QA Process part 2, Inspections/Code Review

Claire Le Goues

November 19, 2020

Administrivia

- No lecture next week.

Learning Goals

- Overview of concepts how to enforce QA techniques in a process
- Select when and how to integrate tools and policies into the process: daily builds, continuous integration, test automation, static analysis, issue tracking, …
- Understand human and social challenges of adopting QA techniques
- Understand how process and tool improvement can solve the dilemma between features and quality
- Understand different forms of peer reviews with different formality levels.
- Engage in constructive modern code review using a typical commit review system.
- Describe the benefits and properties of good checklists in code review.
- Mitigate social and cultural issues in code review.
- Contrast motivations for and benefits of commit review at modern tech companies.

2010: Agile

- Web-based services and C++ evolution requires faster iteration
- Embrace of agile methods
- Massive reduction of testing team (from two testers per developers toward one): developers now expected to do their own testing

Annotation

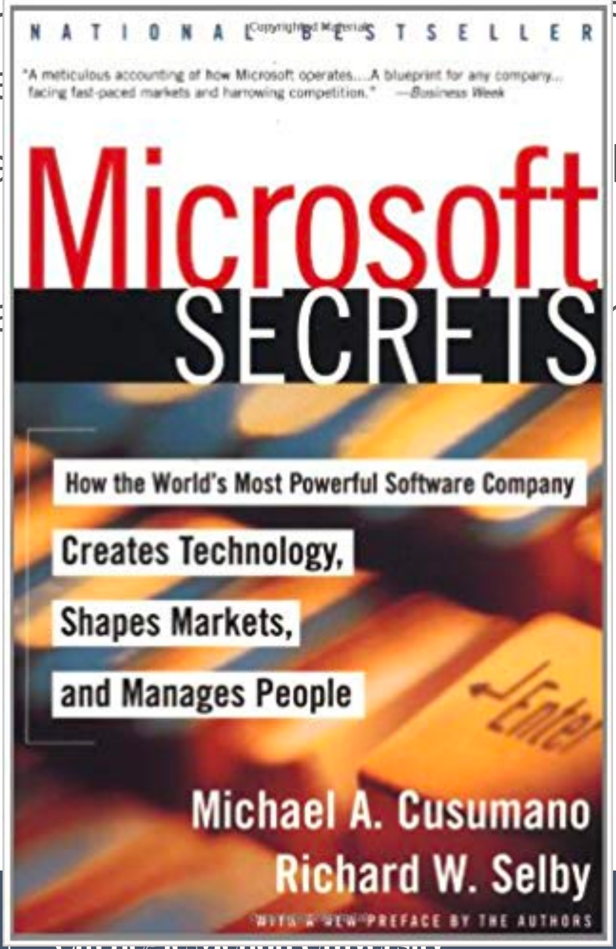
- How to motivate developers, especially with millions of lines of unannotated code?
- Microsoft approach:
    - Require annotations at checkin (e.g., Reject code that has a char* with no __ecount())
    - Make annotations natural, like what you would put in a comment anyway
        - But now machine checkable
        - Avoid formality with poor match to engineering practices
    - Incrementality
        - Check code ↔ design consistency on every compile
        - Rewards programmers for each increment of effort
            - Provide benefit for annotating partial code
            - Can focus on most important parts of the code first
            - Avoid excuse: I'll do it after the deadline
    - Build tools to infer annotations
        - Inference is approximate and so annotations may need to be changed, but saves work overall.
        - Unfortunately not yet available outside Microsoft

Bug prediction

- Metrics
- Mining software repositories
- Example results:
  - Distributed development not critical, but organizational distance is
- Now prioritizing testing effort

Case Study: Microsoft

- Microsoft plans software in features
- 3-4 mile
- After ea hich features should still be implem
- Stabiliz nilestone



NATIONAL BESTSELLER

Copyrighted Material

"A meticulous accounting of how Microsoft operates....A blueprint for any company facing fast-paced markets and harrowing competition." —*Business Week*

Microsoft SECRETS

How the World's Most Powerful Software Company Creates Technology, Shapes Markets, and Manages People

Michael A. Cusumano

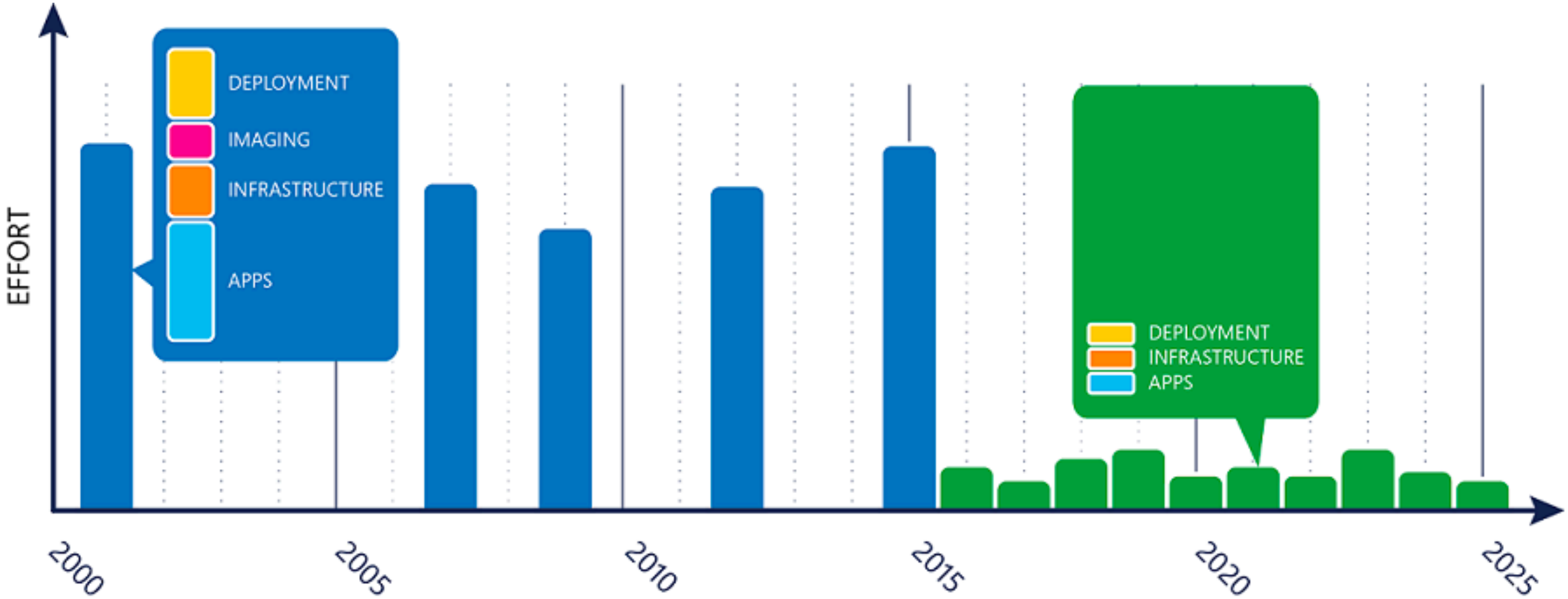
Richard W. Selby

WITH A NEW PREFACE BY THE AUTHORS

Cusumano and Selby. Microsoft Secrets.

Prepare servicing strategy for Windows 10 updates

07/26/2017 • 6 minutes to read • Contributors



QA WITHIN THE PROCESS

QA Process Considerations

- We covered several QA techniques:
  - Formal verification (15-112)
  - Unit testing, test driven development
  - Various forms of advanced testing for quality attributes (GUI testing, fuzz testing, …)
  - Static analysis
  - Dynamic analysis
  - Formal inspections and other forms of code reviews
- But: When to use? Which techniques? How much? How to introduce? Automation? How to establish a quality culture? How to ensure compliance? Social issues? What about external components?

Qualities and Risks, tradeoffs

- What qualities are required? (requirements engineering)
- What risks are expected?

- Align QA strategy based on qualities and risks

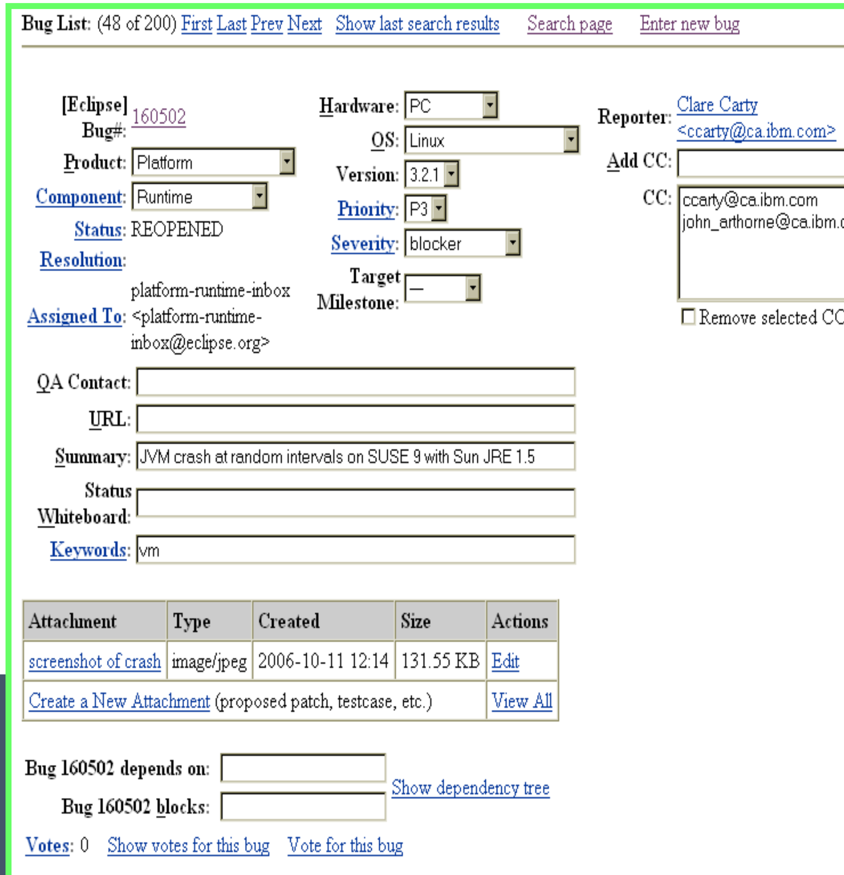
- Understand limitations of QA approaches
  - e.g. testing vs static analysis,
    formal verification vs inspection, …
- Mix and match techniques for different qualities

QA as part of the process

- Have QA deliverables at milestones (management policy)
  - Inspection / test report before milestone
- Change development practices (req. developer buy-in)
  - e.g., continuous integration, pair programming, reviewed checkins, zero-bug static analysis before checking
- Static analysis part of code review (Google)
- Track bugs and other quality metrics

Defect tracking

- Issues: Bug, feature request, query
- Basis for measurement
  - reported in which phase
  - duration to repair, difficulty
  - categorization
    -> root cause analysis
- Facilitates communication
  - questions back to reporter
  - ensures reports are not forgotten
- Accountability



Bug List: (48 of 200) First Last Prev Next Show last search results Search page Enter new bug

[Eclipse] Bug#: 160502
Product: Platform
Component: Runtime
Status: REOPENED
Resolution:
Assigned To: platform-runtime-inbox <platform-runtime-inbox@eclipse.org>

Hardware: PC
OS: Linux
Version: 3.2.1
Priority: P3
Severity: blocker
Target Milestone: --

Reporter: Clare Carty <ccarty@ca.ibm.com>
Add CC:
CC: ccarty@ca.ibm.com
john_arthorne@ca.ibm.c
□ Remove selected CC

QA Contact:
URL:
Summary: JVM crash at random intervals on SUSE 9 with Sun JRE 1.5
Status Whiteboard:
Keywords: vm

Attachment	Type	Created	Size	Actions
screenshot of crash	image/jpeg	2006-10-11 12:14	131.55 KB	Edit
Create a New Attachment (proposed patch, testcase, etc.)				View All

