Agile Quality Assurance

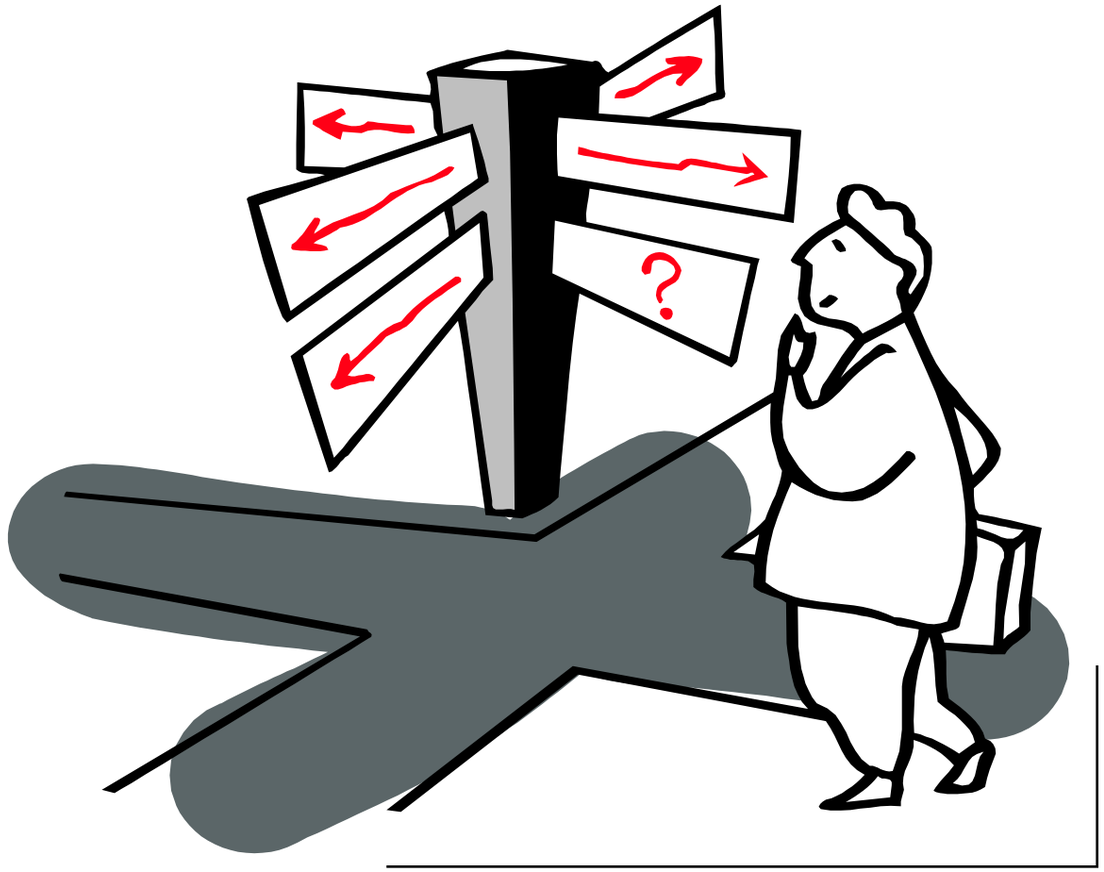
"in vivo" research in software evolution

Prof.Dr. Serge Demeyer

PEM Colloquium — CWI (Amsterdam) — January 2012

Universiteit Antwerpen

Table Of Contents



Innovation

Underlying Technology

1908 — patent on paper filte

194 — commerical piston espresso machine

2001 — senseo

Business Models

1475 — Kiva han coffee house (Constantinople)

(Vienna)
1529 — European coffee house

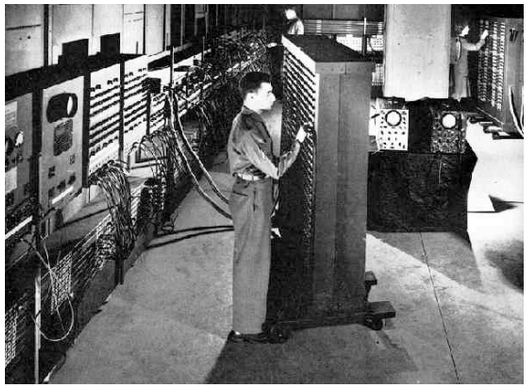
1971 — Starbucks (seattle)

Technology changes every 20 years

...

Underlying business models rarely change !

Innovation in ICT



ENIAC, 1945

IBM PC, 1981

Technology changes every 5 years

...

Underlying business models change often !

Underlying Technology

Embedded

Google

YouTube

Drupal

facebook

twitter

Internet

Market pressure in ICT

YAHOO!

altavista

Google

Measure of innovation

- # products in portfolio younger than 5 years
  + in ICT usually more than 1/2 the portfolio

Significant investment in R&D

- more products … faster

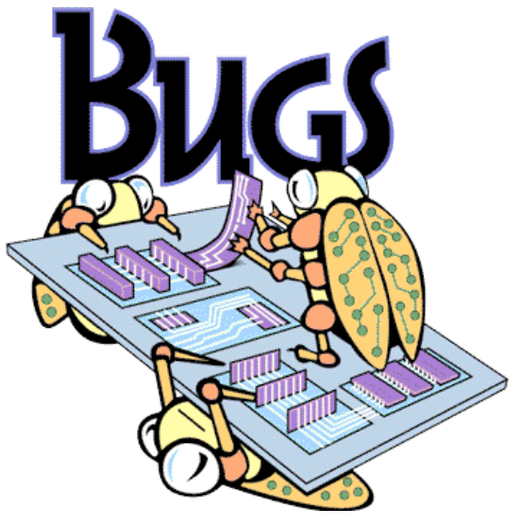
RELIABILITY ⟷ AGILITY

Reliability vs. Agility

Software is vital to our society ⇒ Software must be reliable

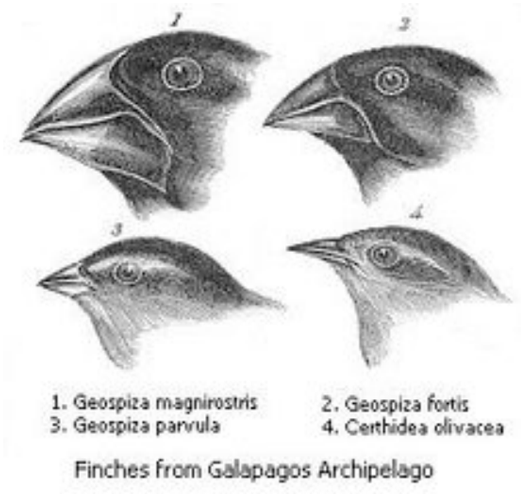
Traditional Software Engineering

Reliable = Software without bugs



Today's Software Engineering

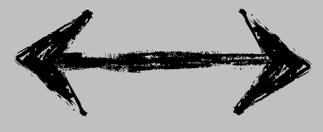
Reliable = Easy to Adapt



On the Origin of Species

Striving for RELIABILITY

(Optimise for *perfection*)



Striving for AGILITY

(Optimise for *development speed*)

