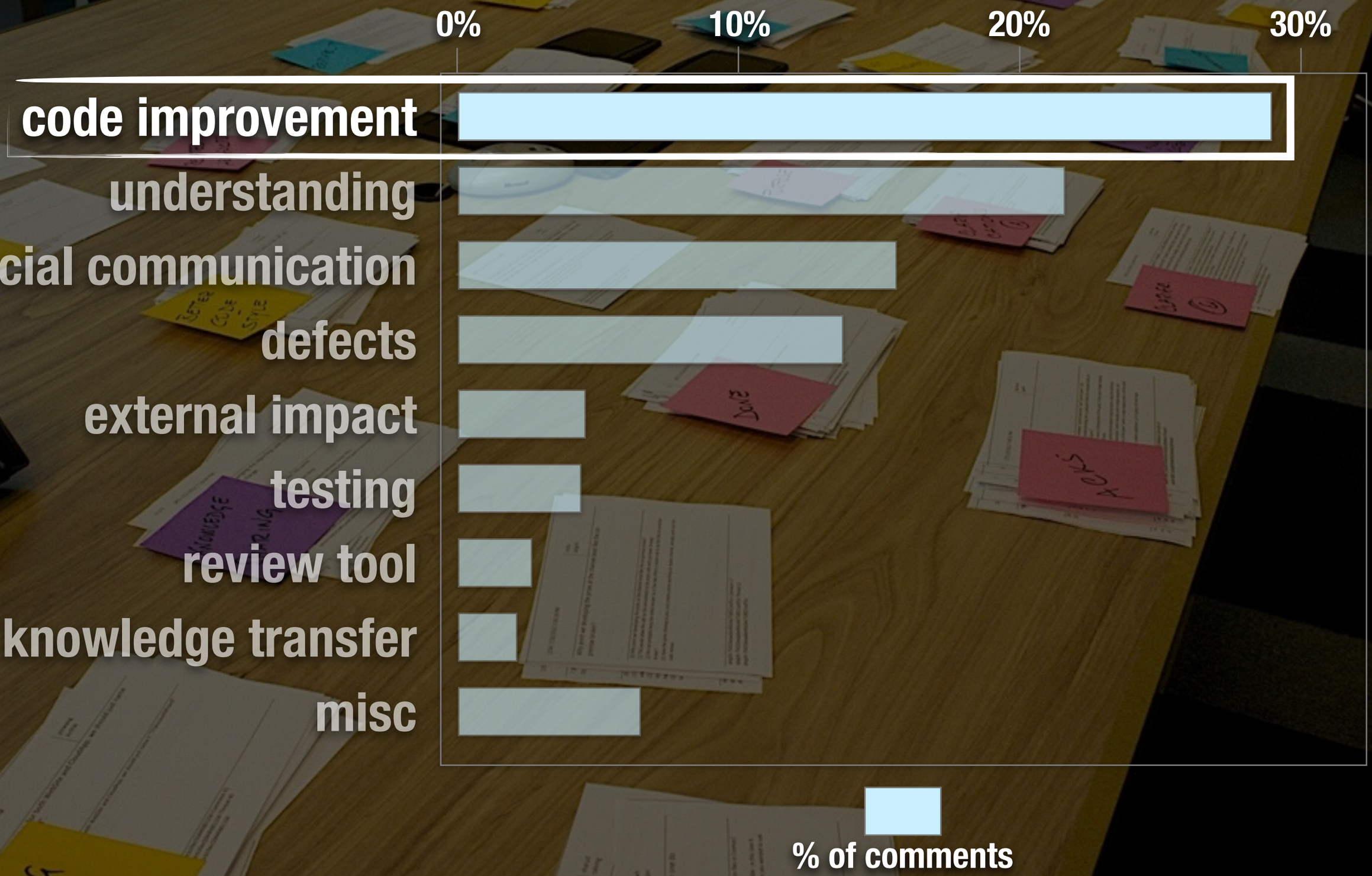


Card sort almost completed

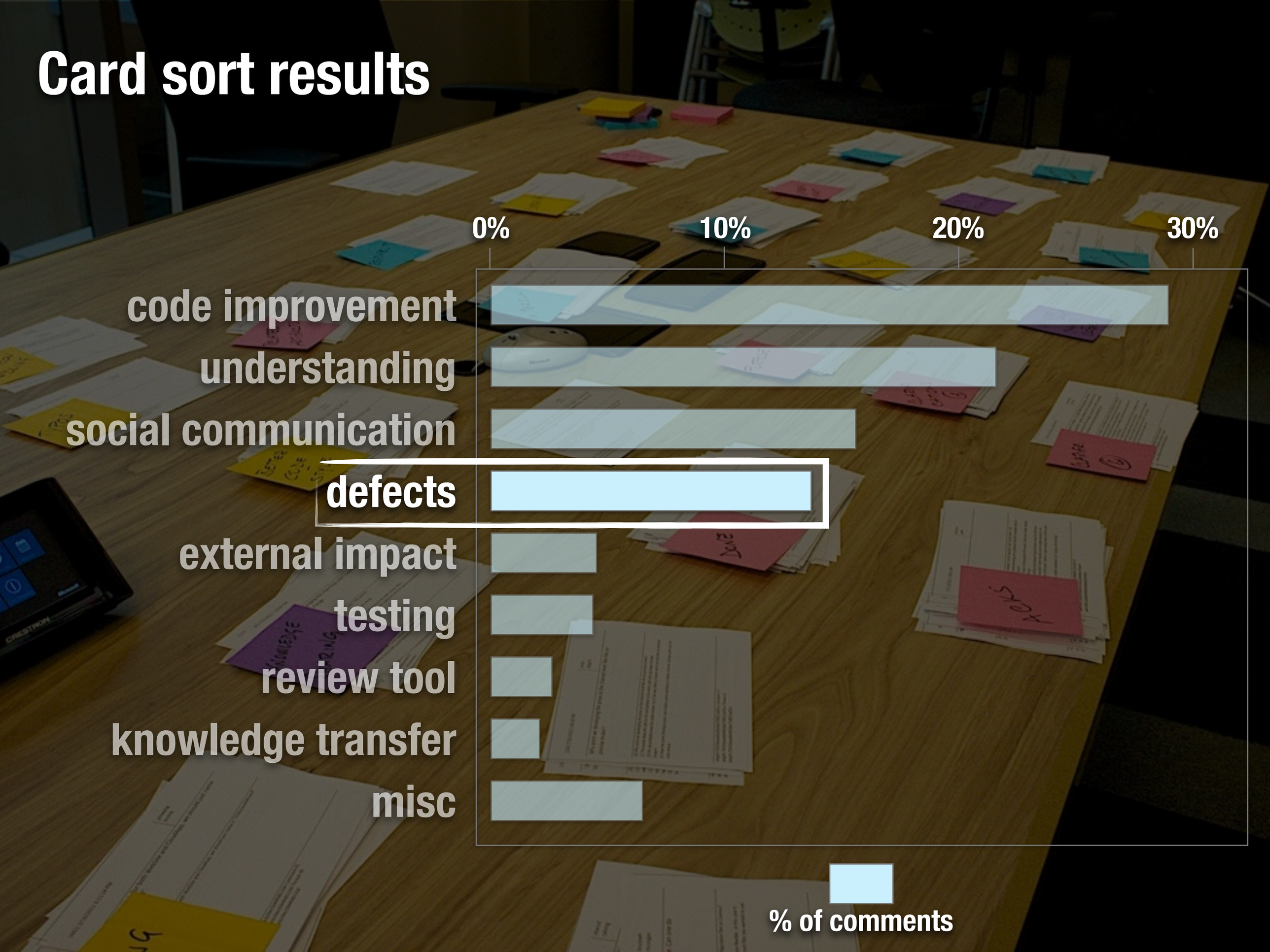
Card sort results



Card sort results



Card sort results



Card sort results defects

"what if they are all used?"

"is it possible that this statement never match?"