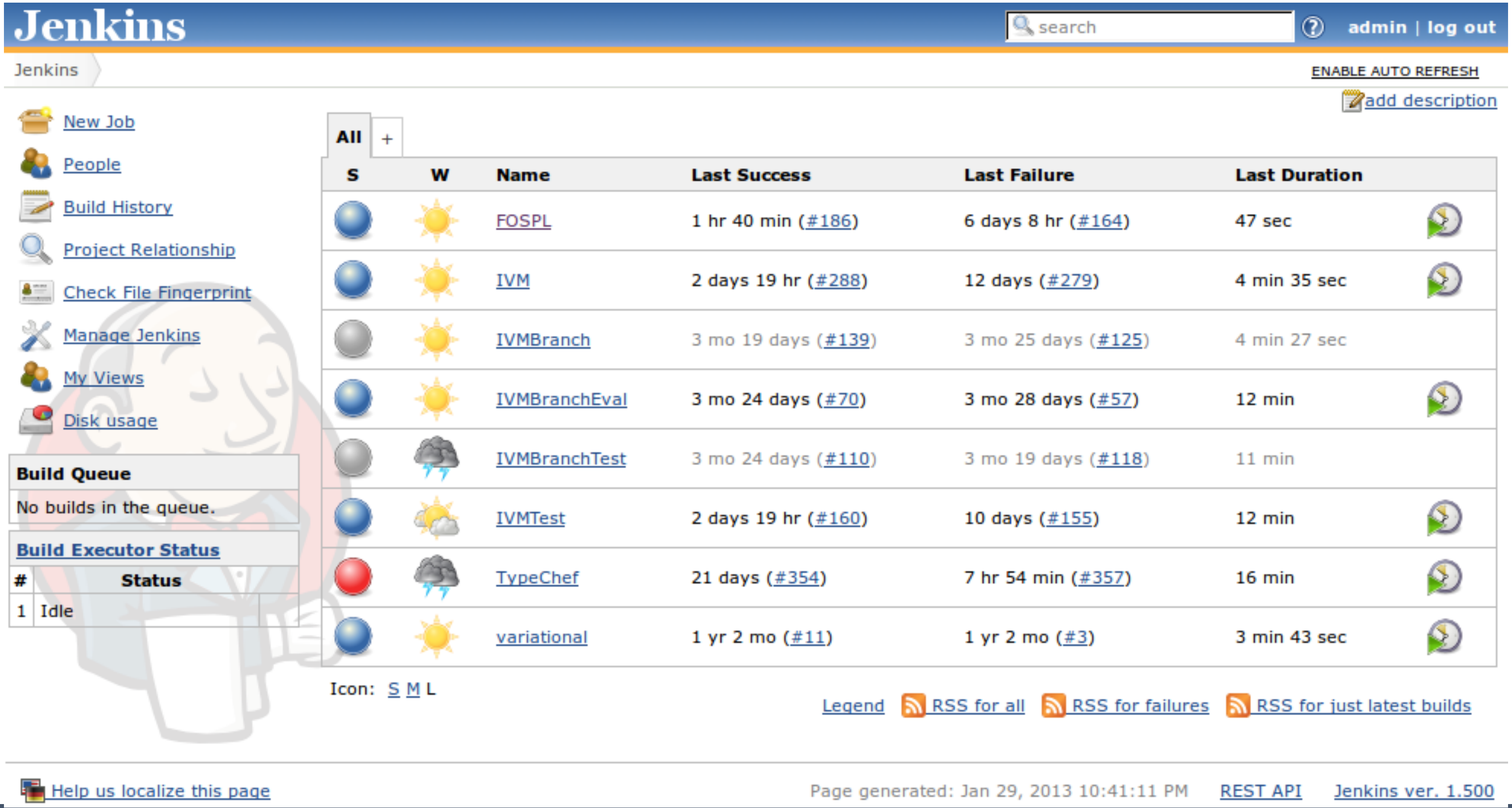
Bug 160502 depends on:
Bug 160502 blocks:
Show dependency tree

Votes: 0 Show votes for this bug Vote for this bug

Enforcement

- Microsoft: check in gates
  - Cannot check in code unless analysis suite has been run and produced no errors (test coverage, dependency violation, insufficient/bad design intent, integer overflow, allocation arithmetic, buffer overruns, memory errors, security issues)
- eBay: dev/QA handoff
  - Developers run FindBugs on desktop
  - QA runs FindBugs on receipt of code, posts results, require high-priority fixes.
- Google: static analysis on commits, shown in review
- Requirements for success
  - Low false positives
  - A way to override false positive warnings (typically through inspection).
  - Developers must buy into static analysis first

Reminder: Continuous Integration



Jenkins

search | admin | log out

Jenkins

ENABLE AUTO REFRESH

add description

- New Job
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views
- Disk usage

Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle

All | +

S	W	Name	Last Success	Last Failure	Last Duration
🔵	☀	FOSPL	1 hr 40 min (#186)	6 days 8 hr (#164)	47 sec
🔵	☀	IVM	2 days 19 hr (#288)	12 days (#279)	4 min 35 sec
⚪	☀	IVMBranch	3 mo 19 days (#139)	3 mo 25 days (#125)	4 min 27 sec
🔵	☀	IVMBranchEval	3 mo 24 days (#70)	3 mo 28 days (#57)	12 min
⚪	⛈	IVMBranchTest	3 mo 24 days (#110)	3 mo 19 days (#118)	11 min
🔵	⛅	IVMTest	2 days 19 hr (#160)	10 days (#155)	12 min
🔴	⛈	TypeChef	21 days (#354)	7 hr 54 min (#357)	16 min
🔵	☀	variational	1 yr 2 mo (#11)	1 yr 2 mo (#3)	3 min 43 sec

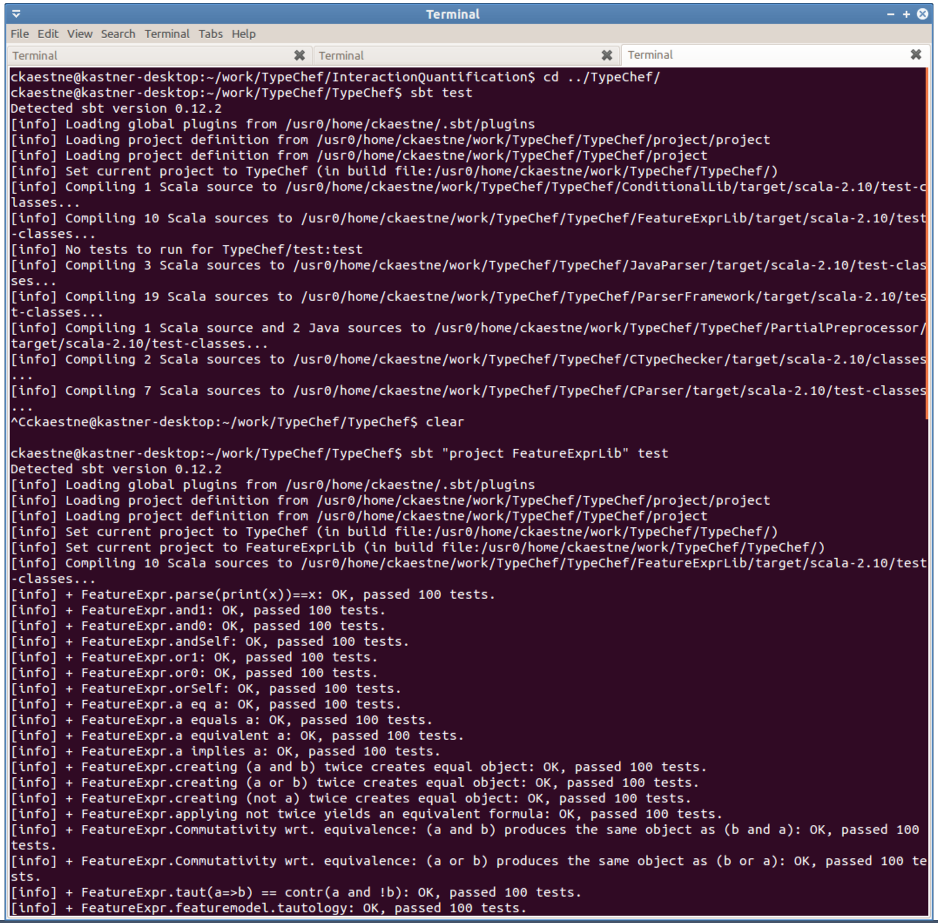
Icon: S M L

Legend | RSS for all | RSS for failures | RSS for just latest builds

Help us localize this page

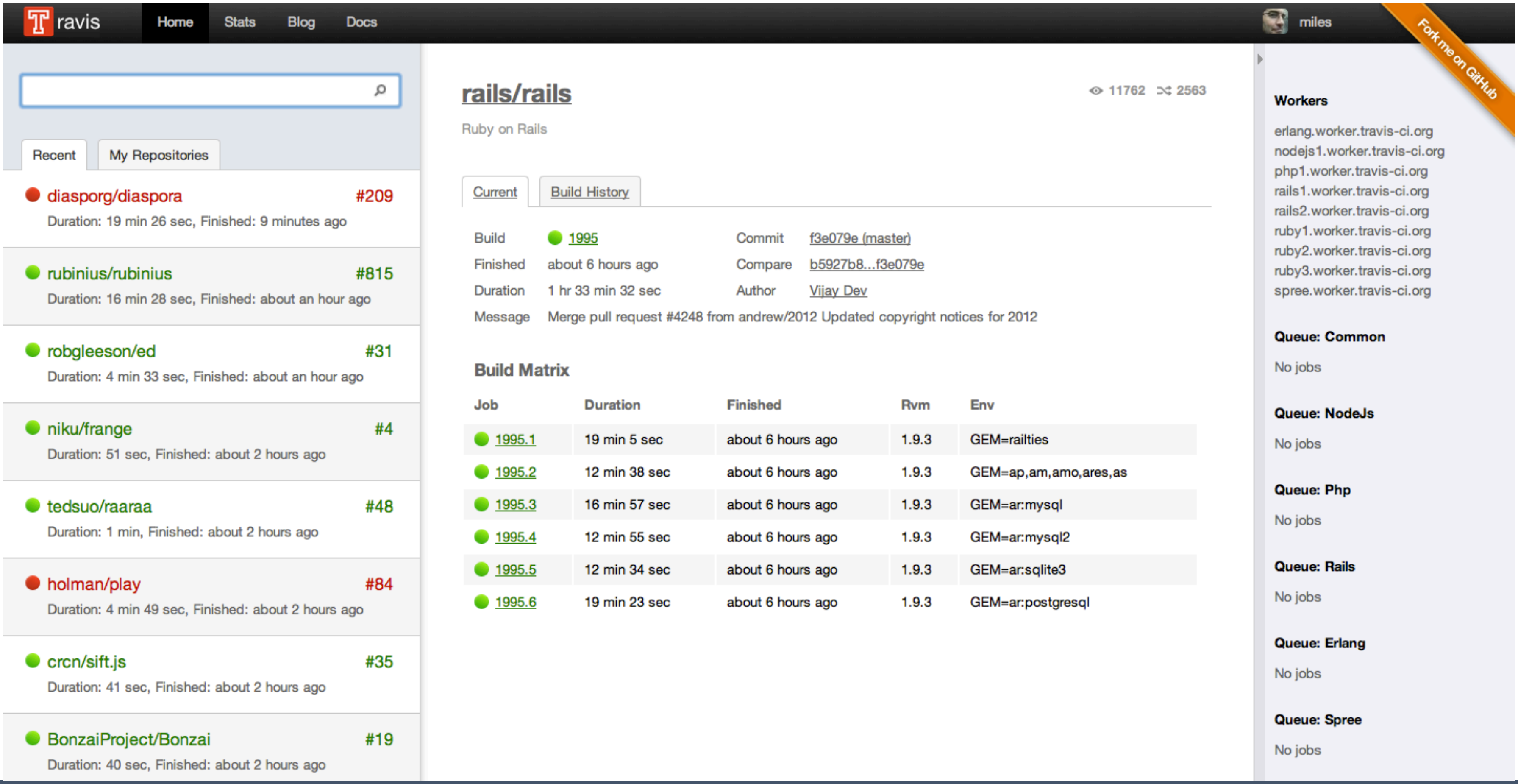
Page generated: Jan 29, 2013 10:41:11 PM | REST API | Jenkins ver. 1.500