Reliability vs. Agility … no single truth

www.taxonweb.be

twitter

YouTube

Google

facebook

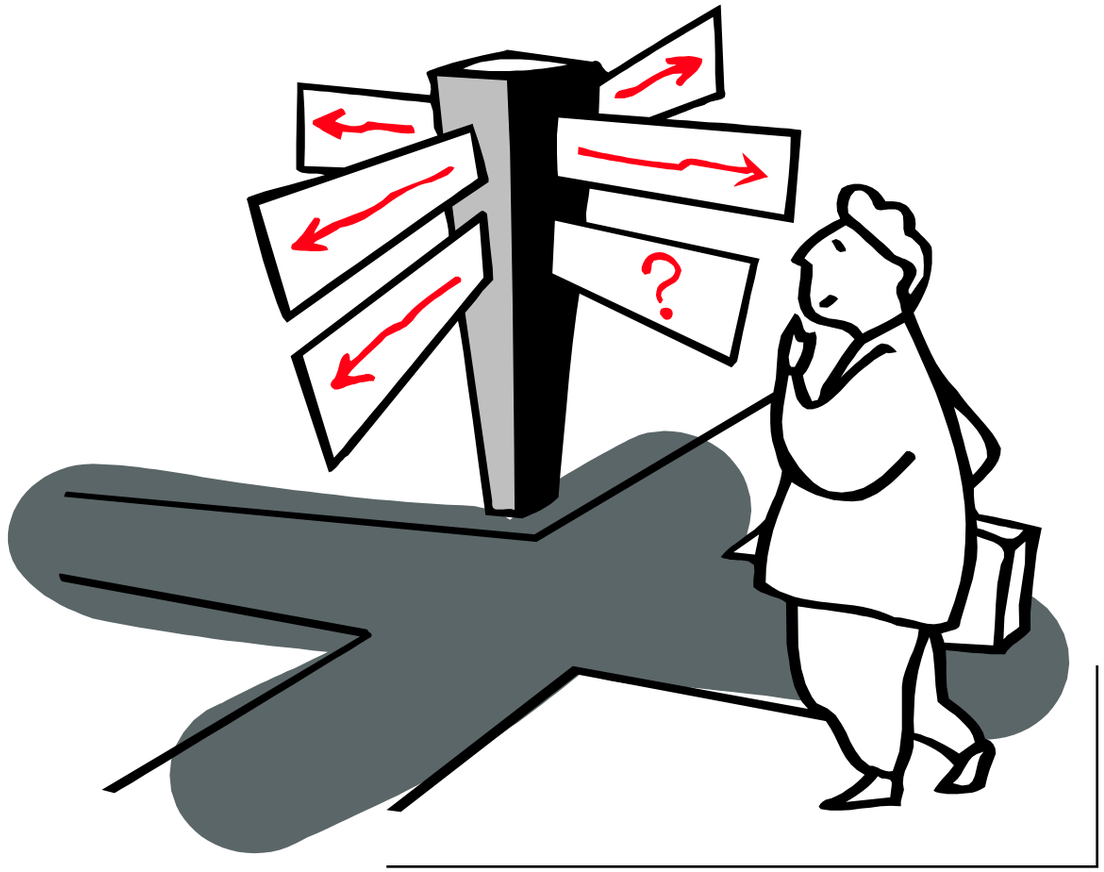
CMMI

RUP

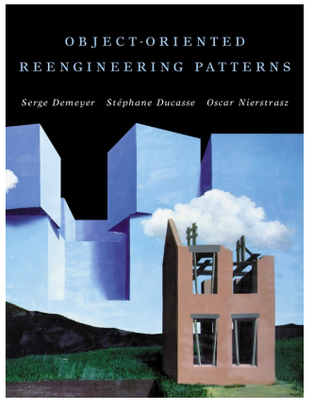
Scrum

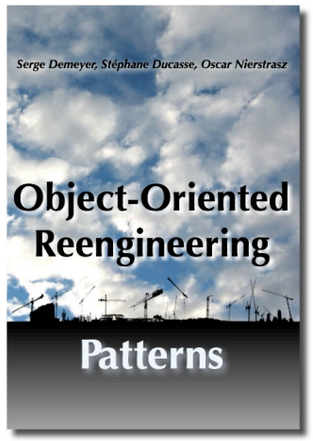
Extreme Programming

Table Of Contents



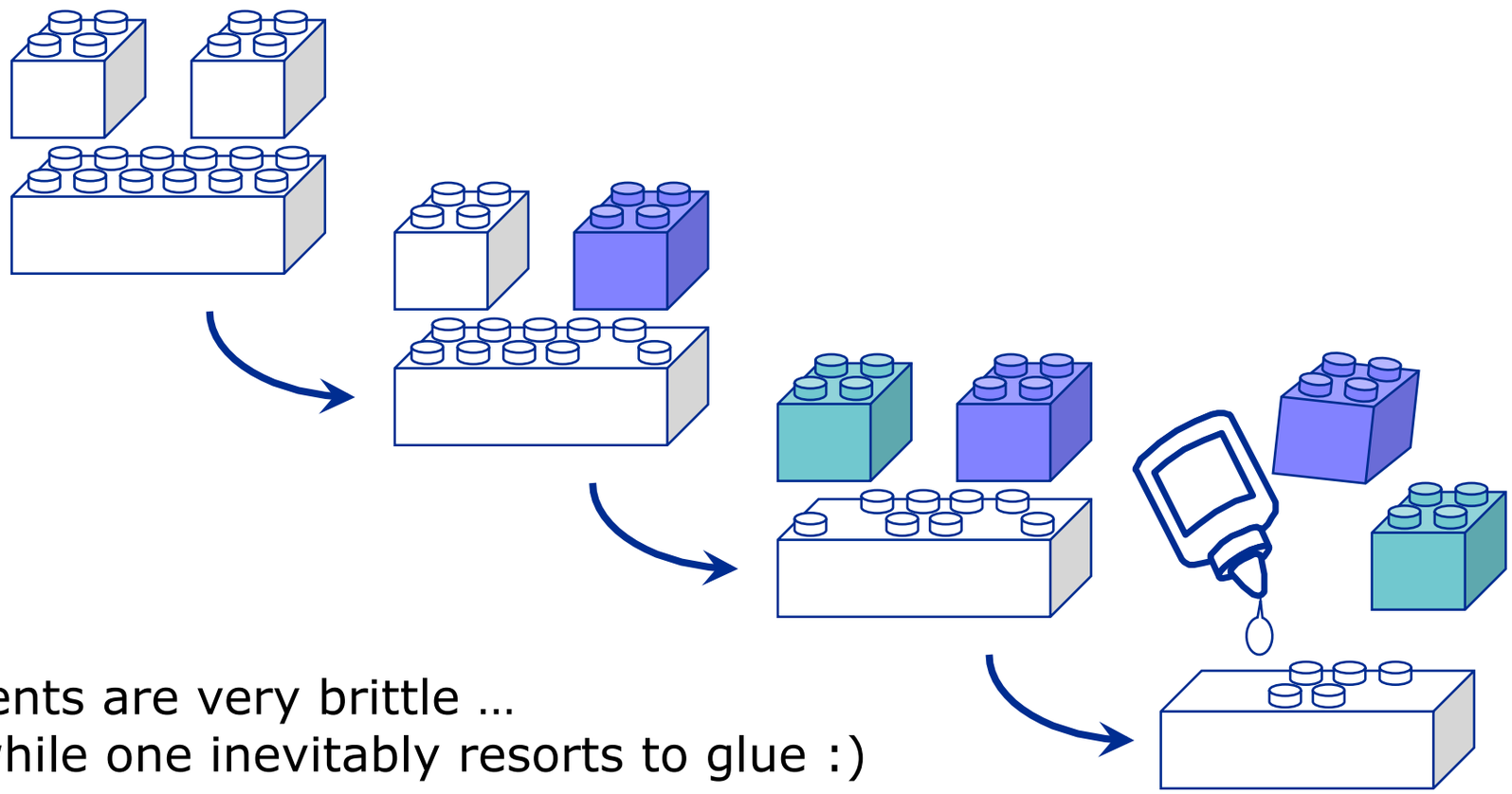
Software Evolution





It is not age that turns a piece of software into a legacy system, but the rate at which it has been developed and adapted without being reengineered.

[Demeyer, Ducasse and Nierstrasz: Object-Oriented Reengineering Patterns]



Components are very brittle …
After a while one inevitably resorts to glue :)

Software Repositories & Archives

Version Control

- CVS, Subversion, …
- Rational ClearCase
- Perforce,
- Visual Source Safe
- …

Automate the Bui

- make
- Ant, Maven
- MSBuild
- OpenMake
- Build Forge

All of a sudden empirical research has what any empirical science needs: a large corpus of objects to analyze.

[Bertrand Meyer's technology blog]

Issue Tracking

- Bugzilla
- BugTracker.NET
- ClearQuest
- JIRA
- Mant
- Visual Studio Team Foundation Server
- …

Automated Testing

- HP QuickTest Professional
- IBM Rational Functional Tester
- Maveryx
- Selenium
- TestComplete
- Visual Studio Test Professional Microsoft 2010
- …

… mailing archives, newsgroups, chat-boxes, facebook, twitter, …

Mining Software Repositories

The Mining Software Repositories (MSR) field analyzes the rich data available in software repositories to uncover interesting and actionable information about software systems and projects.

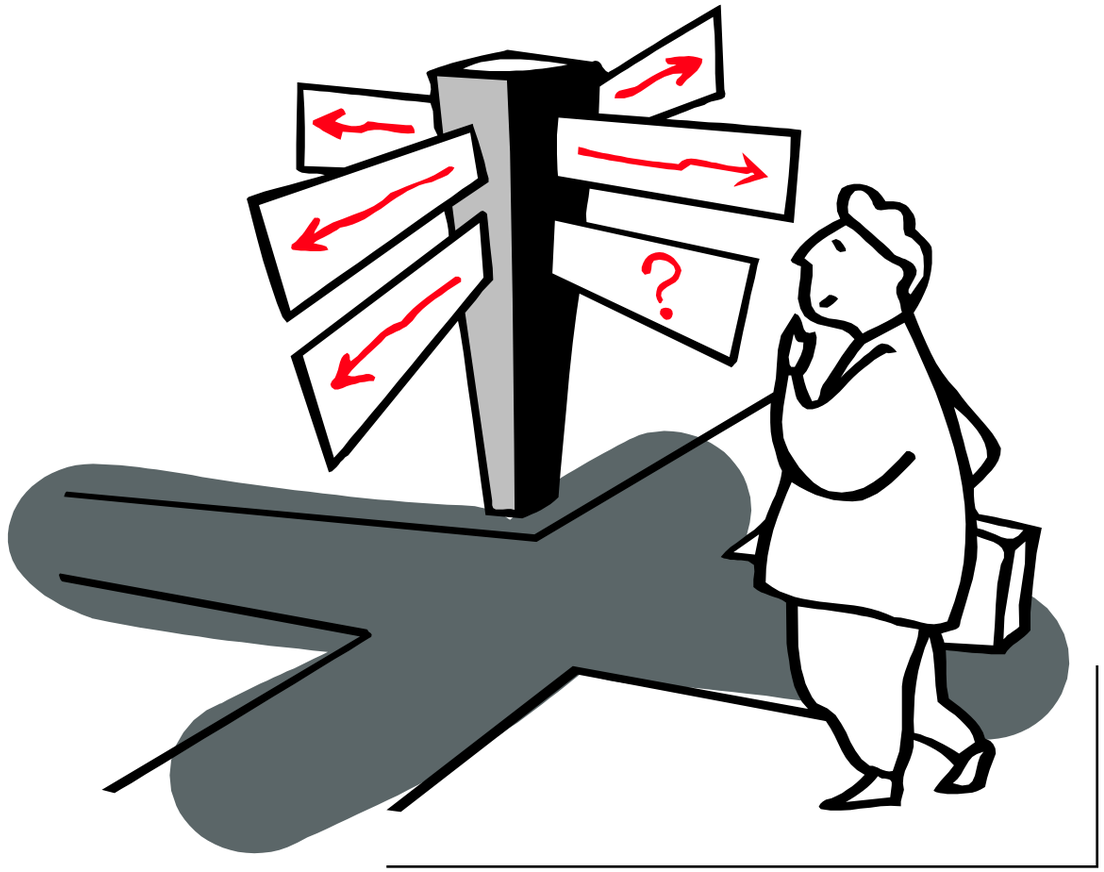
Conferences

- 2012—9th edition, Zürich, CH
- 2011—8th edition, Honolulu, HI, USA
- 2010—7th edition, Cape Town, ZAF
- 2009—6th edition, Vancouver, CAN
- 2008—5th edition, Leipzig, DEU
- 2007—4th edition, Minneapolis, MN, USA
- 2006—3rd edition, Shanghai, CHN
- 2005—2nd edition, Saint Luis, MO, USA
- 2004—1st edition, Edinburgh, UK

Hall of Fame—Mining Challenge Winners

- 2011—Apples Vs. Oranges? An exploration of the challenges of comparing the source code of two software systems (Daniel M. German and Julius Davies)
- 2010—Cloning and Copying between GNOME Projects (Jens Krinke, Nicolas Gold, Yue Jia, and David Binkley)
- 2009—On the use of Internet Relay Chat (IRC) meeting by developers of the GNOME GTK+ project (Emad Shihab, Zhen Ming Jiang and Ahmed E. Hassan)
- 2008—A newbie's guide to Eclipse APIs (Reid Holmes and Robert J. Walker)
- 2007—Mining Eclipse Developer Contributions via Author-Topic Models (Erik Linstead, Paul Rigor, Sushil Bajracharya, Cristina Lopes, and Pierre Baldi)
- 2006—A study of the contributors of PostgreSQL (Daniel M. German)