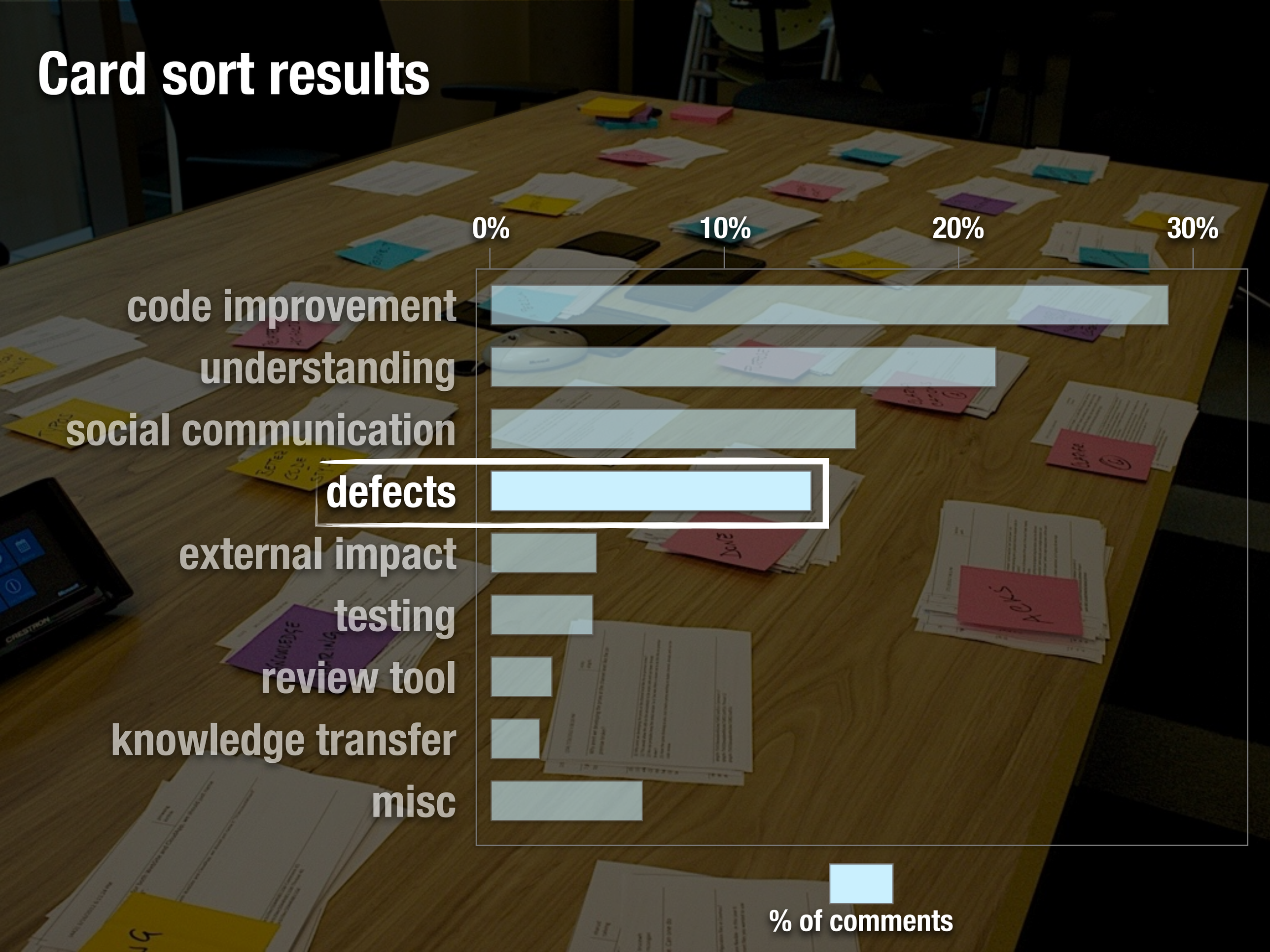
"should this end date be current date?"

"does it work if you put 0 here?"

"any doubt about the precedence here?"

"should be &&?"

Card sort results



Code review at Microsoft: Expectations vs. Reality

What is the outcome of code review in OSS?

Moritz Beller

TU Delft

What is the outcome of code review in OSS?

GROMACS FAST. FLEXIBLE. FREE.