Automating Test Execution



```
ckaestne@kastner-desktop:~/work/TypeChef/InteractionQuantification$ cd ../TypeChef/
ckaestne@kastner-desktop:~/work/TypeChef/TypeChef$ sbt test
Detected sbt version 0.12.2
[info] Loading global plugins from /usr0/home/ckaestne/.sbt/plugins
[info] Loading project definition from /usr0/home/ckaestne/work/TypeChef/TypeChef/project/project
[info] Loading project definition from /usr0/home/ckaestne/work/TypeChef/TypeChef/project
[info] Set current project to TypeChef (in build file:/usr0/home/ckaestne/work/TypeChef/TypeChef/)
[info] Compiling 1 Scala source to /usr0/home/ckaestne/work/TypeChef/TypeChef/ConditionalLib/target/scala-2.10/test-c
lasses...
[info] Compiling 10 Scala sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/FeatureExprLib/target/scala-2.10/test
-classes...
[info] No tests to run for TypeChef/test:test
[info] Compiling 3 Scala sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/JavaParser/target/scala-2.10/test-clas
ses...
[info] Compiling 19 Scala sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/ParserFramework/target/scala-2.10/tes
t-classes...
[info] Compiling 1 Scala source and 2 Java sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/PartialPreprocessor/
target/scala-2.10/test-classes...
[info] Compiling 2 Scala sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/CTypeChecker/target/scala-2.10/classes
...
[info] Compiling 7 Scala sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/CParser/target/scala-2.10/test-classes
...
^Cckaestne@kastner-desktop:~/work/TypeChef/TypeChef$ clear

ckaestne@kastner-desktop:~/work/TypeChef/TypeChef$ sbt "project FeatureExprLib" test
Detected sbt version 0.12.2
[info] Loading global plugins from /usr0/home/ckaestne/.sbt/plugins
[info] Loading project definition from /usr0/home/ckaestne/work/TypeChef/TypeChef/project/project
[info] Loading project definition from /usr0/home/ckaestne/work/TypeChef/TypeChef/project
[info] Set current project to TypeChef (in build file:/usr0/home/ckaestne/work/TypeChef/TypeChef/)
[info] Set current project to FeatureExprLib (in build file:/usr0/home/ckaestne/work/TypeChef/TypeChef/)
[info] Compiling 10 Scala sources to /usr0/home/ckaestne/work/TypeChef/TypeChef/FeatureExprLib/target/scala-2.10/test
-classes...
[info] + FeatureExpr.parse(print(x))==x: OK, passed 100 tests.
[info] + FeatureExpr.and1: OK, passed 100 tests.
[info] + FeatureExpr.and0: OK, passed 100 tests.
[info] + FeatureExpr.andSelf: OK, passed 100 tests.
[info] + FeatureExpr.or1: OK, passed 100 tests.
[info] + FeatureExpr.or0: OK, passed 100 tests.
[info] + FeatureExpr.orSelf: OK, passed 100 tests.
[info] + FeatureExpr.a eq a: OK, passed 100 tests.
[info] + FeatureExpr.a equals a: OK, passed 100 tests.
[info] + FeatureExpr.a equivalent a: OK, passed 100 tests.
[info] + FeatureExpr.a implies a: OK, passed 100 tests.
[info] + FeatureExpr.creating (a and b) twice creates equal object: OK, passed 100 tests.
[info] + FeatureExpr.creating (a or b) twice creates equal object: OK, passed 100 tests.
[info] + FeatureExpr.creating (not a) twice creates equal object: OK, passed 100 tests.
[info] + FeatureExpr.applying not twice yields an equivalent formula: OK, passed 100 tests.
[info] + FeatureExpr.Commutativity wrt. equivalence: (a and b) produces the same object as (b and a): OK, passed 100
tests.
[info] + FeatureExpr.Commutativity wrt. equivalence: (a or b) produces the same object as (b or a): OK, passed 100 te
sts.
[info] + FeatureExpr.taut(a=>b) == contr(a and !b): OK, passed 100 tests.
[info] + FeatureExpr.featuremodel.tautology: OK, passed 100 tests.
```

Continuous Integration with Travis-CI



● diasporg/diaspora #209
Duration: 19 min 26 sec, Finished: 9 minutes ago

● rubinius/rubinius #815
Duration: 16 min 28 sec, Finished: about an hour ago

● robgleeson/ed #31
Duration: 4 min 33 sec, Finished: about an hour ago

● niku/frange #4
Duration: 51 sec, Finished: about 2 hours ago

● tedsuo/raaraa #48
Duration: 1 min, Finished: about 2 hours ago

● holman/play #84
Duration: 4 min 49 sec, Finished: about 2 hours ago

● crcn/sift.js #35
Duration: 41 sec, Finished: about 2 hours ago

● BonzaiProject/Bonzai #19
Duration: 40 sec, Finished: about 2 hours ago

rails/rails 11762 2563

Ruby on Rails

Build	● 1995	Commit	f3e079e (master)
Finished	about 6 hours ago	Compare	b5927b8...f3e079e
Duration	1 hr 33 min 32 sec	Author	Vijay Dev
Message	Merge pull request #4248 from andrew/2012 Updated copyright notices for 2012		

Build Matrix

Job	Duration	Finished	Rvm	Env
● 1995.1	19 min 5 sec	about 6 hours ago	1.9.3	GEM=railties
● 1995.2	12 min 38 sec	about 6 hours ago	1.9.3	GEM=ap,am,amo,ares,as
● 1995.3	16 min 57 sec	about 6 hours ago	1.9.3	GEM=ar:mysql
● 1995.4	12 min 55 sec	about 6 hours ago	1.9.3	GEM=ar:mysql2
● 1995.5	12 min 34 sec	about 6 hours ago	1.9.3	GEM=ar:sqlite3
● 1995.6	19 min 23 sec	about 6 hours ago	1.9.3	GEM=ar:postgresql

Workers

erlang.worker.travis-ci.org
nodejs1.worker.travis-ci.org
php1.worker.travis-ci.org
rails1.worker.travis-ci.org
rails2.worker.travis-ci.org
ruby1.worker.travis-ci.org
ruby2.worker.travis-ci.org
ruby3.worker.travis-ci.org
spree.worker.travis-ci.org

Queue: Common

No jobs

Queue: NodeJs

No jobs

Queue: Php

No jobs

Queue: Rails

No jobs

Queue: Erlang

No jobs

Queue: Spree

No jobs

SOCIAL ASPECTS

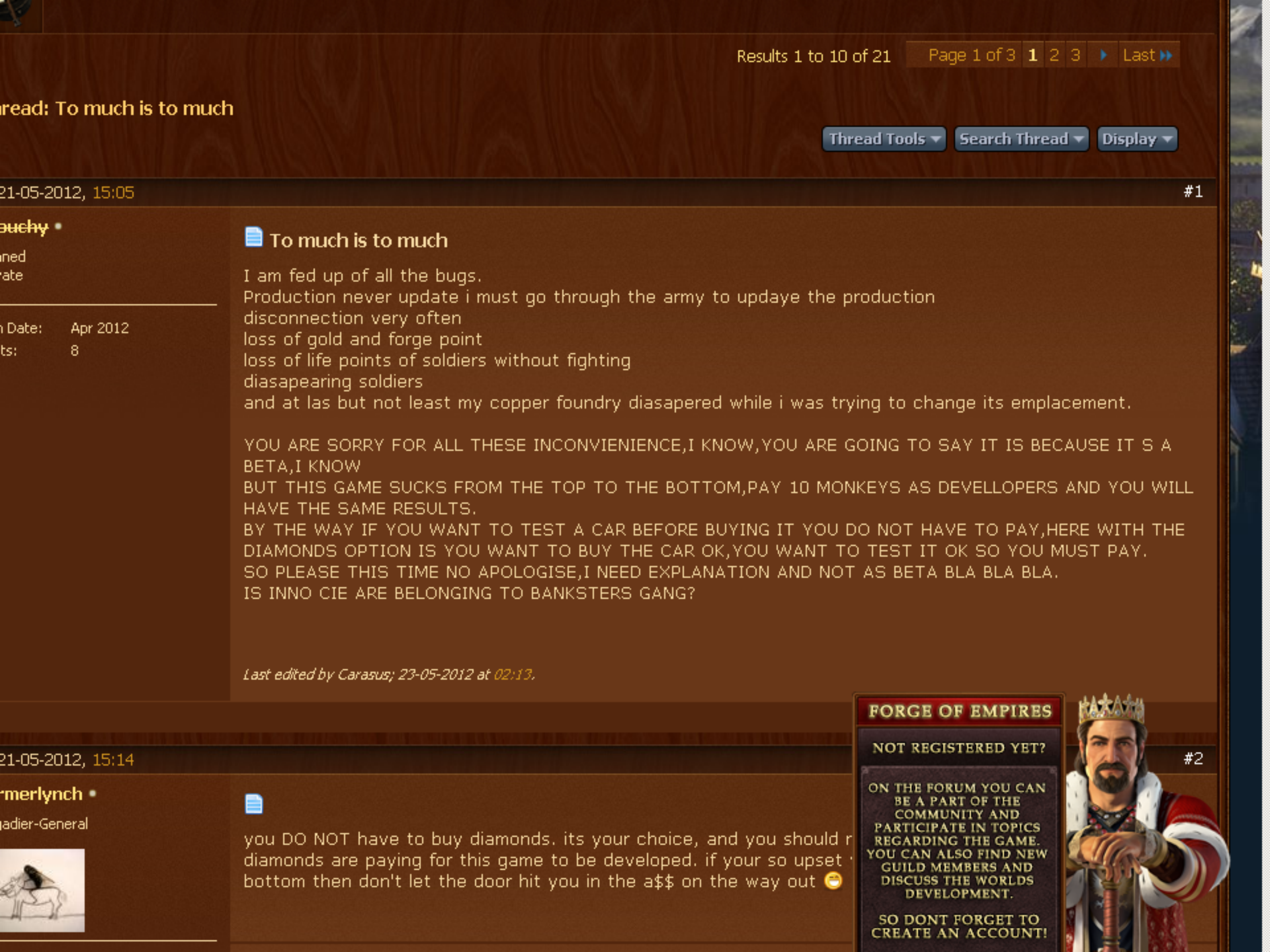
Social issues

- Developer attitude toward defects
- Developer education about security
- Using peer pressure to enforce QA practices
  - Breaking the build – various rules

- Developer vs tester culture
  - Testers tend to deliver bad news
- Defects in performance evaluations?
- Issues vs defects
- Good test suits raise confidence, encourage shared code ownership

Reporting Defects

- Reproducible defects
- Simple and general
- One defect per report
- Non-antagonistic
  - (testers usually bring bad news)
  - State the problem
  - Don't blame



Results 1 to 10 of 21 Page 1 of 3 **1** 2 3 Last

read: To much is to much

21-05-2012, 15:05 #1

~~auchy~~

ned
rate

n Date: Apr 2012
ts: 8

To much is to much

I am fed up of all the bugs.
Production never update i must go through the army to updaye the production
disconnection very often
loss of gold and forge point
loss of life points of soldiers without fighting
diasapearing soldiers
and at las but not least my copper foundry diasapered while i was trying to change its emplacement.

YOU ARE SORRY FOR ALL THESE INCONVIENIENCE,I KNOW,YOU ARE GOING TO SAY IT IS BECAUSE IT S A BETA,I KNOW
BUT THIS GAME SUCKS FROM THE TOP TO THE BOTTOM,PAY 10 MONKEYS AS DEVELLOPERS AND YOU WILL HAVE THE SAME RESULTS.
BY THE WAY IF YOU WANT TO TEST A CAR BEFORE BUYING IT YOU DO NOT HAVE TO PAY,HERE WITH THE DIAMONDS OPTION IS YOU WANT TO BUY THE CAR OK,YOU WANT TO TEST IT OK SO YOU MUST PAY.
SO PLEASE THIS TIME NO APOLOGISE,I NEED EXPLANATION AND NOT AS BETA BLA BLA BLA.
IS INNO CIE ARE BELONGING TO BANKSTERS GANG?