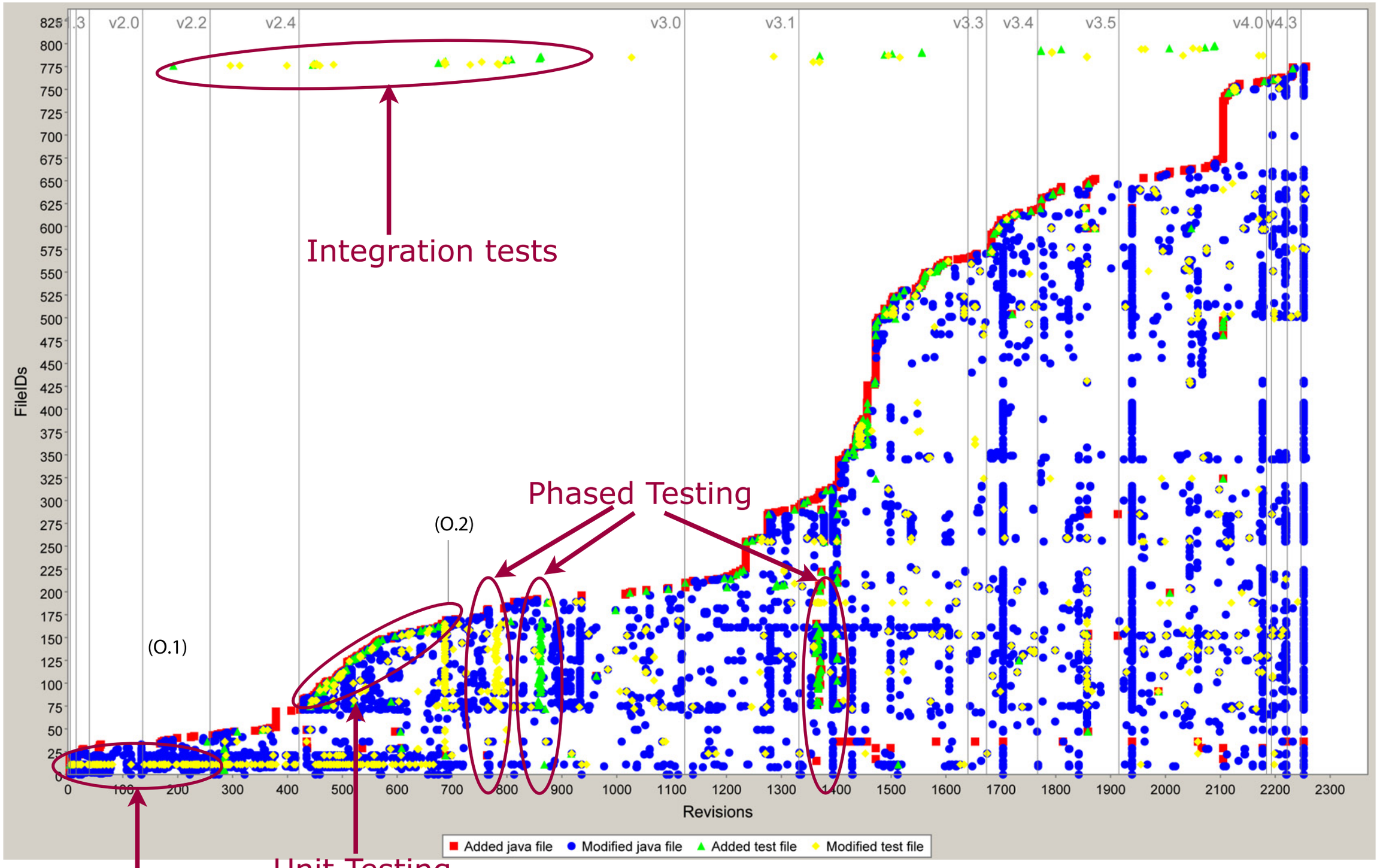
Table Of Contents



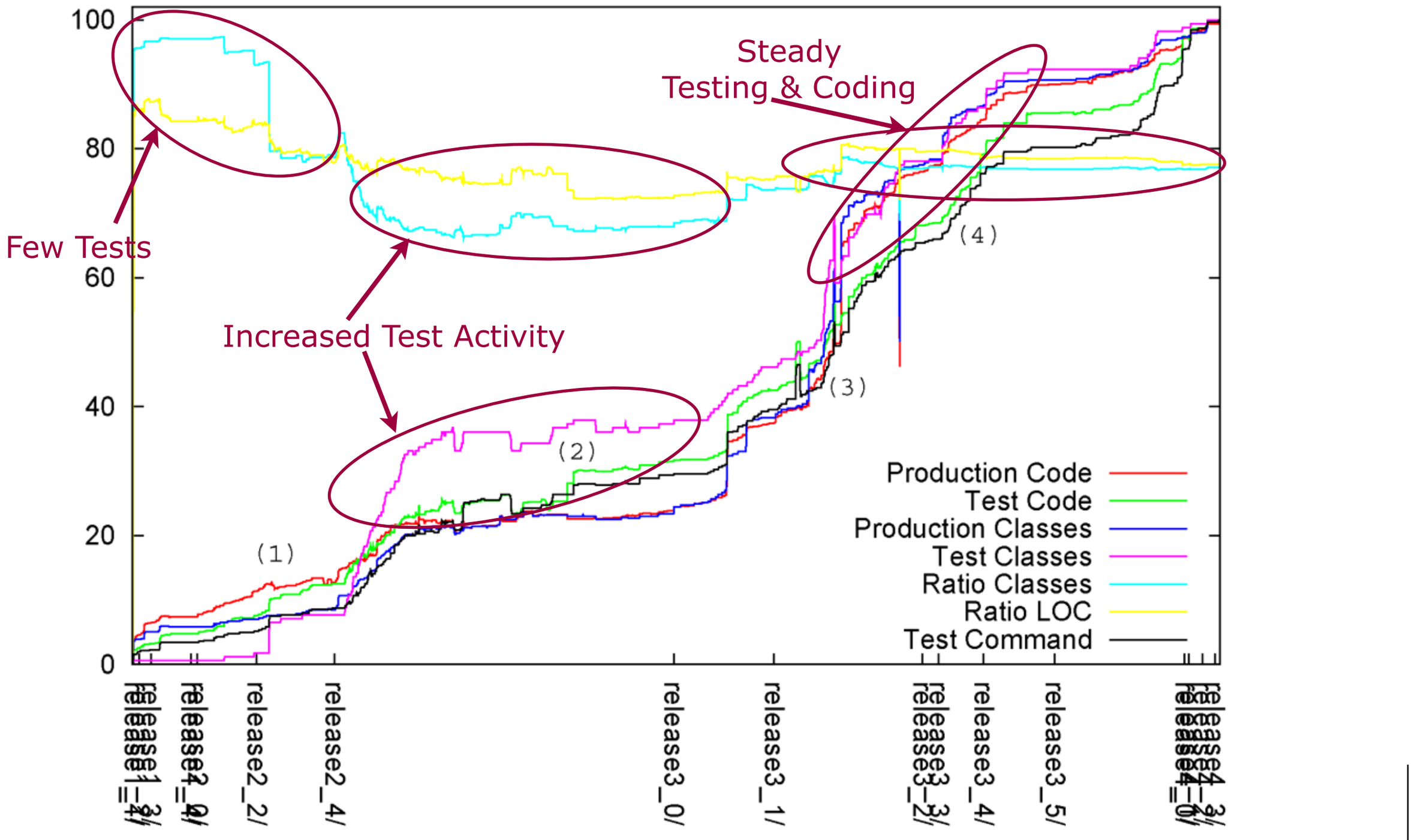
Test Monitor — Change History

http://swerl.tudelft.nl/bin/view/Main/TestHistory

Case = Checkstyle



Test Monitor — Growth History



Test Monitor — Coverage Evolution

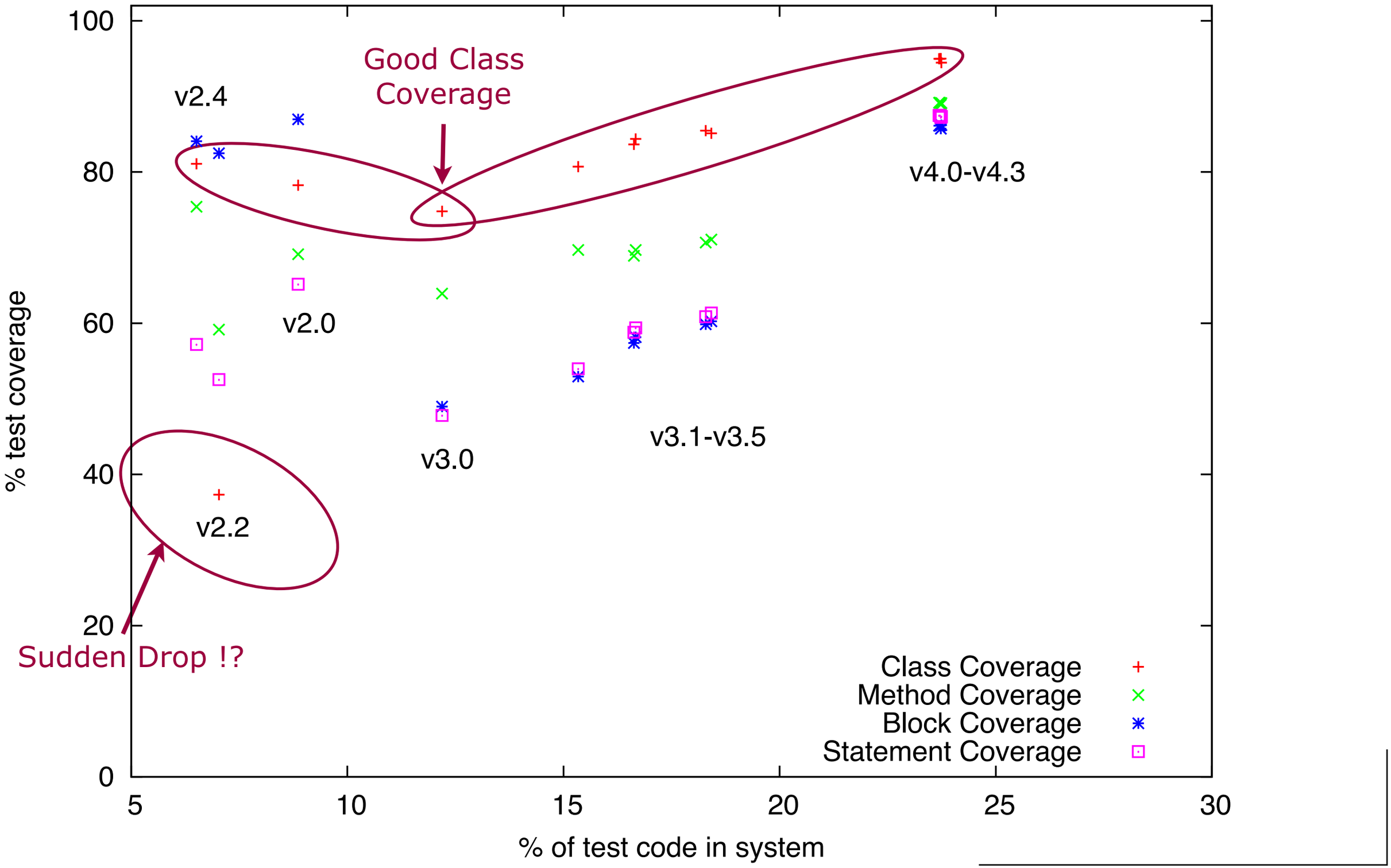


Table Of Contents

Introduction

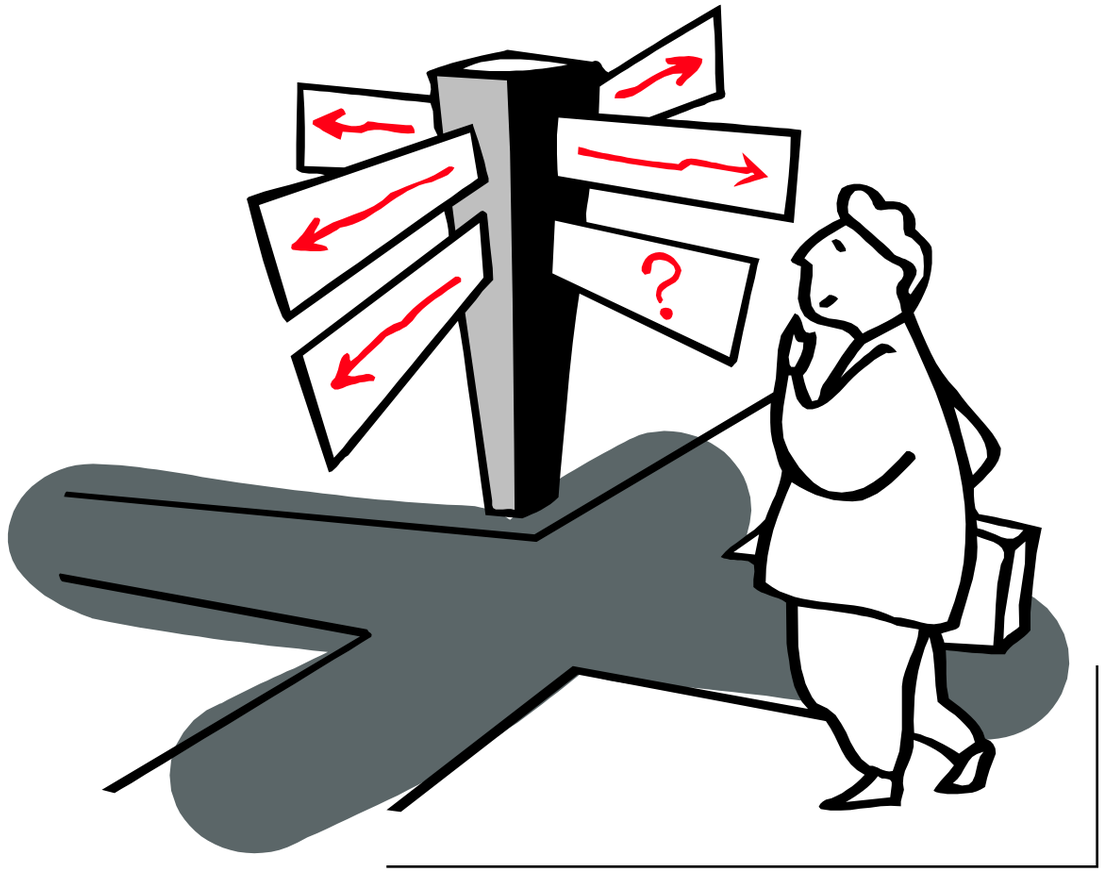
- Reliability vs. Agility

Mining Software Repositories

- Tests (= visualisation)
  + How good was our testing process ?
- Bugs (= text mining)
  + Who should fix this bug ?
  + How long will it take to fix this bug ?
  + What is the severity of this bug ?
- Expertise (= social network analysis)
  + Who are the key personalities ?
  + Who can help me with this file ?
  + Where should we focus our (regression) tests ?

Conclusion

- The future



Enter Bug: OAW4

http://www.openarchitectureware.org/bugzilla/enter_bug.cgi?product=OAW4

Meistbesuchte Seiten | openArchitectureW... | LEO | Karsten Thoms | Fornax | .Net Braindrops | TinyURL!

Bugzilla – Enter Bug: OAW4

Home | New | Search | Find | Reports | My Requests | My Votes | Preferences | Log out karsten.thoms@itemis.de

Before reporting a bug, please read the bug writing guidelines, please look at the list of most frequently reported bugs, and please search for the bug.

Reporter: karsten.thoms@itemis.de

Version: 4.2.1 / 4.3.0 / 4.3.1 / 4.3.1 RC1 / 4.3.1 RC2

Product/Component Specific vocabulary

Product: OAW4

Component: oAW-adapter / oAW-build / oAW-check / oAW-classic / oAW-docs

Severity: enhancement

