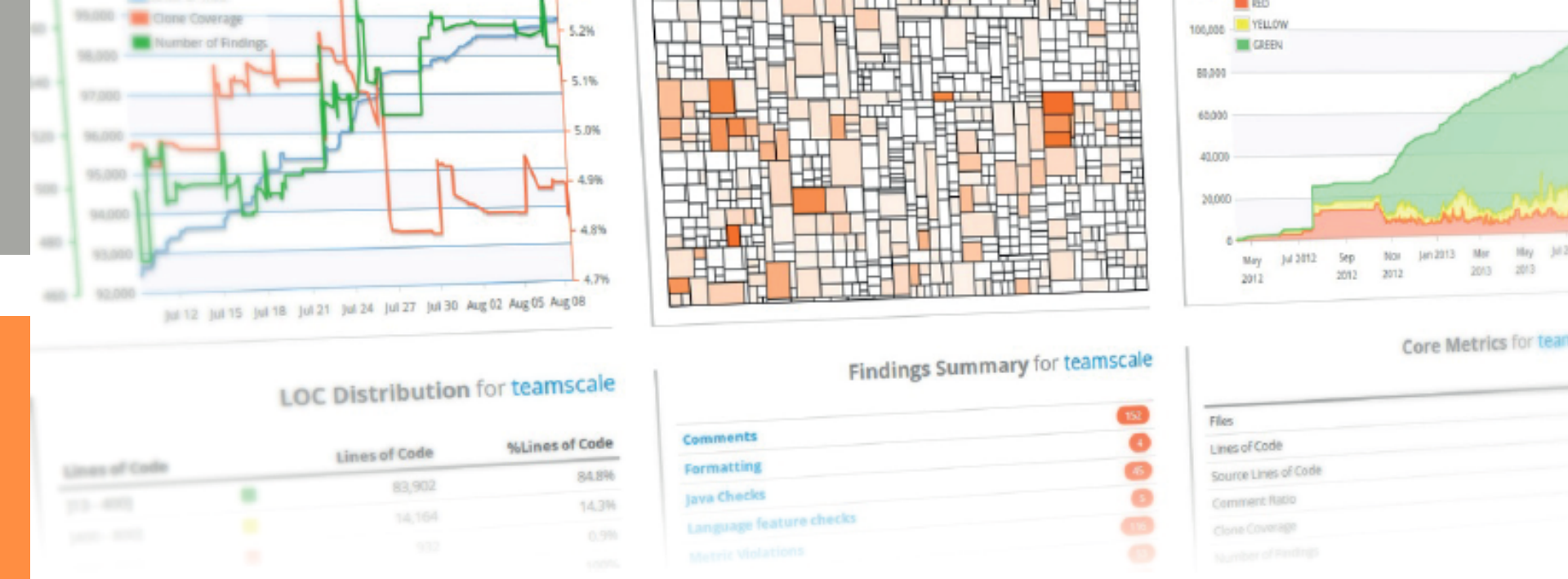


Was kann man aus der Versionshistorie eines Softwaresystems lernen?

JUG Hamburg - 15. Februar 2017

Dr. Dennis Pagano
@dennispagano

CQSE
Continuous Quality in Software Engineering

Ausschnitt aus meiner Historie

Forschung

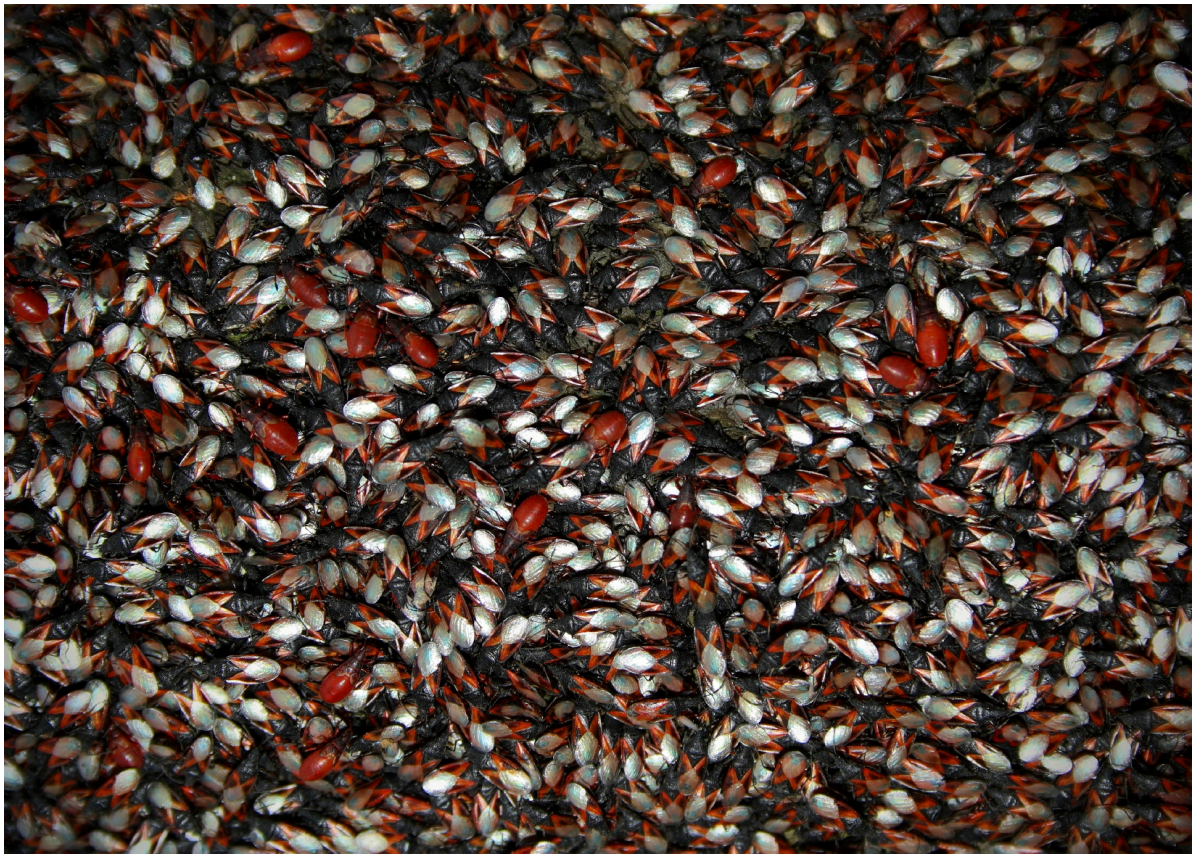
- Analyse von Open Source Communities
- Historie von Code und Kommunikationsartefakten
- Kontinuierliches Einbeziehen von Nutzerfeedback in die Entwicklung

Praxis

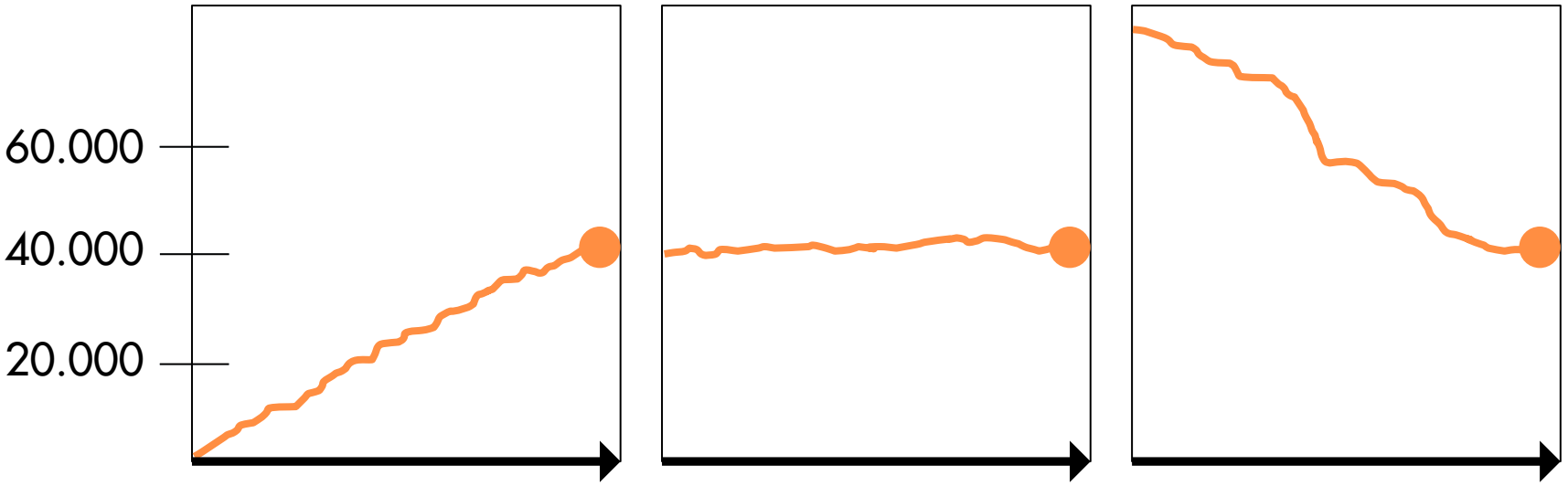
- Software Audits: Qualitätsbewertung
- Quality und Test Control
- Entwicklung: Historisierte Qualitätsanalysen in Echtzeit

CQSE

Hilfe, unser System hat 42.000 Findings!

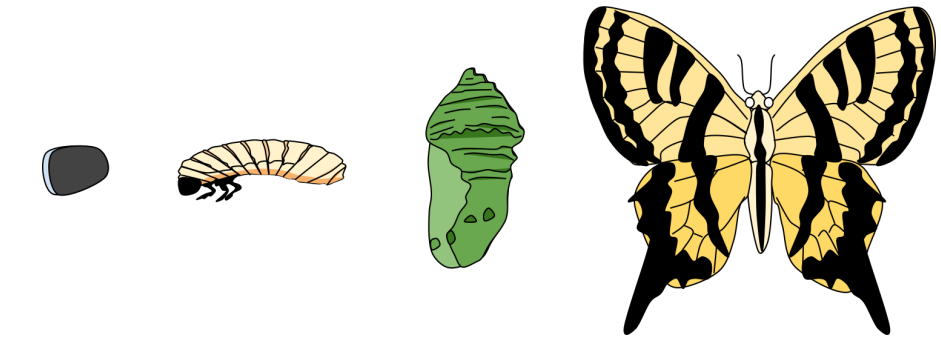


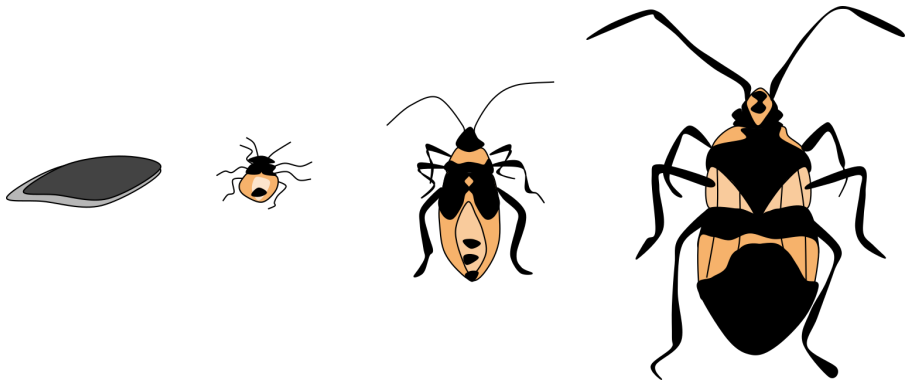
Sind 42.000 Findings schlecht?

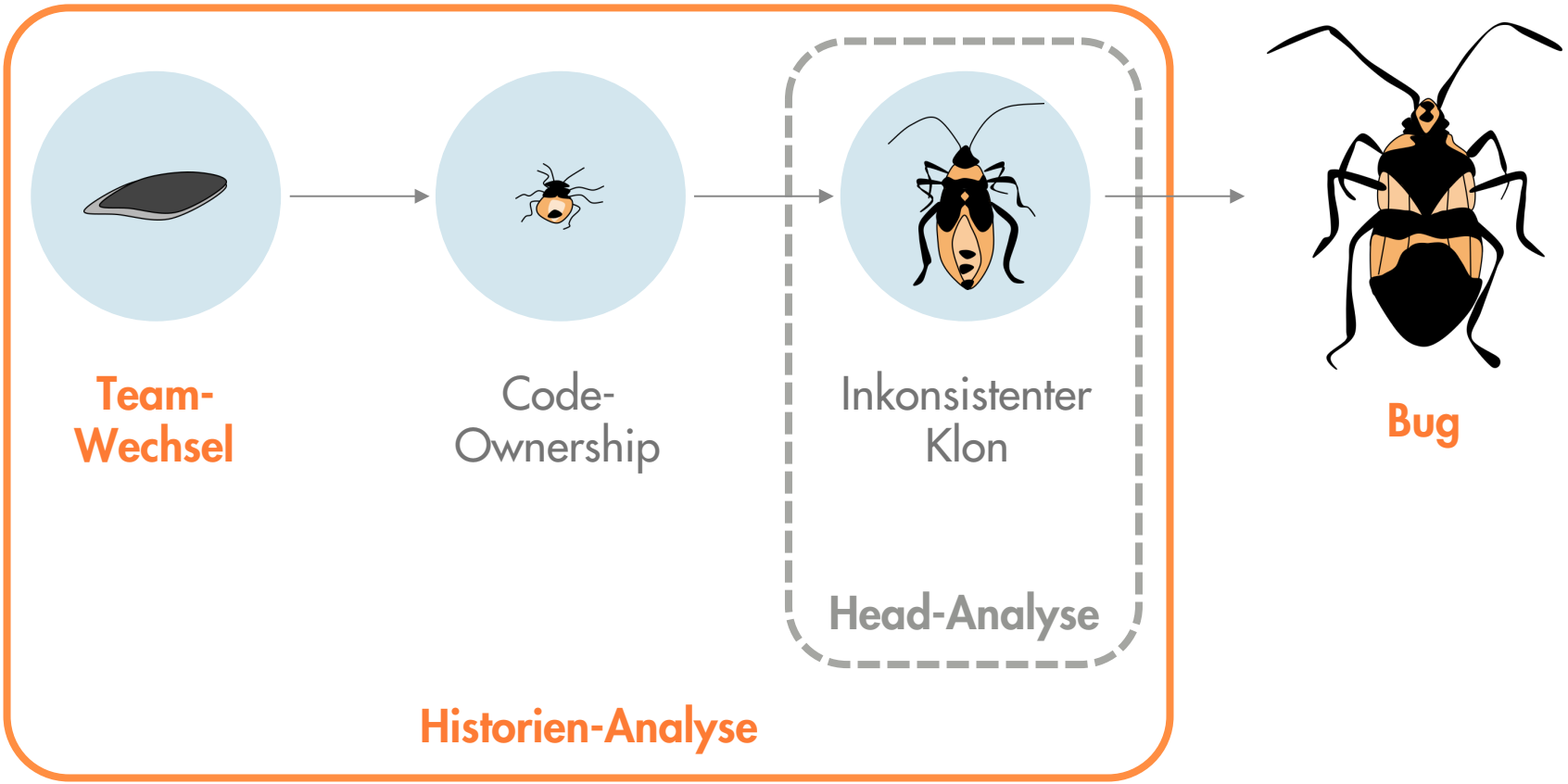


Qualitätsziel?

Die Entstehung der aktuellen Version ist entscheidend für die richtige Reaktion



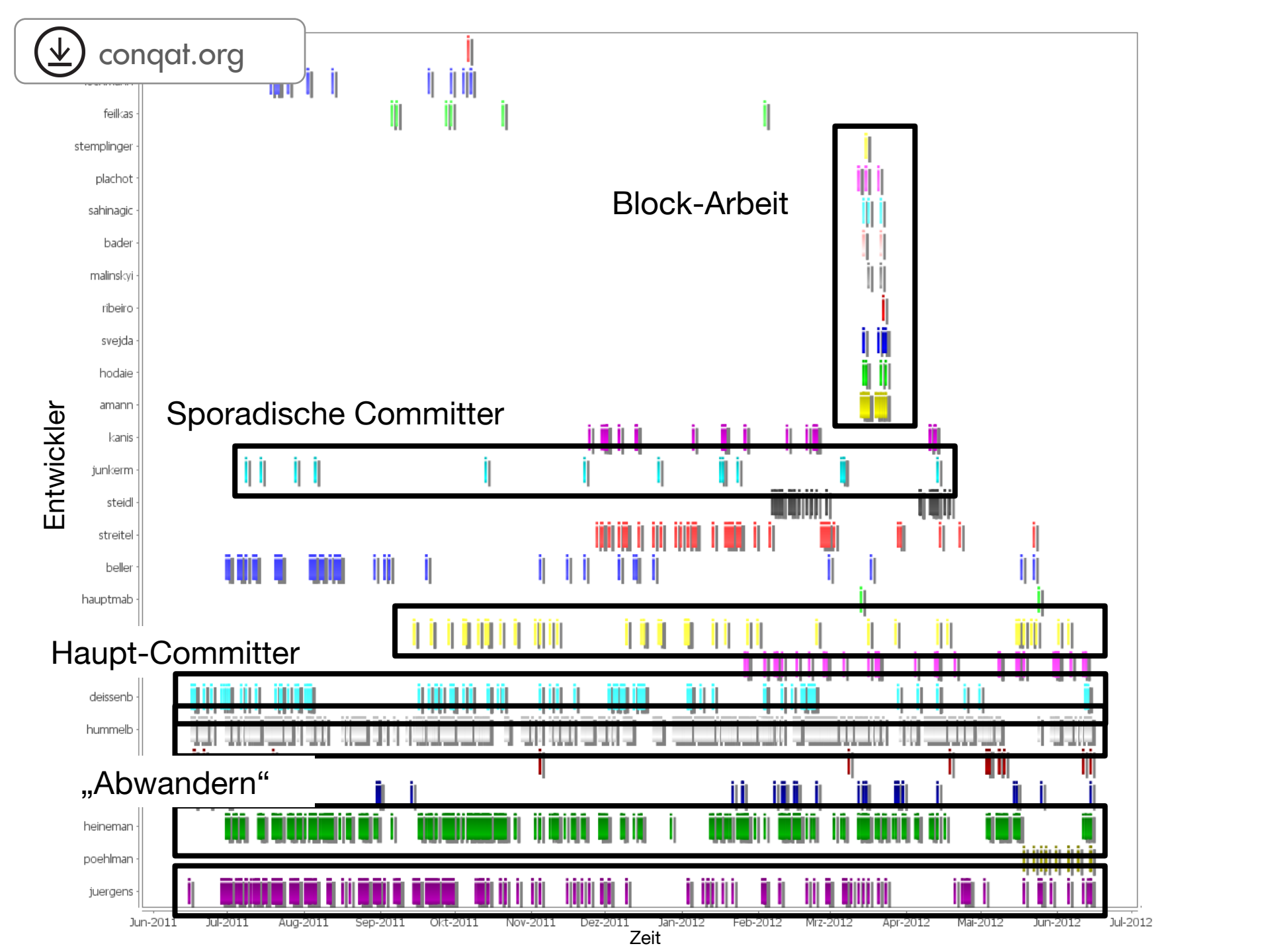


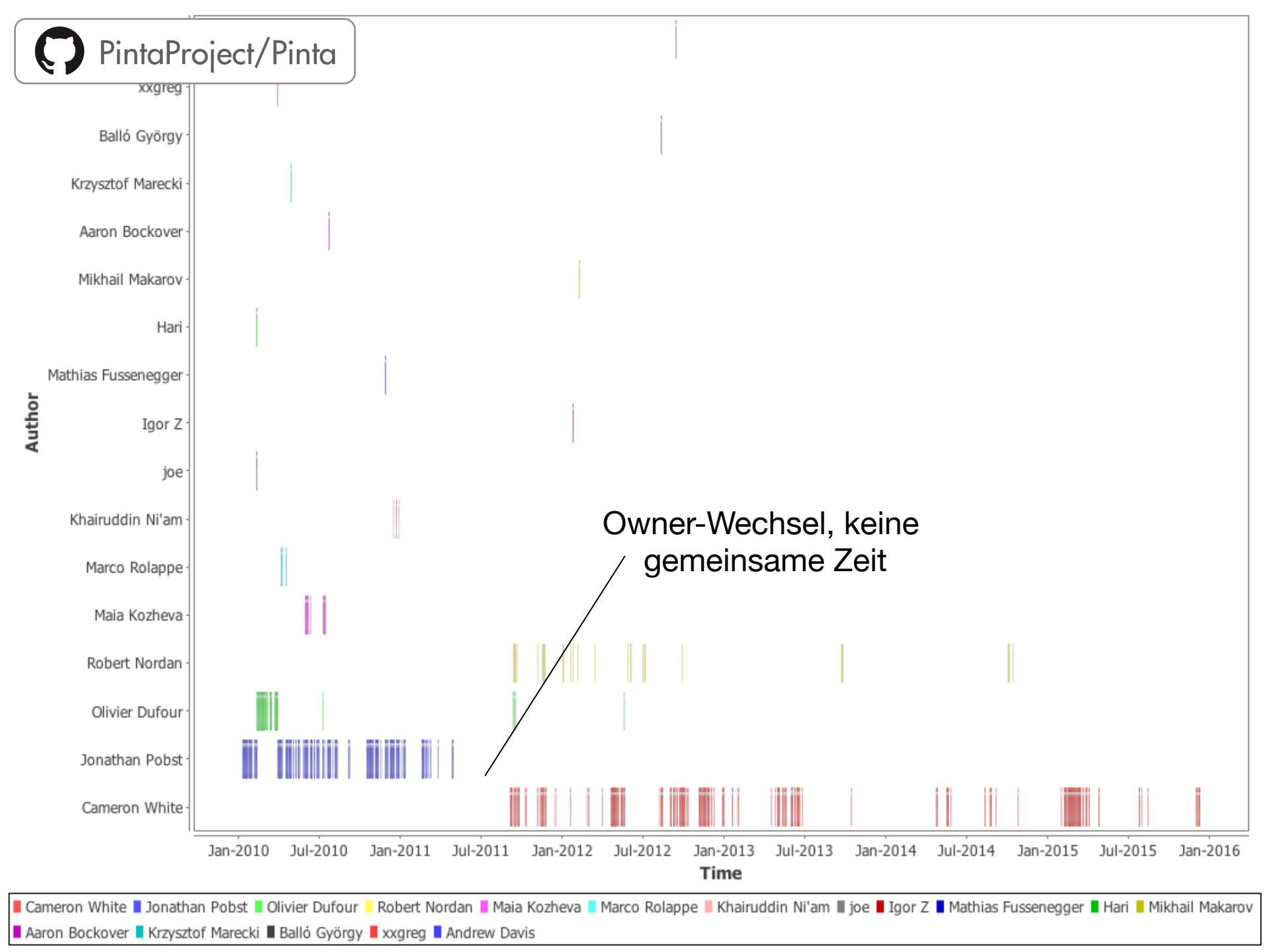

Team-
Wechsel
Code-
Ownership
Inkonsistenter
Klon
Head-Analyse
Historien-Analyse
Bug