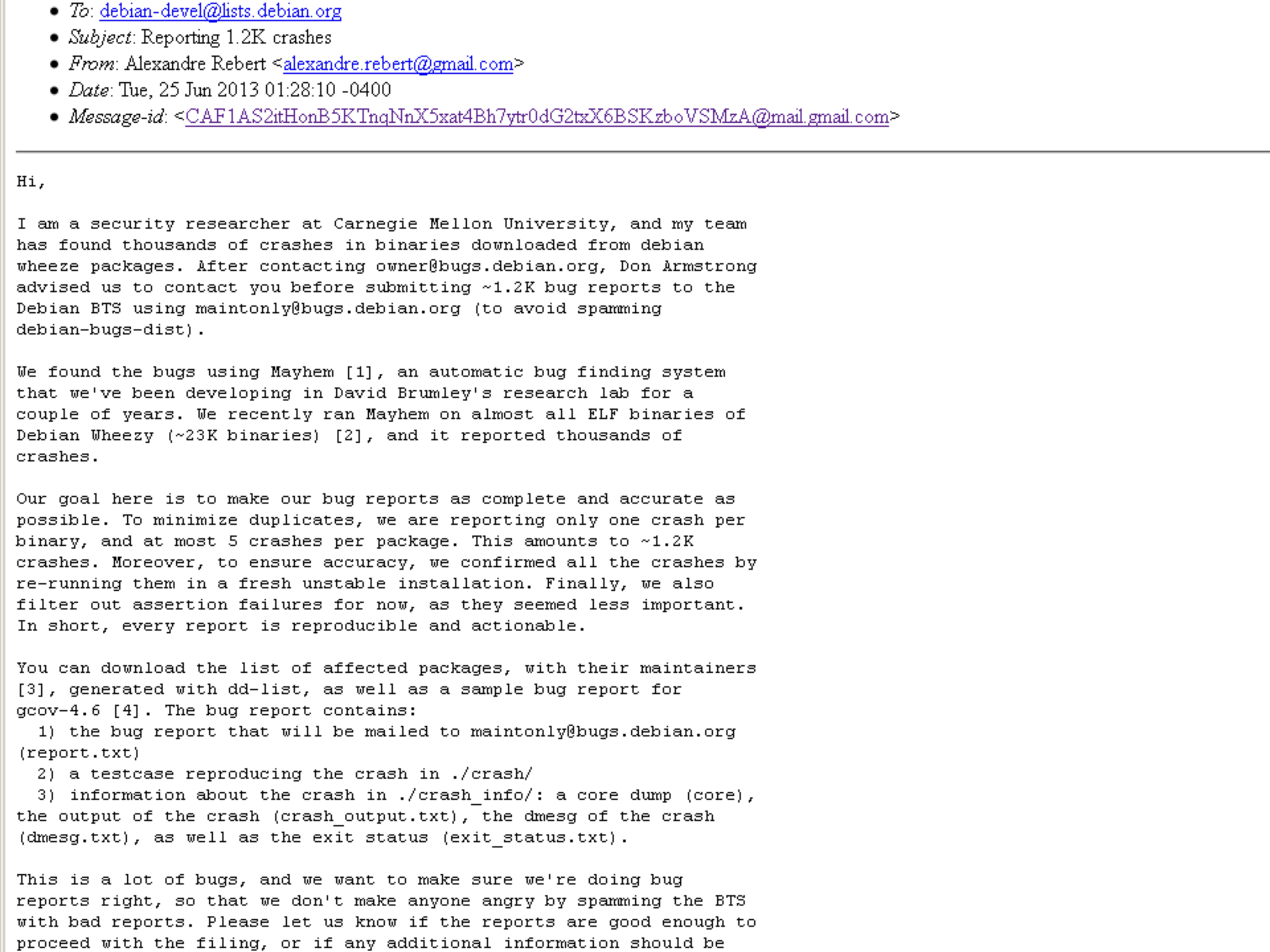
Last edited by Carasus; 23-05-2012 at 02:13.

21-05-2012, 15:14 #2

rmerlynch

adier-General

you DO NOT have to buy diamonds. its your choice, and you should r
diamonds are paying for this game to be developed. if your so upset
bottom then don't let the door hit you in the a$$ on the way out 😁



- *To*: debian-devel@lists.debian.org
- *Subject*: Reporting 1.2K crashes
- *From*: Alexandre Rebert <alexandre.rebert@gmail.com>
- *Date*: Tue, 25 Jun 2013 01:28:10 -0400
- *Message-id*: <CAF1AS2itHonB5KTnqNnX5xat4Bh7ytr0dG2txX6BSKzboVSMzA@mail.gmail.com>

Hi,

I am a security researcher at Carnegie Mellon University, and my team has found thousands of crashes in binaries downloaded from debian wheeze packages. After contacting owner@bugs.debian.org, Don Armstrong advised us to contact you before submitting ~1.2K bug reports to the Debian BTS using maintonly@bugs.debian.org (to avoid spamming debian-bugs-dist).

We found the bugs using Mayhem [1], an automatic bug finding system that we've been developing in David Brumley's research lab for a couple of years. We recently ran Mayhem on almost all ELF binaries of Debian Wheezy (~23K binaries) [2], and it reported thousands of crashes.

Our goal here is to make our bug reports as complete and accurate as possible. To minimize duplicates, we are reporting only one crash per binary, and at most 5 crashes per package. This amounts to ~1.2K crashes. Moreover, to ensure accuracy, we confirmed all the crashes by re-running them in a fresh unstable installation. Finally, we also filter out assertion failures for now, as they seemed less important. In short, every report is reproducible and actionable.

You can download the list of affected packages, with their maintainers [3], generated with dd-list, as well as a sample bug report for gcov-4.6 [4]. The bug report contains:
  1) the bug report that will be mailed to maintonly@bugs.debian.org (report.txt)
  2) a testcase reproducing the crash in ./crash/
  3) information about the crash in ./crash_info/: a core dump (core), the output of the crash (crash_output.txt), the dmesg of the crash (dmesg.txt), as well as the exit status (exit_status.txt).

This is a lot of bugs, and we want to make sure we're doing bug reports right, so that we don't make anyone angry by spamming the BTS with bad reports. Please let us know if the reports are good enough to proceed with the filing, or if any additional information should be

Code Reviews and Inspection

“Many eyes make all bugs shallow”

Standard Refrain in Open Source

“Have peers, rather than customers, find defects”

Karl Wiegers

Isn’t testing sufficient?

- Errors can mask other errors
- Only completed implementations can be tested (esp. scalability, performance)
- Design documents cannot be tested
- Tests don’t check code quality
- Many quality attributes (eg., security, compliance, scalability) are difficult to test

A second pair of eyes

- Different background, different experience
- No preconceived idea of correctness
- Not biased by “what was intended”

FORMAL INSPECTIONS

Formal Inspections

- Idea popularized in 70s at IBM
- Broadly adopted in 80s, much research
  - Sometimes replacing component testing
- Group of developers meets to formally review code or other artifacts
- Most effective approach to find bugs
  - Typically 60-90% of bugs found with inspections
- Expensive and labor-intensive

(see textbook Chapter 22.2)

Inspection Team and Roles

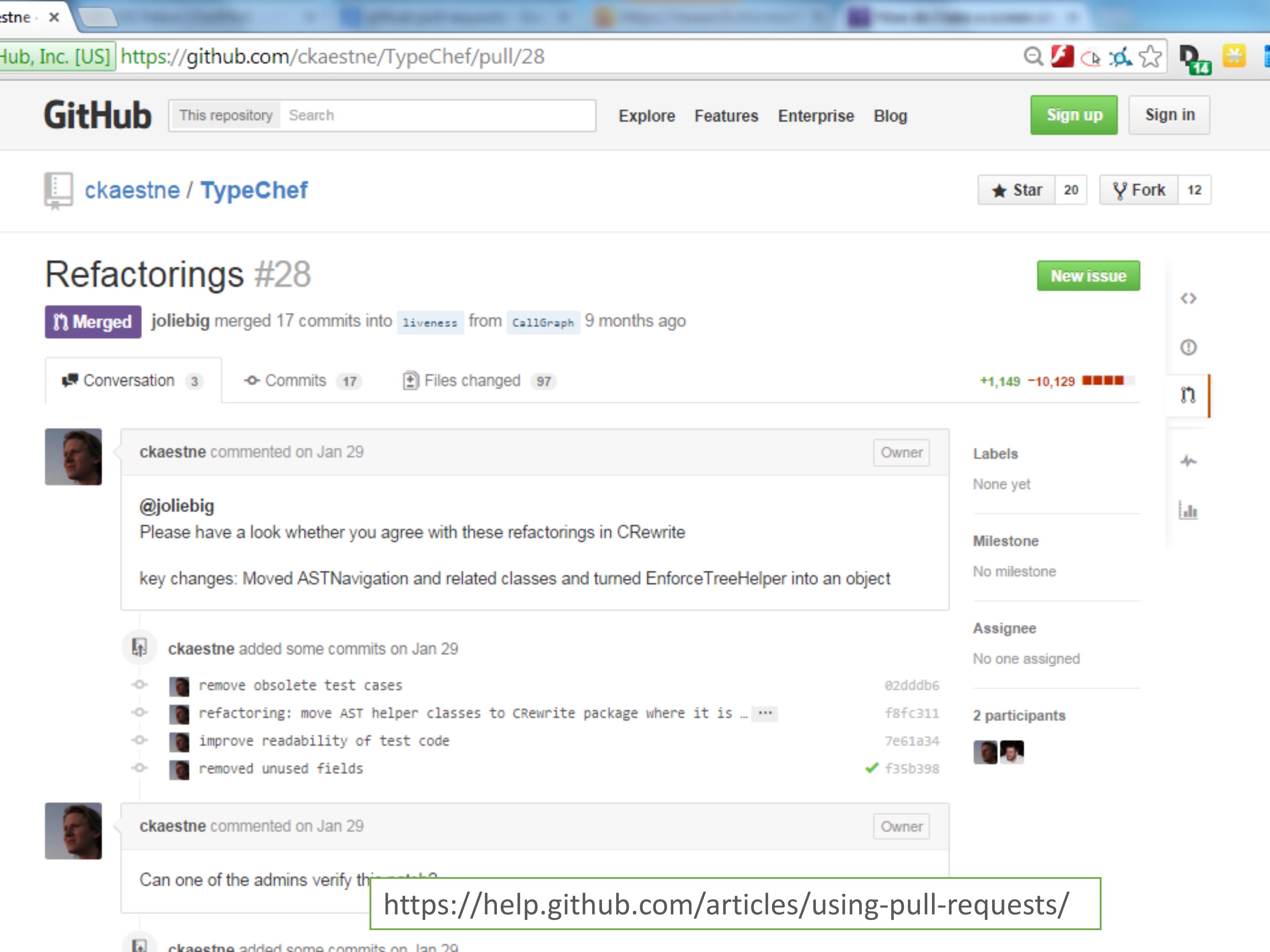
- Typically 4-5 people (min 3)
- Author
- Inspector(s)
  - Find faults and broader issues
- Reader
  - Presents the code or document at inspection meeting
- Scribe
  - Records results
- Moderator
  - Manages process, facilitates, reports

Checklists

- Reminder what to look for
- Include issues detected in the past
- Preferably focus on few important items
- Examples:
    - Are all variables initialized before use?
    - Are all variables used?
    - Is the condition of each if/while statement correct?
    - Does each loop terminate?
    - Do function parameters have the right types and appear in the right order?
    - Are linked lists efficiently traversed?
    - Is dynamically allocated memory released?
    - Can unexpected inputs cause corruption?
    - Have all possible error conditions been handled?
    - Are strings correctly sanitized?