Priority: P5

Suggestions ?

Platform: PC

OS: Mac OS

Initial State: NEW

Assign To:

Cc:

Default CC:

Estimated Hours: 0.0

Deadline: (YYYY-MM-DD)

URL: http://

Summary:

Description:

Description ⇒ text Mining

Attachment: Add an attachment

Stack Traces ⇒ Link to source code

Depends on:

Blocks:

Commit | Remember values as bookmarkable template

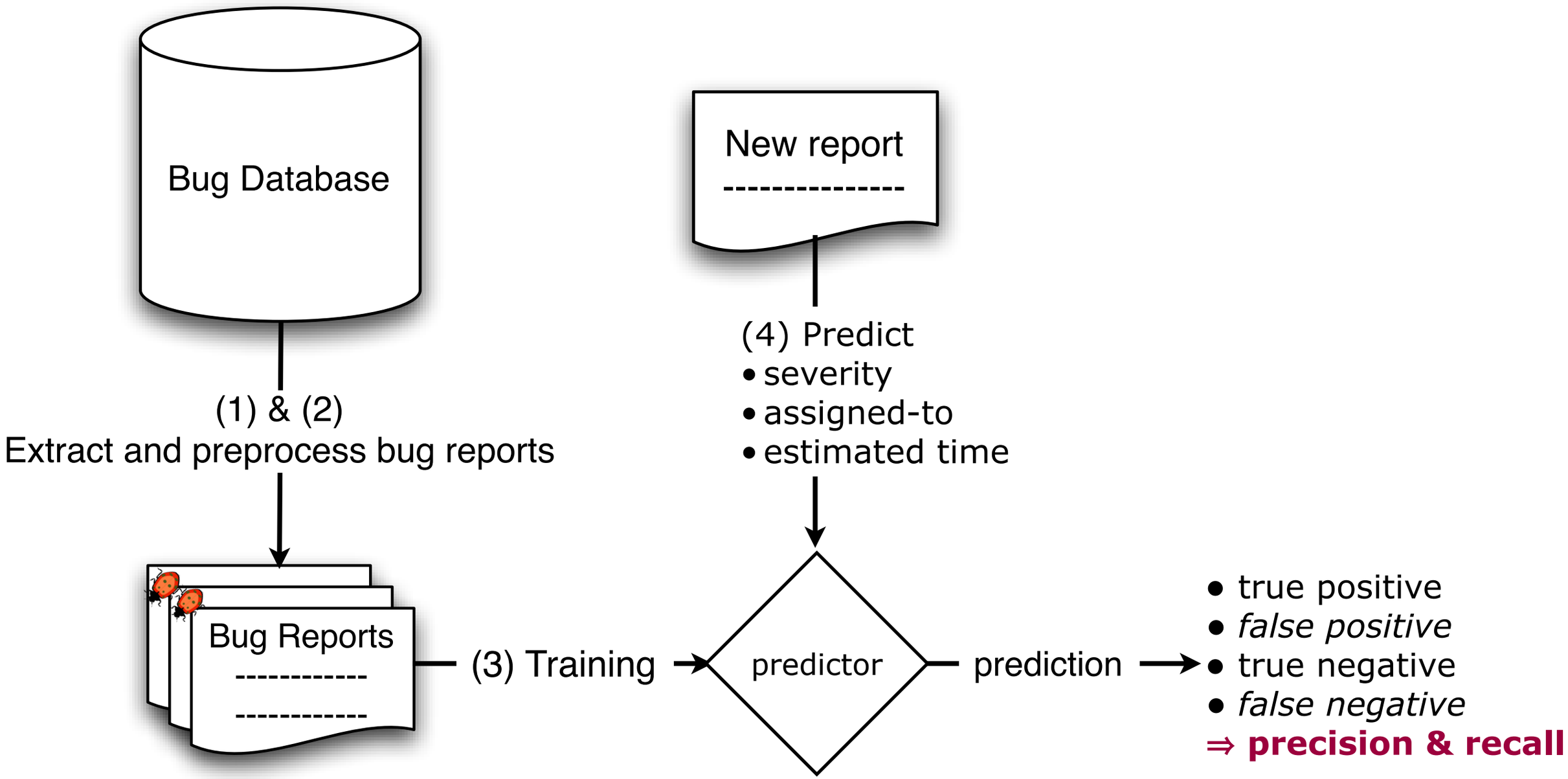
We've made a guess at your operating system and platform. Please check them and, if we got it wrong, email karsten.thoms@itemis.de.

Actions: Home | New | Search | Find | Reports | My Requests | My Votes | Preferences | Log out karsten.thoms@itemis.de

Edit: Parameters | Default Preferences | Sanity Check | Users | Products | Flags | Custom Fields | Field Values | Groups | Keywords | Whining

Saved Searches: My Bugs

Text Mining


Bug Database
(1) & (2)
Extract and preprocess bug reports
Bug Reports

---
(3) Training
New report
---
(4) Predict
• severity
• assigned-to
• estimated time
predictor
prediction
• true positive
• *false positive*
• true negative
• *false negative*
⇒ **precision & recall**

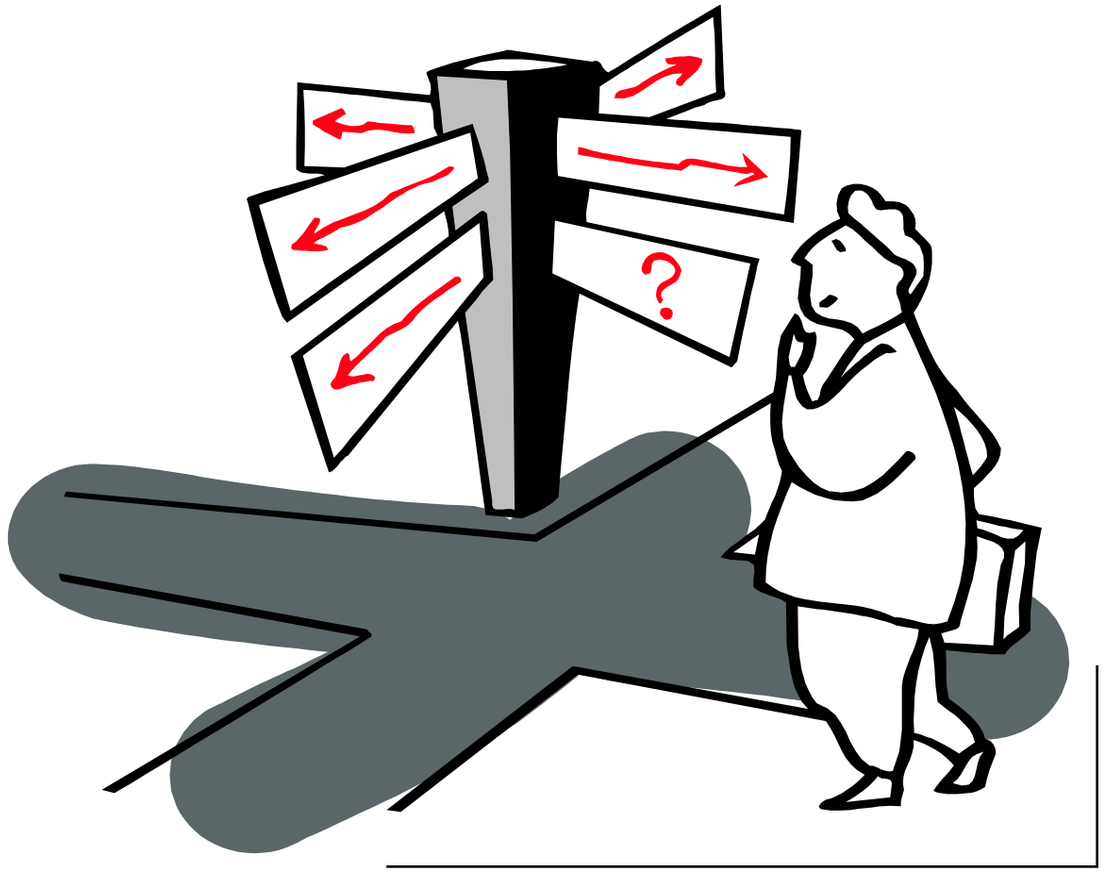
Results

Question	Cases	Precision	Recall
Who should fix this bug ?	Eclipse, Firefox, gcc	eclipse: 57% firefox: 64% gcc: 6%	—
How long will it take to fix this bug ?	JBoss	depends on the component many similar reports: off by one hour few similar reports: off by 7 hours	
What is the severity of this bug ?	Mozilla, Eclipse, Gnome	mozilla, eclipse: 67% - 73% gnome: 75%-82%	mozilla, eclipse: 50% - 75% gnome: 68%-84%