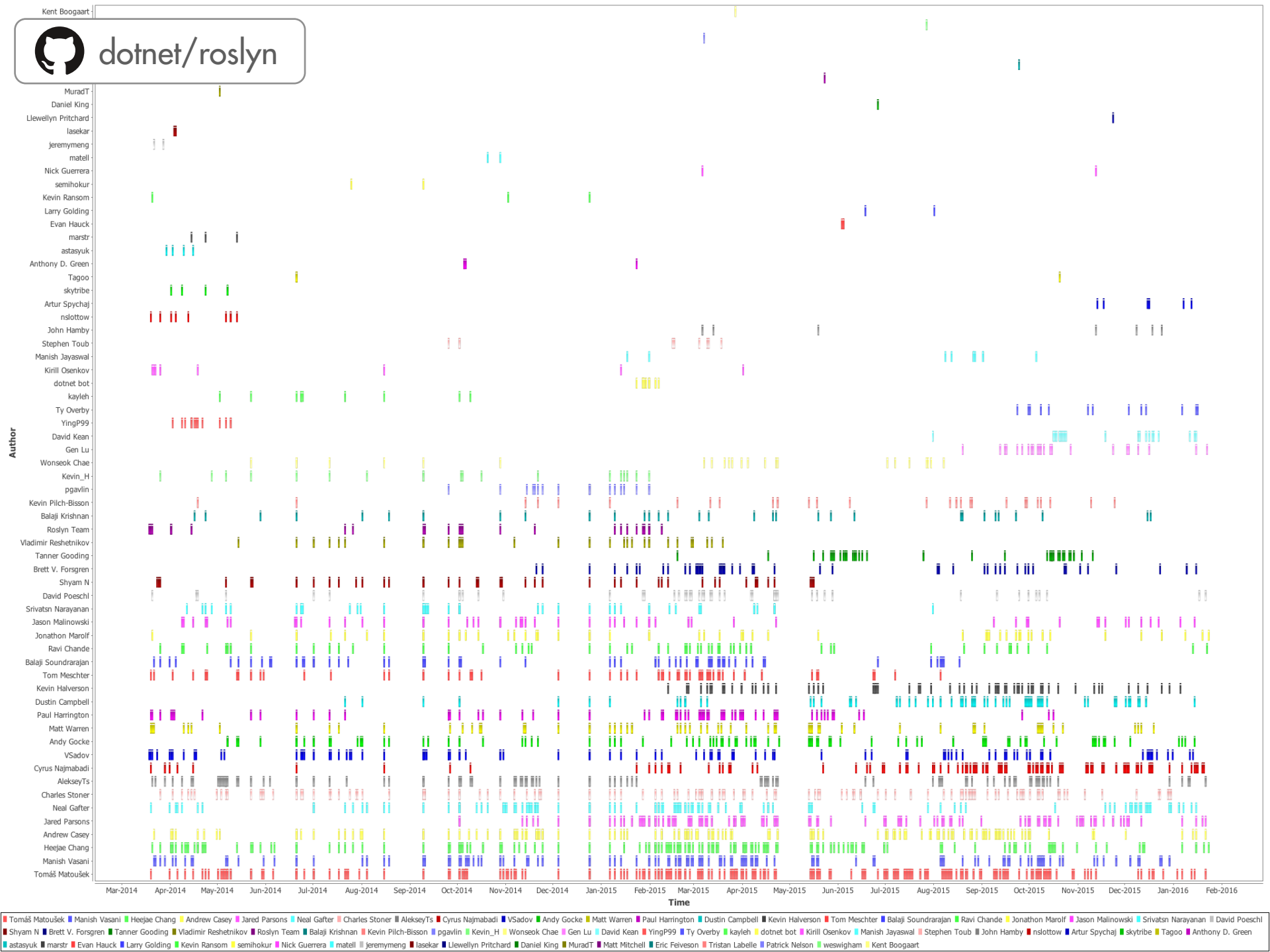
Agenda

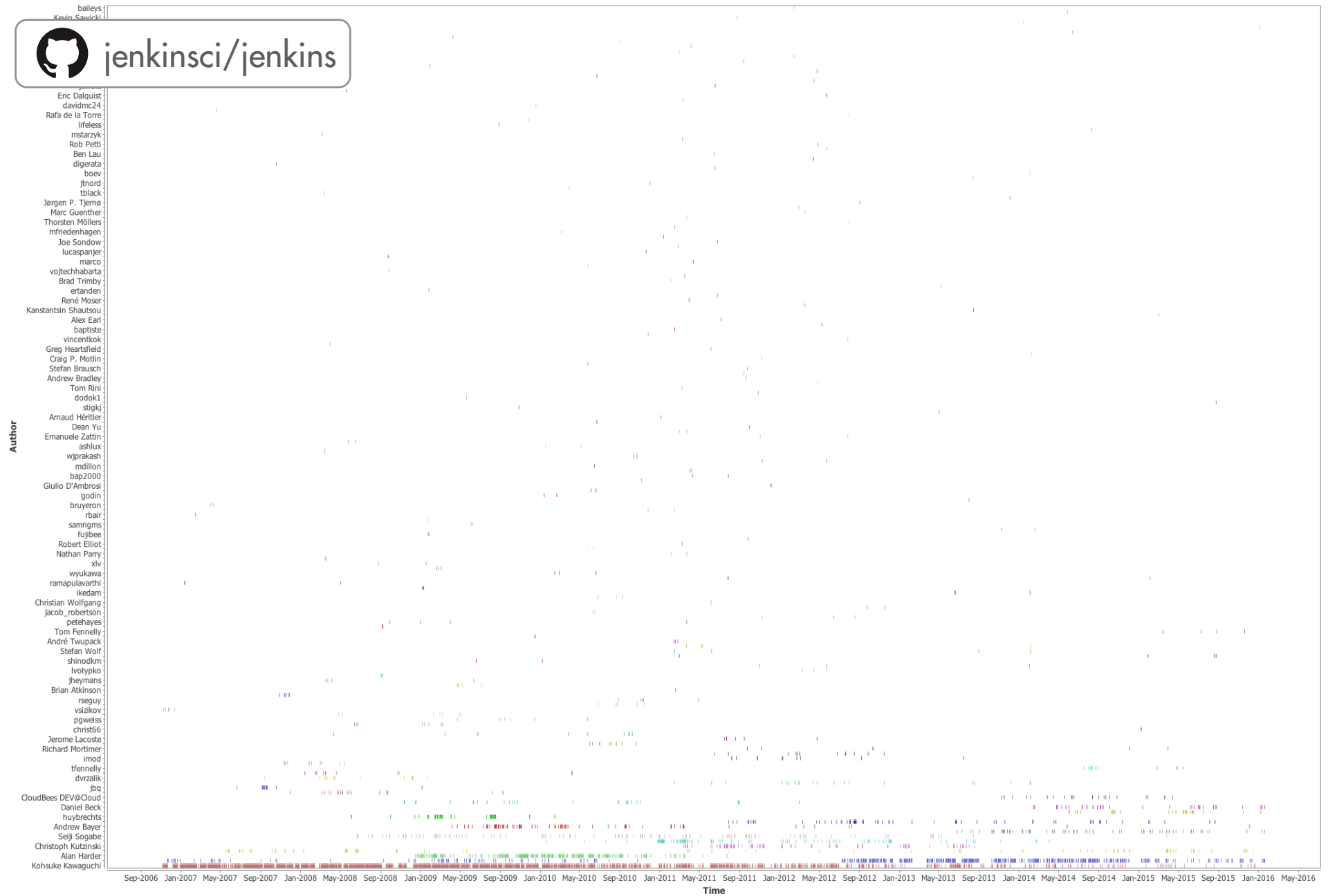
1. Welche Probleme verrät die Historie?
2. Was kann man im eigenen Projekt machen?

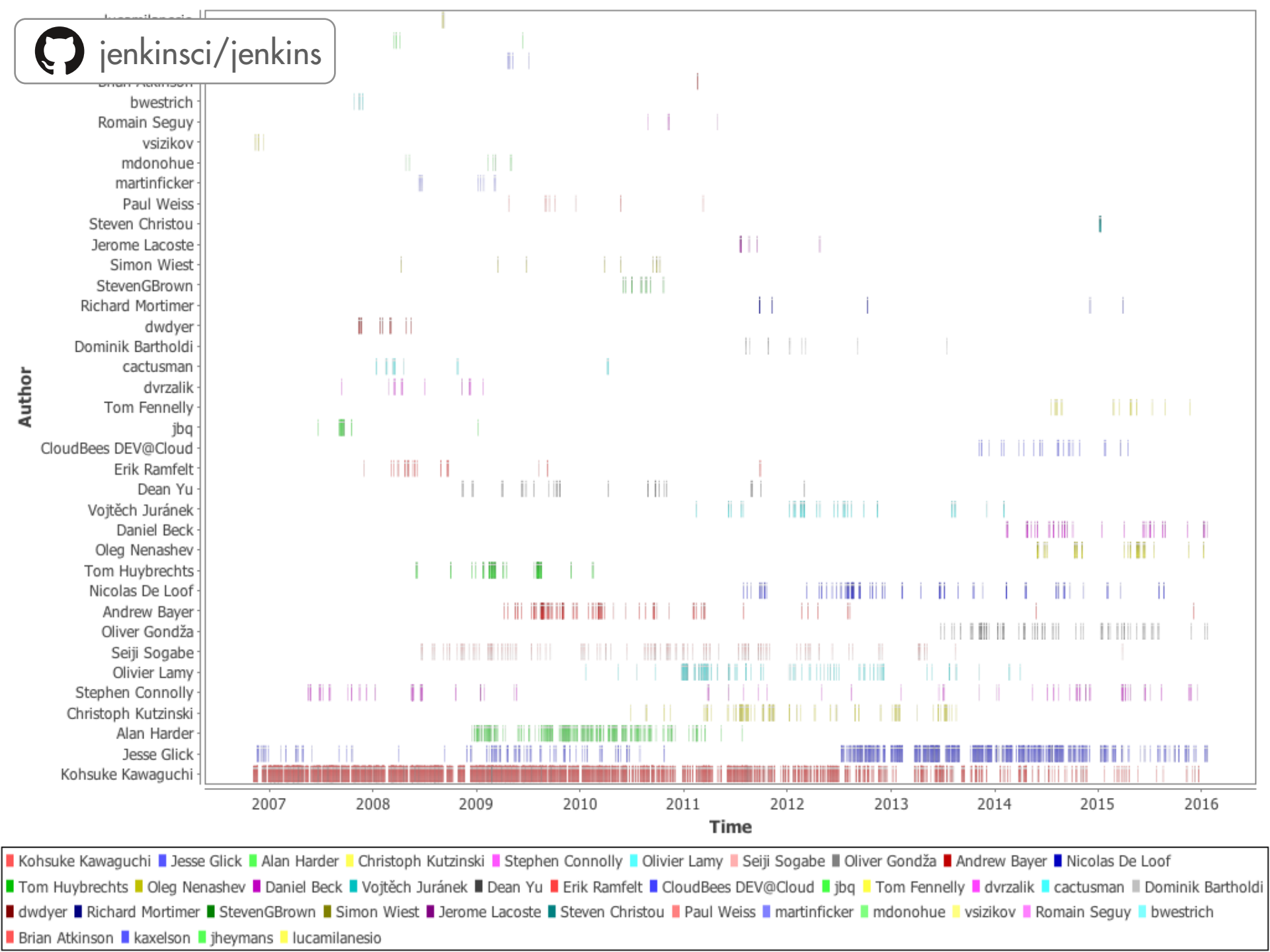
Die Historie verrät, wie sich das Team entwickelt hat.


conqat.org
Block-Arbeit
Sporadische Committer
Haupt-Committer
„Abwandern“
Entwickler
feilkas
stemplinger
plachot
sahinagic
bader
malinskyi
ribeiro
svejda
hodaie
amann
kanis
junkerm
steidl
streitel
beller
hauptmab
deissenb
hummelb
heineman
poehlman
juergens
Jun-2011
Jul-2011
Aug-2011
Sep-2011
Okt-2011
Nov-2011
Dez-2011
Jan-2012
Feb-2012
Mrz-2012
Apr-2012
Mai-2012
Jun-2012
Jul-2012
Zeit


PintaProject/Pinta
xxgreg
Balló György
Krzysztof Marecki
Aaron Bockover
Mikhail Makarov
Hari
Mathias Fussenegger
Igor Z
joe
Khairuddin Ni'am
Marco Rolappe
Maia Kozheva
Robert Nordan
Olivier Dufour
Jonathan Pobst
Cameron White
Author
Owner-Wechsel, keine gemeinsame Zeit
Jan-2010
Jul-2010
Jan-2011
Jul-2011
Jan-2012
Jul-2012
Jan-2013
Jul-2013
Jan-2014
Jul-2014
Jan-2015
Jul-2015
Jan-2016
Time
Cameron White
Jonathan Pobst
Olivier Dufour
Robert Nordan
Maia Kozheva
Marco Rolappe
Khairuddin Ni'am
joe
Igor Z
Mathias Fussenegger
Hari
Mikhail Makarov
Aaron Bockover
Krzysztof Marecki
Balló György
xxgreg
Andrew Davis

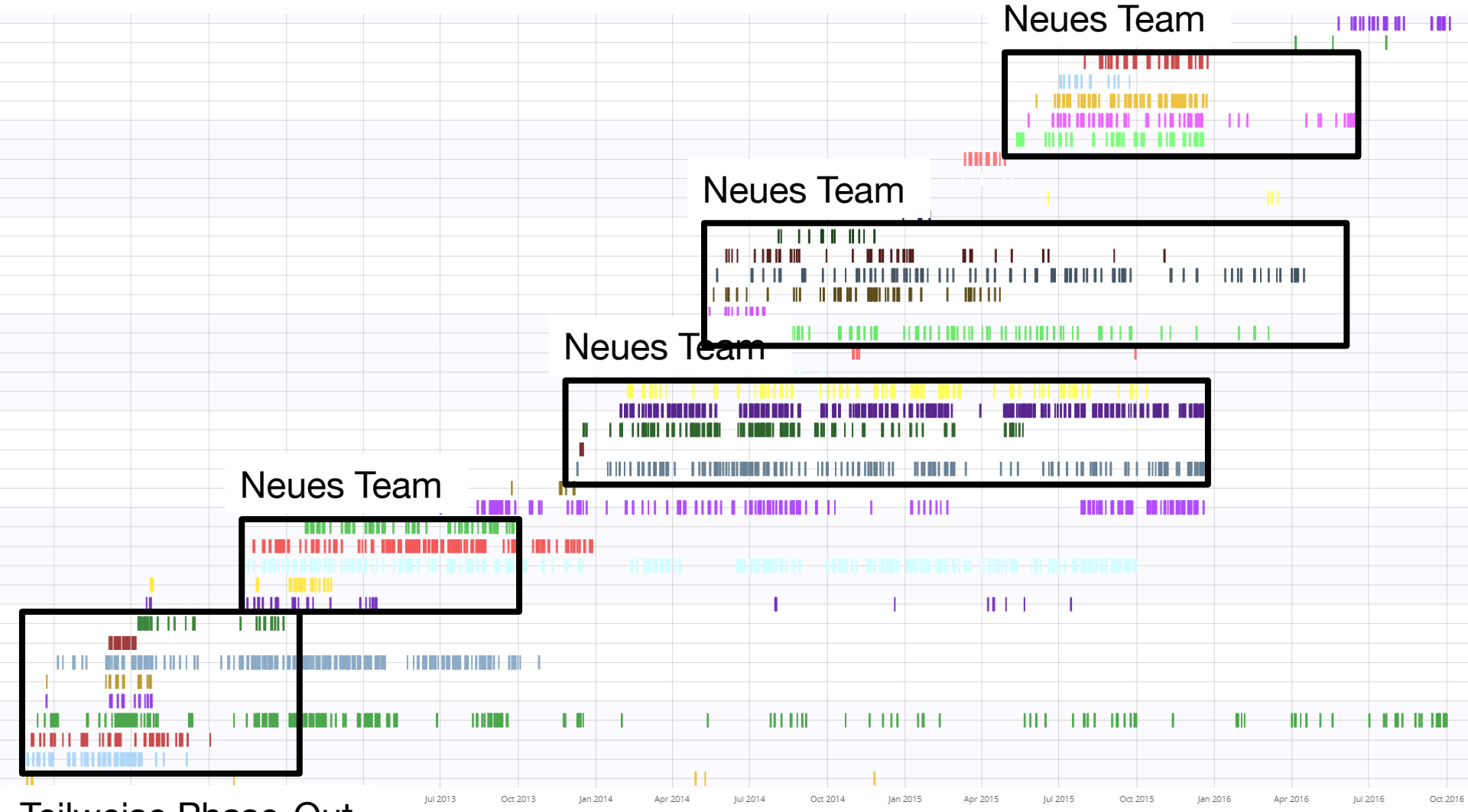

dotnet/roslyn
Author
Time
Mar-2014
Apr-2014
May-2014
Jun-2014
Jul-2014
Aug-2014
Sep-2014
Oct-2014
Nov-2014
Dec-2014
Jan-2015
Feb-2015
Mar-2015
Apr-2015
May-2015
Jun-2015
Jul-2015
Aug-2015
Sep-2015
Oct-2015
Nov-2015
Dec-2015
Jan-2016
Feb-2016
Tomáš Matoušek Manish Vasani Heejae Chang Andrew Casey Jared Parsons Neal Gafter Charles Stoner AlekseyTs Cyrus Najmabadi VSadov Andy Gocke Matt Warren Paul Harrington Dustin Campbell Kevin Halverson Tom Meschter Balaji Soundrarajan Ravi Chande Jonathon Marolf Jason Malinowski Srivatsn Narayanan David Poeschl
Shyam N Brett V. Forsgren Tanner Gooding Vladimir Reshetnikov Roslyn Team Balaji Krishnan Kevin Pilch-Bisson pgavlin Kevin_H Wonseok Chae Gen Lu David Kean YingP99 Ty Overby kayleh dotnet bot Kirill Osenkov Manish Jayaswal Stephen Toub John Hamby nslottow Artur Spychaj skytribe Tagoo Anthony D. Green
astasyuk marstr Evan Hauck Larry Golding Kevin Ransom semihokur Nick Guerrera matell jeremymeng lasekar Llewellyn Pritchard Daniel King MuradT Matt Mitchell Eric Feiveson Tristan Labelle Patrick Nelson weswigham Kent Boogaart