Process details

- Authors do not explain or defend the code – not objective
  - Author != moderator, != scribe, !=reader
  - Author should still join the meeting to observe questions and misunderstandings and clarify issues if necessary
- Reader (optional) walks through the code line by line, explaining it
  - Reading the code aloud requires deeper understanding
  - Verbalizes interpretations, thus observing differences in interpretation



ckaestne / TypeChef

Refactorings #28

Merged joliebig merged 17 commits into `liveness` from `CallGraph` 9 months ago

ckaestne commented on Jan 29

@joliebig
Please have a look whether you agree with these refactorings in CRewrite

key changes: Moved ASTNavigation and related classes and turned EnforceTreeHelper into an object

ckaestne added some commits on Jan 29

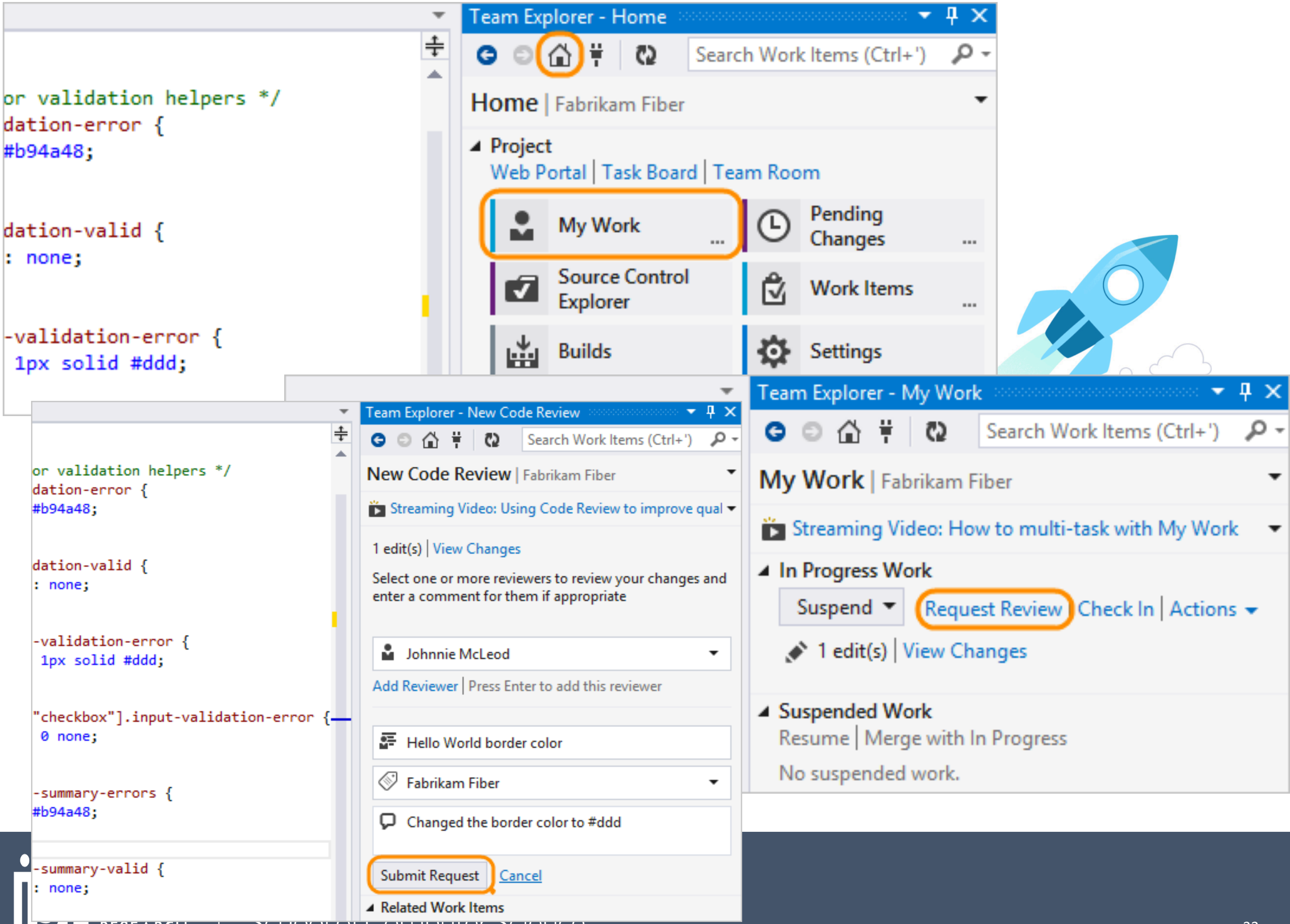
```
remove obsolete test cases                                            02dddb6
refactoring: move AST helper classes to CRewrite package where it is … …   f8fc311
improve readability of test code                                      7e61a34
removed unused fields                                               ✓ f35b398
```

ckaestne commented on Jan 29

Can one of the admins verify th

ckaestne added some commits on Jan 29

https://help.github.com/articles/using-pull-requests/



```
or validation helpers */
dation-error {
#b94a48;

dation-valid {
: none;

-validation-error {
 1px solid #ddd;
```

Team Explorer - Home

Search Work Items (Ctrl+')

Home | Fabrikam Fiber

Project

Web Portal | Task Board | Team Room

- My Work
- Pending Changes
- Source Control Explorer
- Work Items
- Builds
- Settings

```
or validation helpers */
dation-error {
#b94a48;

dation-valid {
: none;

-validation-error {
 1px solid #ddd;

"checkbox"].input-validation-error {
 0 none;

-summary-errors {
#b94a48;

-summary-valid {
: none;
```

Team Explorer - New Code Review

Search Work Items (Ctrl+')

New Code Review | Fabrikam Fiber

Streaming Video: Using Code Review to improve qual

1 edit(s) | View Changes

Select one or more reviewers to review your changes and enter a comment for them if appropriate

Johnnie McLeod

Add Reviewer | Press Enter to add this reviewer

Hello World border color

Fabrikam Fiber

Changed the border color to #ddd

Submit Request Cancel

Related Work Items

Team Explorer - My Work

Search Work Items (Ctrl+')

My Work | Fabrikam Fiber

Streaming Video: How to multi-task with My Work

In Progress Work

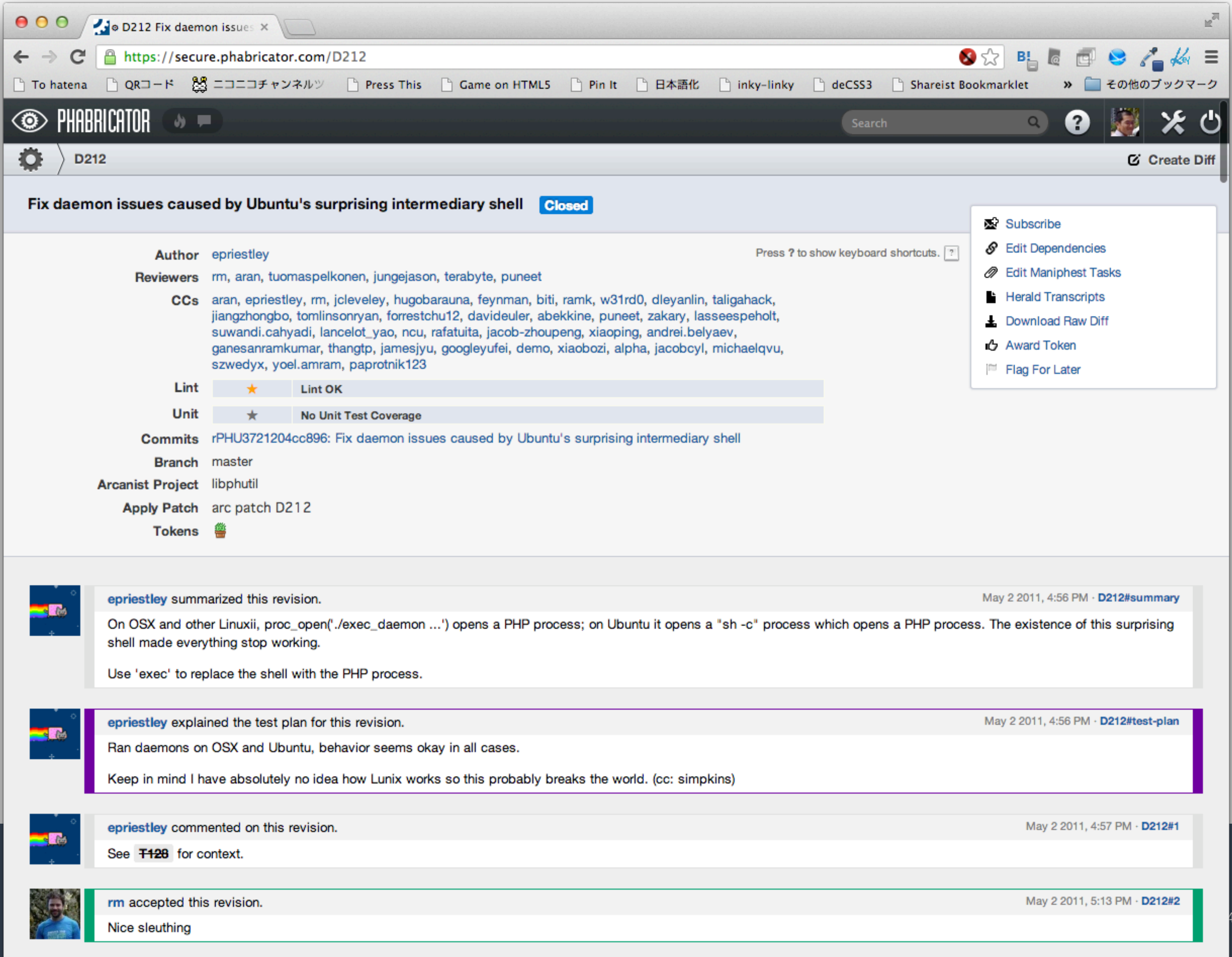
Suspend | Request Review | Check In | Actions

1 edit(s) | View Changes

Suspended Work

Resume | Merge with In Progress

No suspended work.



Fix daemon issues caused by Ubuntu's surprising intermediary shell Closed

Author	epriestley
Reviewers	rm, aran, tuomaspelkonen, jungejason, terabyte, puneet
CCs	aran, epriestley, rm, jcleveley, hugobarauna, feynman, biti, ramk, w31rd0, dleyanlin, taligahack, jiangzhongbo, tomlinsonryan, forrestchu12, davideuler, abekkine, puneet, zakary, lasseespeholt, suwandi.cahyadi, lancelot_yao, ncu, rafatuita, jacob-zhoupeng, xiaoping, andrei.belyaev, ganesanramkumar, thangtp, jamesjyu, googleyufei, demo, xiaobozi, alpha, jacobcyl, michaelqvu, szwedyx, yoel.amram, paprotnik123
Lint	★ Lint OK
Unit	★ No Unit Test Coverage
Commits	rPHU3721204cc896: Fix daemon issues caused by Ubuntu's surprising intermediary shell
Branch	master
Arcanist Project	libphutil
Apply Patch	arc patch D212
Tokens	

epriestley summarized this revision. May 2 2011, 4:56 PM · D212#summary

On OSX and other Linuxii, proc_open('./exec_daemon ...') opens a PHP process; on Ubuntu it opens a "sh -c" process which opens a PHP process. The existence of this surprising shell made everything stop working.

Use 'exec' to replace the shell with the PHP process.

epriestley explained the test plan for this revision. May 2 2011, 4:56 PM · D212#test-plan

Ran daemons on OSX and Ubuntu, behavior seems okay in all cases.