conQAT

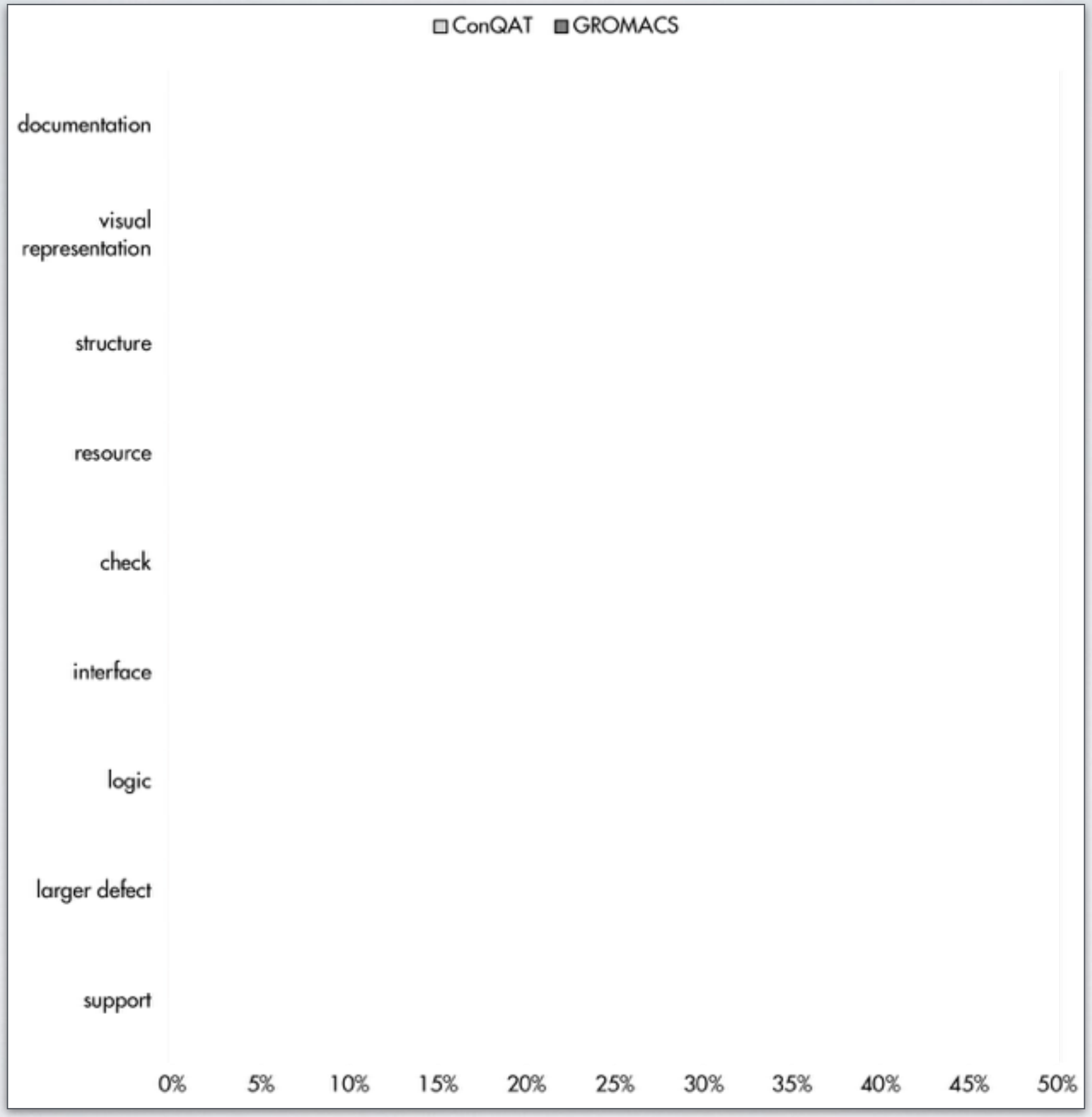
We manually analyzed

1,400 review-induced changes

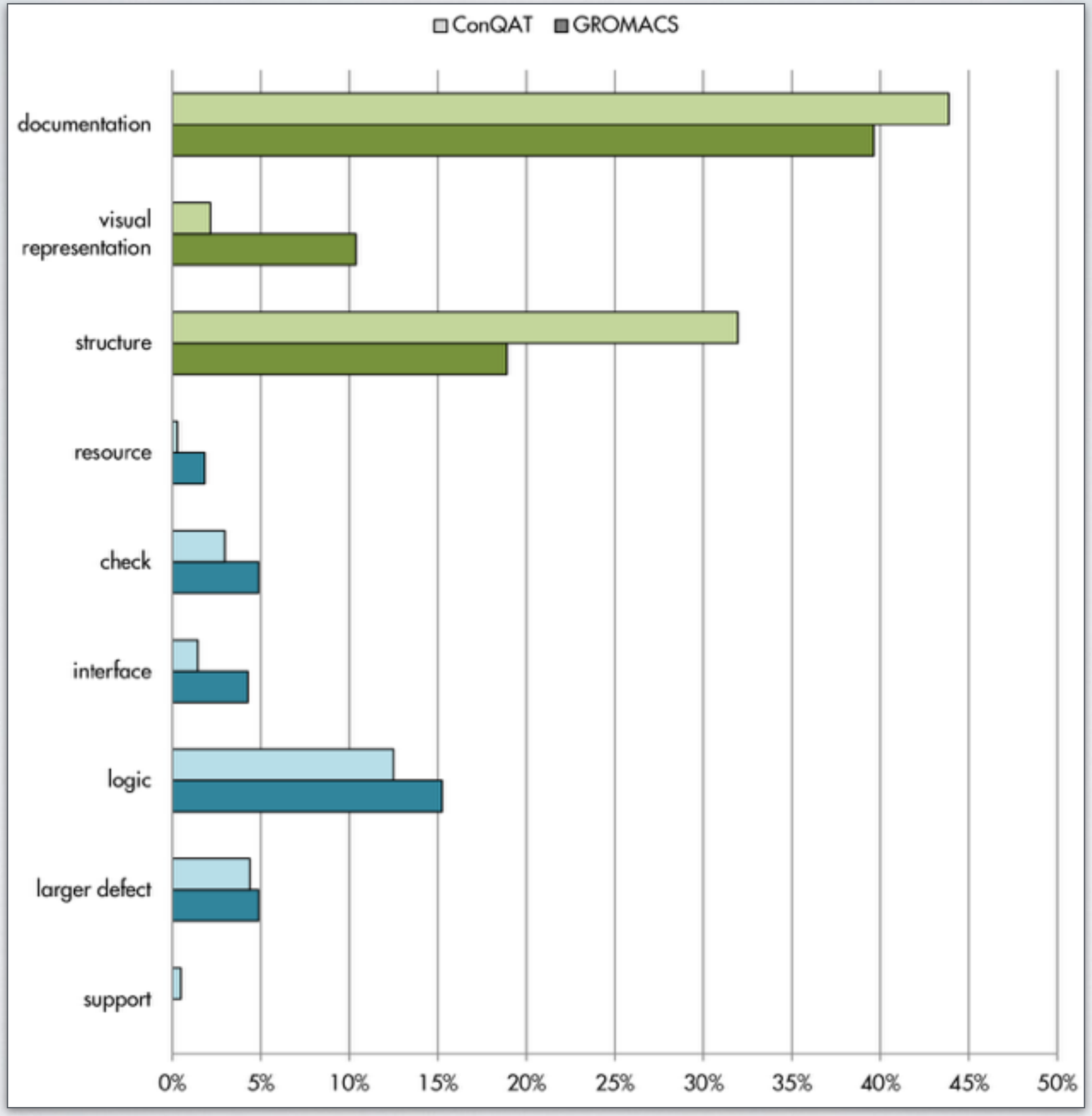
```
108    /** Validity timer for concatinated SMS segment */
       Wink Saville   Maybe reword as: Max time value between concatenated SMS seg...
109    private static final int VALID_TIMER_CONCAT_SEGMENT = 256000;
       Wink Saville   Rename to MAX_CONCATENATED_SEGMENT_TIME
110    private static final int PDU_COLUMN = 0;
111    private static final int SEQUENCE_COLUMN = 1;
112    private static final int DESTINATION_PORT_COLUMN = 2;
113    private static final int DATE_COLUMN = 3; //check validity time
       Wink Saville   Remove comment.
114
                    ... skipped 574 common lines ...+100
575                 if (Arrays.equals(oldPdu, pdu)) {
576                     return Intents.RESULT_SMS_HANDLED;
       Wink Saville   Before returning we should call cursor.close().
577                 }
```

```
108    /** Max time value between concatenated SMS segments */
109    private static final int MAX_CONCATENATED_SEGMENT_TIME =
110    private static final int PDU_COLUMN = 0;
111    private static final int SEQUENCE_COLUMN = 1;
112    private static final int DESTINATION_PORT_COLUMN = 2;
113    private static final int DATE_COLUMN = 3;
114
                    ... skipped 574 common lines ...+100
575                 if (Arrays.equals(oldPdu, pdu)) {
576                     cursor.close();
577                     return Intents.RESULT_SMS_HANDLED;
578                 }
```

What is the outcome of code review in OSS?



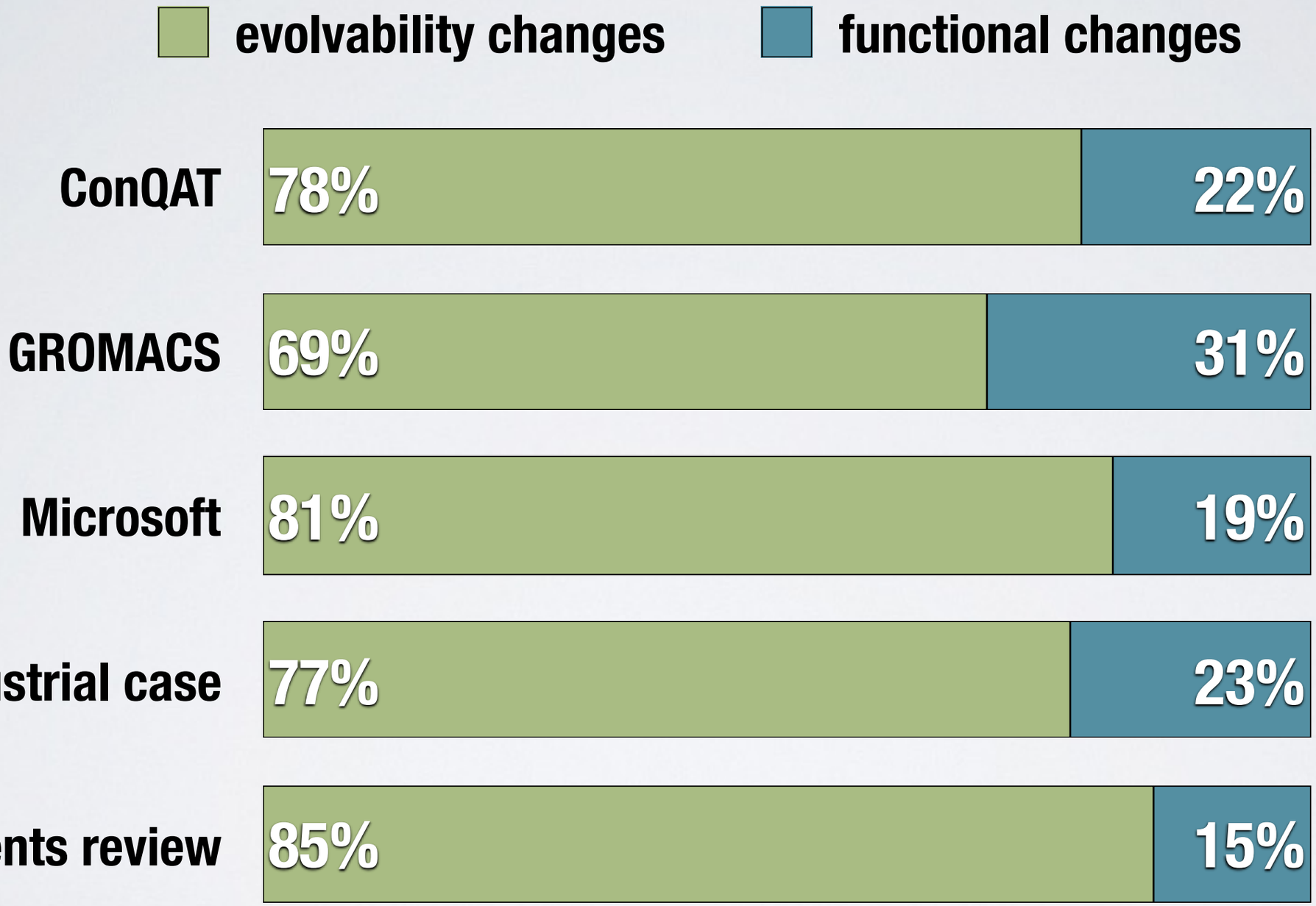
What is the outcome of code review in OSS?