jenkinsci/jenkins
Author
baileys
Kevin Sawicki
Eric Dalquist
davidmc24
Rafa de la Torre
lifeless
mstarzyk
Rob Petti
Ben Lau
digerata
boev
jtnord
tblack
Jørgen P. Tjernø
Marc Guenther
Thorsten Möllers
mfriedenhagen
Joe Sondow
lucaspanjer
marco
vojtechhabarta
Brad Trimby
ertanden
René Moser
Kanstantsin Shautsou
Alex Earl
baptiste
vincentkok
Greg Heartsfield
Craig P. Motlin
Stefan Brausch
Andrew Bradley
Tom Rini
dodok1
stigkj
Arnaud Héritier
Dean Yu
Emanuele Zattin
ashlux
wjprakash
mdillon
bap2000
Giulio D'Ambrosi
godin
bruyeron
rbair
samngms
fujibee
Robert Elliot
Nathan Parry
xlv
wyukawa
ramapulavarthi
ikedam
Christian Wolfgang
jacob_robertson
petehayes
Tom Fennelly
André Twupack
Stefan Wolf
shinodkm
lvotypko
jheymans
Brian Atkinson
rseguy
vsizikov
pgweiss
christ66
Jerome Lacoste
Richard Mortimer
imod
tfennelly
dvrzalik
jbq
CloudBees DEV@Cloud
Daniel Beck
huybrechts
Andrew Bayer
Seiji Sogabe
Christoph Kutzinski
Alan Harder
Kohsuke Kawaguchi
Sep-2006
Jan-2007
May-2007
Sep-2007
Jan-2008
May-2008
Sep-2008
Jan-2009
May-2009
Sep-2009
Jan-2010
May-2010
Sep-2010
Jan-2011
May-2011
Sep-2011
Jan-2012
May-2012
Sep-2012
Jan-2013
May-2013
Sep-2013
Jan-2014
May-2014
Sep-2014
Jan-2015
May-2015
Sep-2015
Jan-2016
May-2016
Time
Kohsuke Kawaguchi
Jesse Glick
Alan Harder
Stephen Connolly
Christoph Kutzinski
Olivier Lamy
Seiji Sogabe
Oliver Gondža
Andrew Bayer
Nicolas De Loof
huybrechts
Oleg Nenashev
Daniel Beck
dty
CloudBees DEV@Cloud
redsolo
jbq
Vojtech Juranek
dvrzalik
cactusman
tfennelly
dwdyer
imod
vjuranek
Richard Mortimer
StevenGBrown
Jerome Lacoste
swiest
christ66
martinficker
pgweiss
mdonohue
vsizikov
kutzi
rseguy
bwestrich
Brian Atkinson
kaxelson
jheymans
lucamilanesio
lvotypko
Fred G
shinodkm
James Nord
Stefan Wolf
Ulli Hafner
André Twupack
lacostej
Tom Fennelly
jjasper
petehayes
OHTAKE Tomohiro
jacob_robertson
Erik Molekamp
Christian Wolfgang
km
ikedam
brucechapman
ramapulavarthi
Ryan Campbell
wyukawa
draco2k8
xlv
jasonchaffee
Nathan Parry
jpederzolli
Robert Elliot
Erik Ramfelt
fujibee
Harald Albers
samngms
rtyler
rbair
Jesse Farinacci
bruyeron
Vojtěch Juránek
godin
mode
Giulio D'Ambrosi
wolfs
bap2000
Martijn Baay
mdillon
Vincent Latombe
wjprakash
drulli
ashlux
frizbog
Emanuele Zattin
Tomer Cohen
Dean Yu
teilo
Arnaud Héritier
dnadolny
stigkj
Sam Van Oort
dodok1
Dominik Bartholdi
Tom Rini
Reynald Borer
Andrew Bradley
Stephen Ware
Stefan Brausch
kbertelson
Craig P. Motlin
Michael Strasser
Greg Heartsfield
elefevre
vincentkok
Alexandre Garnier
baptiste
David Reiss
Alex Earl
Johno Crawford
Kanstantsin Shautsou
Sami Tikka
René Moser
Richard Bywater
ertanden
olivergondza
Brad Trimby
Olav Reinert
vojtechhabarta
mcrooney
marco
david_calavera
lucaspanjer
Nigel Magnay
Joe Sondow
mleinart
mfriedenhagen
Ken Bertelson
Thorsten Möllers
Mike Robinet
Marc Guenther
Andrew Melo
Jørgen P. Tjernø
Josh Gibbs
tblack
vlatombe
jtnord
Michael Clarke
boev
dwelden
digerata
Bruno P. Kinoshita
Ben Lau
Dave Hunt
Rob Petti
Paul Weiss
mstarzyk
Ben McCann
lifeless
javadude
Rafa de la Torre
vglushenkov
davidmc24
Paul Sandoz
Eric Dalquist
davidseymore
jsiirola
Bruno Meneguello
GLundh
Christoph Thelen
f_cavarretta
Ciprian Ciubotariu
Hayato ITO
Stuart McCulloch
Lucien Weller
willemv
jhm
Surya Gaddipati
valentina
Jonathan A. Sternberg
Kevin Sawicki
Yoann Dubreuil
baileys


jenkinsci/jenkins
bwestrich
Romain Seguy
vsizikov
mdonohue
martinficker
Paul Weiss
Steven Christou
Jerome Lacoste
Simon Wiest
StevenGBrown
Richard Mortimer
dwdyer
Dominik Bartholdi
cactusman
dvrzalik
Tom Fennelly
jbq
CloudBees DEV@Cloud
Erik Ramfelt
Dean Yu
Vojtěch Juránek
Daniel Beck
Oleg Nenashev
Tom Huybrechts
Nicolas De Loof
Andrew Bayer
Oliver Gondža
Seiji Sogabe
Olivier Lamy
Stephen Connolly
Christoph Kutzinski
Alan Harder
Jesse Glick
Kohsuke Kawaguchi
Author
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
Time
Kohsuke Kawaguchi
Jesse Glick
Alan Harder
Christoph Kutzinski
Stephen Connolly
Olivier Lamy
Seiji Sogabe
Oliver Gondža
Andrew Bayer
Nicolas De Loof
Tom Huybrechts
Oleg Nenashev
Daniel Beck
Vojtěch Juránek
Dean Yu
Erik Ramfelt
CloudBees DEV@Cloud
jbq
Tom Fennelly
dvrzalik
cactusman
Dominik Bartholdi
dwdyer
Richard Mortimer
StevenGBrown
Simon Wiest
Jerome Lacoste
Steven Christou
Paul Weiss
martinficker
mdonohue
vsizikov
Romain Seguy
bwestrich
Brian Atkinson
kaxelson
jheymans
lucamilanesio

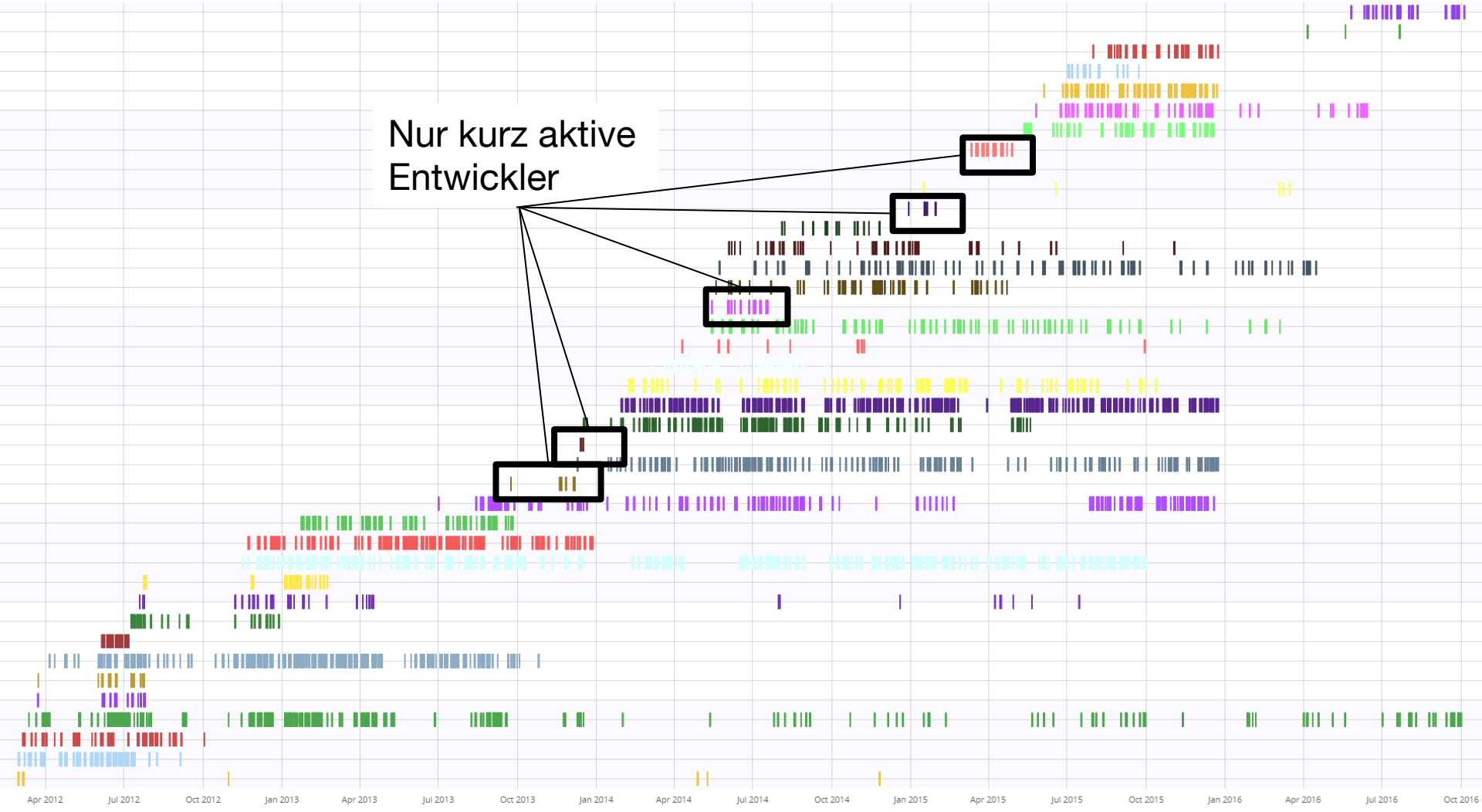
Kundenprojekt (C#)

Kundenprojekt (C#)

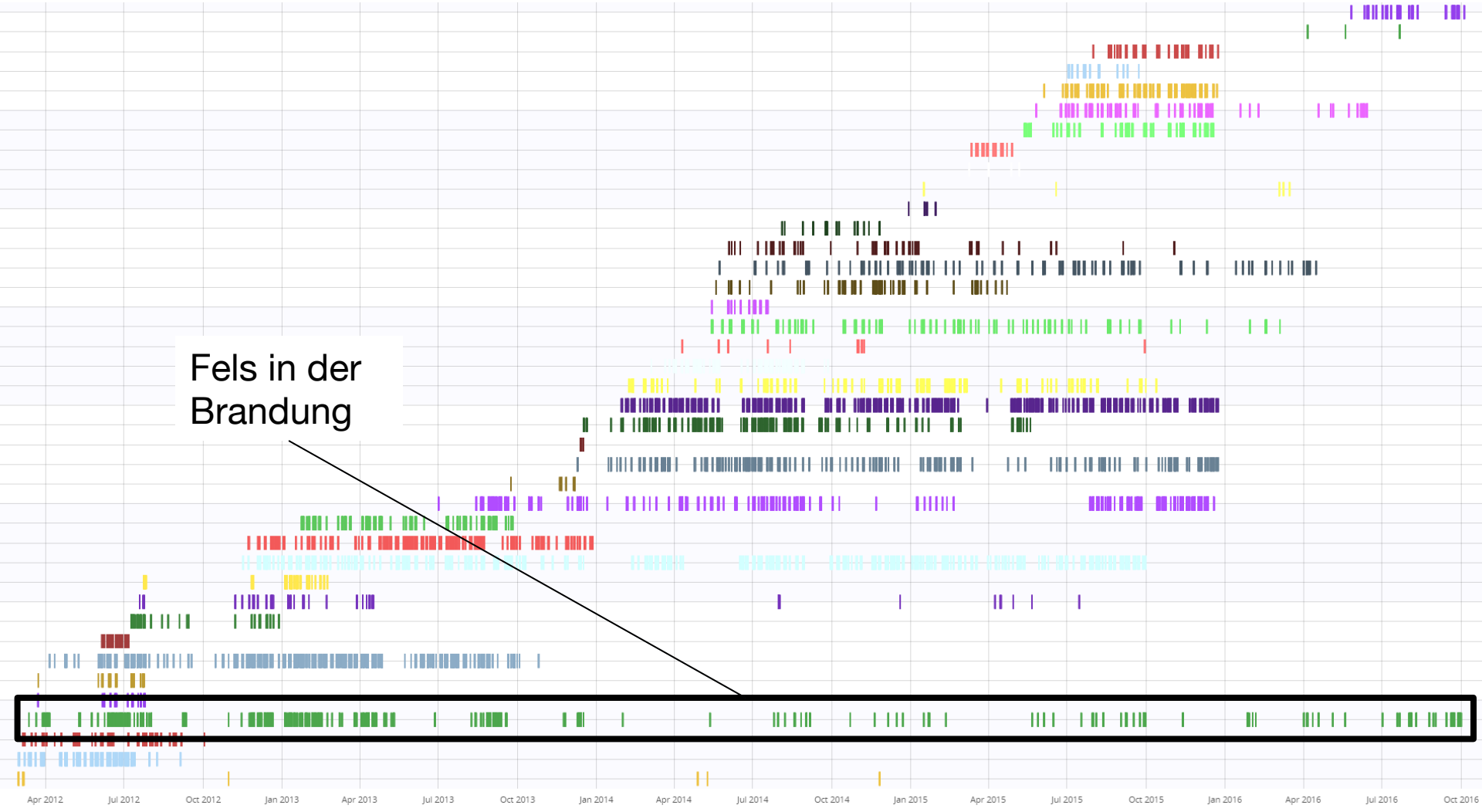


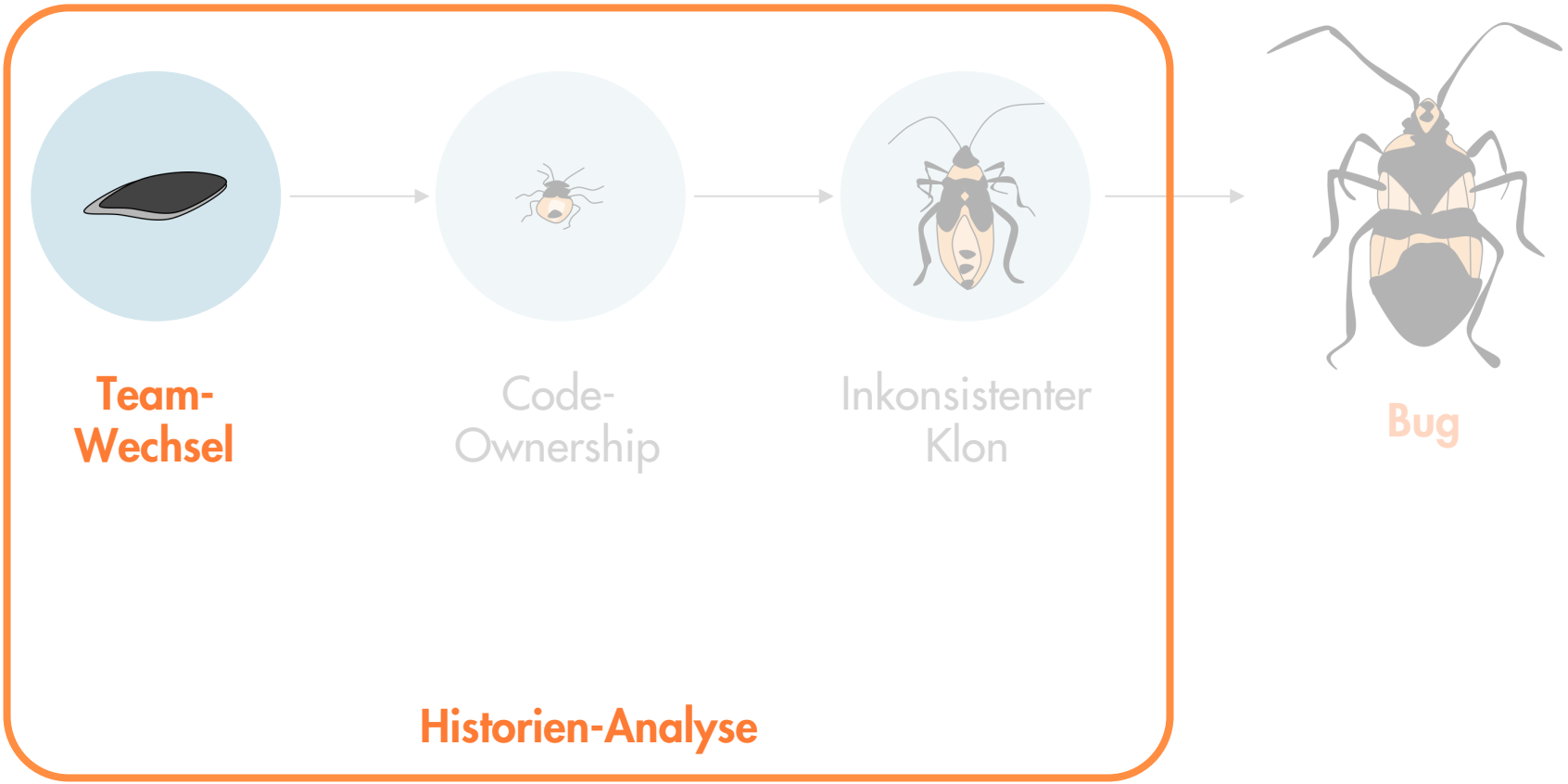
Teilweise Phase-Out

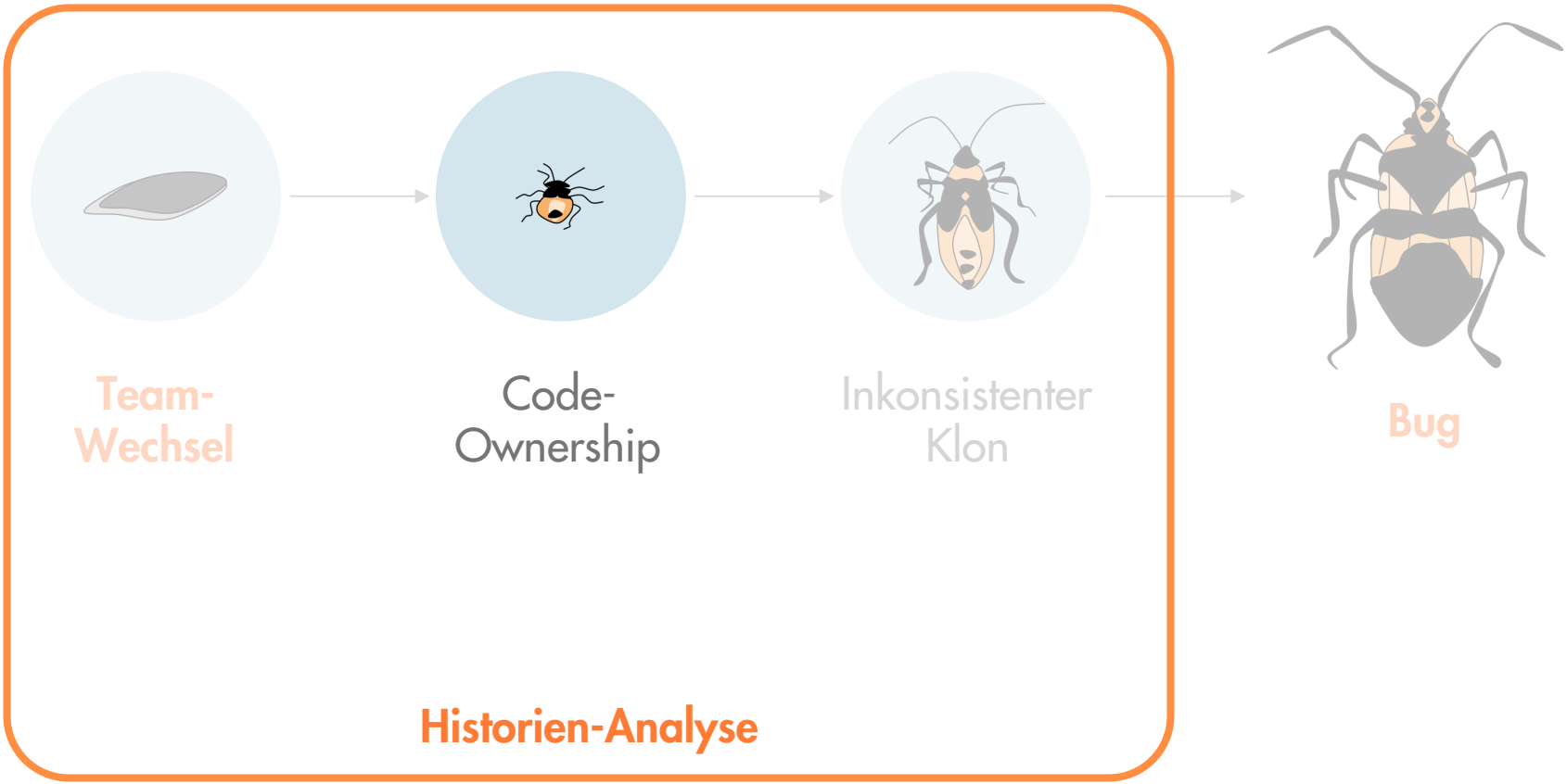
Kundenprojekt (C#)



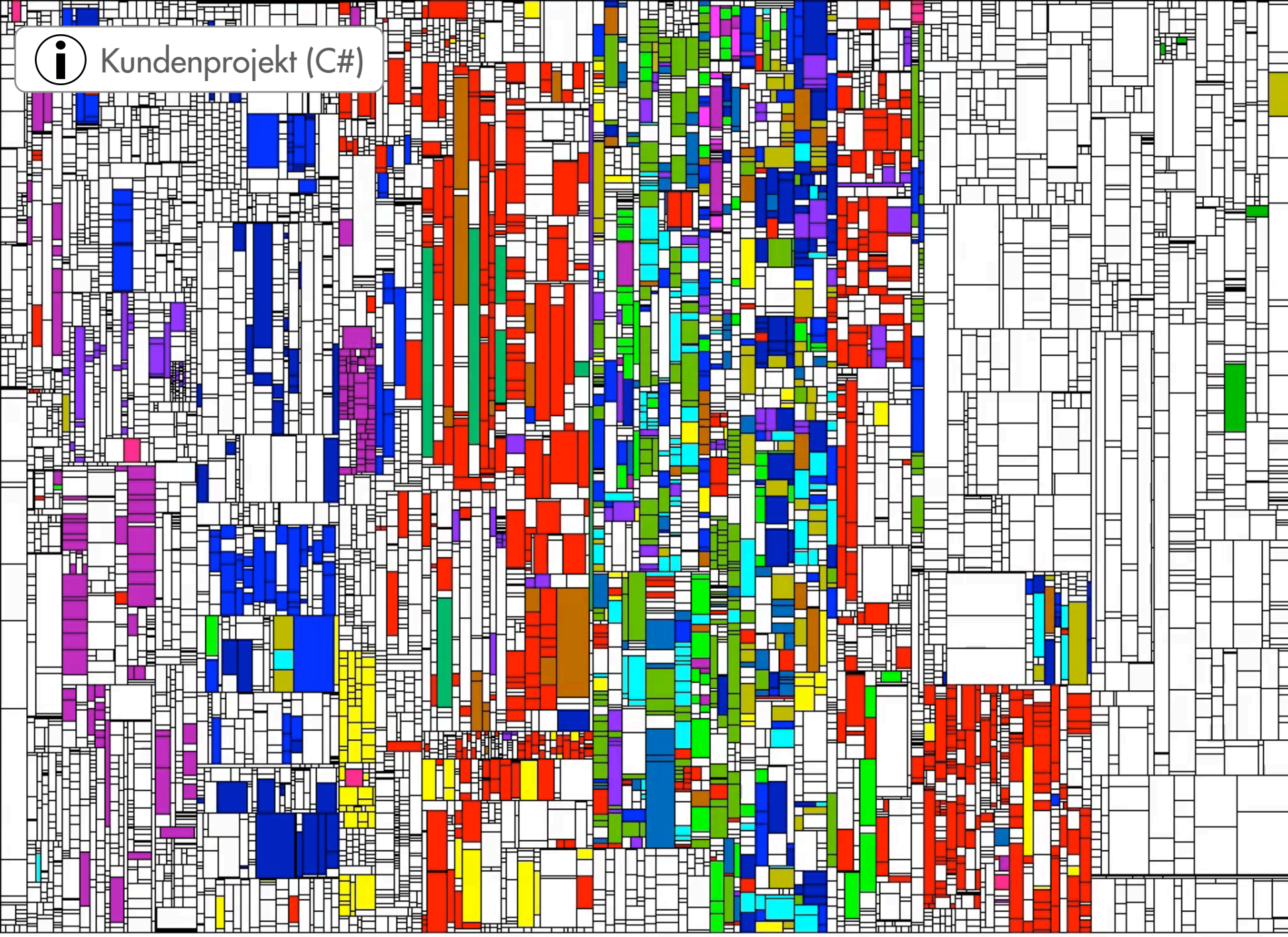
Kundenprojekt (C#)