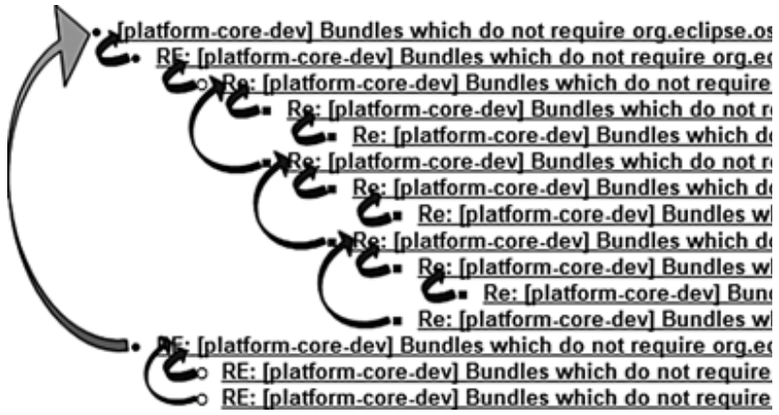
Promising results but …
- how much training is needed ?
- how reliable is the data ?
  (estimates, severity, assigned-to)
- does this generalize ? (on industrial scale ?)

⇒ replication is needed

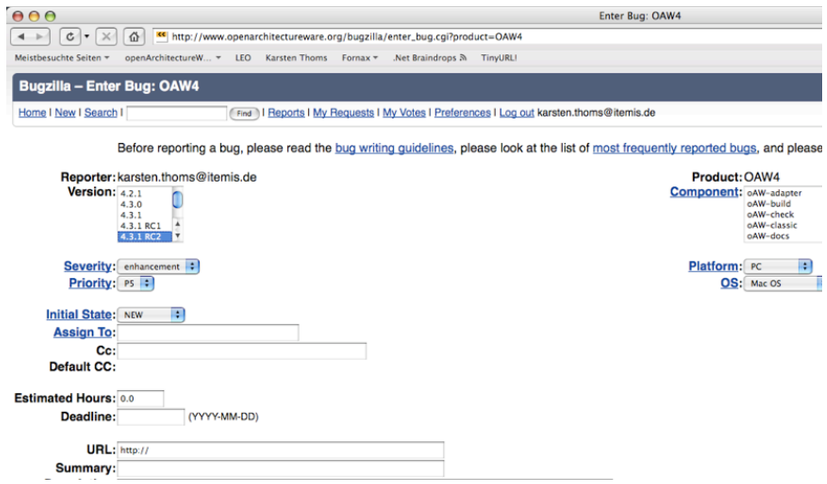
Table Of Contents

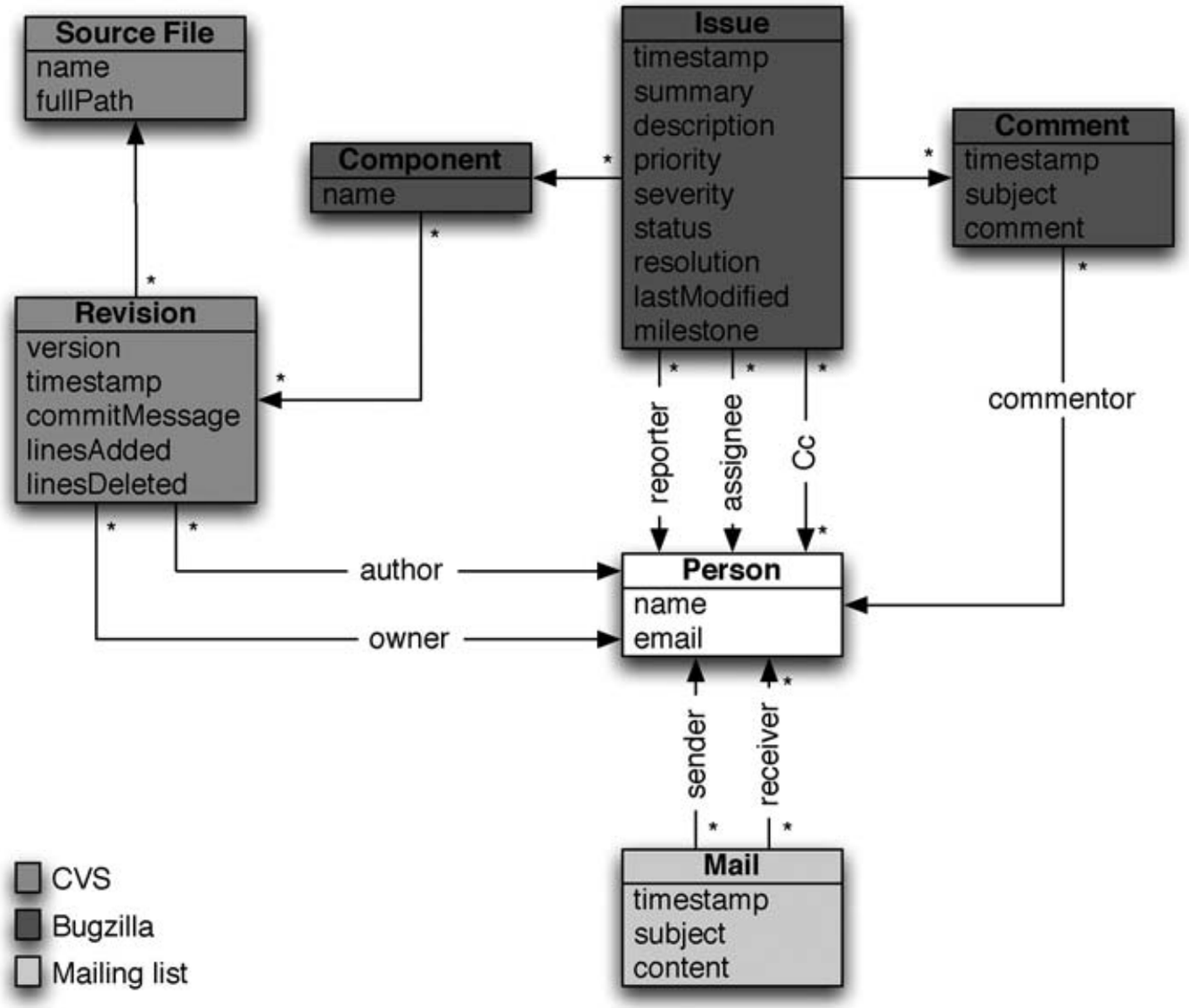


Construct Social Network

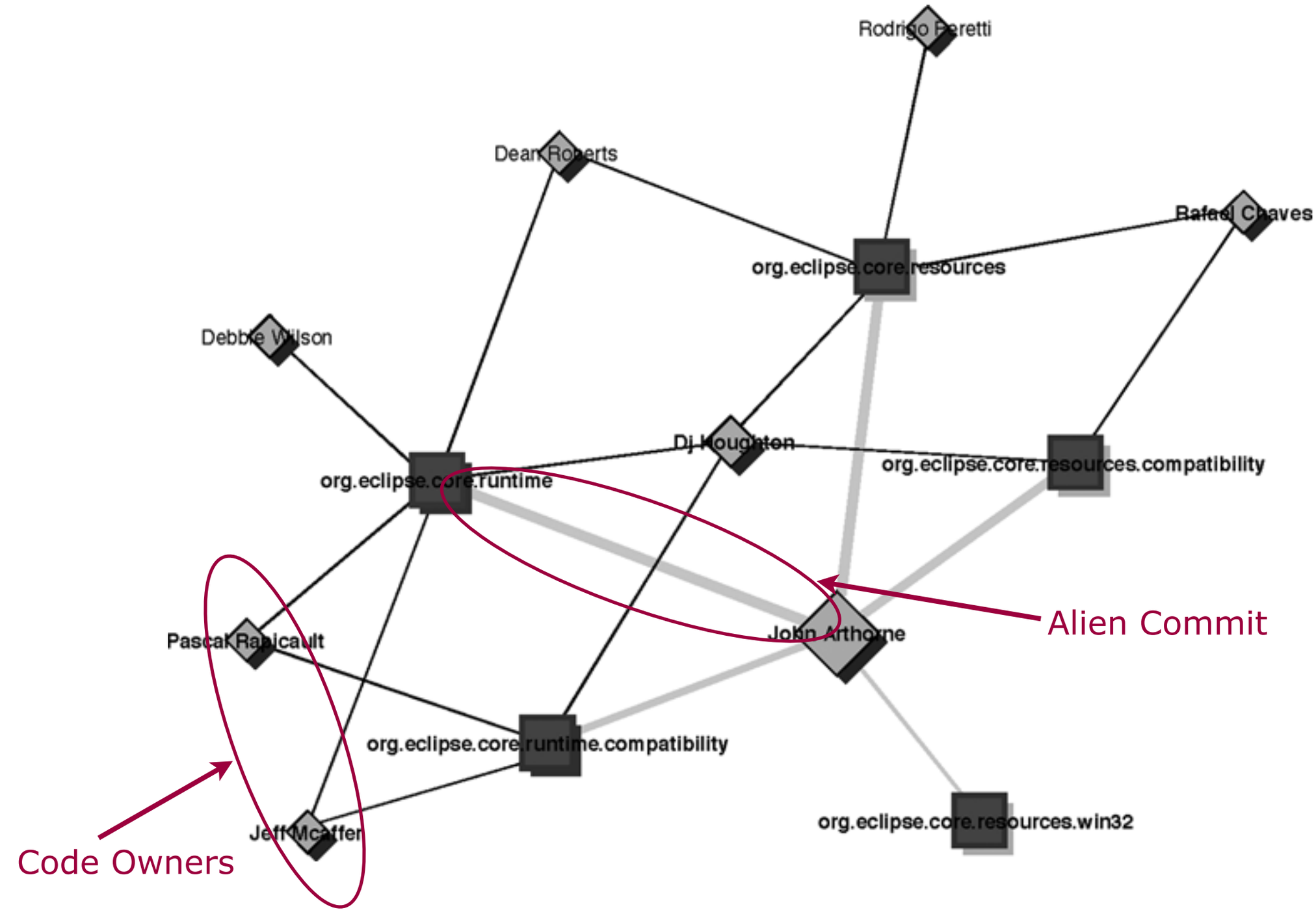


Mailing lists
+ Bug reports ⇒
+ CVS Logs

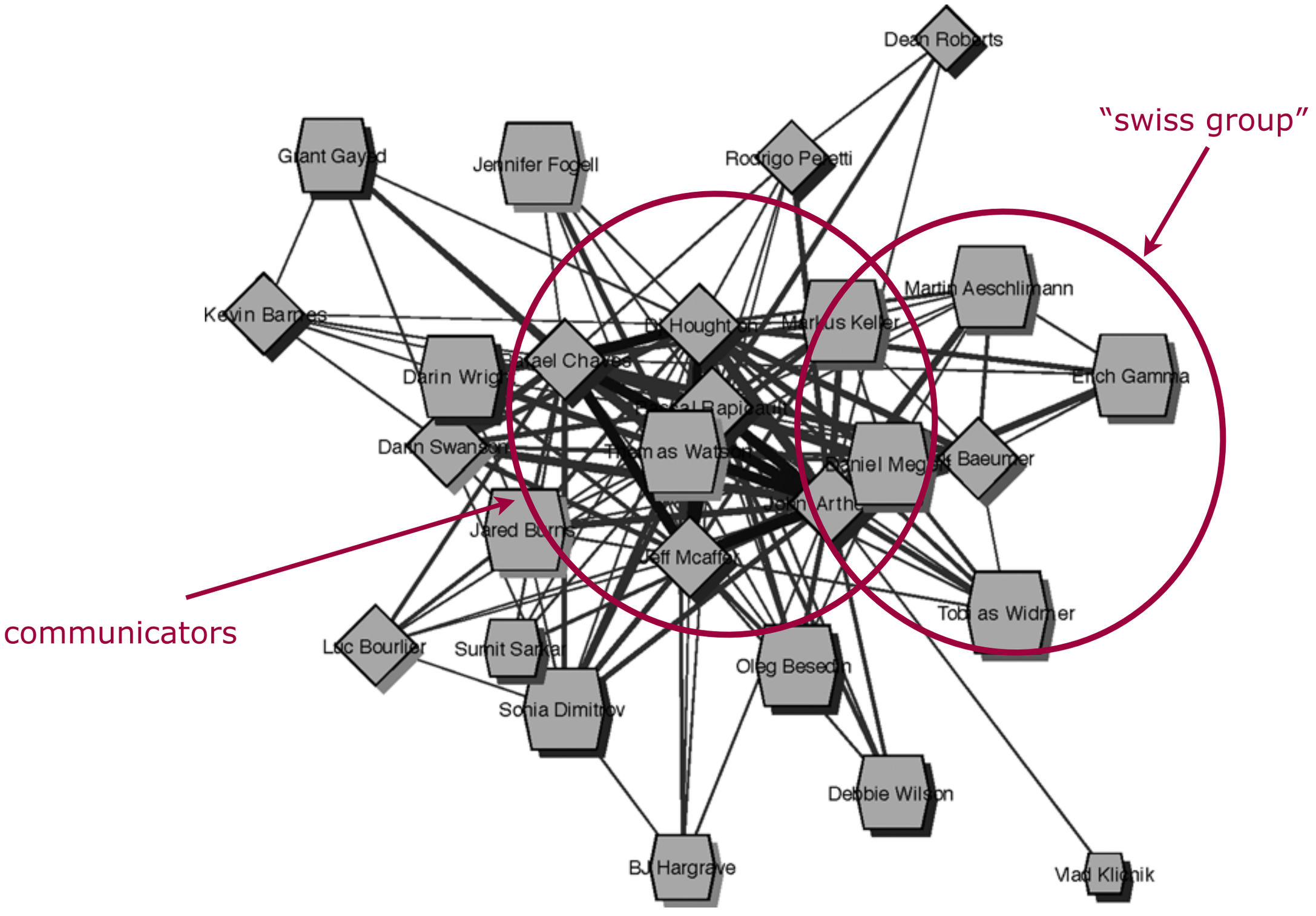




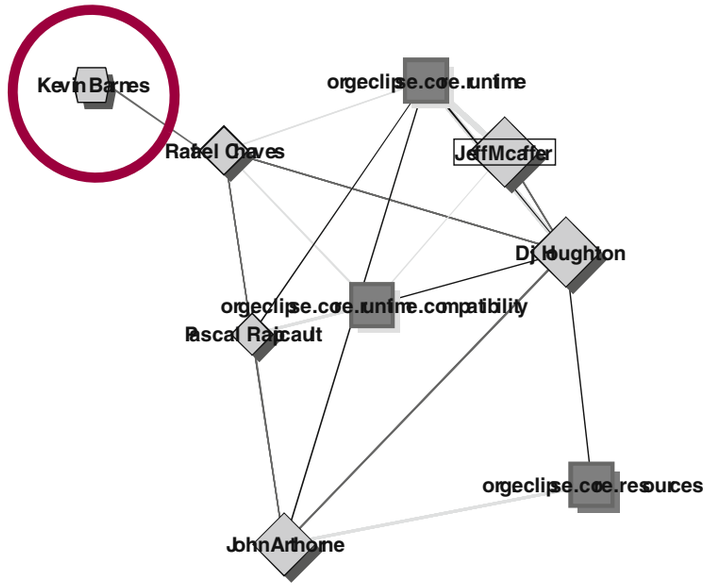
Code Ownership (& Alien Commits)