What is the outcome of code review in OSS?

What is the outcome of code review?

What is the outcome of code review?



What Do Code Reviews at Microsoft and in Open Source Projects Have in Common?

What Do Code Reviews at Microsoft and in Open Source Projects Have in Common?

The outcome

Why do expectations not match reality?

Code reviews

… “[if] executed properly, [they] find bugs faster and more effectively than testing or other known debugging techniques”

— Jason Cohen, 2011

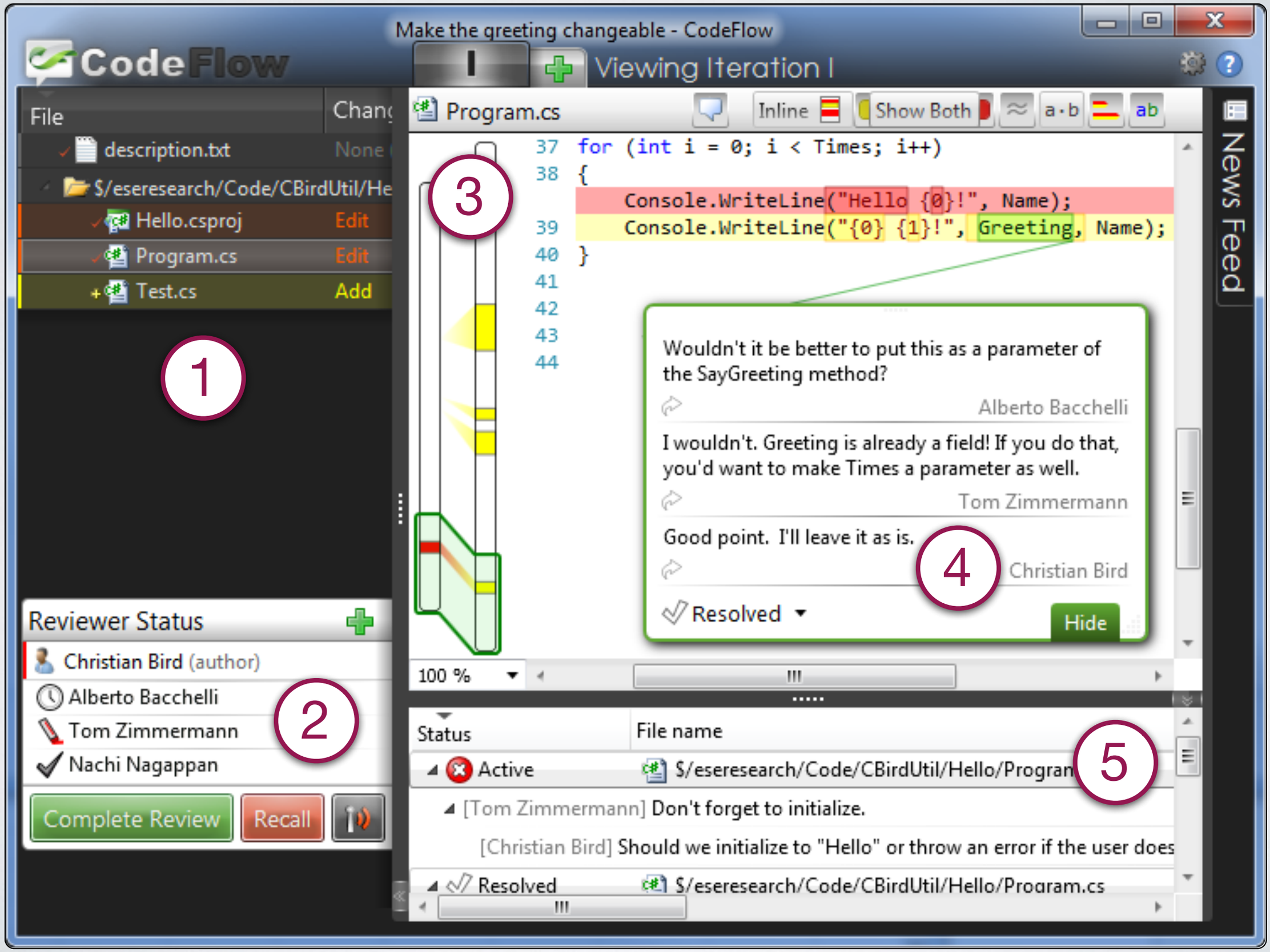
Code reviews

… “[if] executed properly, [they] find bugs faster and more effectively than testing or other known debugging techniques**—but when done inefficiently they can quickly become unproductive. ”**

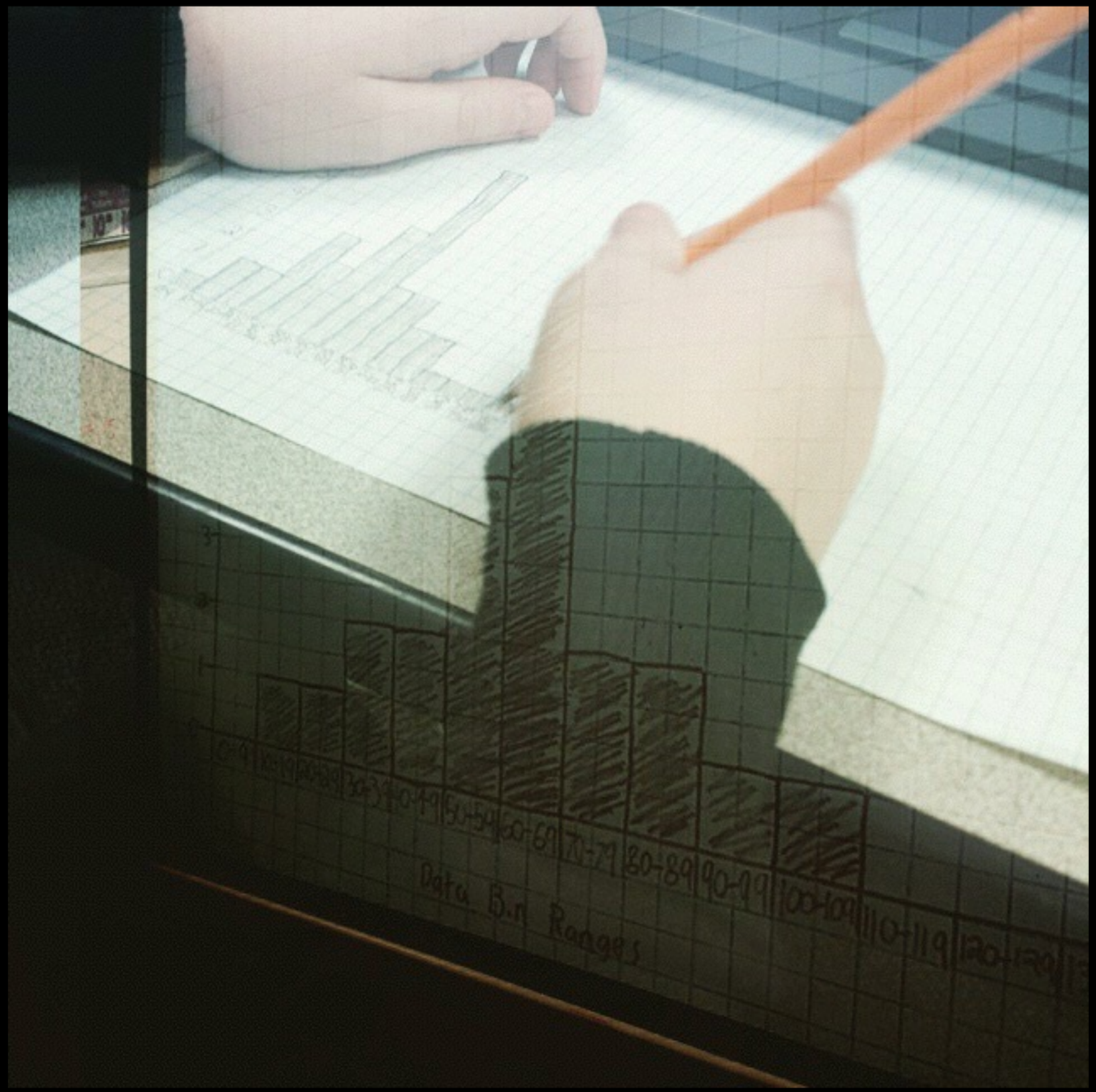
— Jason Cohen, 2011

Code review is (still) a fully manual task

Tools only supports logistics of code review



Let’s look at the survey’s answers!