Team-
Wechsel
Code-
Ownership
Inkonsistenter
Klon
Bug
Historien-Analyse

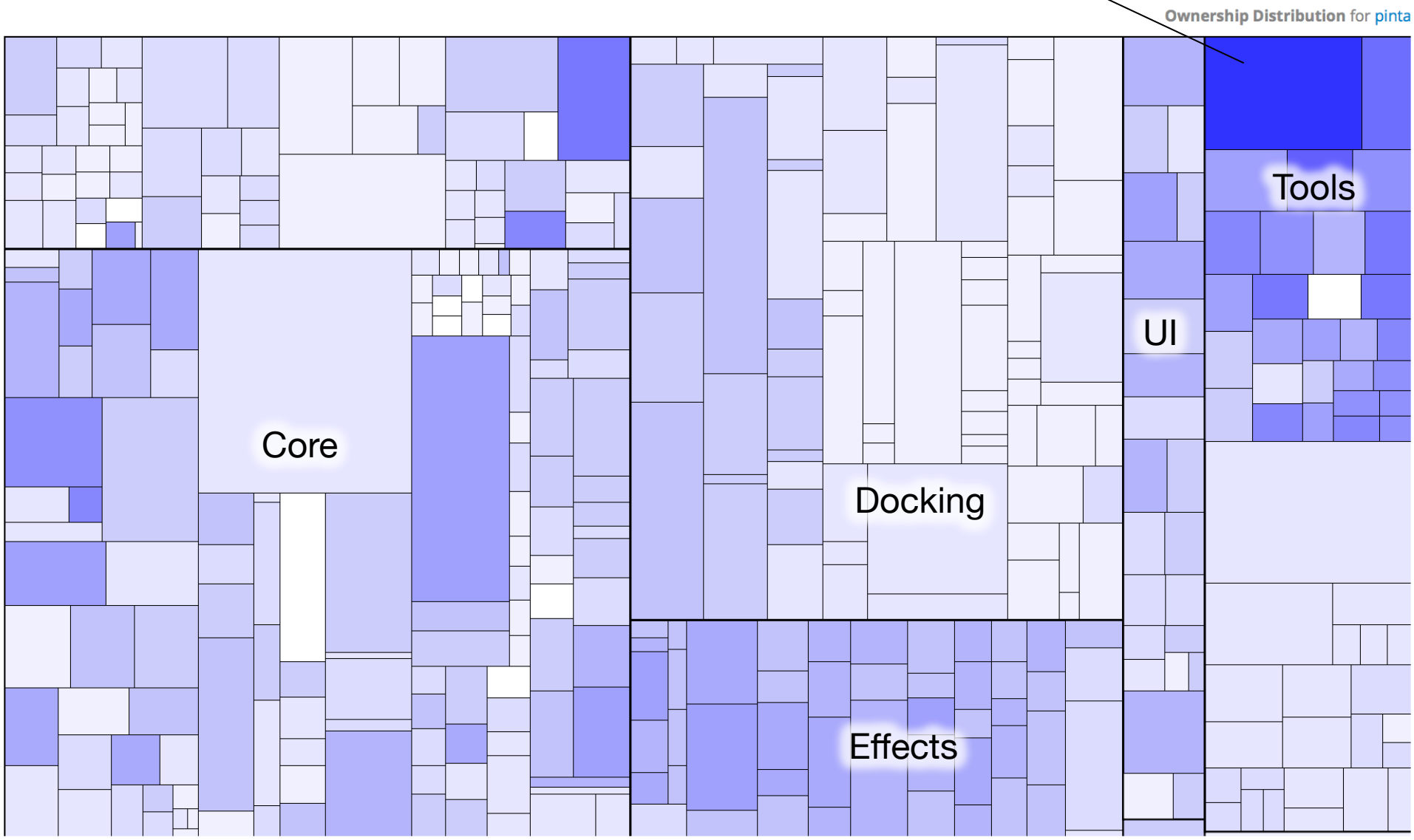

Team-
Wechsel
Code-
Ownership
Inkonsistenter
Klon
Bug
Historien-Analyse

Die Historie verrät, wer welchen Code kennt.


Kundenprojekt (C#)

PintaProject/Pinta

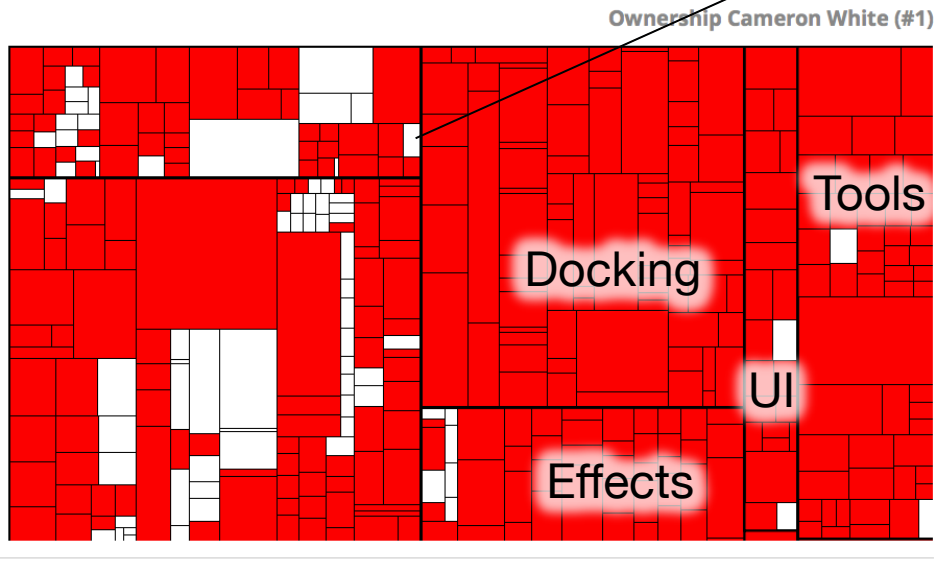
`TextTool.cs` (17 Entwickler)

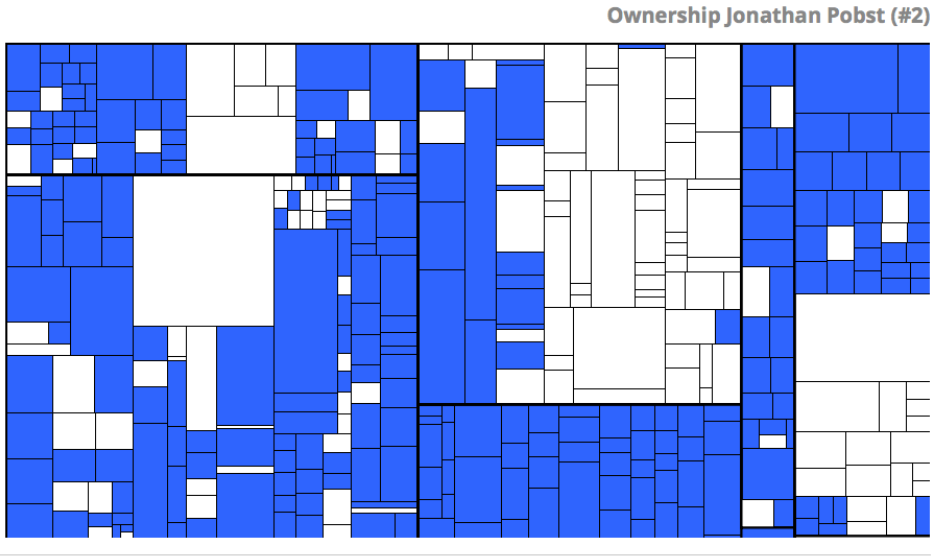


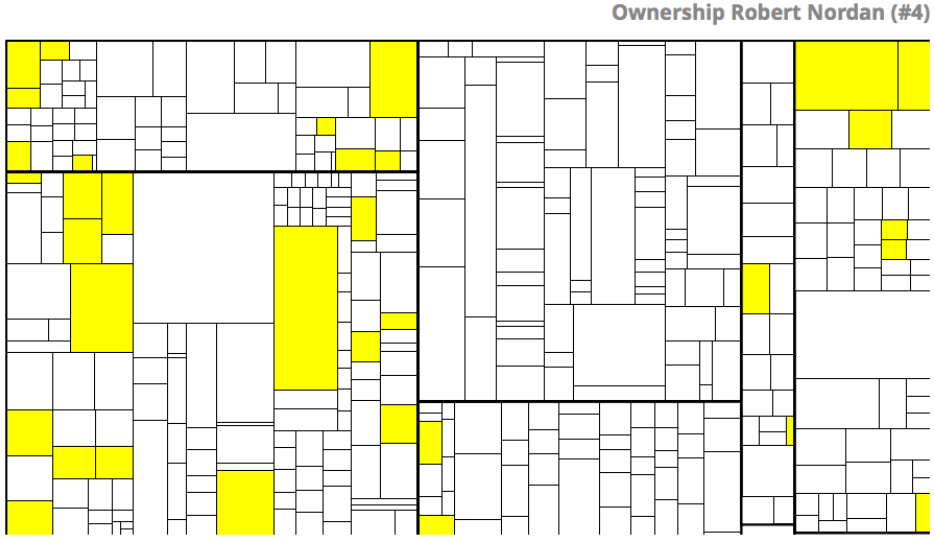
PintaProject/Pinta

Wissenslücke

„Abgewandert“

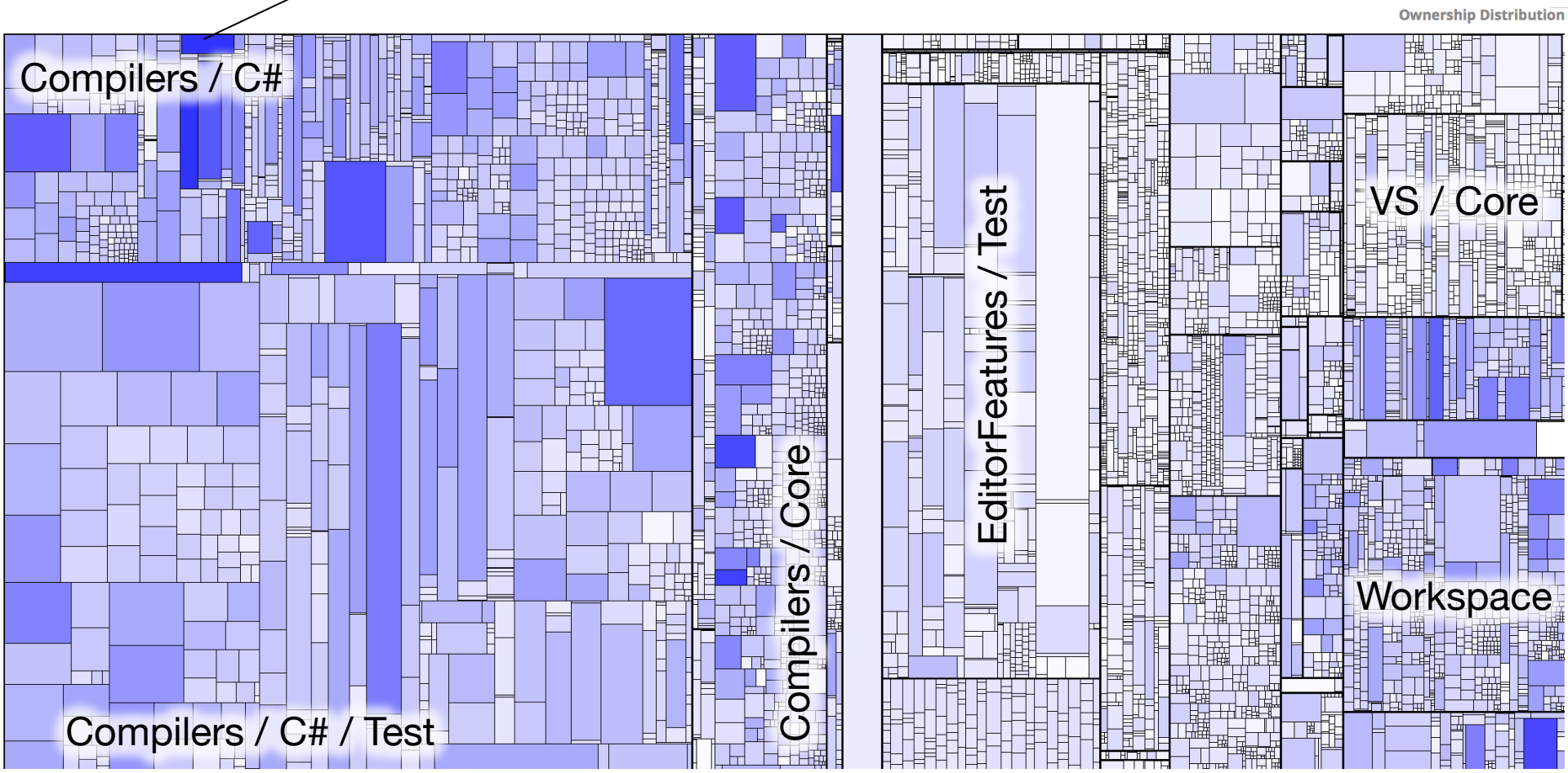




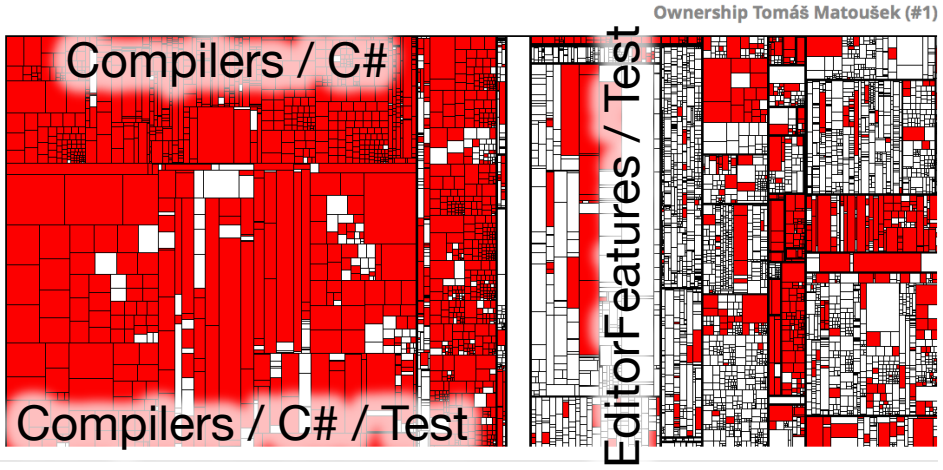


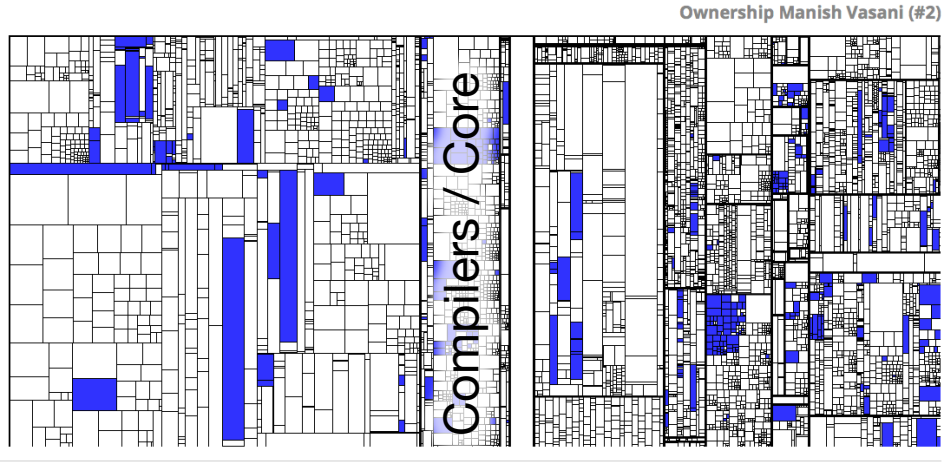
dotnet/roslyn

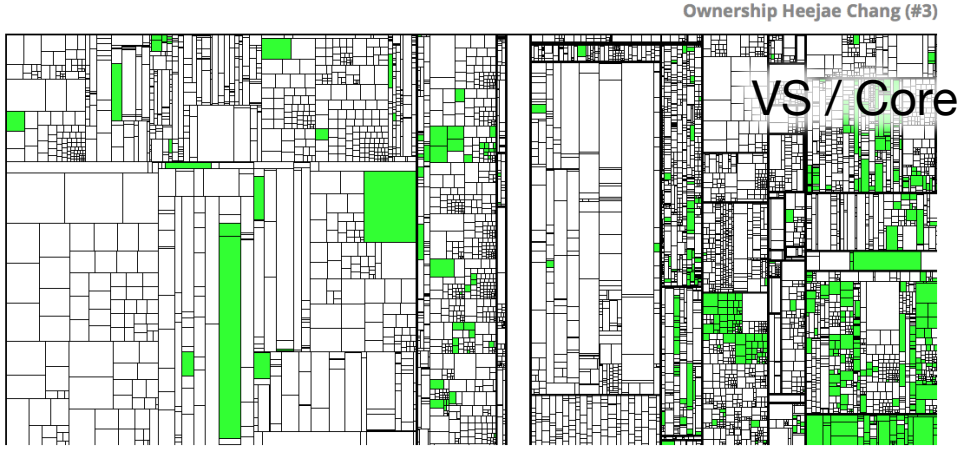
`CSharpCommandLineParser.cs`
(47 Entwickler)