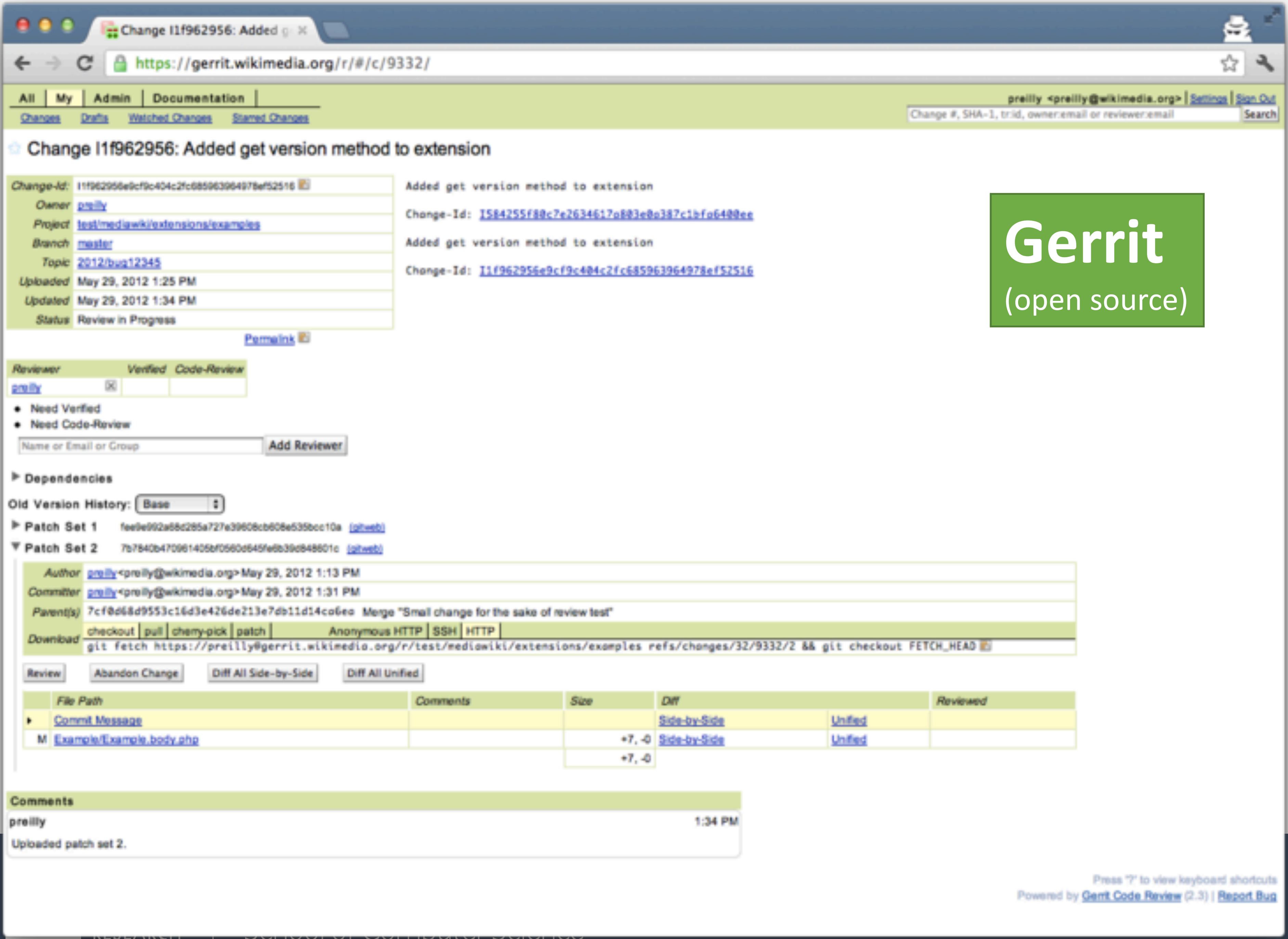
Keep in mind I have absolutely no idea how Lunix works so this probably breaks the world. (cc: simpkins)

epriestley commented on this revision. May 2 2011, 4:57 PM · D212#1

See ~~T128~~ for context.

rm accepted this revision. May 2 2011, 5:13 PM · D212#2

Nice sleuthing



Change I1f962956: Added g
https://gerrit.wikimedia.org/r/#/c/9332/
All | My | Admin | Documentation
Changes Drafts Watched Changes Starred Changes
preilly <preilly@wikimedia.org> | Settings | Sign Out
Change #, SHA-1, tr:id, owner:email or reviewer:email
Search

Change I1f962956: Added get version method to extension

Change-Id:	I1f962956e9cf9c404c2fc685963964978ef52516
Owner	preilly
Project	test/mediawiki/extensions/examples
Branch	master
Topic	2012/bug12345
Uploaded	May 29, 2012 1:25 PM
Updated	May 29, 2012 1:34 PM
Status	Review in Progress

Permalink

```
Added get version method to extension

Change-Id: I584255f80c7e2634617a803e0a387c1bfa6400ee

Added get version method to extension

Change-Id: I1f962956e9cf9c404c2fc685963964978ef52516
```

Reviewer	Verified	Code-Review
preilly ☒		

- Need Verified
- Need Code-Review

▶ Dependencies

Old Version History: Base

▶ Patch Set 1 fee9e992a68d285a727e39608cb608e535bcc10a (gitweb)

▼ Patch Set 2 7b7840b470961405bf0560d645fe6b39d848601c (gitweb)

Author	preilly<preilly@wikimedia.org> May 29, 2012 1:13 PM
Committer	preilly<preilly@wikimedia.org> May 29, 2012 1:31 PM
Parent(s)	7cf0d68d9553c16d3e426de213e7db11d14ca6ea Merge "Small change for the sake of review test"
Download	checkout | pull | cherry-pick | patch Anonymous HTTP | SSH | HTTP

```
git fetch https://preilly@gerrit.wikimedia.org/r/test/mediawiki/extensions/examples refs/changes/32/9332/2 && git checkout FETCH_HEAD
```

	File Path	Comments	Size	Diff		Reviewed
▸	Commit Message			Side-by-Side	Unified	
M	Example/Example.body.php		+7, -0	Side-by-Side	Unified	
			+7, -0			

Comments

preilly 1:34 PM

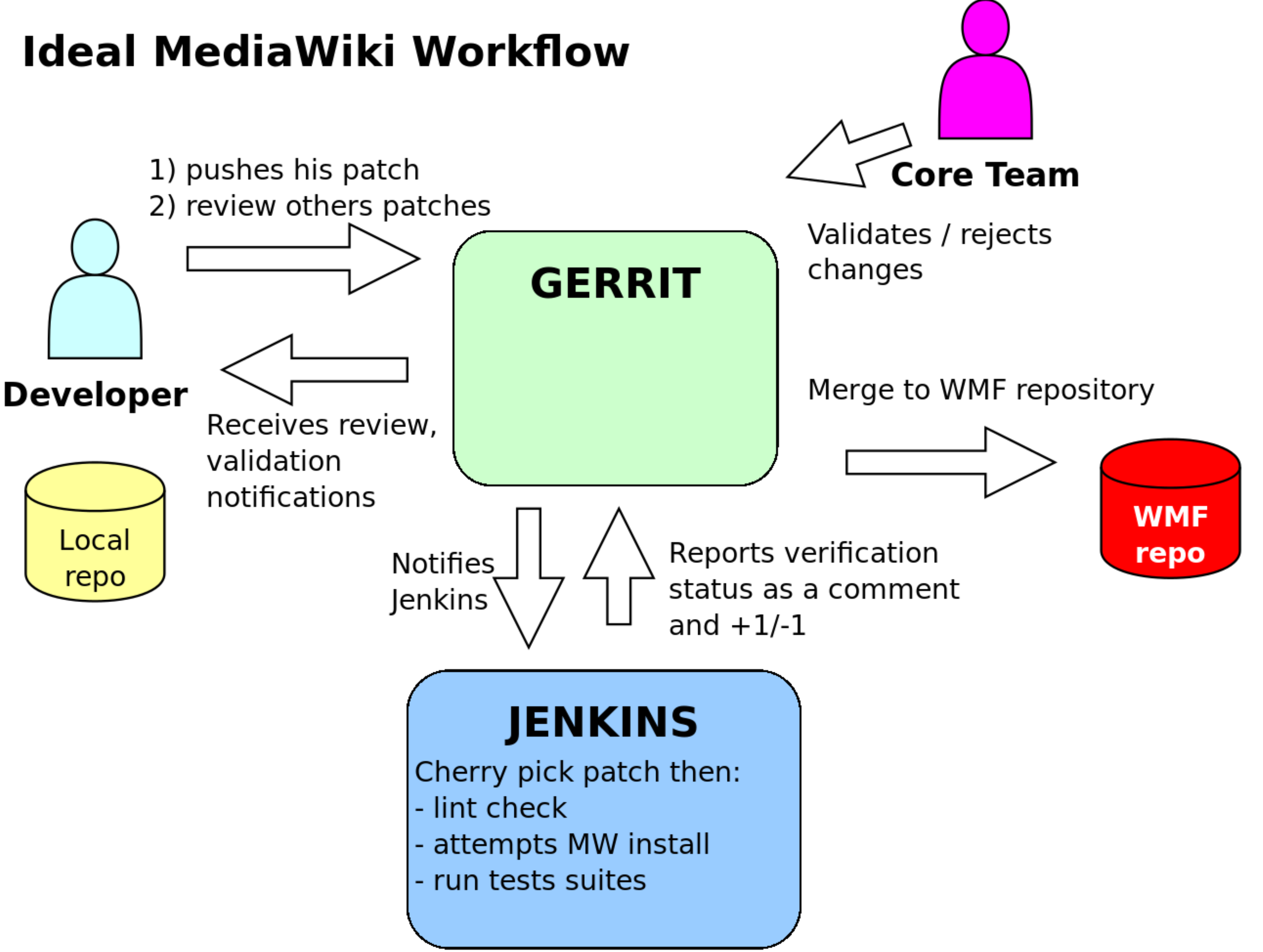
Uploaded patch set 2.

Press '?' to view keyboard shortcuts
Powered by Gerrit Code Review (2.3) | Report Bug

Gerrit

(open source)

Ideal MediaWiki Workflow



Core Team

Validates / rejects changes

1) pushes his patch
2) review others patches

GERRIT

Developer

Receives review, validation notifications

Local repo

Merge to WMF repository

WMF repo

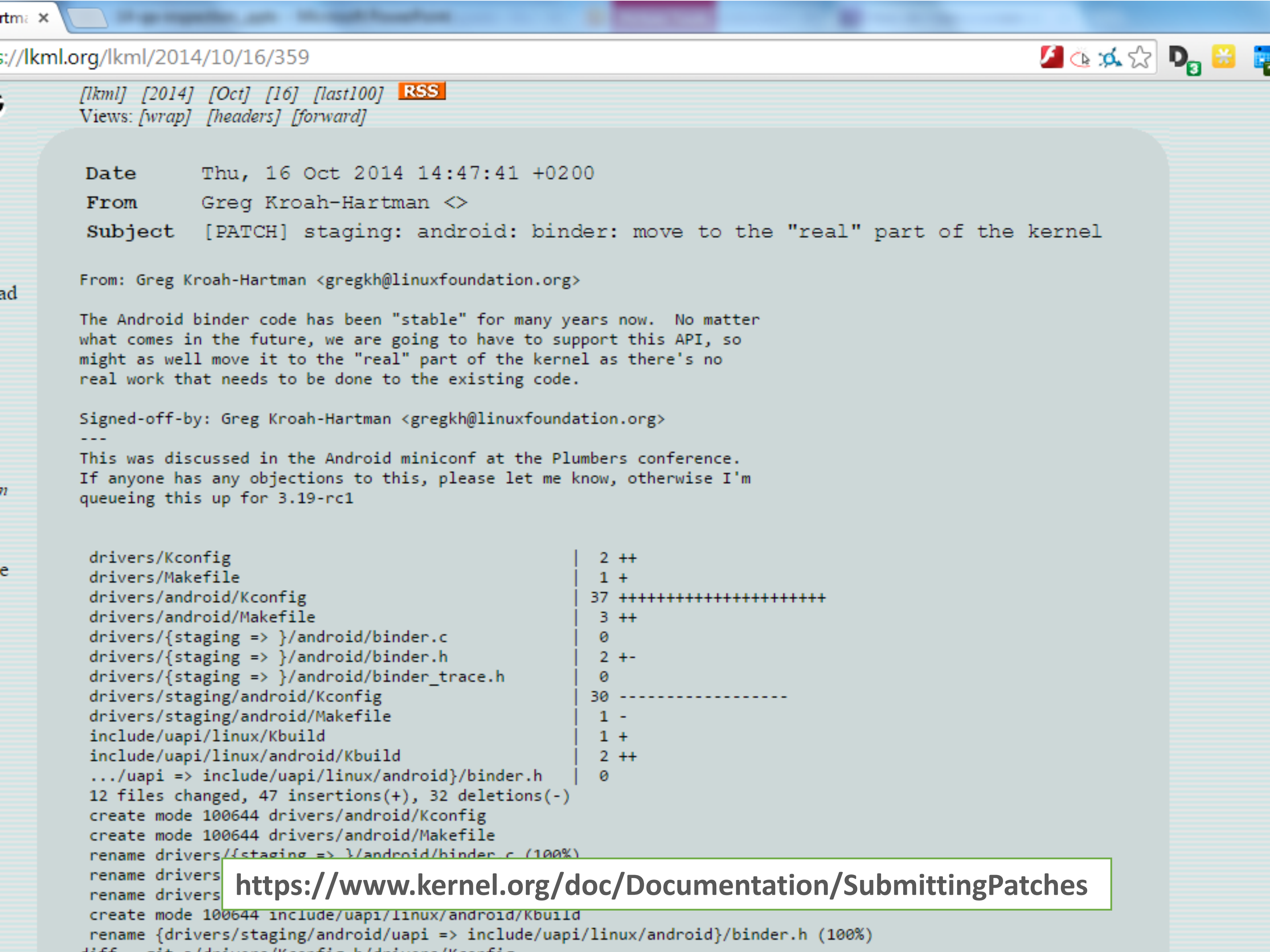
Notifies Jenkins

Reports verification status as a comment and +1/-1

JENKINS

Cherry pick patch then:
- lint check
- attempts MW install
- run tests suites

http://www.mediawiki.org/wiki/Gerrit/Advanced_usage



[lkml] [2014] [Oct] [16] [last100] RSS
Views: [wrap] [headers] [forward]