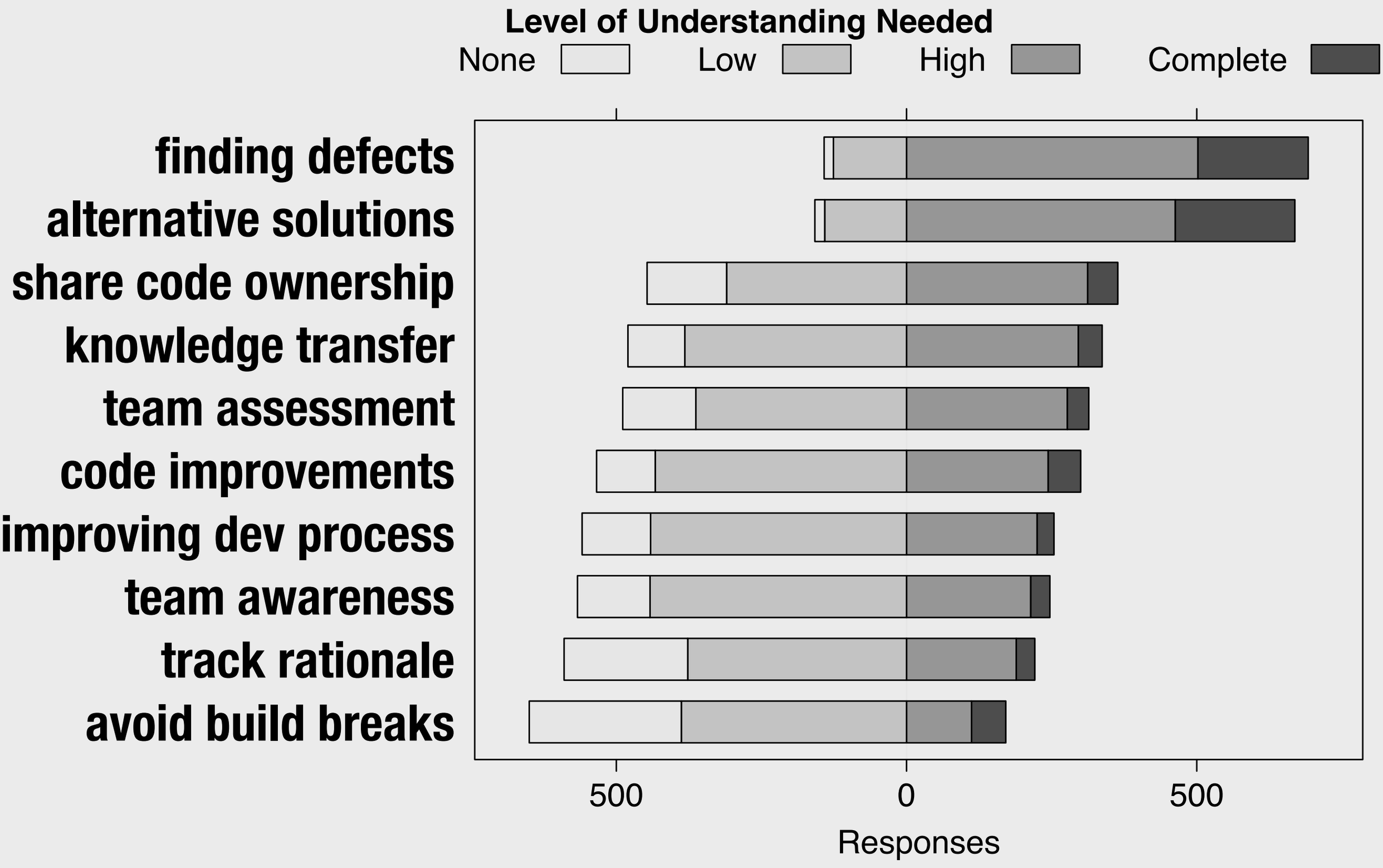


What are the challenges of code review at MSFT?

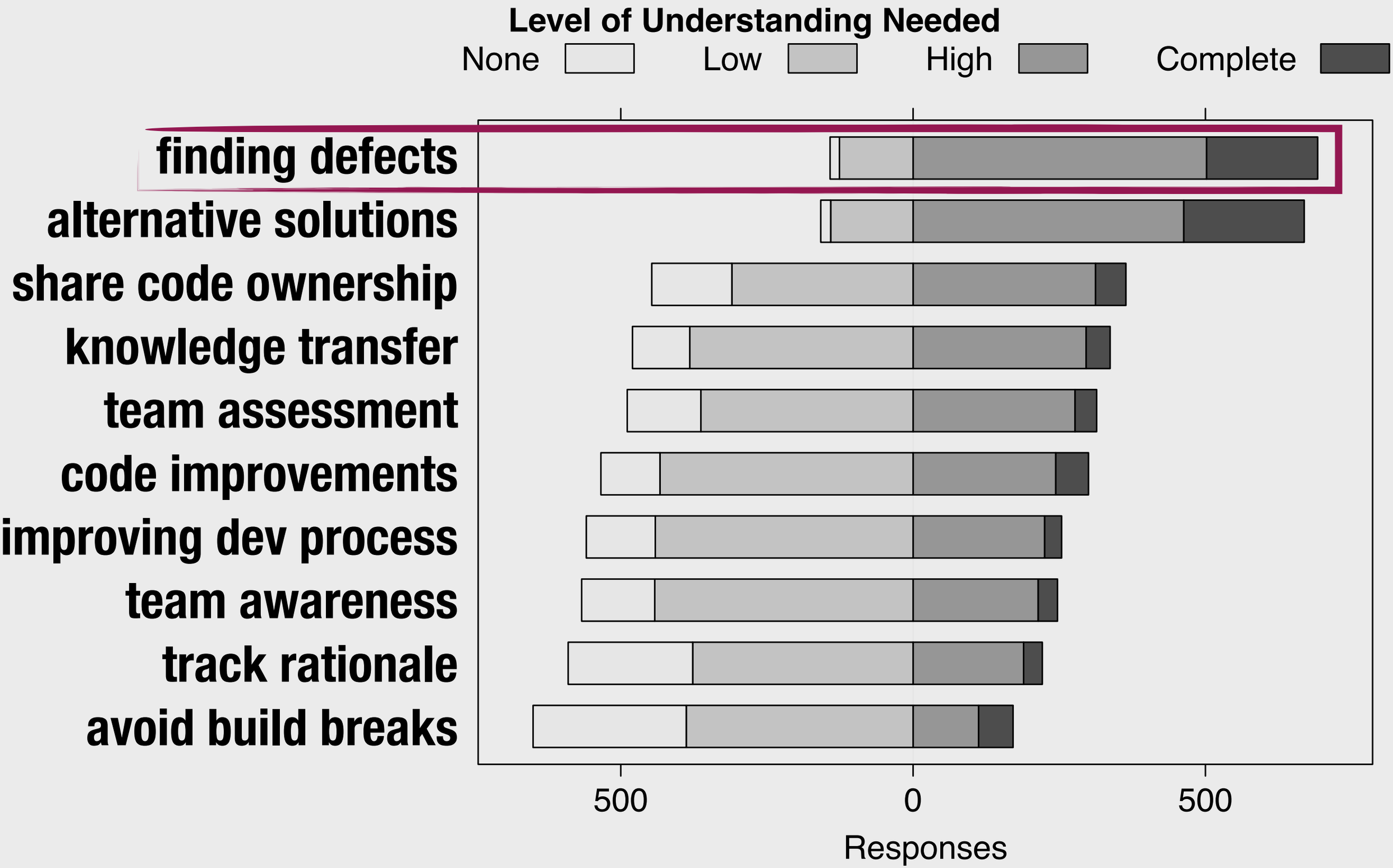
What are the challenges of code review at MSFT?