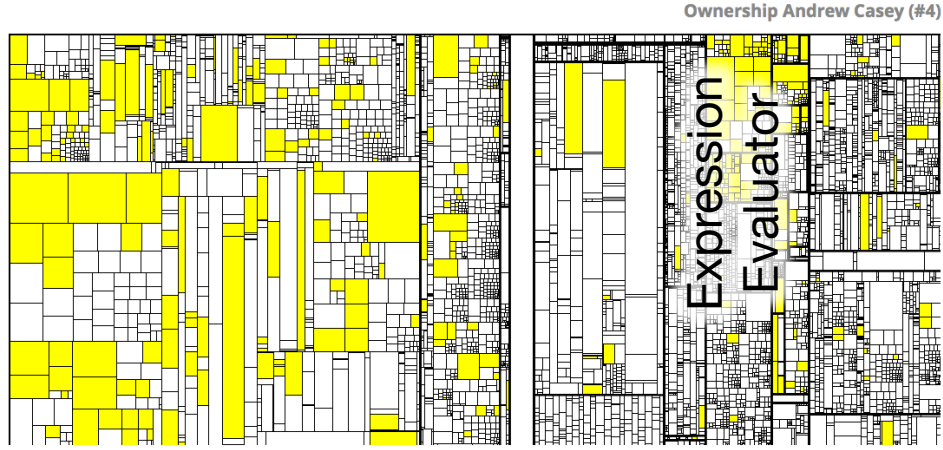


dotnet/roslyn



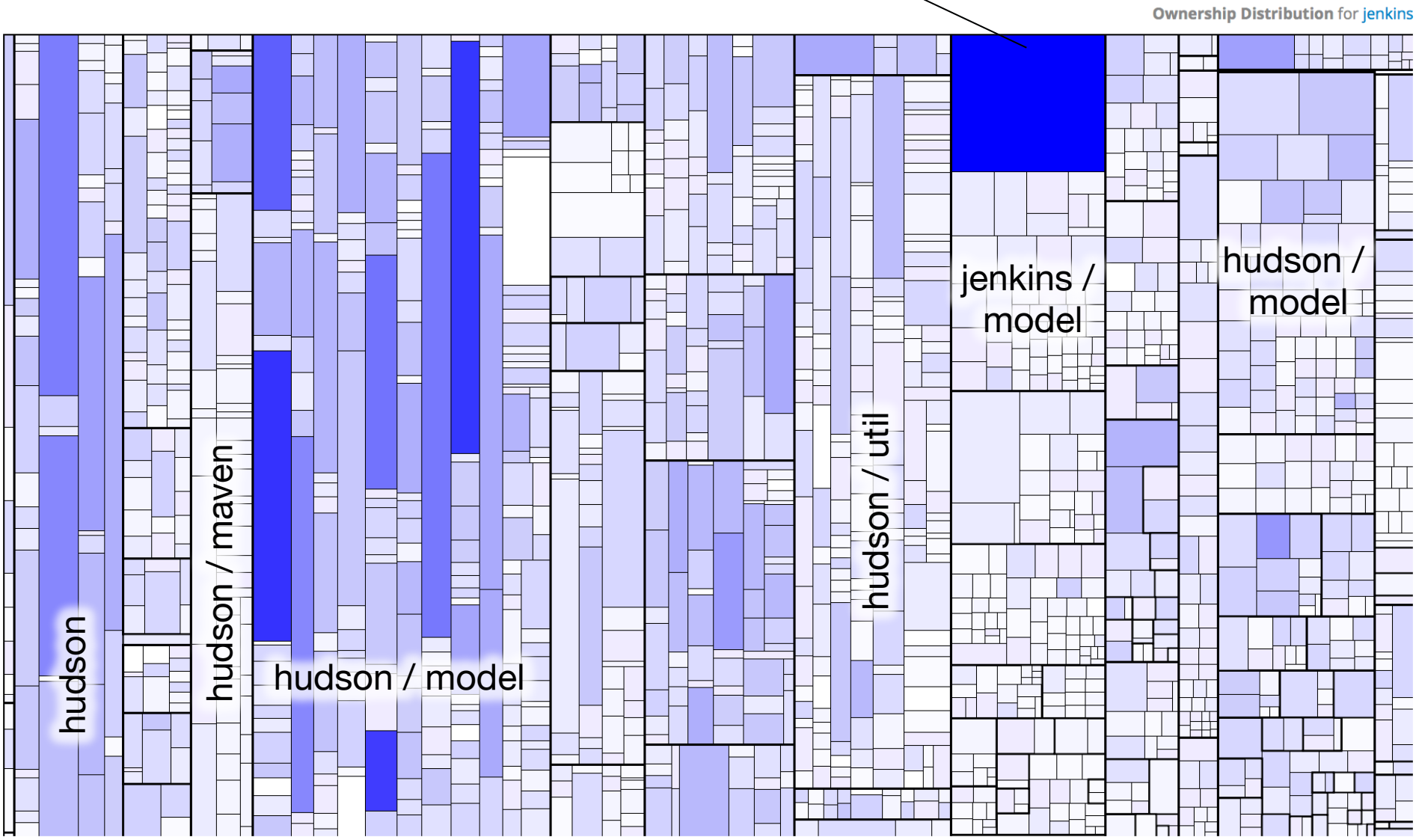






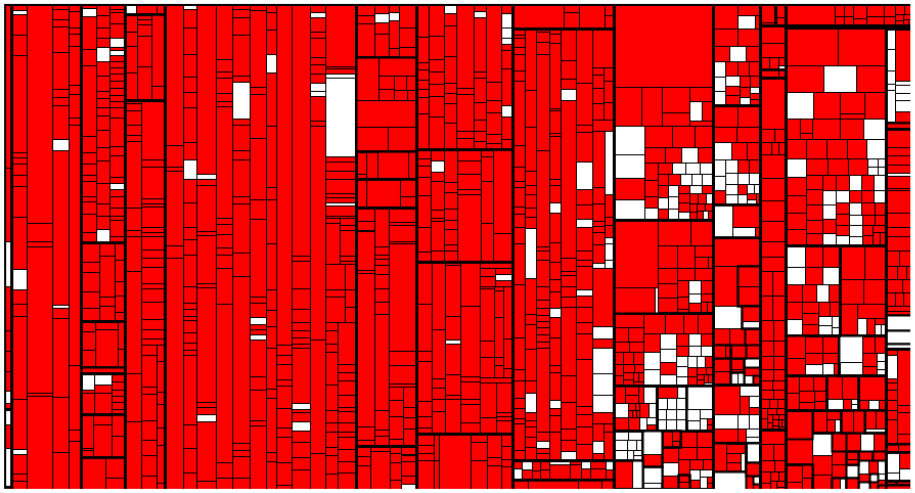
jenkinsci/jenkins

`jenkins/model/Jenkins.java`
(48 Entwickler)

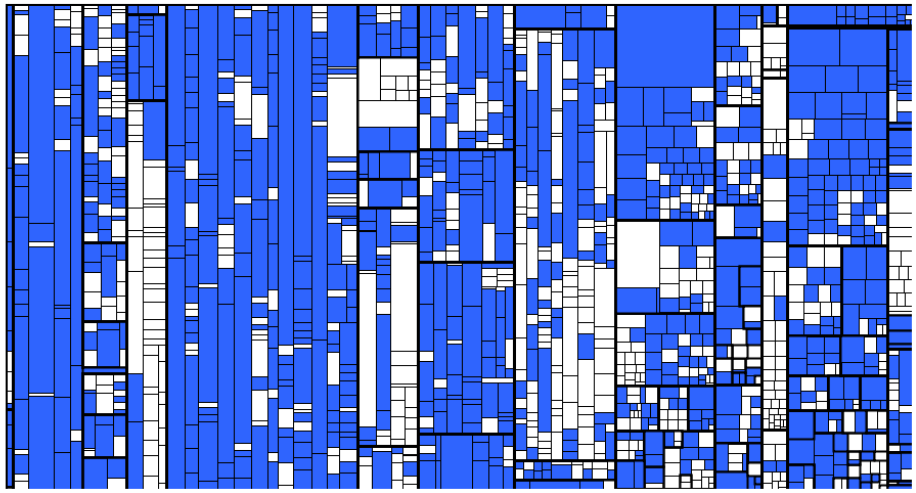


jenkinsci/jenkins

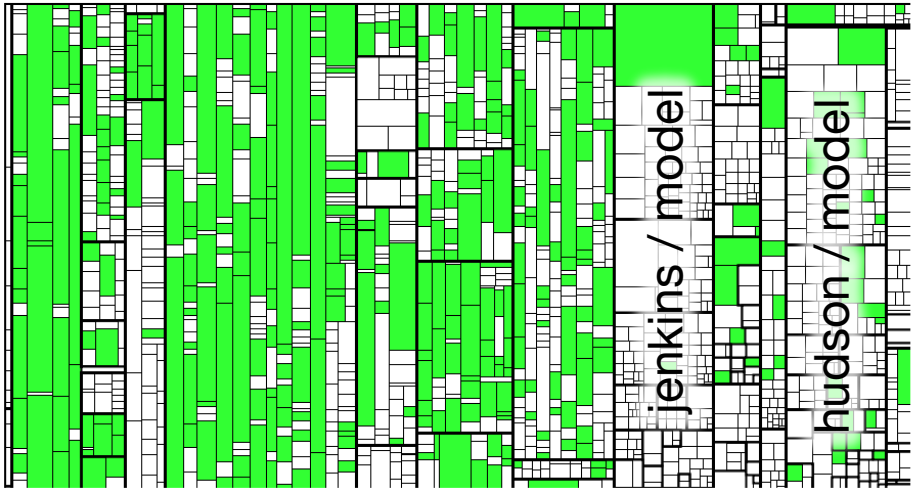
Ownership Kohsuke Kawaguchi (#1)



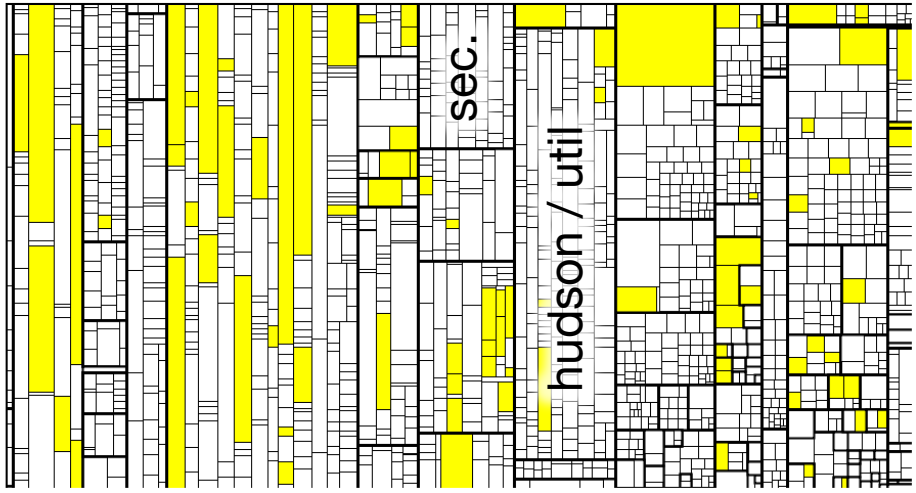
Ownership Jesse Glick (#2)

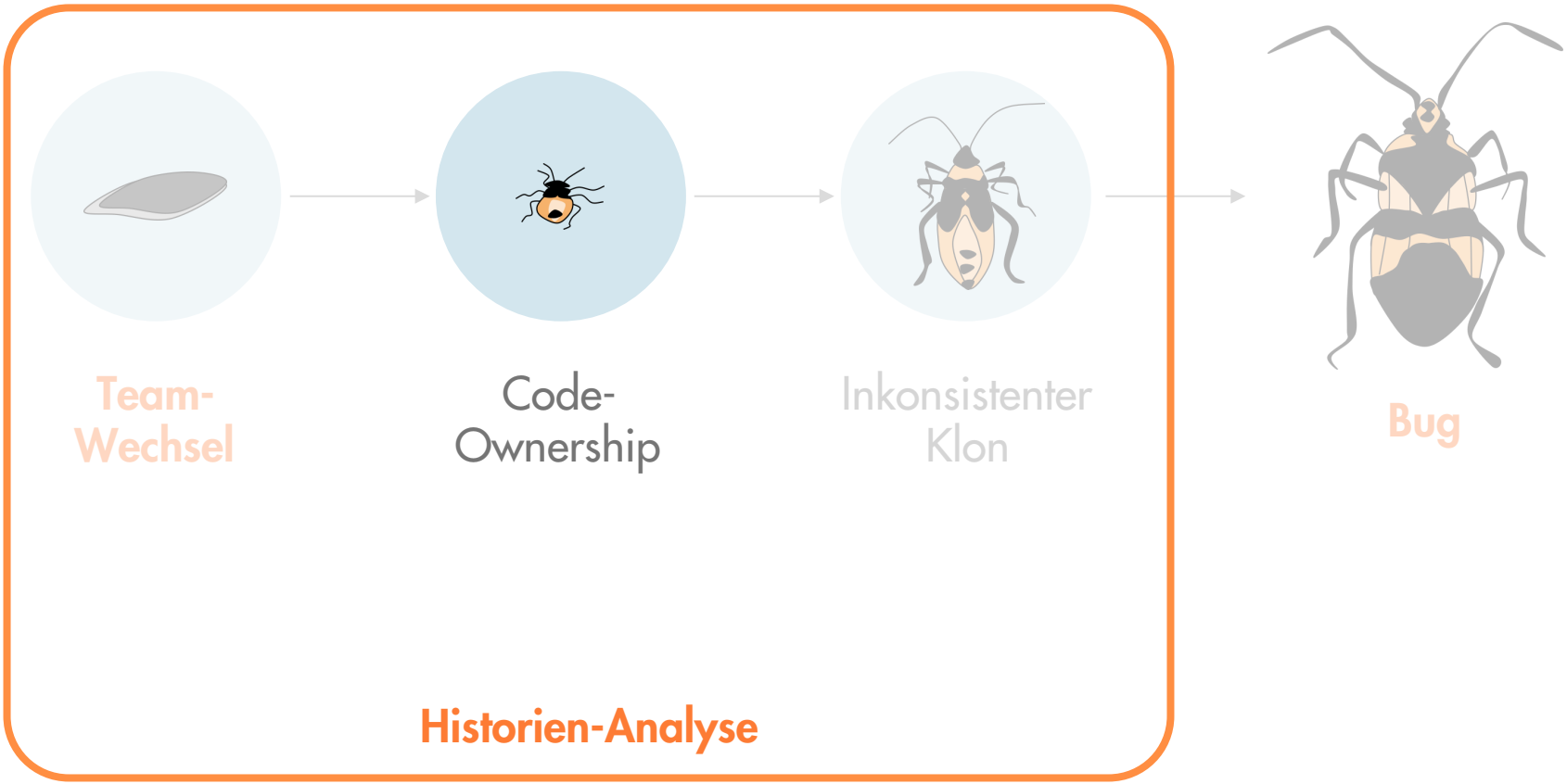


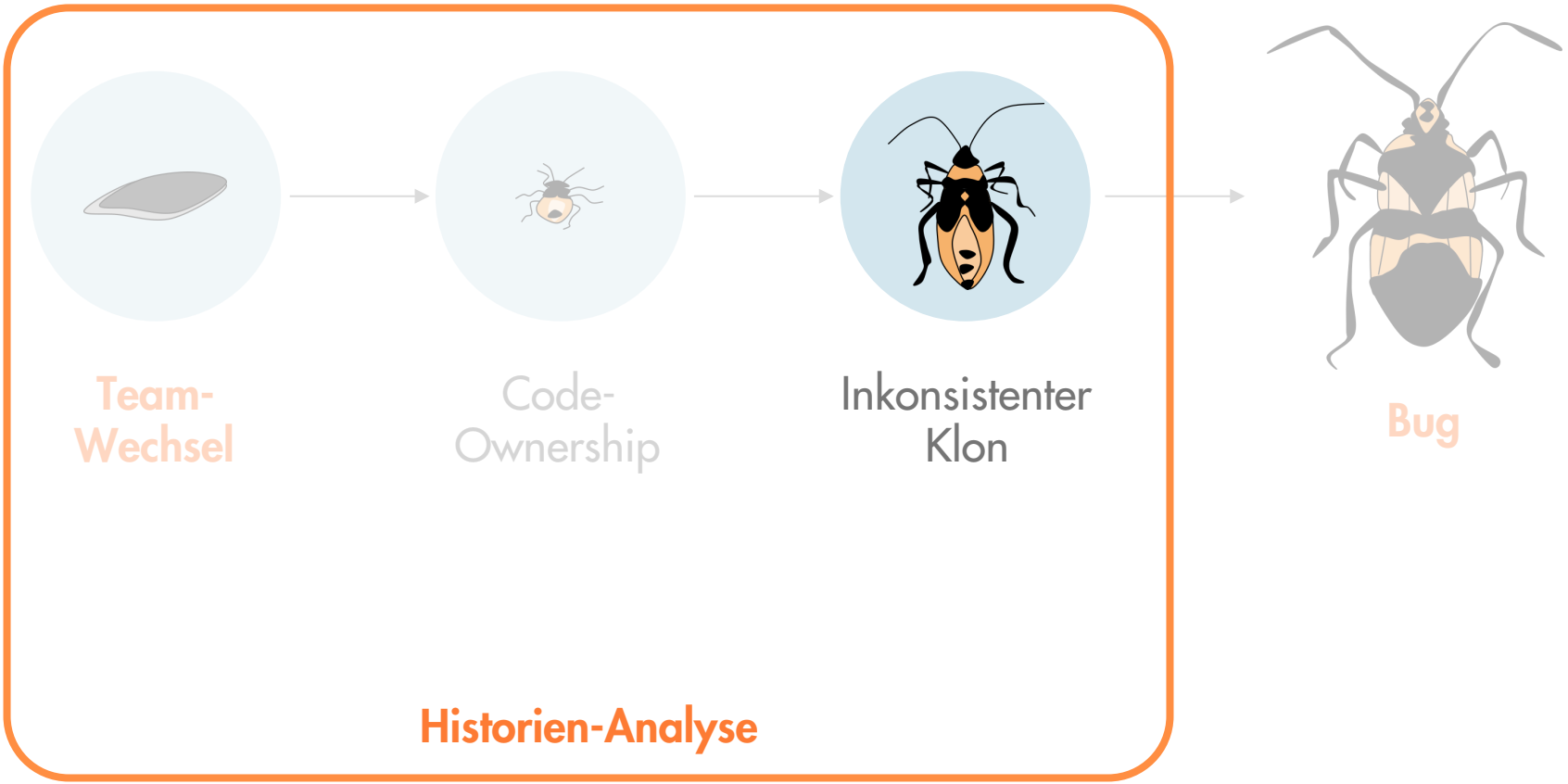
Ownership Alan Harder (#3)



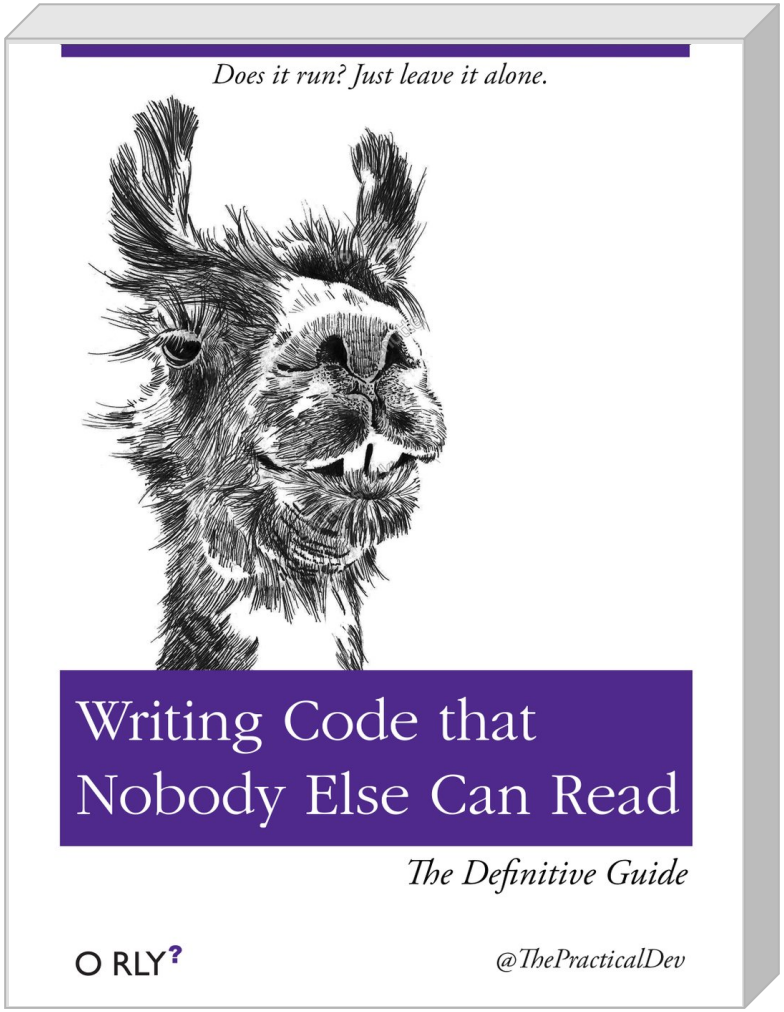
Ownership Christoph Kutzinski (#4)



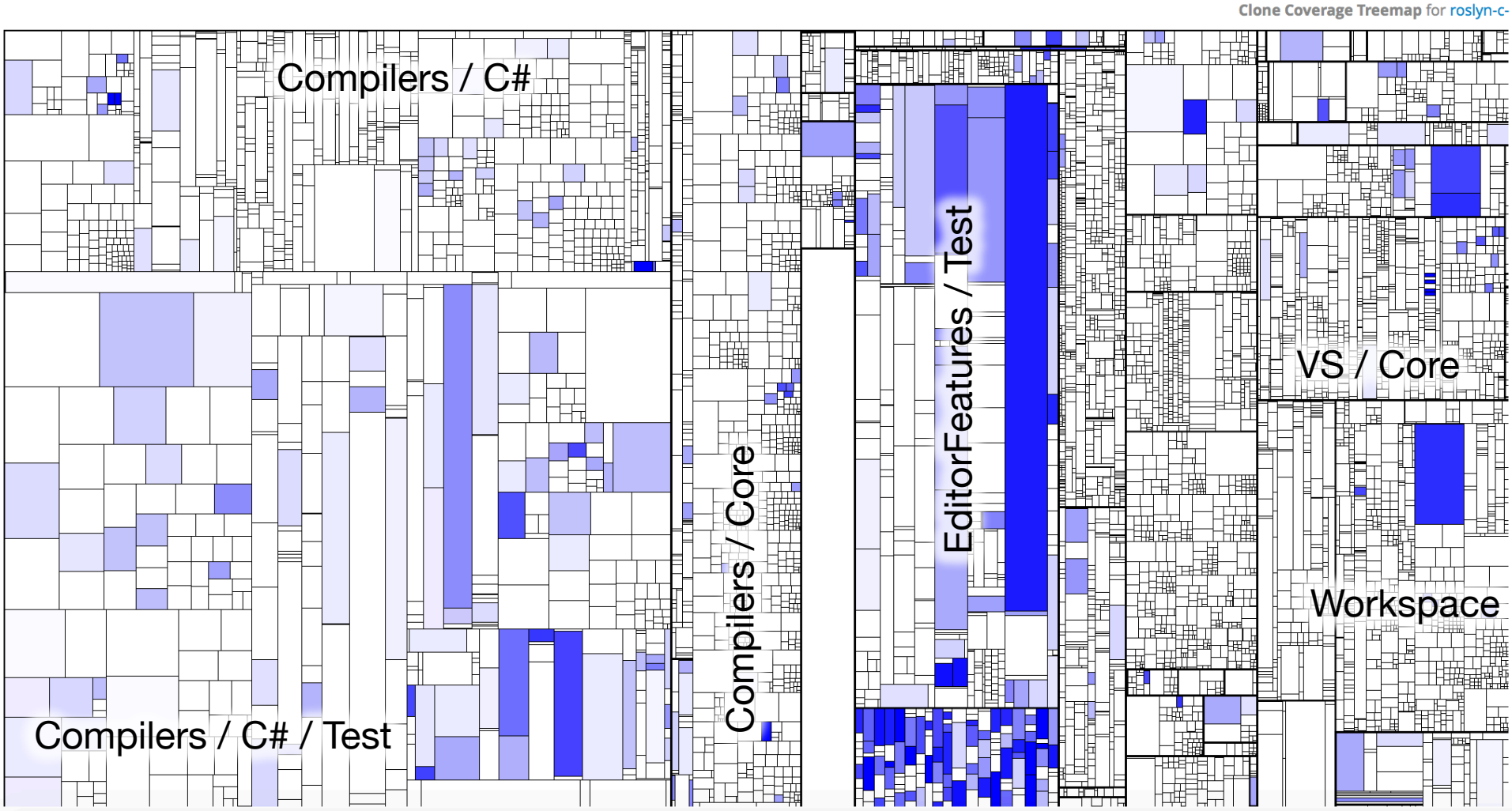

Team-
Wechsel
Code-
Ownership
Inkonsistenter
Klon
Bug
Historien-Analyse


Team-
Wechsel
Code-
Ownership
Inkonsistenter
Klon
Bug
Historien-Analyse

Die Historie verrät, wie sich die Qualität entwickelt hat.