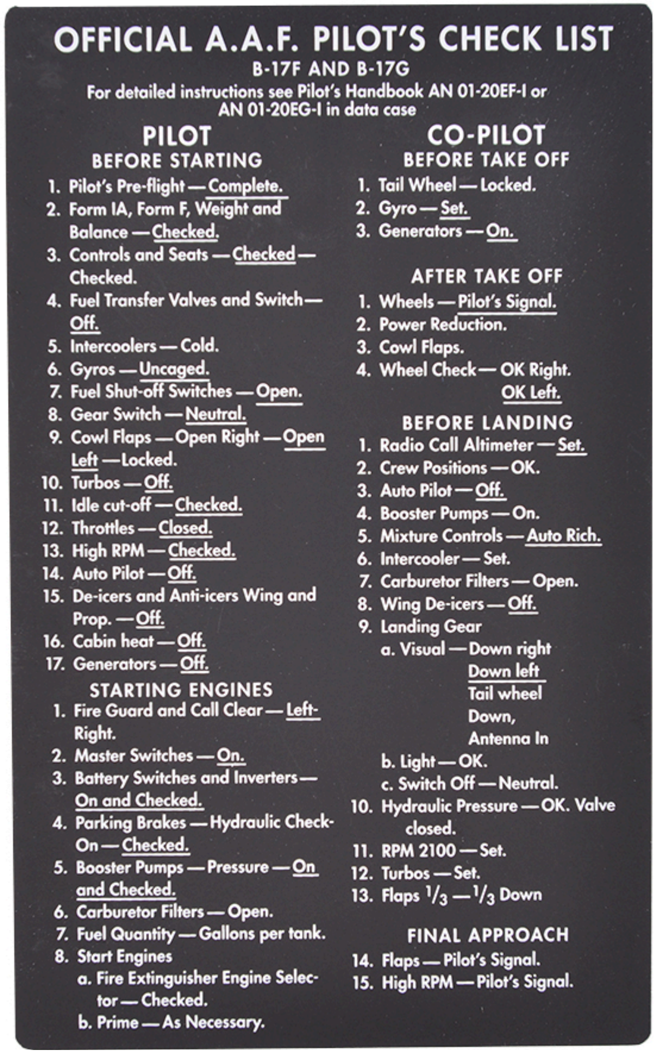
```
Date     Thu, 16 Oct 2014 14:47:41 +0200
From     Greg Kroah-Hartman <>
Subject  [PATCH] staging: android: binder: move to the "real" part of the kernel
```

```
From: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

The Android binder code has been "stable" for many years now.  No matter
what comes in the future, we are going to have to support this API, so
might as well move it to the "real" part of the kernel as there's no
real work that needs to be done to the existing code.

Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
---
This was discussed in the Android miniconf at the Plumbers conference.
If anyone has any objections to this, please let me know, otherwise I'm
queueing this up for 3.19-rc1


 drivers/Kconfig                                      |  2 ++
 drivers/Makefile                                     |  1 +
 drivers/android/Kconfig                              | 37 +++++++++++++++++++++++
 drivers/android/Makefile                             |  3 ++
 drivers/{staging => }/android/binder.c               |  0
 drivers/{staging => }/android/binder.h               |  2 +-
 drivers/{staging => }/android/binder_trace.h         |  0
 drivers/staging/android/Kconfig                      | 30 ------------------
 drivers/staging/android/Makefile                     |  1 -
 include/uapi/linux/Kbuild                            |  1 +
 include/uapi/linux/android/Kbuild                    |  2 ++
 .../uapi => include/uapi/linux/android}/binder.h     |  0
 12 files changed, 47 insertions(+), 32 deletions(-)
 create mode 100644 drivers/android/Kconfig
 create mode 100644 drivers/android/Makefile
 rename drivers/{staging => }/android/binder.c (100%)
 rename drivers
 rename drivers
 create mode 100644 include/uapi/linux/android/Kbuild
 rename {drivers/staging/android/uapi => include/uapi/linux/android}/binder.h (100%)
diff --git a/drivers/Kconfig b/drivers/Kconfig
```

https://www.kernel.org/doc/Documentation/SubmittingPatches

Process: Checklists!

https://en.wikipedia.org/wiki/File:B17_-_Chino_Airshow_2014_(framed).jpg



OFFICIAL A.A.F. PILOT'S CHECK LIST

B-17F AND B-17G

For detailed instructions see Pilot's Handbook AN 01-20EF-I or AN 01-20EG-I in data case

PILOT

BEFORE STARTING

1. Pilot's Pre-flight — Complete.
2. Form IA, Form F, Weight and Balance — Checked.
3. Controls and Seats — Checked — Checked.
4. Fuel Transfer Valves and Switch — Off.
5. Intercoolers — Cold.
6. Gyros — Uncaged.
7. Fuel Shut-off Switches — Open.
8. Gear Switch — Neutral.
9. Cowl Flaps — Open Right — Open Left — Locked.
10. Turbos — Off.
11. Idle cut-off — Checked.
12. Throttles — Closed.
13. High RPM — Checked.
14. Auto Pilot — Off.
15. De-icers and Anti-icers Wing and Prop. — Off.
16. Cabin heat — Off.
17. Generators — Off.

STARTING ENGINES

1. Fire Guard and Call Clear — Left-Right.
2. Master Switches — On.
3. Battery Switches and Inverters — On and Checked.
4. Parking Brakes — Hydraulic Check-On — Checked.
5. Booster Pumps — Pressure — On and Checked.
6. Carburetor Filters — Open.
7. Fuel Quantity — Gallons per tank.
8. Start Engines
   - a. Fire Extinguisher Engine Selector — Checked.
   - b. Prime — As Necessary.

CO-PILOT

BEFORE TAKE OFF

1. Tail Wheel — Locked.
2. Gyro — Set.
3. Generators — On.

AFTER TAKE OFF

1. Wheels — Pilot's Signal.
2. Power Reduction.
3. Cowl Flaps.
4. Wheel Check — OK Right.
   OK Left.

BEFORE LANDING

1. Radio Call Altimeter — Set.
2. Crew Positions — OK.
3. Auto Pilot — Off.
4. Booster Pumps — On.
5. Mixture Controls — Auto Rich.
6. Intercooler — Set.
7. Carburetor Filters — Open.
8. Wing De-icers — Off.
9. Landing Gear
   - a. Visual — Down right
     Down left
     Tail wheel
     Down,
     Antenna In
   - b. Light — OK.
   - c. Switch Off — Neutral.
10. Hydraulic Pressure — OK. Valve closed.
11. RPM 2100 — Set.
12. Turbos — Set.
13. Flaps $^{1}/_{3}$ — $^{1}/_{3}$ Down

FINAL APPROACH

14. Flaps — Pilot's Signal.
15. High RPM — Pilot's Signal.

The Checklist: https://www.newyorker.com/magazine/2007/12/10/the-checklist

Activity

DEVELOP CHECKLIST FOR CODE REVIEW

EXPECTATIONS AND OUTCOMES OF MODERN CODE REVIEWS

Reasons for Code Reviews

- Finding defects
  - both low-level and high-level issues
  - requirements/design/code issues
  - security/performance/… issues
- Code improvement
  - readability, formatting, commenting, consistency, dead code removal, naming
  - enforce to coding standards
- Identifying alternative solutions
- Knowledge transfer
  - learn about API usage, available libraries, best practices, team conventions, system design, "tricks", …
  - "developer education", especially for junior developers

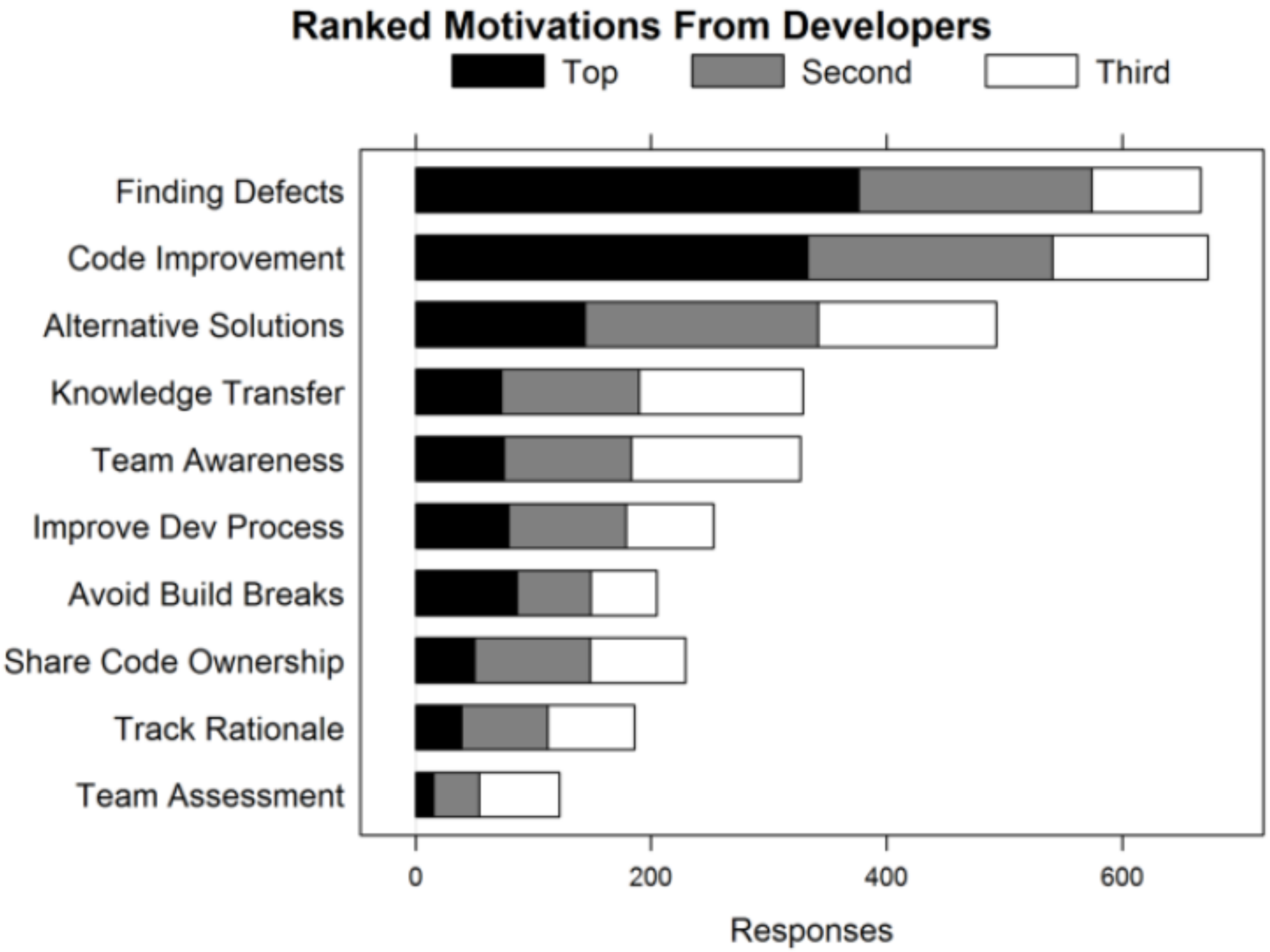
Bacchelli, Alberto, and Christian Bird. "Expectations, outcomes, and challenges of modern code review." *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013.