“understanding the code takes most of the reviewing time.”

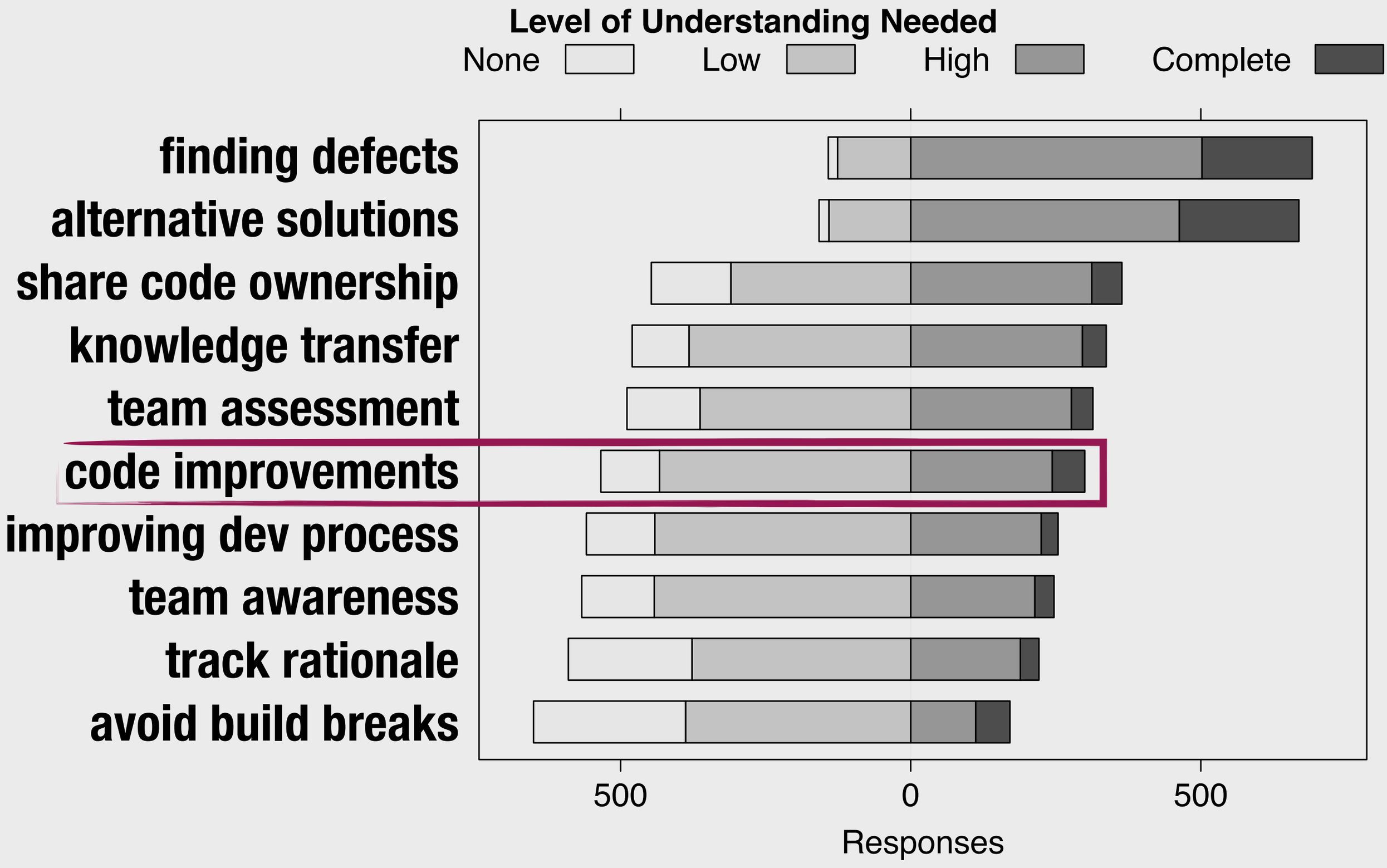
Understanding needs, by outcome



Understanding needs, by outcome



Understanding needs, by outcome



Effect of code ownership on reviews

Does it take longer to review files that you are not familiar with (or files that are new)?

"YES"

798 developers (91%)

Is there a difference in comments/feedback you receive when a reviewer is very familiar with or the owner of the files you changed in a code review?

"YES"

716 developers (82%)

Effect of code ownership on reviews

Is there a difference in comments/feedback you receive when a reviewer is very familiar with or the owner of the files you changed in a code review?

"YES"

716 developers (82%)

"Comments reflect their deeper understanding – more likely to find subtle defects, feedback is more conceptual (better ideas, approaches) instead of superficial (naming, mechanical style, etc.)"

Code review

needs the right data and the right tools

Software analytics

… “is analytics on software data for managers and software engineers with the aim of empowering software development individuals and teams to gain and share insight from their data to make better decisions.”

— Menzies and Zimmermann, 2013

FOCUS: **GUEST EDITORS’ INTRODUCTION**

Software Analytics: So What?

Tim Menzies, West Virginia University

Thomas Zimmermann, Microsoft Research

IN THIS SPECIAL issue of *IEEE Software*, we invited submissions that reflected the benefits (and drawbacks) of software analytics. The response was overwhelming. Software analytics is an area of explosive growth, and we had so many excellent submissions that we had to split this special issue into two volumes—you’ll see even more content in the September/October issue. We divided the articles on conceptual grounds, so both volumes will feature equally excellent work.

To better frame these articles, we offer some definitions and historical perspectives on software analytics. Specifically, we describe where the field was, where it is, and where it might be going.

What Is Software Analytics?

Thanks to the Internet and open source, there’s now so much data about software projects that it’s impossible to manually browse through it all:

- As of late 2012, our Web searches show that Mozilla Firefox had 800,000 bug reports, and platforms such as Sourceforge.net and GitHub hosted 324,000 and 11.2 million projects, respectively.

0740-7459/13/$31.00 © 2013 IEEE

Software analytics

… “**is analytics on software data** for managers and software engineers with the aim of **empowering software development individuals and teams** to gain and share insight from their data to make better decisions.”

— Menzies and Zimmermann, 2013

FOCUS: **GUEST EDITORS’ INTRODUCTION**

Software Analytics: So What?

Tim Menzies, West Virginia University

Thomas Zimmermann, Microsoft Research

IN THIS SPECIAL issue of *IEEE Software*, we invited submissions that reflected the benefits (and drawbacks) of software analytics. The response was overwhelming. Software analytics is an area of explosive growth, and we had so many excellent submissions that we had to split this special issue into two volumes—you’ll see even more content in the September/October issue. We divided the articles on conceptual grounds, so both volumes will feature equally excellent work.

To better frame these articles, we offer some definitions and historical perspectives on software analytics. Specifically, we describe where the field was, where it is, and where it might be going.

What Is Software Analytics?

Thanks to the Internet and open source, there’s now so much data about software projects that it’s impossible to manually browse through it all:

- As of late 2012, our Web searches show that Mozilla Firefox had 800,000 bug reports, and platforms such as Sourceforge.net and GitHub hosted 324,000 and 11.2 million projects, respectively.

0740-7459/13/$31.00 © 2013 IEEE


IDE
Versioning system
01:09 Action A1
01:09 Action A2
01:09 Action A3
01:09 Action A4
01:09 Action A5
01:09 Action
Execution
Debugger
Issue tracking system
Mailing system

Software analytics' workflow

software engineering tasks helped

programming

debugging

maintenance

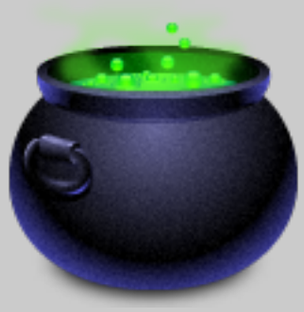
testing

...

data mining and software analysis techniques

classification

patterns



clustering

...

software data

source code

versioning system

issue tracking system

...