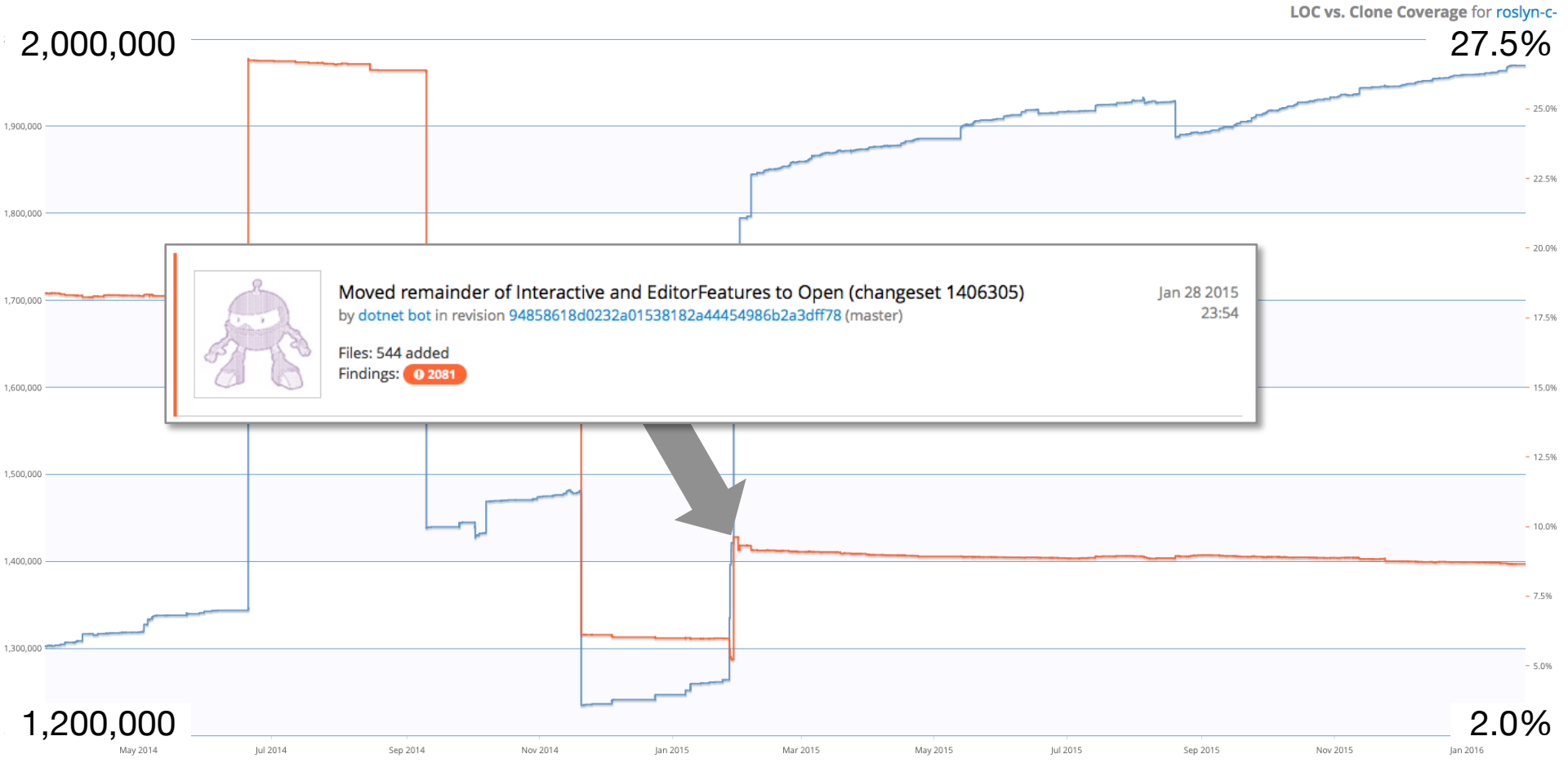


dotnet/roslyn



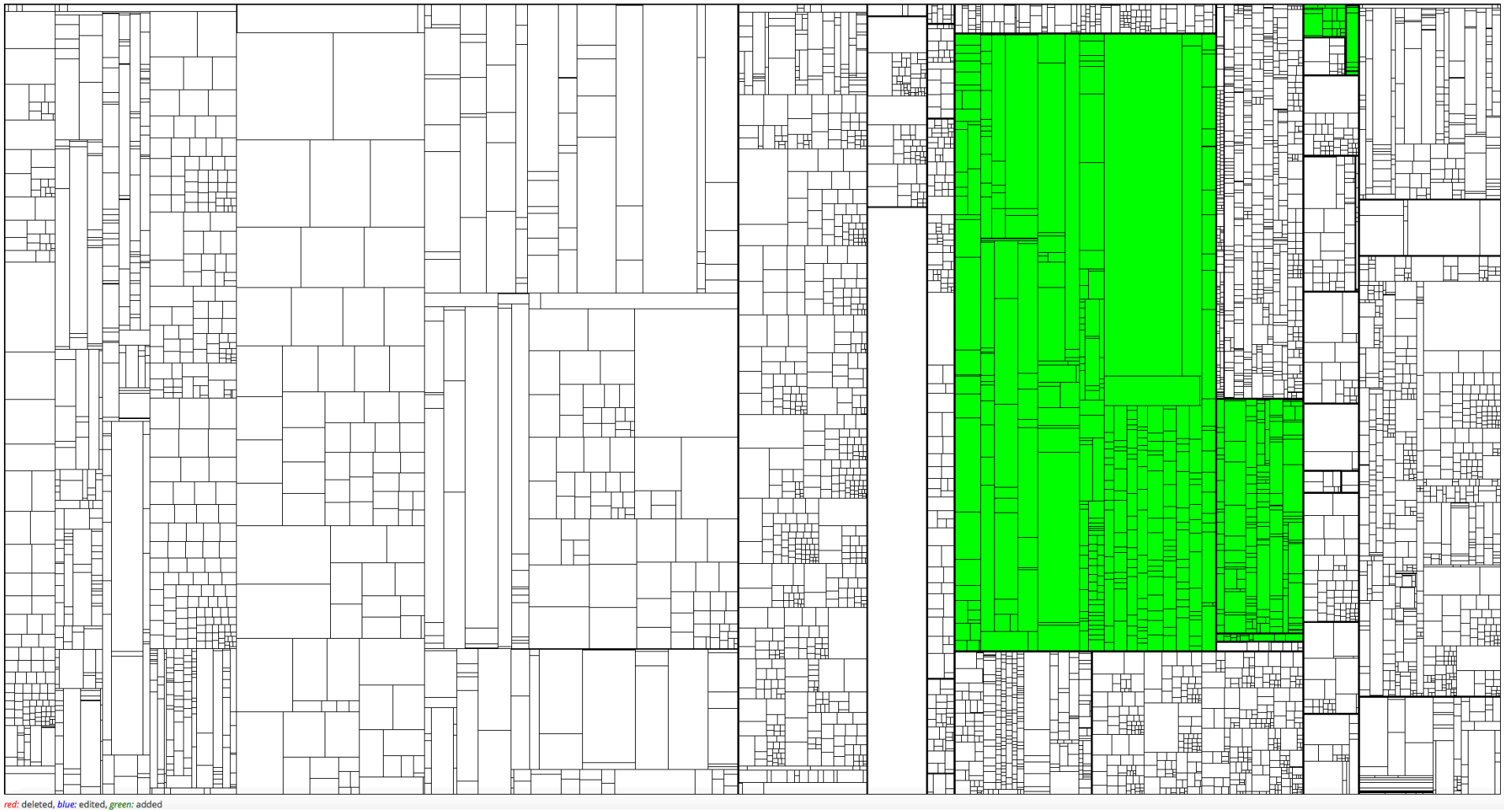
Head (Feb 2016)
Klonüberdeckung: 8,7%

dotnet/roslyn



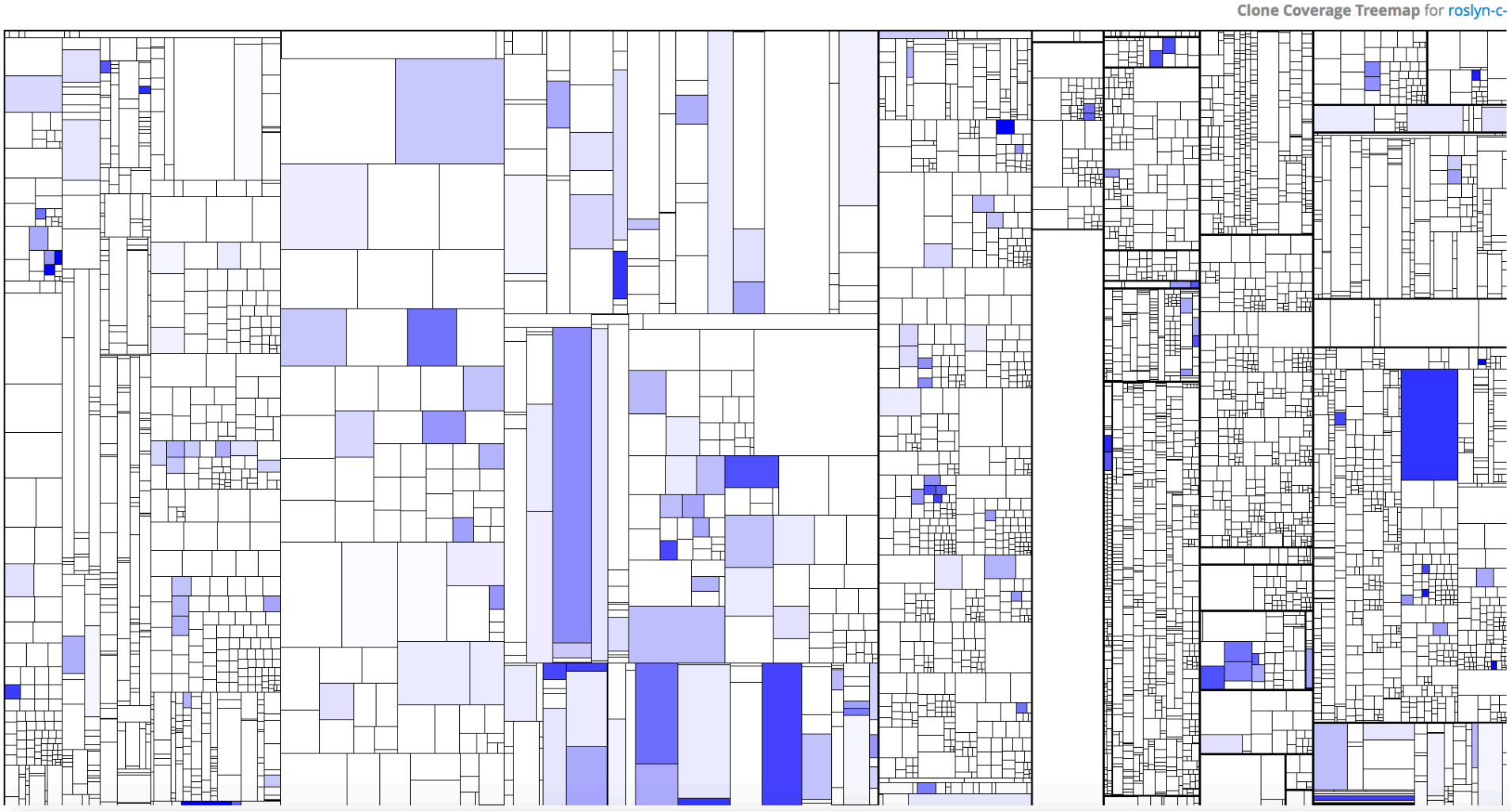
Lines of Code
Clone Coverage

dotnet/roslyn



Änderungen

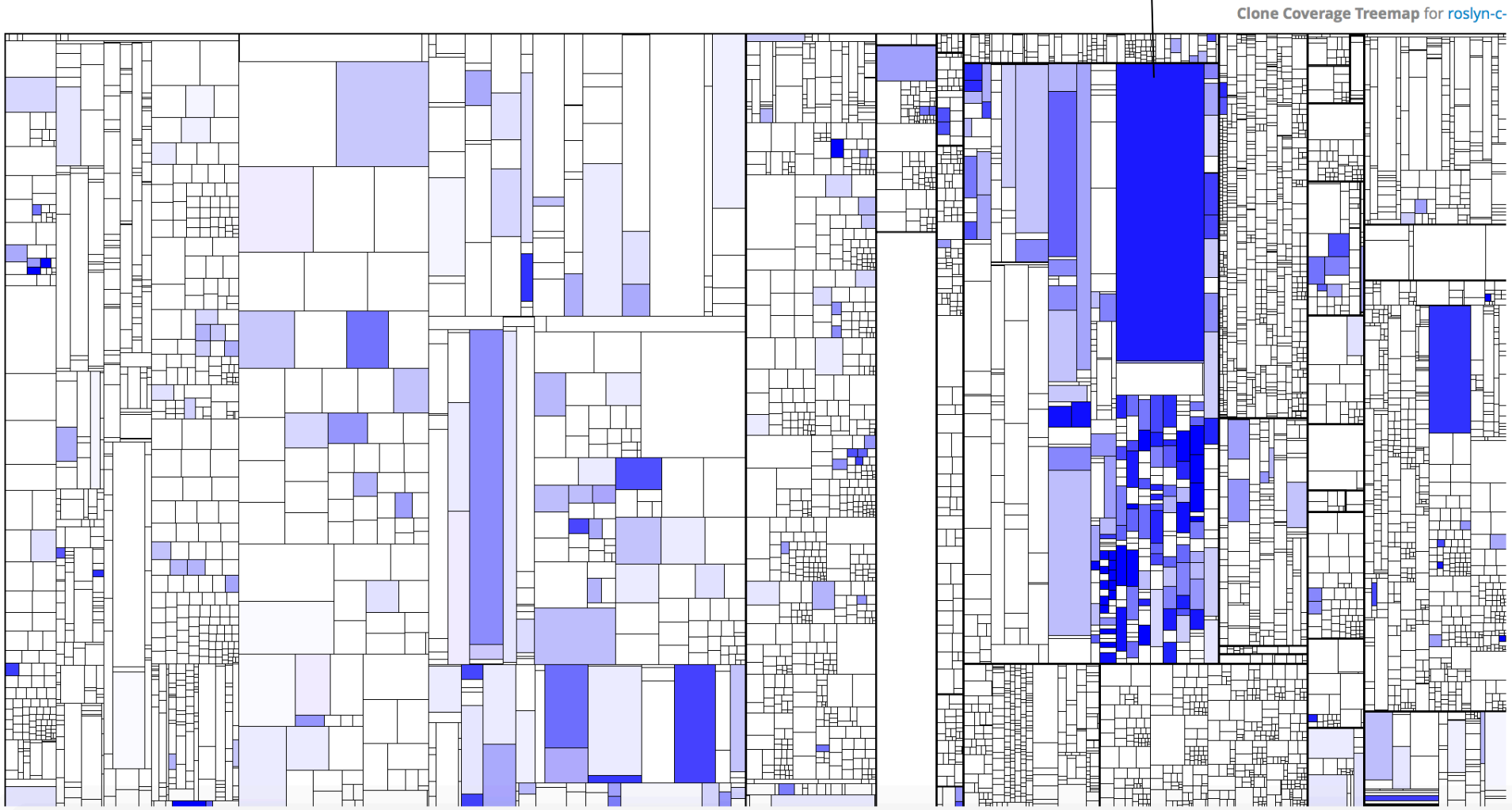
dotnet/roslyn



Vorher
(Klonüberdeckung 5,2%)

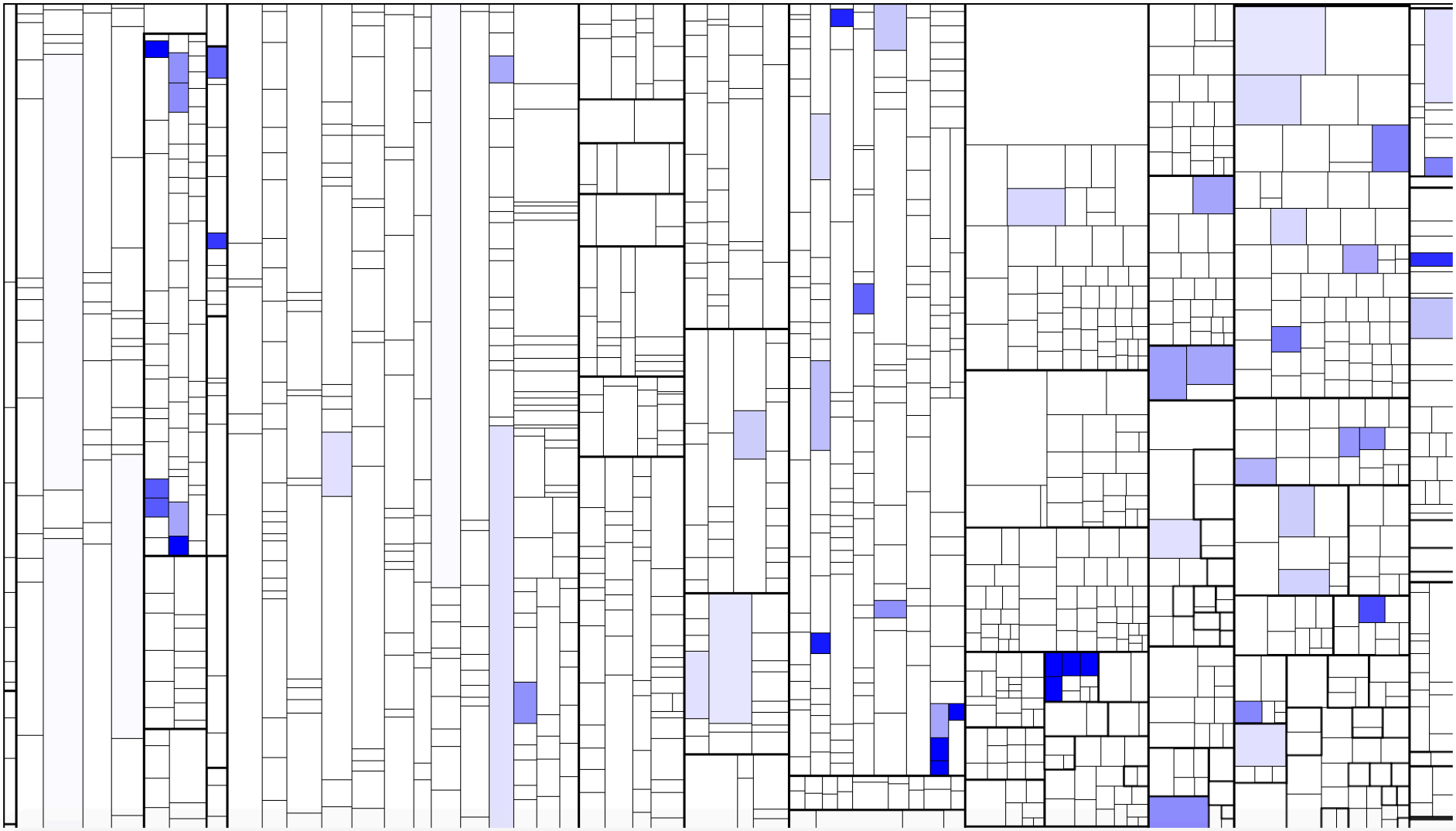
dotnet/roslyn

Performance-Test Case
(89% Klonüberdeckung)