Reasons for Code Reviews (continued)

- Team awareness and transparency
  - let others "double check" changes
  - announce changes to specific developers or entire team ("FYI")
  - general awareness of ongoing changes and new functionality
- Shared code ownership
  - shared understanding of larger part of the code base
  - openness toward critique and changes
  - makes developers "less protective" of their code

Bacchelli, Alberto, and Christian Bird. "Expectations, outcomes, and challenges of modern code review." *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013.

Code Review at Microsoft

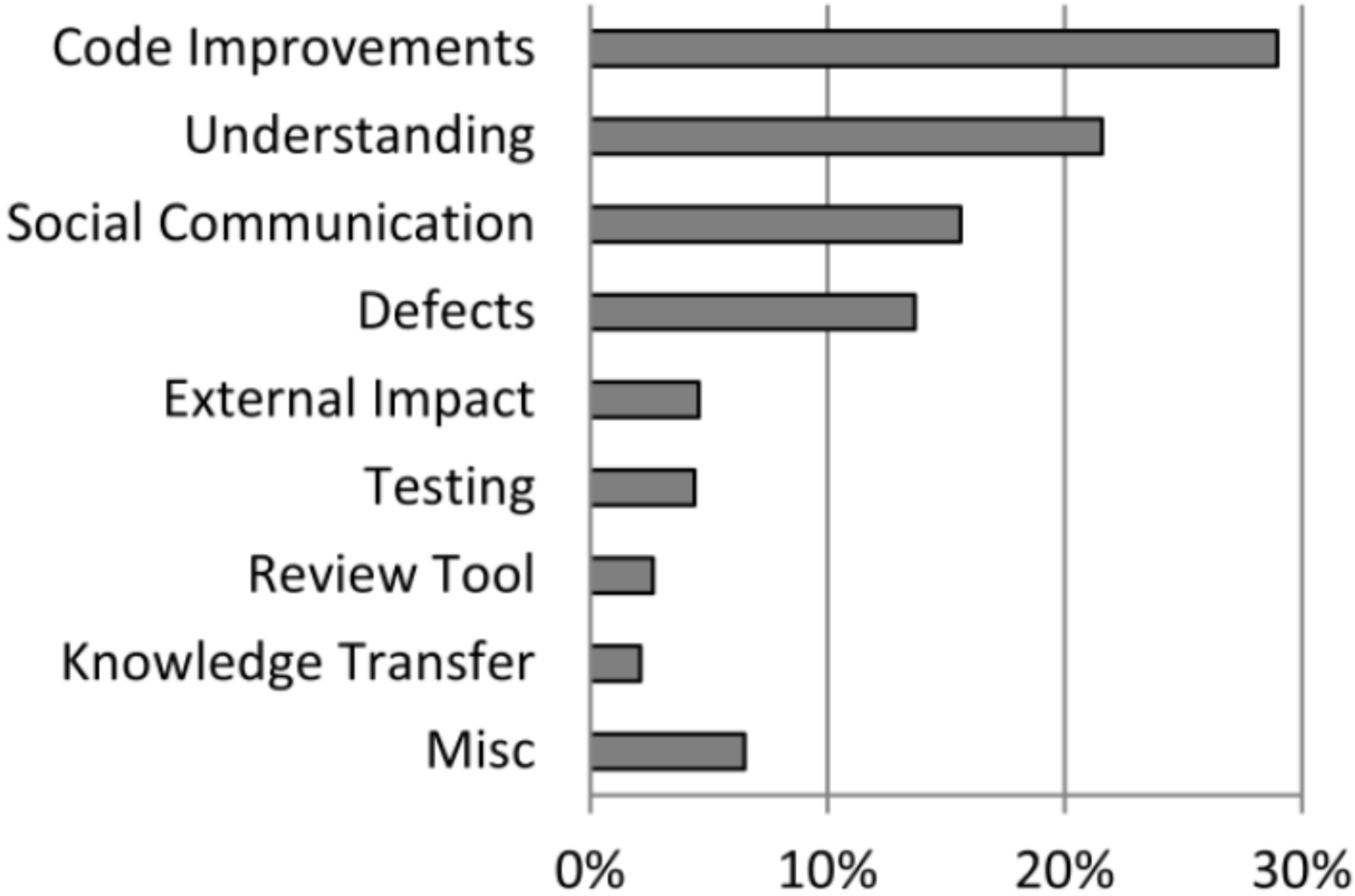


Bacchelli, Alberto, and Christian Bird. "Expectations, outcomes, and challenges of modern code review." *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013.

Outcomes (at Microsoft analyzing 200 reviews with 570 comments)

- Most frequently code improvements (29%)
  - 58 better coding practices
  - 55 removing unused/dead code
  - 52 improving readability
- Defect finding (14%)
  - 65 logical issues ("uncomplicated logical errors, eg., corner cases, common configuration values, operator precedence)
  - 6 high-level issues
  - 5 security issues
  - 3 wrong exception handling
- Knowledge transfer
  - 12 pointers to internal/external documentation etc

Bacchelli, Alberto, and Christian Bird. "Expectations, outcomes, and challenges of modern code review." *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013.



Bacchelli, Alberto, and Christian Bird. "Expectations, outcomes, and challenges of modern code review." *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013.

Mismatch of Expectations and Outcomes

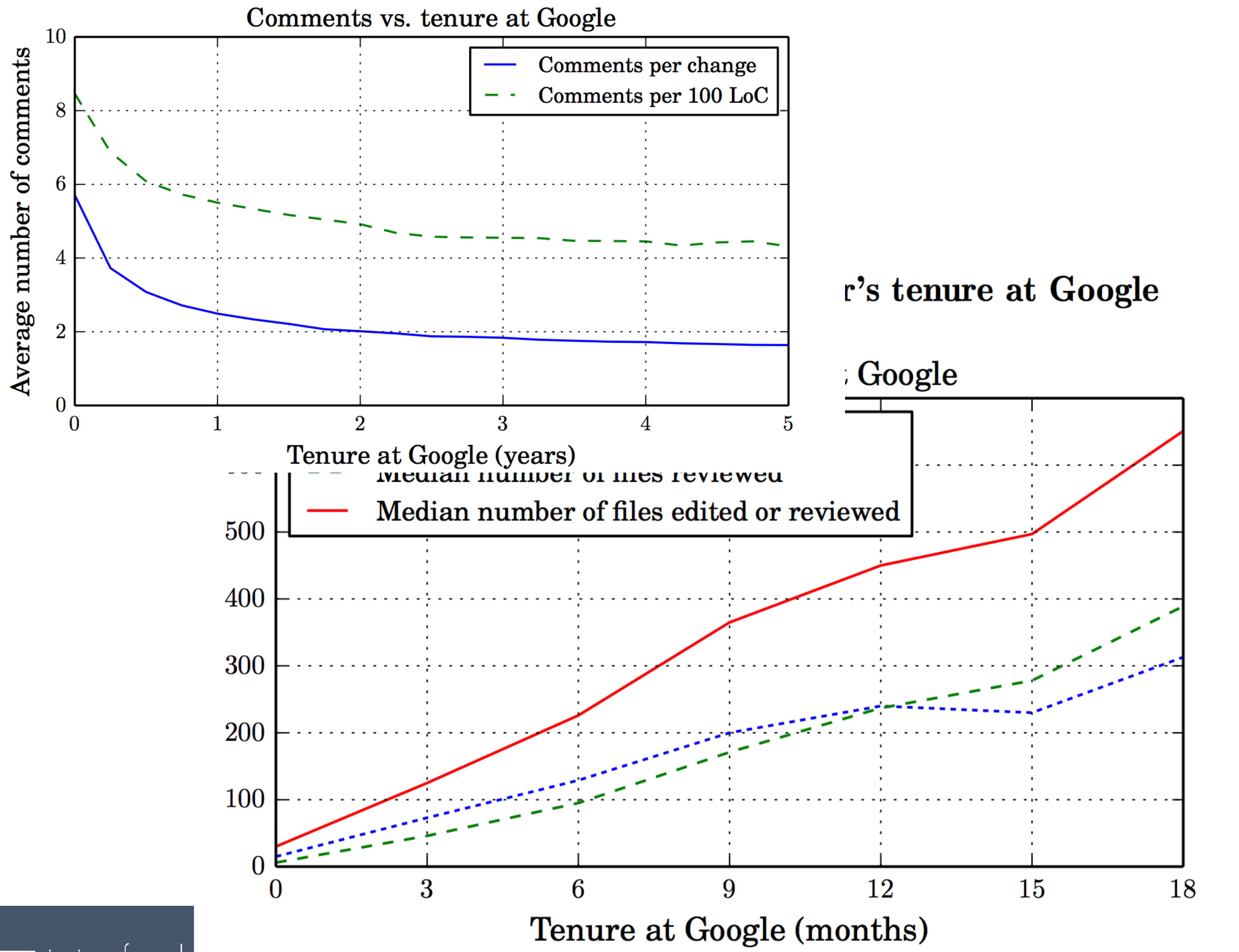
- Low quality of code reviews
  - Reviewers look for easy errors, as formatting issues
  - Miss serious errors
- Understanding is the main challenge
  - Understanding the reason for a change
  - Understanding the code and its context
  - Feedback channels to ask questions often needed
- No quality assurance on the outcome

Bacchelli, Alberto, and Christian Bird. "Expectations, outcomes, and challenges of modern code review." *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013.

Code Review at Google

- Introduced to *“force developers to write code that other developers could understand”*
- 3 Found benefits:
  - checking the consistency of style and design
  - ensuring adequate tests
  - improving security by making sure no single developer can commit arbitrary code without oversight

Caitlin Sadowski, Emma Söderberg, Luke Church, Michal Sipko and Alberto Bacchelli. 2018. Modern Code Review: A Case Study at Google. International Conference on Software Engineering


Comments vs. tenure at Google
Comments per change
Comments per 100 LoC
Average number of comments
0
2
4
6
8
10
0
1
2
3
4
5
Tenure at Google (years)
r's tenure at Google
Google
Median number of files reviewed
Median number of files edited or reviewed
0
100
200
300
400
500
0
3
6
9
12
15
18
Tenure at Google (months)

Social issues: Egos in Inspections

- Author’s self-worth in artifacts
- Identify defects, not alternatives; do not criticize authors
    - “you didn’t initialize variable a” -> “I don’t see where variable a is initialized”
- Avoid defending code; avoid discussions of solutions/alternatives
- Reviewers should not “show off” that they are better/smarter
- Avoid style discussions if there are no guidelines
- Author decides how to resolve fault

Social issues 2

- Moderator must move discussion along, resolve conflicts
- Meetings should not include management
- Do not use for HR evaluation
  - “finding more than 5 bugs during inspection counts against the author”
  - Leads to avoidance, fragmented submission, not pointing out defects, holding pre-reviews
- Responsibility for quality with authors, not reviewers
  - “why fix this, reviewers will find it”

Summary

- Code reviews effective to identify bugs
- Additional benefits (e.g., knowledge transfer, shared code ownership, awareness)
- Reviews require understanding
- Different review types with different formality
- Formal inspection require planning & social skills, are expensive, but very effective

Further Reading

- Sommerville. Software Engineering. 8th Edition. Addison-Wesley 2007. Chapter 22.2
  - Overview of formal inspections
- Wiegers. Peer Reviews in Software. Addison-Wesley 2002
  - Entire book on formal inspections; how to run them and how to introduce them
- Bacchelli and Bird. "Expectations, outcomes, and challenges of modern code review.“ *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013.
  - Detailed studies of modern code reviews at Microsoft
- Oram and Wilson (ed.). Making Software. O’Reilly 2010. Chapter 18
  - Overview of empirical research on formal inspections