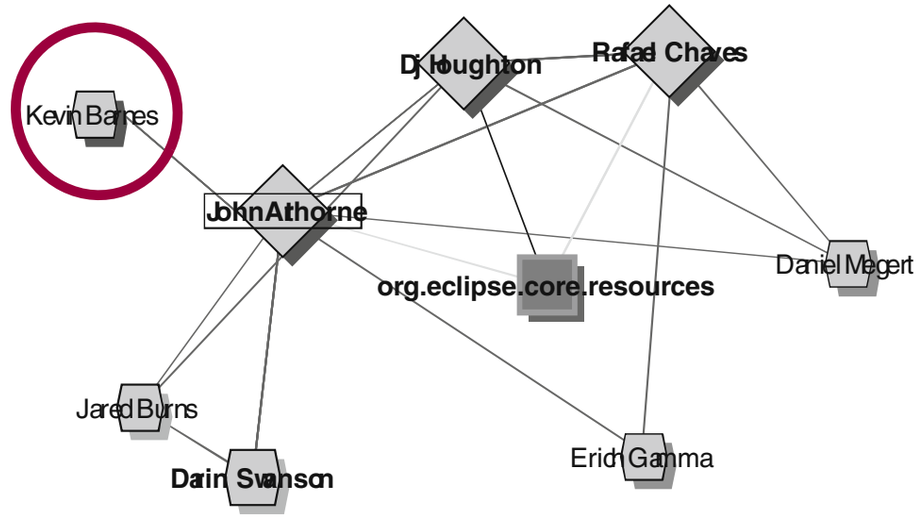
Key personalities



Socialization

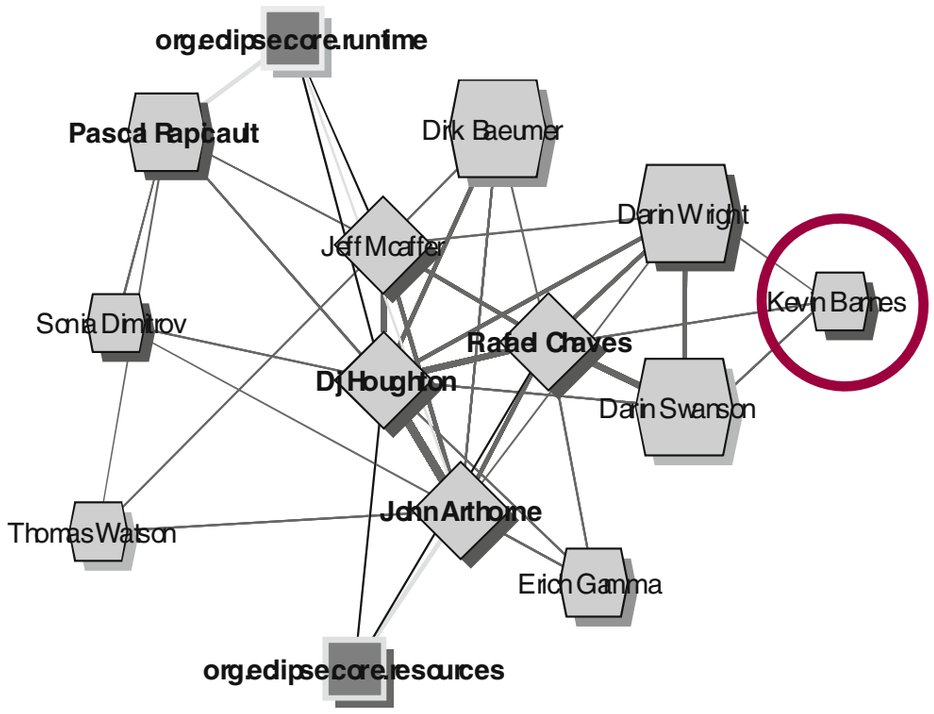


(a) 1st half of April 2004

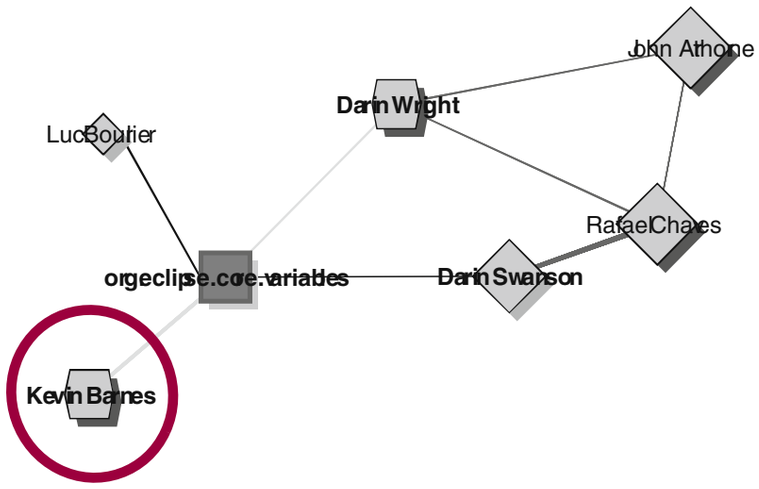


(b) 2nd half of April 2004

Socialization of Kevin Barness in the Eclipse Platform Core project

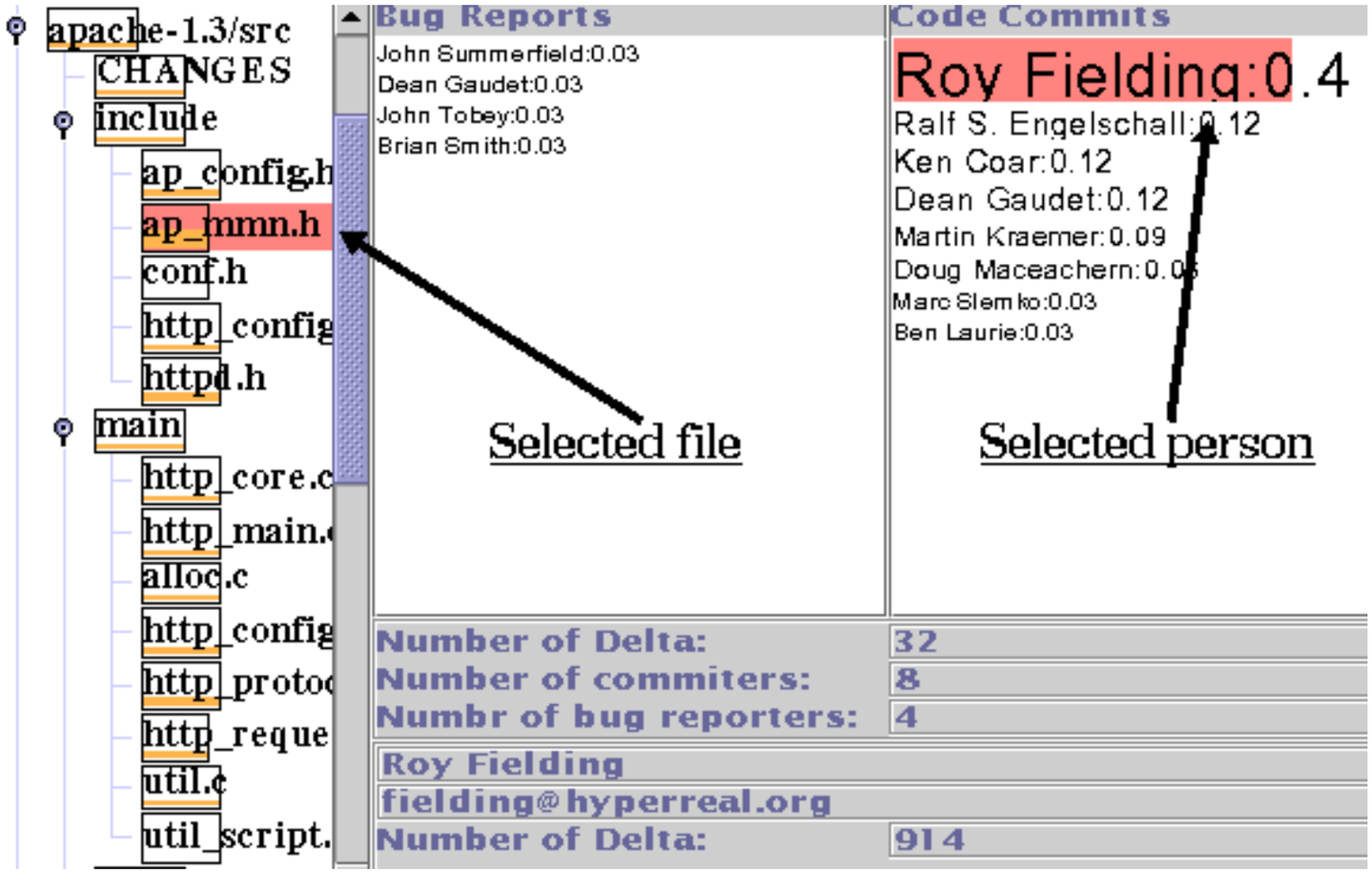


(c) 1st half of June 2004



(d) 2nd half of June 2004

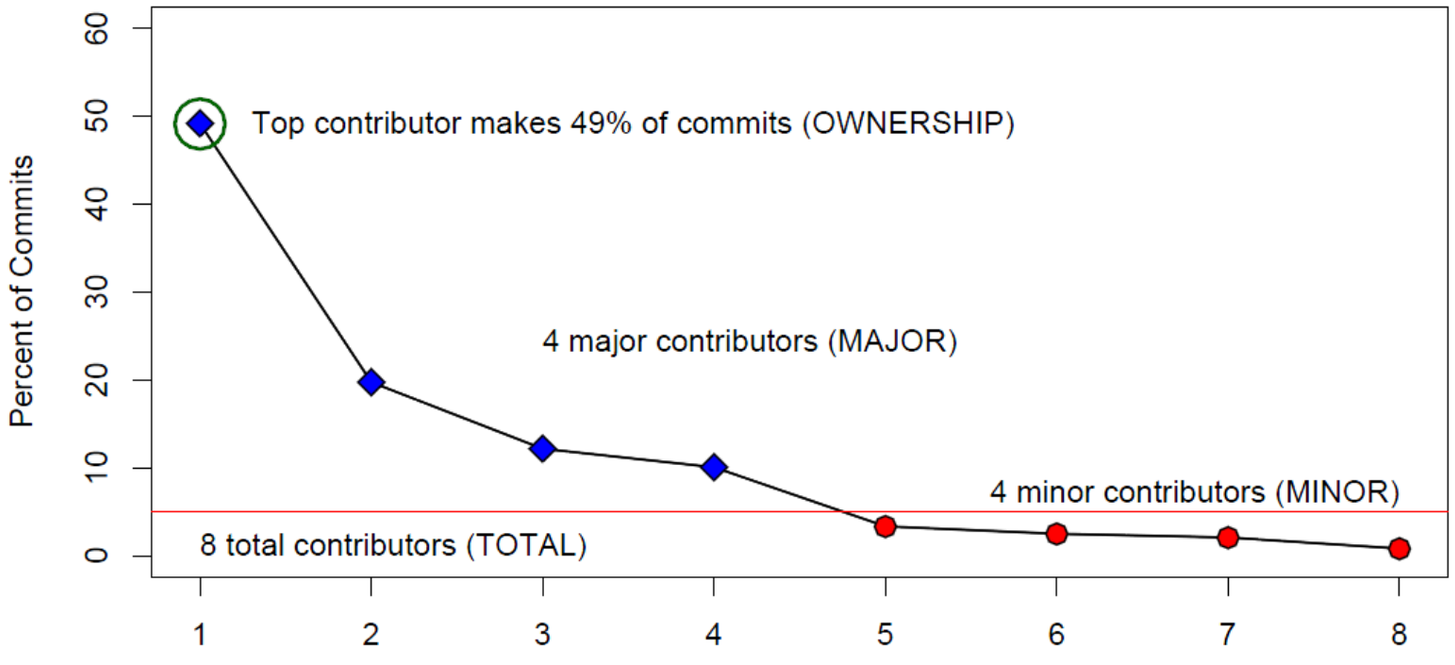
Expertise Browser