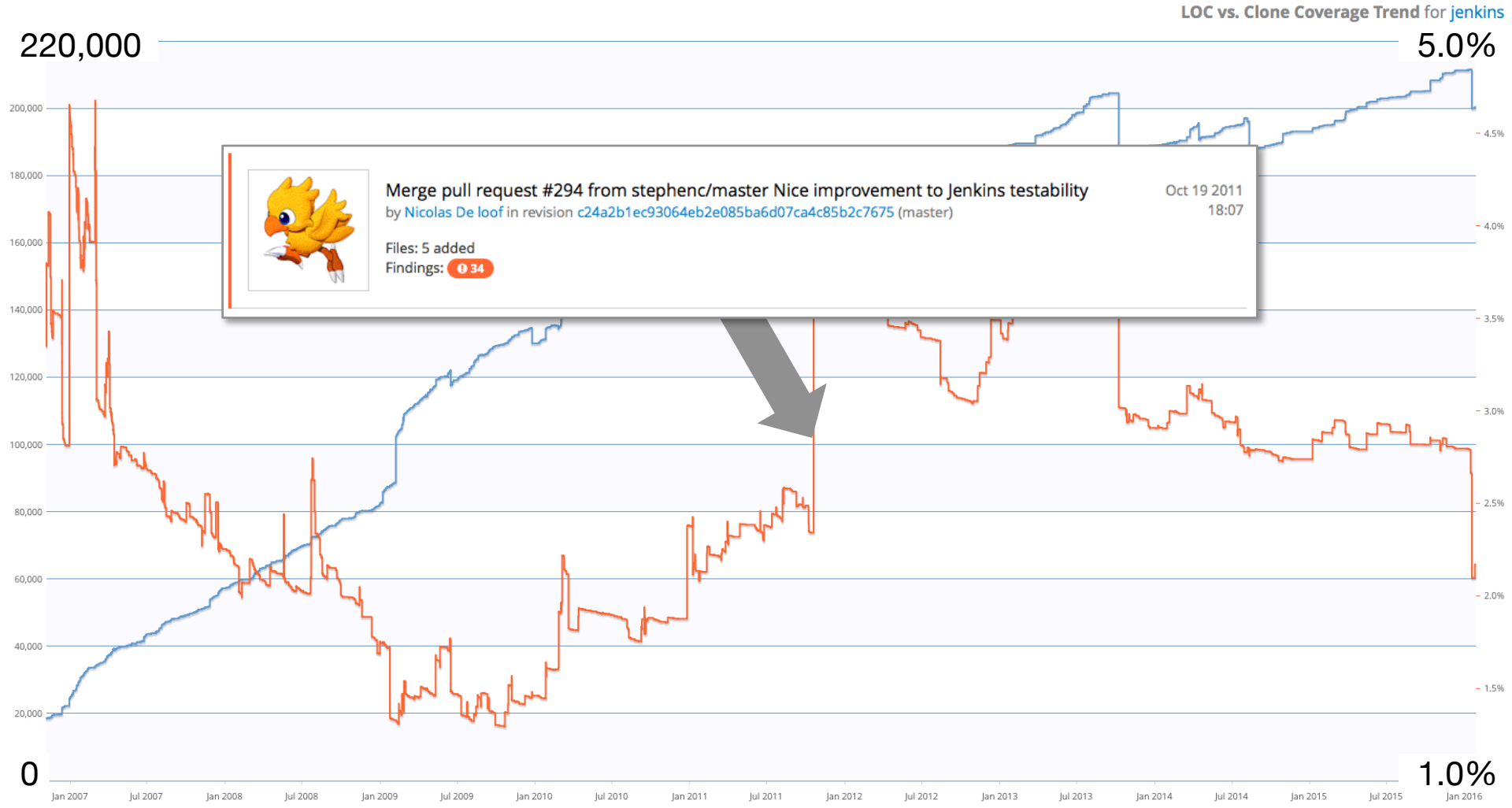
Nachher
(Klonüberdeckung 9,6%)

jenkinsci/jenkins



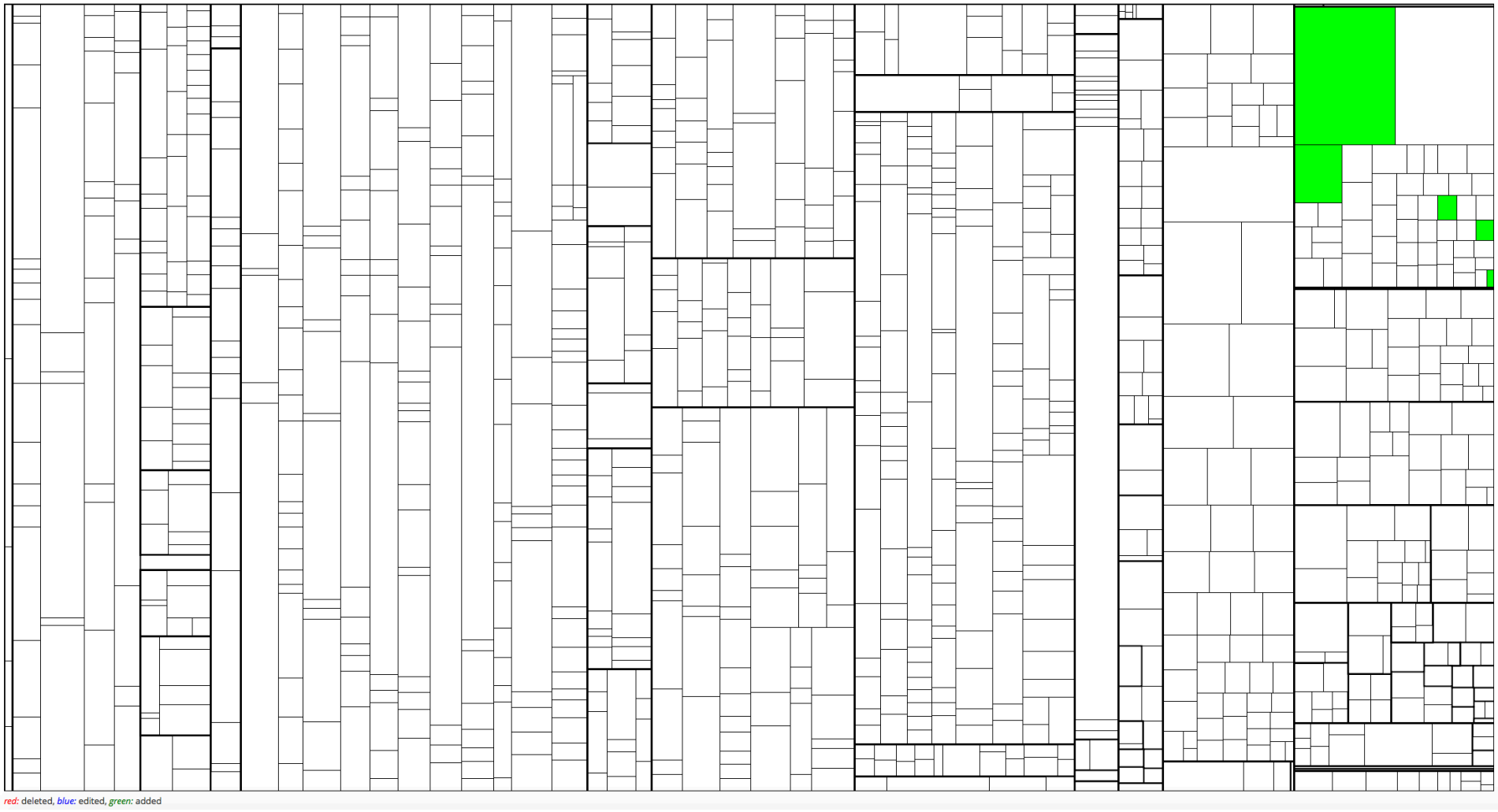
Head (Feb. 2016)
Klonüberdeckung: 2,2%

jenkinsci/jenkins



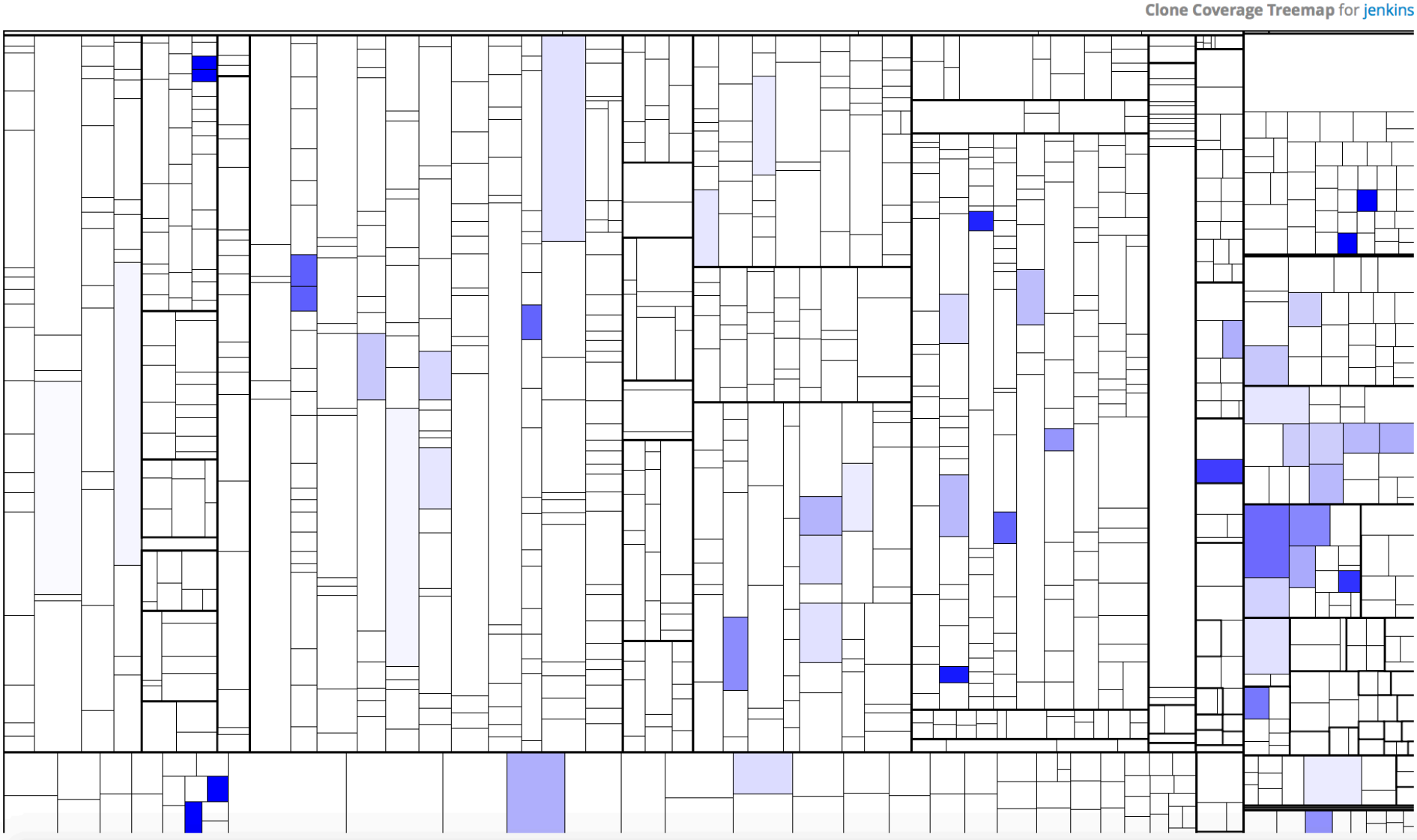
Lines of Code
Clone Coverage

jenkinsci/jenkins



Änderungen

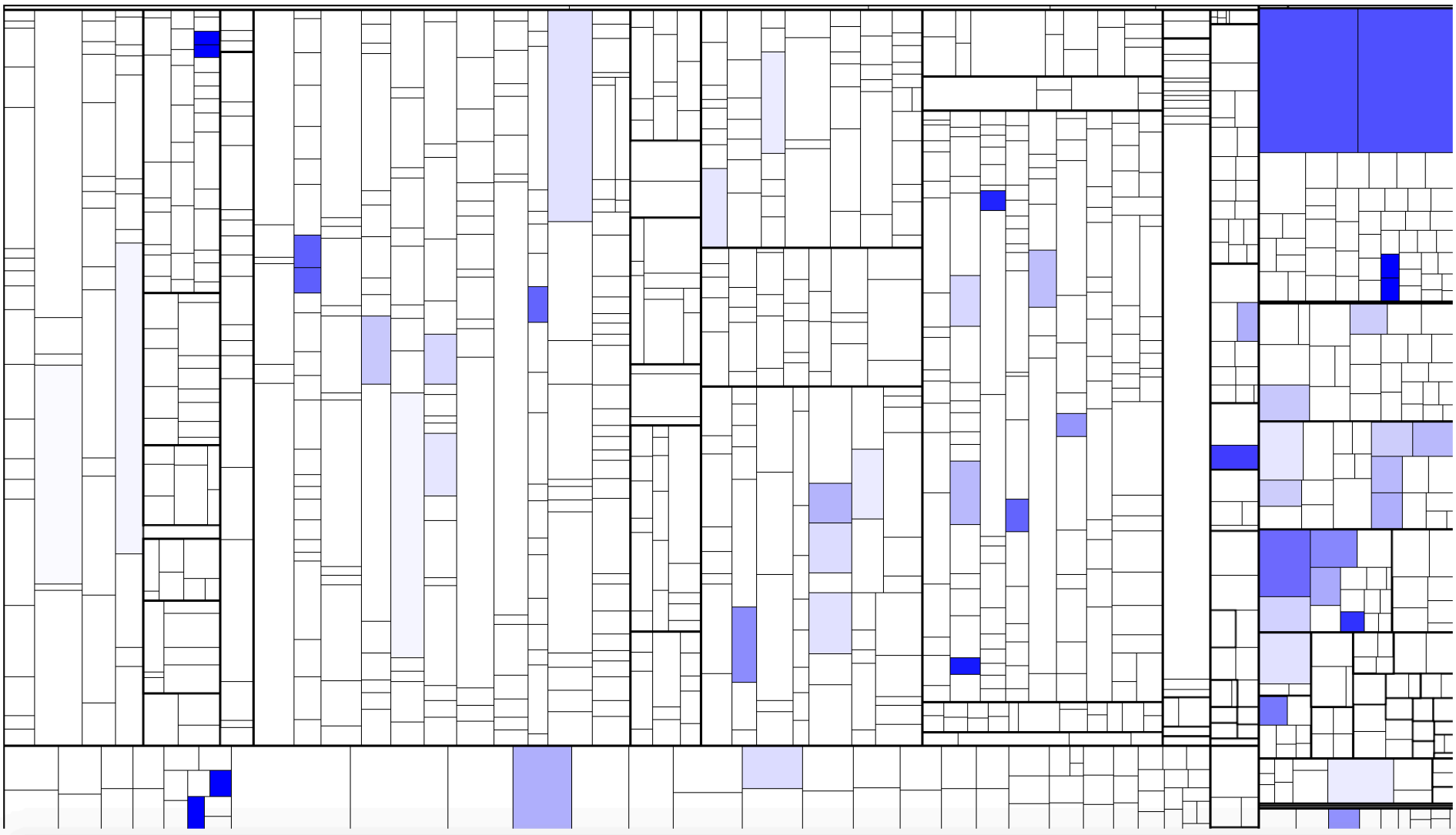
jenkinsci/jenkins



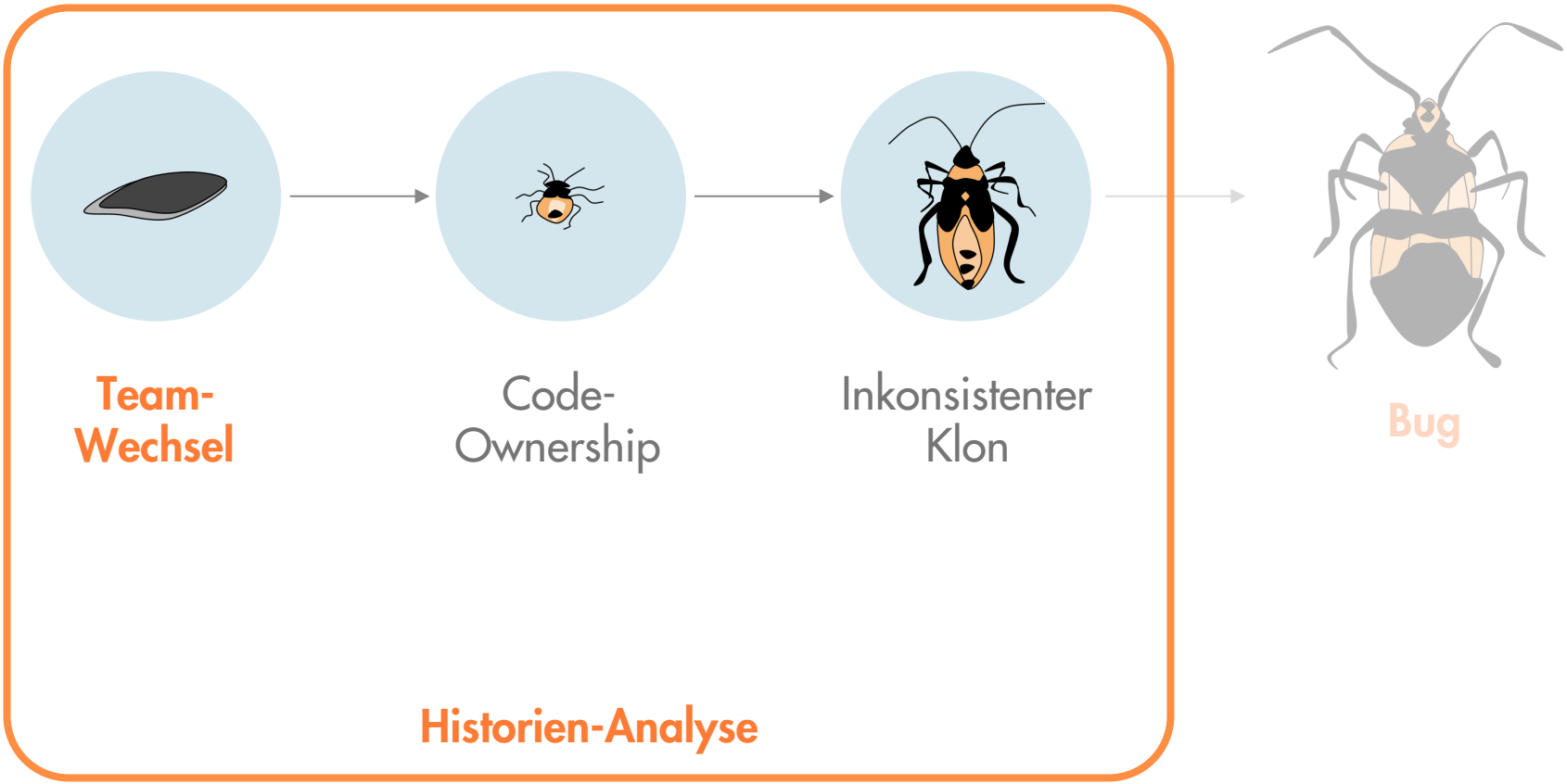
Vorher
(Klonüberdeckung 2,3%)

jenkinsci/jenkins

JenkinsRule.java / HudsonTestCase.java
(68% Klonüberdeckung)



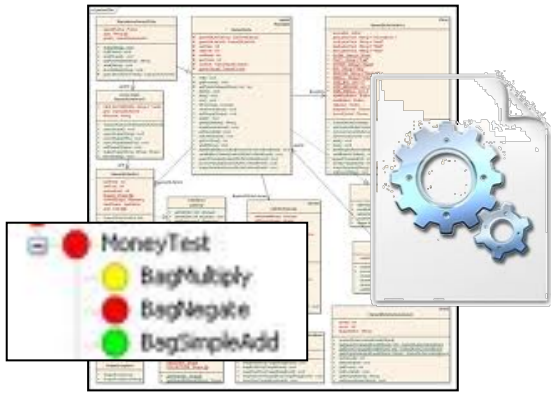
Nachher
(Klonüberdeckung 4,2%)


Team-
Wechsel
Code-
Ownership
Inkonsistenter
Klon
Bug
Historien-Analyse

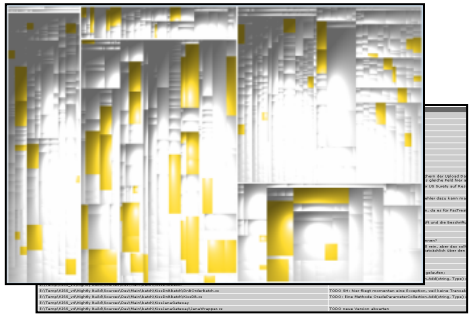
Agenda

1. Welche Probleme verrät die Historie?
2. Was kann man im eigenen Projekt machen?

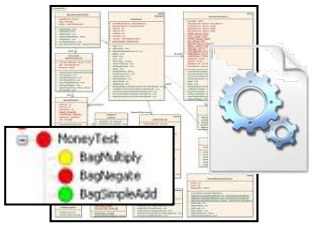
Historienanalysen erfordern neue Werkzeuge.