Software analytics for code reviews

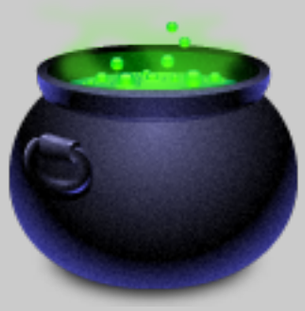
software engineering tasks helped

code review

data mining and software analysis techniques

classification

patterns



clustering

...

software data

source code

versioning system

issue tracking system

review data

...

My research: Data-supported code review

recommender for reviewers

change untangler

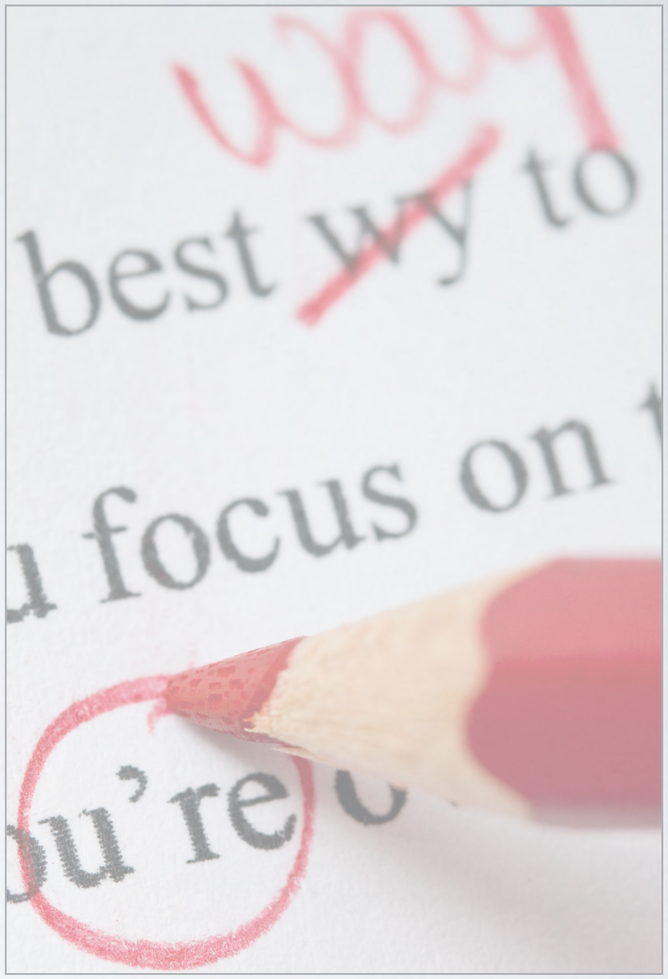
automatic risk detection

My research: Data-supported code review

recommender for reviewers

change untangler

automatic risk detection



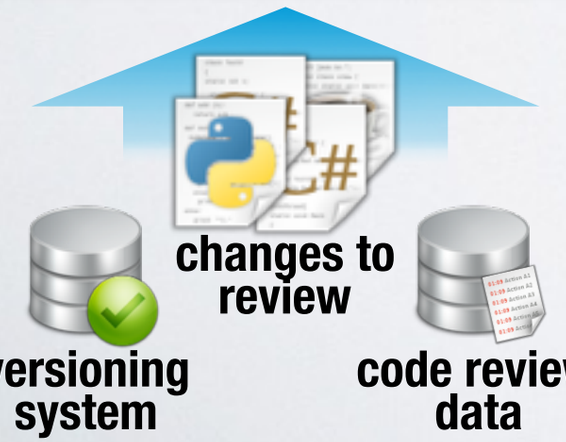
Who should review my code changes?

recommender for reviewers

most appropriate reviewer #1

most appropriate reviewer #3

most appropriate reviewer #2



changes to review

versioning system

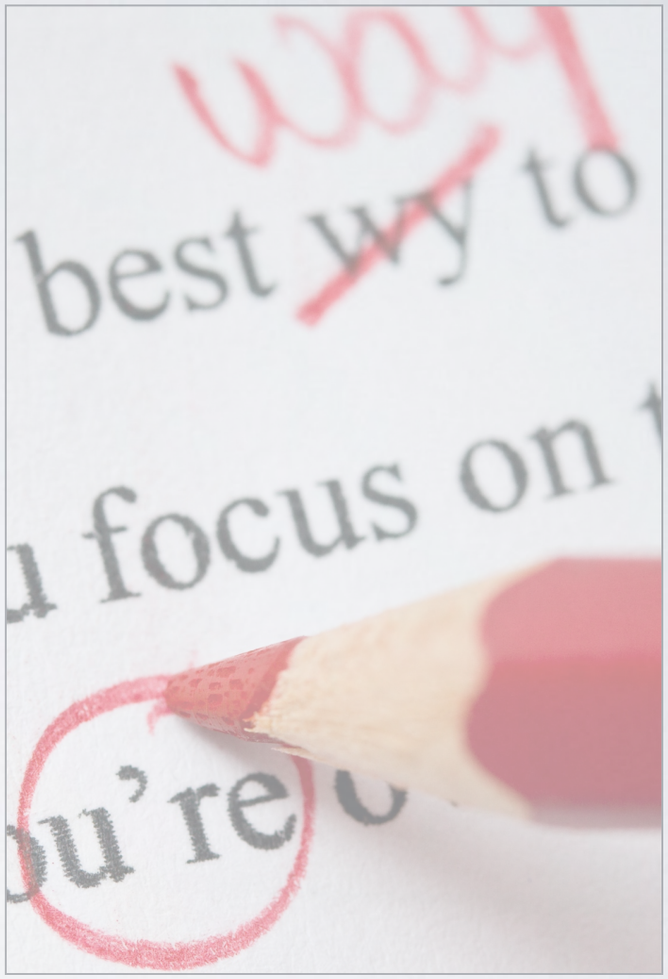
code review data

My research: Data-supported code review

recommender for reviewers

change untangler

automatic risk detection



How should I split my code for easier review?