Used within a geographically dispersed team

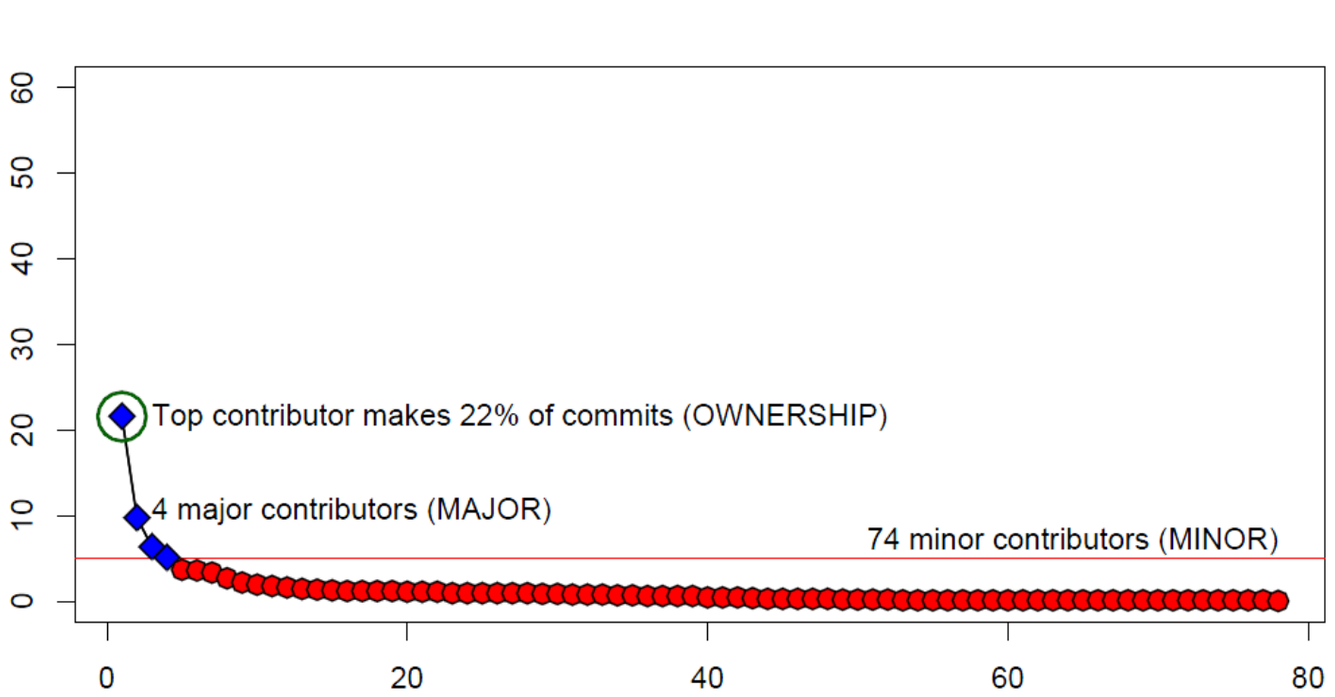
- 120 developers at two sites (Germany and England) grew to 250 developers (incl. satellite site in France)
- satellite teams: locate expertise
- established teams: who is doing what ?

Code Ownership vs. Code Quality



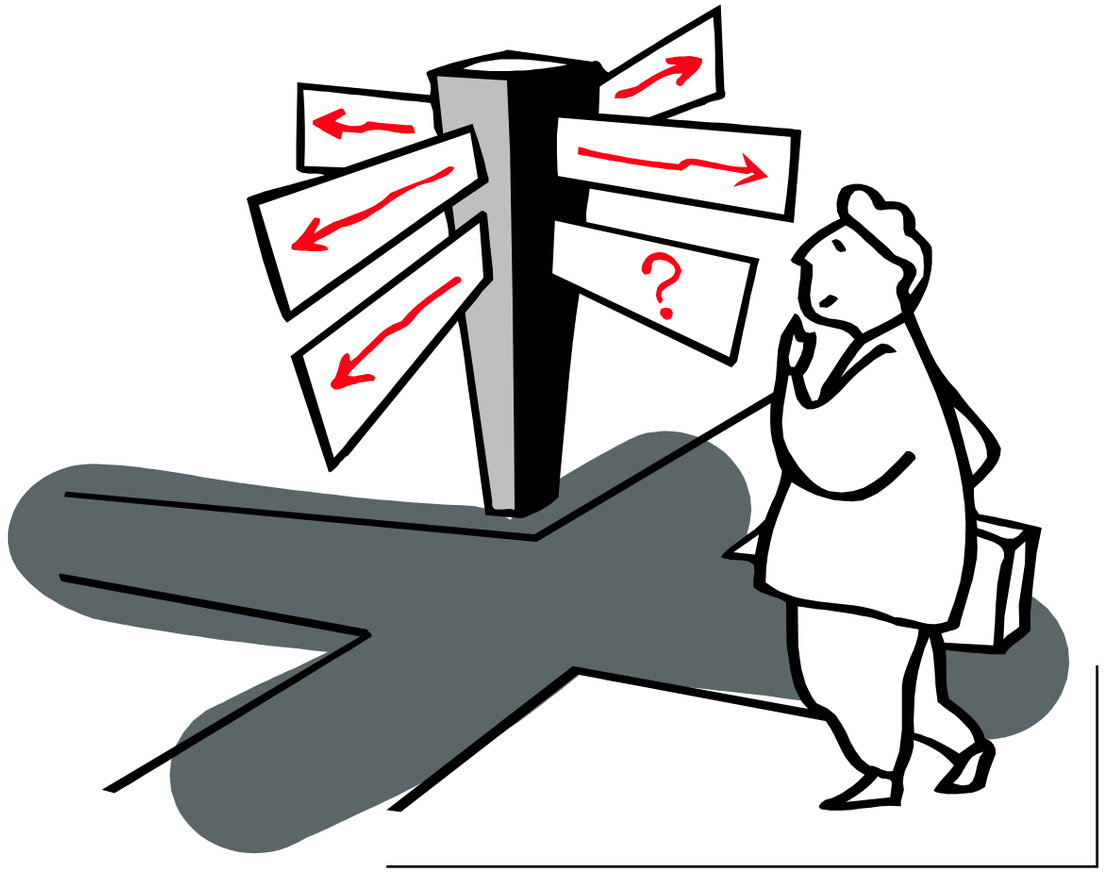
Software components with a high level of ownership will have fewer failures than components with lower top ownership levels.