MoneyTest
BagMultiply
BagNegate
BagSimpleAdd

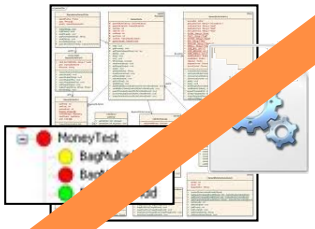

conQAT



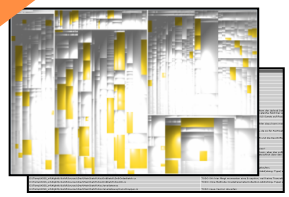
Baseline



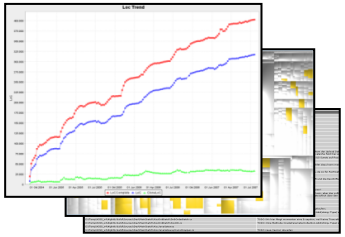
Latest Version



Baseline
Dashboard

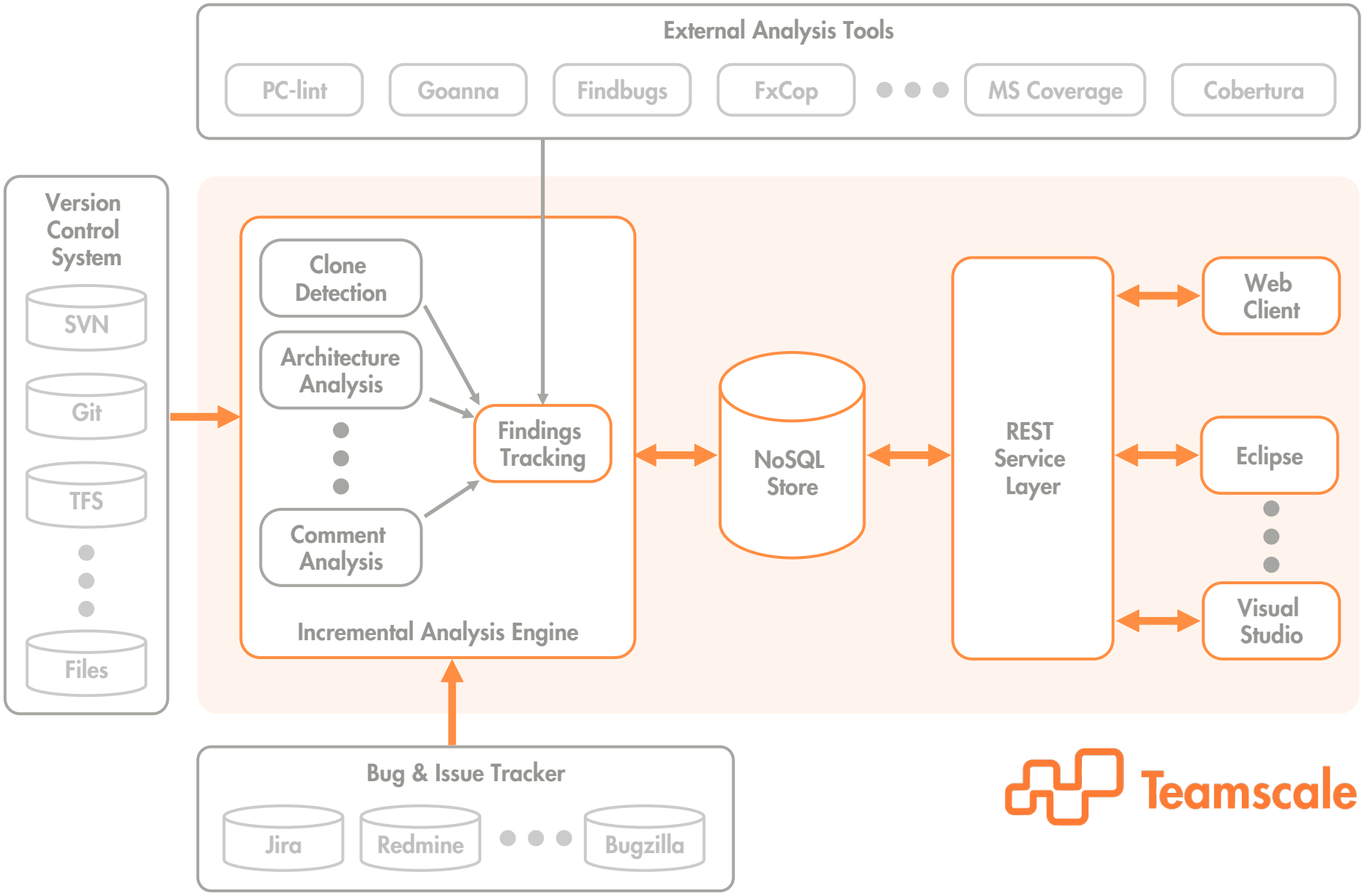


Dashboard
inkl Delta




Teamscale


Free For
Open Source
Software


External Analysis Tools
PC-lint
Goanna
Findbugs
FxCop
MS Coverage
Cobertura
Version Control System
SVN
Git
TFS
Files
Clone Detection
Architecture Analysis
Comment Analysis
Findings Tracking
Incremental Analysis Engine
NoSQL Store
REST Service Layer
Web Client
Eclipse
Visual Studio
Bug & Issue Tracker
Jira
Redmine
Bugzilla
Teamscale

Exclusive and event based gateway can be async as well
by Daniel Lewis in revision e28cd40d7b4ac1e98a370cc6dcacedec6ef1d5b0 (git)

Files: 2 changed
Findings: 0

Jun 25 2015
08:34

Add schema option for Oracle on QA
by Daniel Lewis in revision 93beea0b2623ec52f362d4ca5c7a71039ab14409 (git)

Files: 14 changed, 1 deleted
Findings: 7 6 9

Jun 24 2015
16:35

Merge branch 'master' of https://github.com/mittalabhi2123/Activiti into mittalabhi2123-master
by Isabella Wright in revision 0292ee4f7ace6a1c81af64f71fac78935c39b101 (git)

Files: 4 added, 34 changed
Findings: 44 20 8

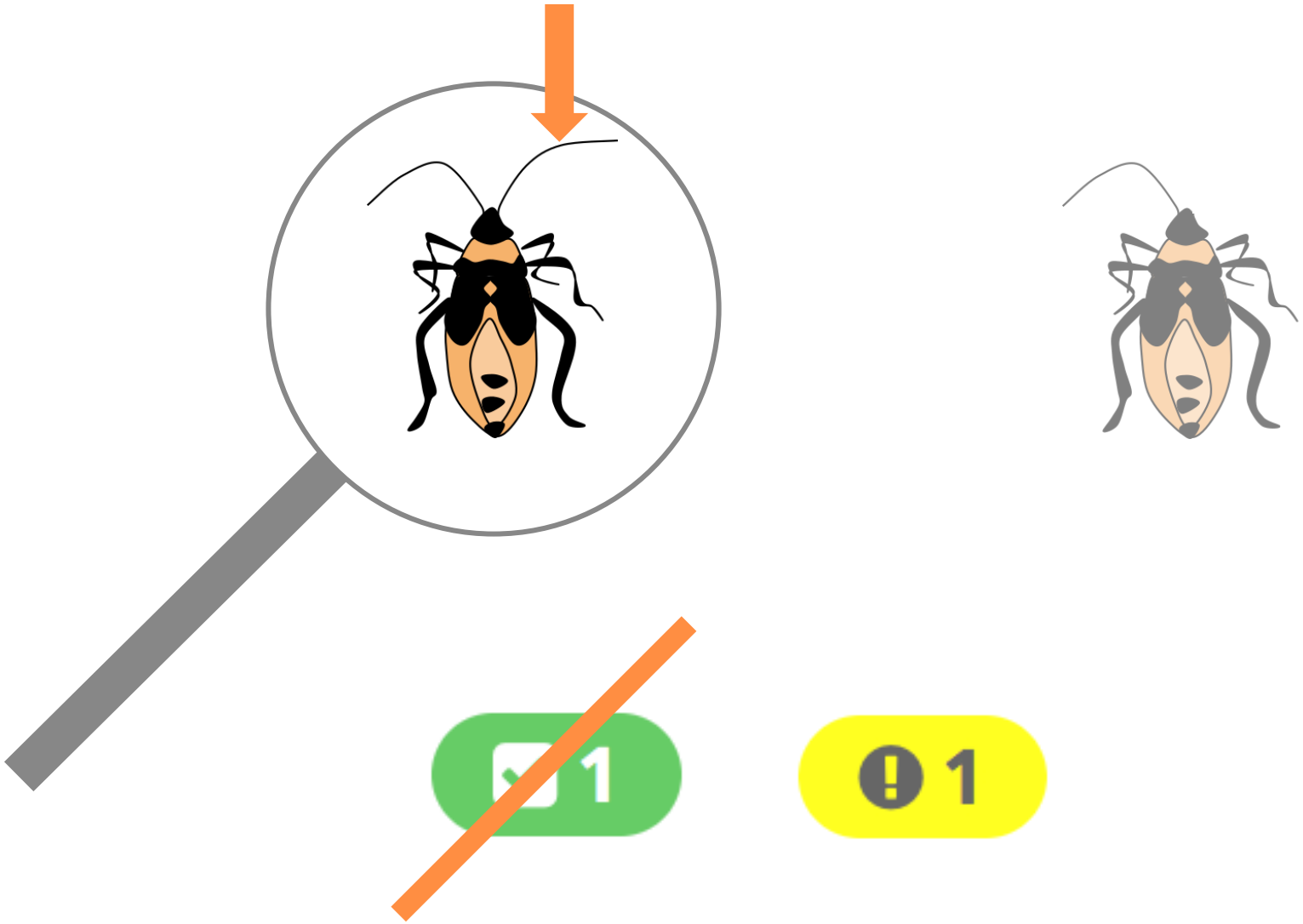
Jun 24 2015
13:07

ACT-4012: IllegalArgumentException when invoking a service task requiring a date (JaxB custom bindings must be used not only when importing the WSDL, but also when creating web-service clients)
by Anya Hill in revision 79db90d23c5af810eea8b830534e3df390aa21a1 (git)

Files: 20 changed, 3 deleted
Findings: 1 15 35

Jun 24 2015
11:06

Mit geerbten Problemen pragmatisch umgehen


1
1

PintaProject/Pinta

Implement a cleaner fix for the gap issues when drawing ellipses.
by Cameron White in revision 10662429ab96515bb2855cde252b04e1fd65212a (master)

Feb 23 2015
01:06

Affected files (4)

Metric changes

1 alerts:

Message	Context
Found potential inconsistent clone change in EllipseEngine.cs	[Broken clone] [Old clone finding] [Code change]

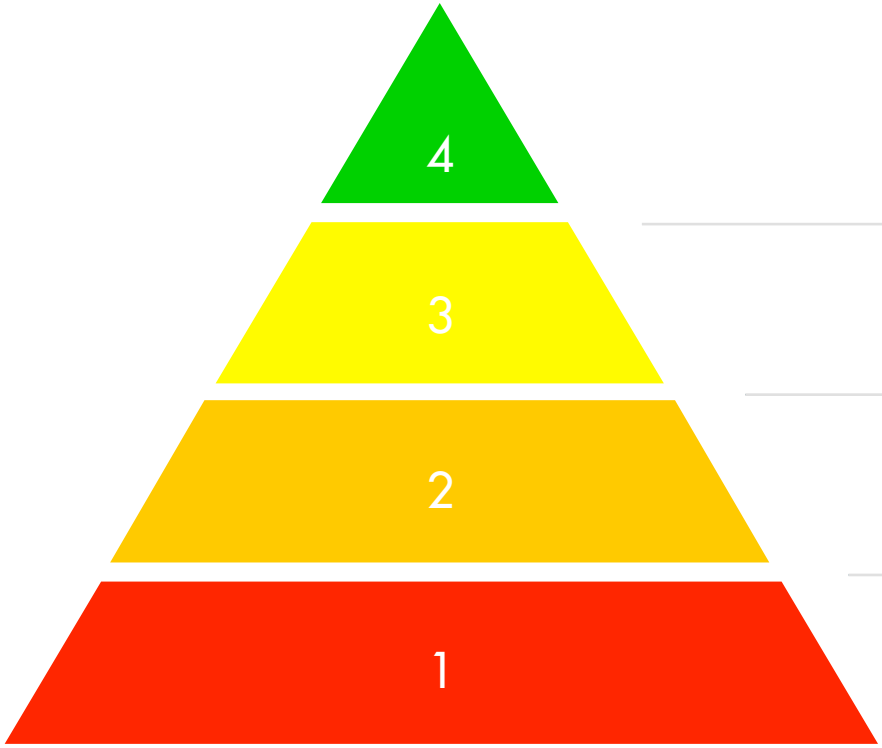
(revision b14ce47783eb93e2a5182cec00bc4ab1e25a54b6)

```
265         }
266
267         /// <summary>
268         /// Calculate each intermediate Point in the specified curve, returning Math.Round(1d / tInterval - 1d)
269         /// </summary>
270         /// <param name="tInterval">The increment value for t (should be between 0-1).</param>
271         /// <param name="x0">Starting point X (not included in the returned Point(s)).</param>
272         /// <param name="y0">Starting point Y (not included in the returned Point(s)).</param>
273         /// <param name="x1">Control point 1 X.</param>
274         /// <param name="y1">Control point 1 Y.</param>
275         /// <param name="x2">Control point 2 X.</param>
276         /// <param name="y2">Control point 2 Y.</param>
277         /// <param name="x3">Ending point X (included in the returned Point(s)).</param>
278         /// <param name="y3">Ending point Y (included in the returned Point(s)).</param>
279         /// <returns></returns>
280         List<IntPoint> CalculateCurvePoints(double tInterval, double x0, double y0, double x1, double y1, double
281         {
282             //Create a new partial Polygon to store the calculated Points.
283             List<IntPoint> calculatedPoints = new List<IntPoint>((int)(1d / tInterval));
284
285             double oneMinusT;
286             double oneMinusTSquared;
287             double oneMinusTCubed;
288
289             double tSquared;
290             double tCubed;
291
292             double oneMinusTSquaredTimesTTimesThree;
293             double oneMinusTTimesTSquaredTimesThree;
294
295             //t will go from tInterval to 1d at the interval of tInterval. t starts
296             //at tInterval instead of 0d because the first Point in the curve is
297             //skipped. This is needed because multiple curves will be placed
298             //sequentially after each other and we don't want to have the same
299             //Point be added to the Polygon twice.
300             for (double t = tInterval; t < 1d; t += tInterval)
301             {
302                 //There are 3 "layers" in a cubic Bezier curve's calculation. These "layers"
303                 //must be calculated for each intermediate Point (for each value of t from
304                 //tInterval to 1d). The Points in each "layer" store [the distance between
305                 //two consecutive Points from the previous "layer" multipled by the value
306                 //of t (which is between 0d-1d)] plus [the position of the first Point of
307                 //the two consecutive Points from the previous "layer"]. This must be
308                 //calculated for the X and Y of every consecutive Point in every layer
309                 //until the last Point possible is reached, which is the Point on the curve.
310
311                 //Note: the code below is an optimized version of the commented explanation above.
312
313                 oneMinusT = 1d - t;
314                 oneMinusTSquared = oneMinusT * oneMinusT;
315                 oneMinusTCubed = oneMinusTSquared * oneMinusT;
316
317                 tSquared = t * t;
318                 tCubed = tSquared * t;
319
320                 oneMinusTSquaredTimesTTimesThree = oneMinusTSquared * t * 3d;
321                 oneMinusTTimesTSquaredTimesThree = oneMinusT * tSquared * 3d;
322
323                 calculatedPoints.Add(new IntPoint(
324                     (long)(oneMinusTCubed * x0 + oneMinusTSquaredTimesTTimesThree * x1 + oneMinusTTimesTSquaredT
```

pinta/Pinta.Tools/Editable/Shapes/EllipseEngine.cs (revision 10662429ab96515bb2855cde252b04e1fd65212a)

```
249         /// <summary>
250         /// Calculate each intermediate Point in the specified curve, returning Math.Round(1d / tInterval - 1d)
251         /// </summary>
252         /// <param name="tInterval">The increment value for t (should be between 0-1).</param>
253         /// <param name="x0">Starting point X (not included in the returned Point(s)).</param>
254         /// <param name="y0">Starting point Y (not included in the returned Point(s)).</param>
255         /// <param name="x1">Control point 1 X.</param>
256         /// <param name="y1">Control point 1 Y.</param>
257         /// <param name="x2">Control point 2 X.</param>
258         /// <param name="y2">Control point 2 Y.</param>
259         /// <param name="x3">Ending point X (included in the returned Point(s)).</param>
260         /// <param name="y3">Ending point Y (included in the returned Point(s)).</param>
261         /// <param name="cPIndex">The index of the previous ControlPoint to the generated points.</param>
262         /// <returns></returns>
263         protected List<GeneratedPoint> calculateCurvePoints(
264             double tInterval,
265             double x0, double y0, double x1, double y1, double x2, double y2, double x3, double y3,
266             int cPIndex)
267         {
268             List<GeneratedPoint> calculatedPoints = new List<GeneratedPoint>((int)(1d / tInterval));
269
270             double oneMinusT;
271             double oneMinusTSquared;
272             double oneMinusTCubed;
273
274             double tSquared;
275             double tCubed;
276
277             double oneMinusTSquaredTimesTTimesThree;
278             double oneMinusTTimesTSquaredTimesThree;
279
280             for (double t = 0; t < 1d; t += tInterval)
281             {
282                 //There are 3 "layers" in a cubic Bezier curve's calculation. These "layers"
283                 //must be calculated for each intermediate Point (for each value of t from
284                 //tInterval to 1d). The Points in each "layer" store [the distance between
285                 //two consecutive Points from the previous "layer" multipled by the value
286                 //of t (which is between 0d-1d)] plus [the position of the first Point of
287                 //the two consecutive Points from the previous "layer"]. This must be
288                 //calculated for the X and Y of every consecutive Point in every layer
289                 //until the last Point possible is reached, which is the Point on the curve.
290
291                 //Note: the code below is an optimized version of the commented explanation above.
292
293                 oneMinusT = 1d - t;
294                 oneMinusTSquared = oneMinusT * oneMinusT;
295                 oneMinusTCubed = oneMinusTSquared * oneMinusT;
296
297                 tSquared = t * t;
298                 tCubed = tSquared * t;
299
300                 oneMinusTSquaredTimesTTimesThree = oneMinusTSquared * t * 3d;
301                 oneMinusTTimesTSquaredTimesThree = oneMinusT * tSquared * 3d;
302
303                 calculatedPoints.Add(new GeneratedPoint(new PointD(
304                     (oneMinusTCubed * x0 + oneMinusTSquaredTimesTTimesThree * x1 + oneMinusTTimesTSquaredTimesTh
305                     (oneMinusTCubed * y0 + oneMinusTSquaredTimesTTimesThree * y1 + oneMinusTTimesTSquaredTimesTh
306                     cPIndex));
307             }
```

Wartungsprobleme schrittweise aufräumen



Keine Defizite

Keine Defizite in geändertem Code

Keine neuen Defizite

Mir doch egal

6 Findings in **geändertem** Code

Add schema (on for Oracle on QA
by Daniel Lewis vision 93beea0b2623ec52f362d4ca5c7a71039ab14409 (git)

Files: 14 changed, deleted
Findings: 7 6 9

Jun 24 2015
16:35

Merge branch 'master' of https://github.com/mittalabhi2123/Activiti into mittalabhi2123-master
by Isabella Wright in revision 0292ee4f7ace6a1c81af64f71fac78935c39b101 (git)

Files: 4 added, 34 changed
Findings: 44 20 8

Jun 24 2015
13:07

ACT-4012: IllegalArgumentException when invoking a service task requiring a date (JaxB custom bindings must be used not only when importing the WSDL, but also when creating web-service clients)
by Anya Hill in revision 79db90d23c5af810eea8b830534e3df390aa21a1 (git)

Files: 20 changed, 3 deleted
Findings: 1 15 35

Jun 24 2015
11:06

Wirksamkeit von Änderungen überprüfen

Source: unknown

jenkinsci/jenkins

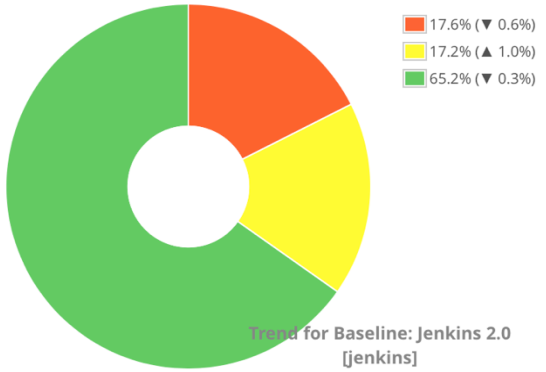
Baseline Delta für Jenkins 2.0

LOC for jenkins

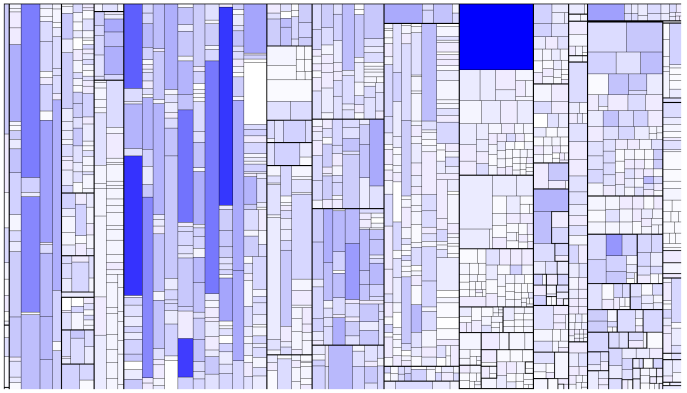
220.7k

+12,903

File Size for jenkins



Ownership for jenkins

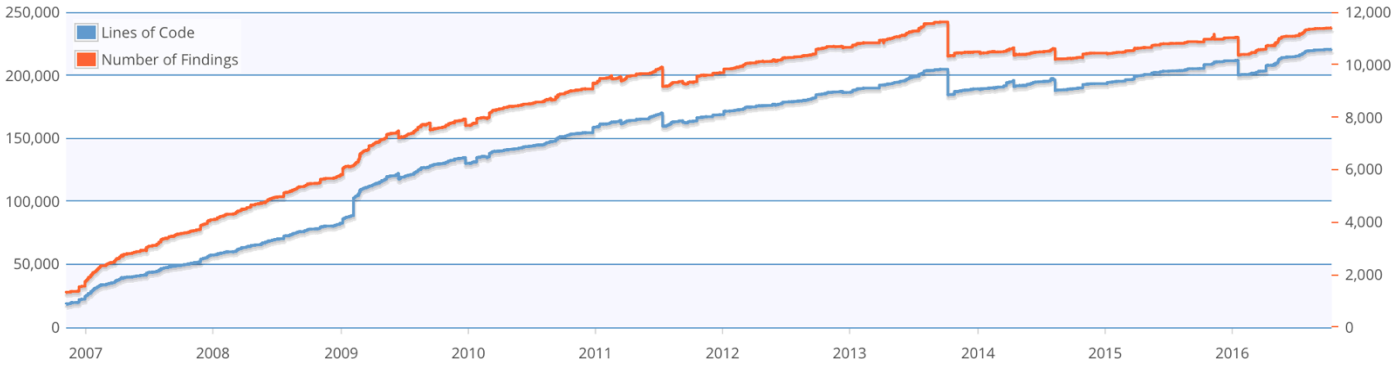


Clone Coverage for jenkins

3.7%

+1.18%

LOC vs. Clone Coverage Trend for jenkins



Fazit

Historienanalysen schaffen ein umfassenderes **Verständnis** als Analysen auf dem aktuellen Stand.

Dadurch erlauben sie die Erkennung und Behebung von **Ursachen**, nicht nur Symptomen.

Kontakt

Ich freue mich auf Diskussionen!

Für eine Evaluierungslizenz von Teamscale (www.teamscale.com),
melden Sie sich bitte bei mir.

Wenn Sie Interesse an einem Job in diesem Bereich haben, bitte auch melden ☺

Dr. Dennis Pagano · pagano@cqse.eu · +49 1590 4062957
@dennispagano

CQSE GmbH
Lichtenbergstraße 8
85748 Garching bei München

CQSE
Continuous Quality in Software Engineering