changes to review

change untangler

versioning system

self-contained change

self-contained change

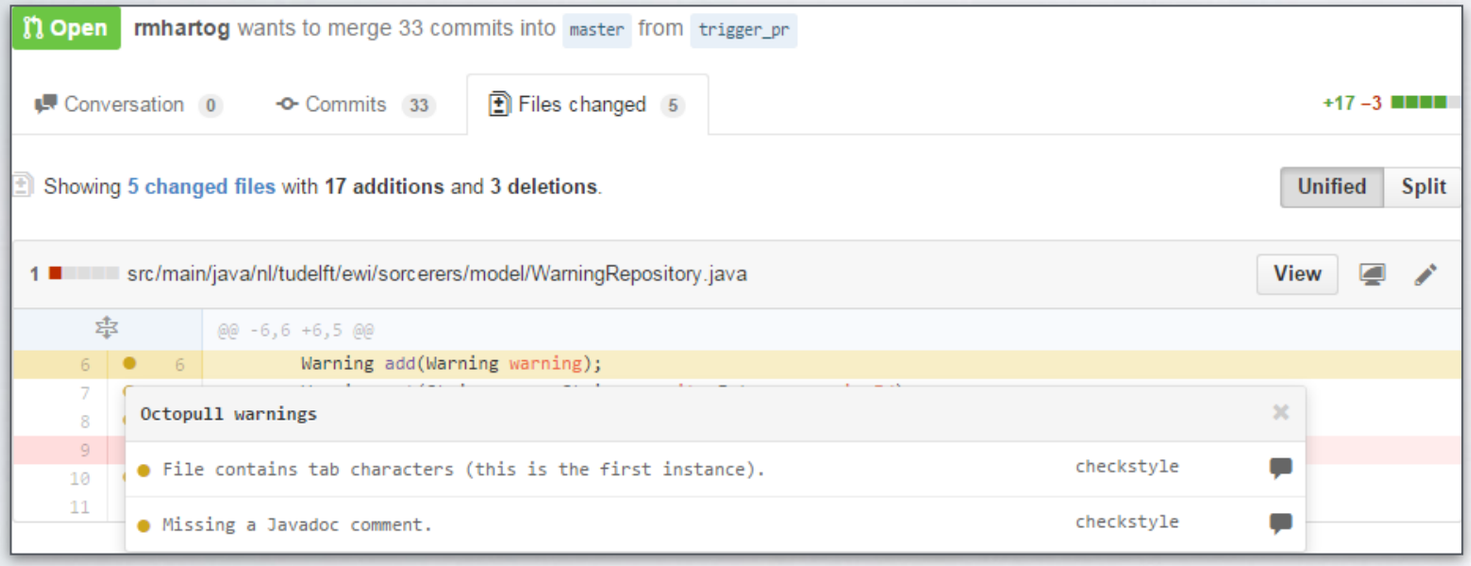
My research: Data-supported code review

recommender for reviewers

change untangler

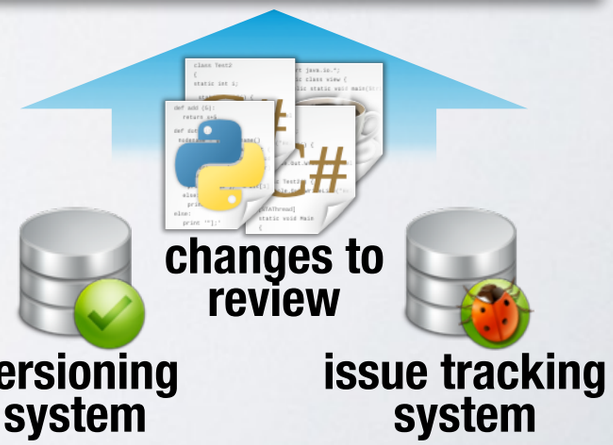
automatic risk detection

Which changes should I review more carefully?



Octopull

automatic risk detection



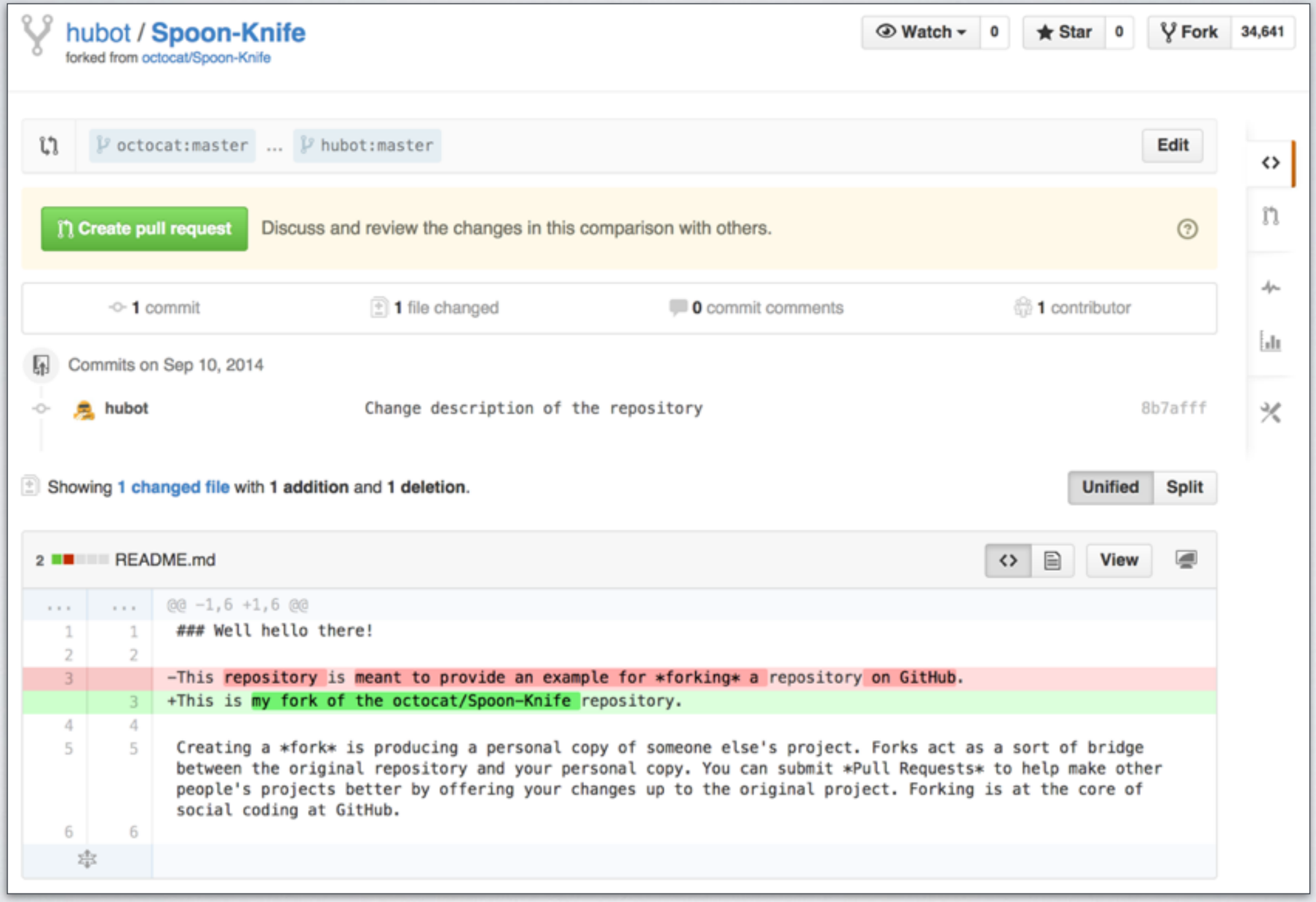
My research: Data-supported code review

recommender for reviewers

change untangler

automatic risk detection

Add code review analytics support to GitHub



Software analytics for code reviews

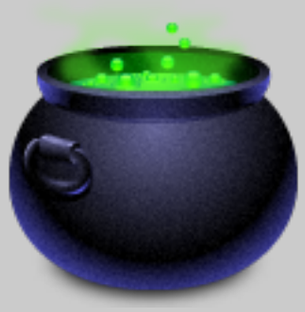
software engineering tasks helped

code review

data mining and software analysis techniques

classification

patterns



clustering

...

software data

source code

versioning system

issue tracking system

review data

...

Alberto Bacchelli
Delft University of Technology
The Netherlands

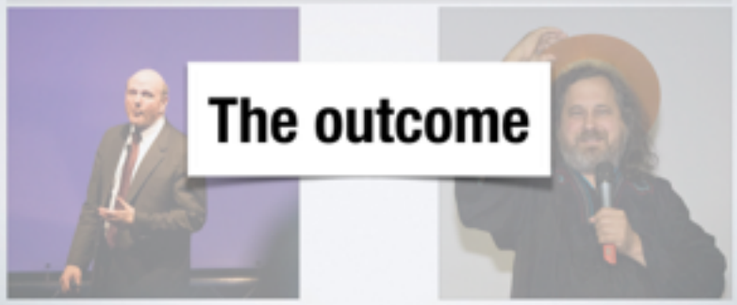
What Do Code Reviews at Microsoft and in Open Source Projects Have in Common?


Why research on modern code review?

Modern Code Review @ Microsoft

Code review at Microsoft: Expectations vs. Reality

What Do Code Reviews at Microsoft and in Open Source Projects Have in Common?



Code review is (still) a fully manual task

My research: Data-supported code review

recommender for reviewers

change untangler

automatic risk detection

Would you like to work on these topics (with me)?

We have 3 fully funded 4-year PhD (or postdoc) positions!

And we are always looking for great students to work on fantastic Master theses!

Find me at the end of the talk, at a.bacchelli@tudelft.nl , or at @sback_

Alberto Bacchelli
Delft University of Technology
The Netherlands

What Do Code Reviews at Microsoft and in Open Source Projects Have in Common?


Why research on modern code review?

Modern Code Review @ Microsoft

Code review at Microsoft: Expectations vs. Reality

What Do Code Reviews at Microsoft and in Open Source Projects Have in Common?

Code review is (still) a fully manual task

My research: Data-supported code review

recommender for reviewers

change untangler

automatic risk detection

Would you like to work on these topics (with me)?

We have 3 fully funded 4-year PhD (or postdoc) positions!

And we are always looking for great students to work on fantastic Master theses!

Find me at the end of the talk, at a.bacchelli@tudelft.nl , or at @sback_