Software components with many minor contributors will have more failures than software components that have fewer.

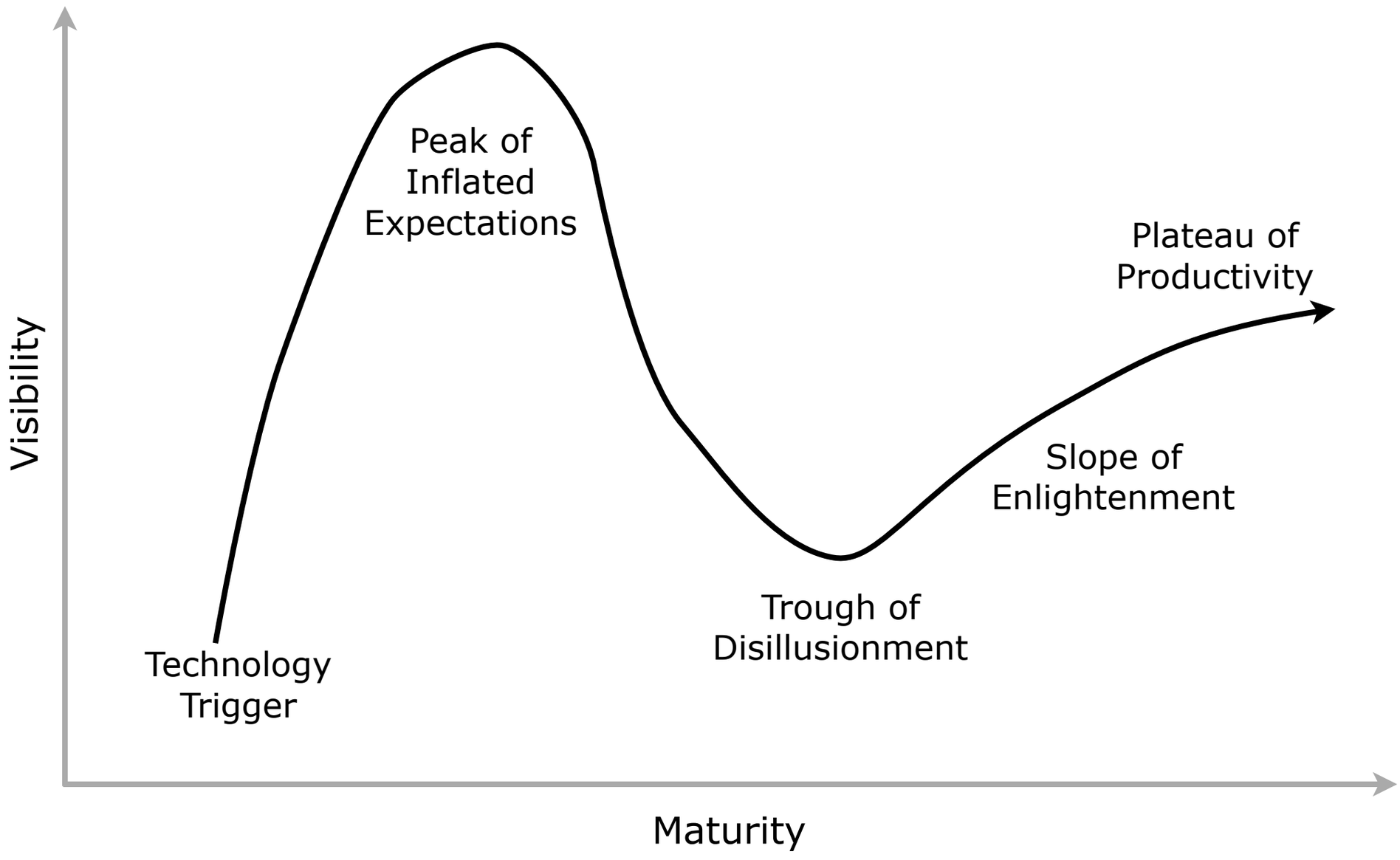


Data from Windows Vista and Windows 7

Table Of Contents



Hype Cycle

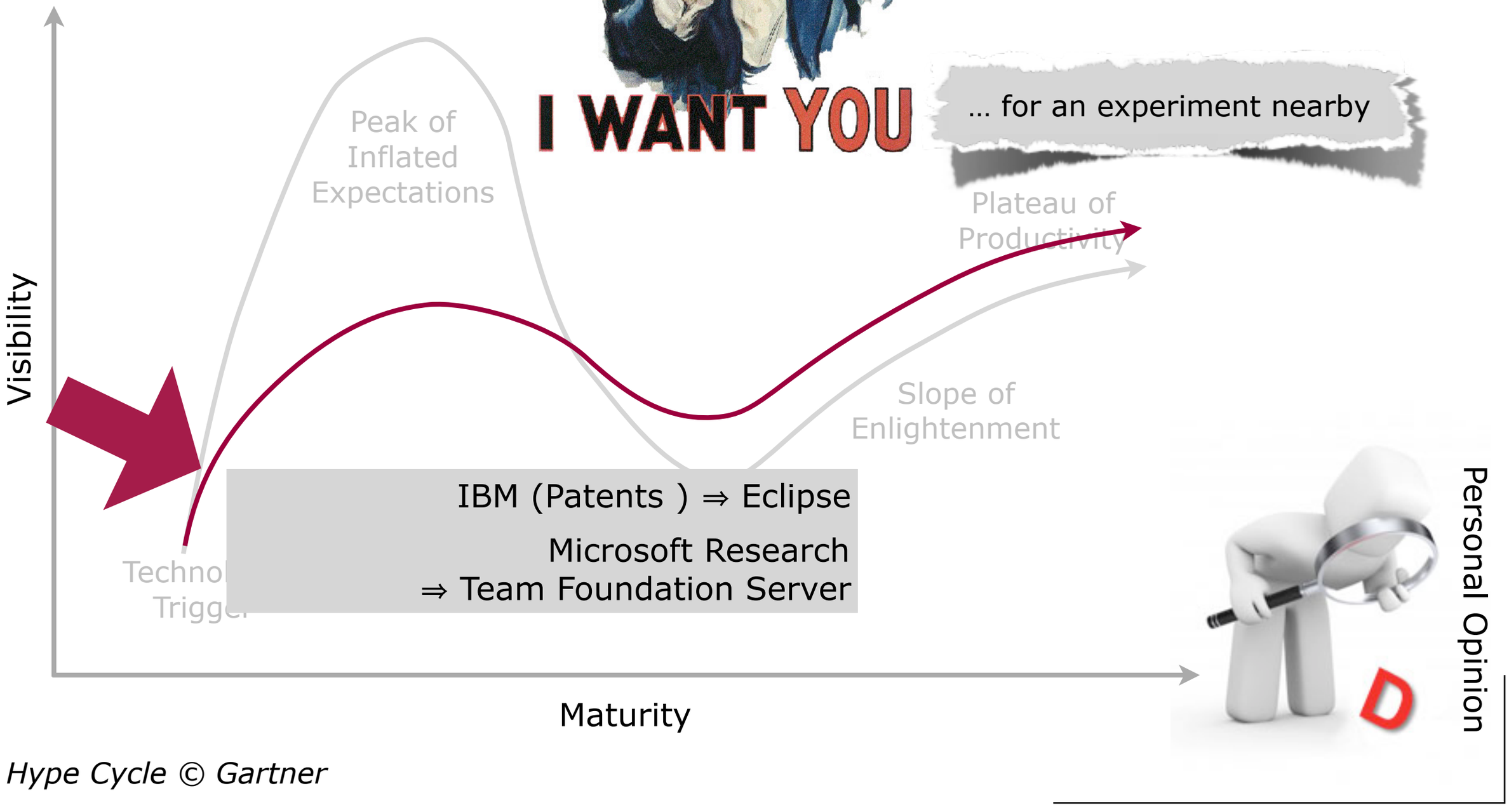


Hype Cycle © Gartner

The Future ?

I WANT YOU

… for an experiment nearby



Hype Cycle © Gartner

Bibliography

Tests

- Andy Zaidman, Bart Van Rompaey, Arie van Deursen, and Serge Demeyer, *Studying the co-evolution of production and test code in open source and industrial developer test processes through repository mining*, In International Journal on Empirical Software Engineering, Volume 16, Number 3, pp. 325--364, 2011

Bugs

- John Anvik, Lyndon Hiew, and Gail C. Murphy. 2006. *Who should fix this bug?* In Proceedings of the 28th international conference on Software engineering (ICSE '06). ACM, New York, NY, USA, 361-370.
- Cathrin Weiss, Rahul Premraj, Thomas Zimmermann, and Andreas Zeller. 2007. *How Long Will It Take to Fix This Bug?* In Proceedings of the Fourth International Workshop on Mining Software Repositories (MSR '07). IEEE Computer Society, Washington, DC, USA
- Ahmed Lamkanfi, Serge Demeyer, Emanuel Giger, and Bart Goethals, *Predicting the Severity of a Reported Bug*, In Proceedings MSR'10 (7th IEEE Working Conference on Mining Software Repositories), May, IEEE Press, 2010

Expertise (Social Networks)

- Martin Pinzger and Harald C. Gall. *Dynamic Analysis of Communication and Collaboration in OSS Projects*. Chapter 13 In Collaborative Software Engineering, I. Mistrík, J. Grundy, A. van der Hoek, J. Whitehead (eds.), pp. 265-284, Springer, 2010.
- Christian Bird, Nachiappan Nagappan, B. Murphy, H. Gall, and P. Devanbu. *Don't Touch My Code! Examining the Effects of Ownership on Software Quality*. In Proceedings of the the eighth joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering, Szeged, Hungary, 2011.
- Audris Mockus and James Herbsleb. *Expertise browser: A quantitative approach to identifying expertise.* In 2002 International Conference on Software Engineering, pages 503-512, Orlando, Florida, May 19-25 2002. ACM Press.