

RISK BASED CODE COVERAGE ANALYSIS



@BartSzulc

RISK BASED CODE COVERAGE ANALYSIS



bartosz.szulc@spartez.com

RISK BASED CODE COVERAGE ANALYSIS



bszulc@atlassian.com



I do:

1. Deliver Jira.
2. Organise Agile & Automation Days.
3. Teach developers quality engineering.



I'm interested in:

1. Scaling best practices.
2. Development productivity.
3. Continuous deployment.

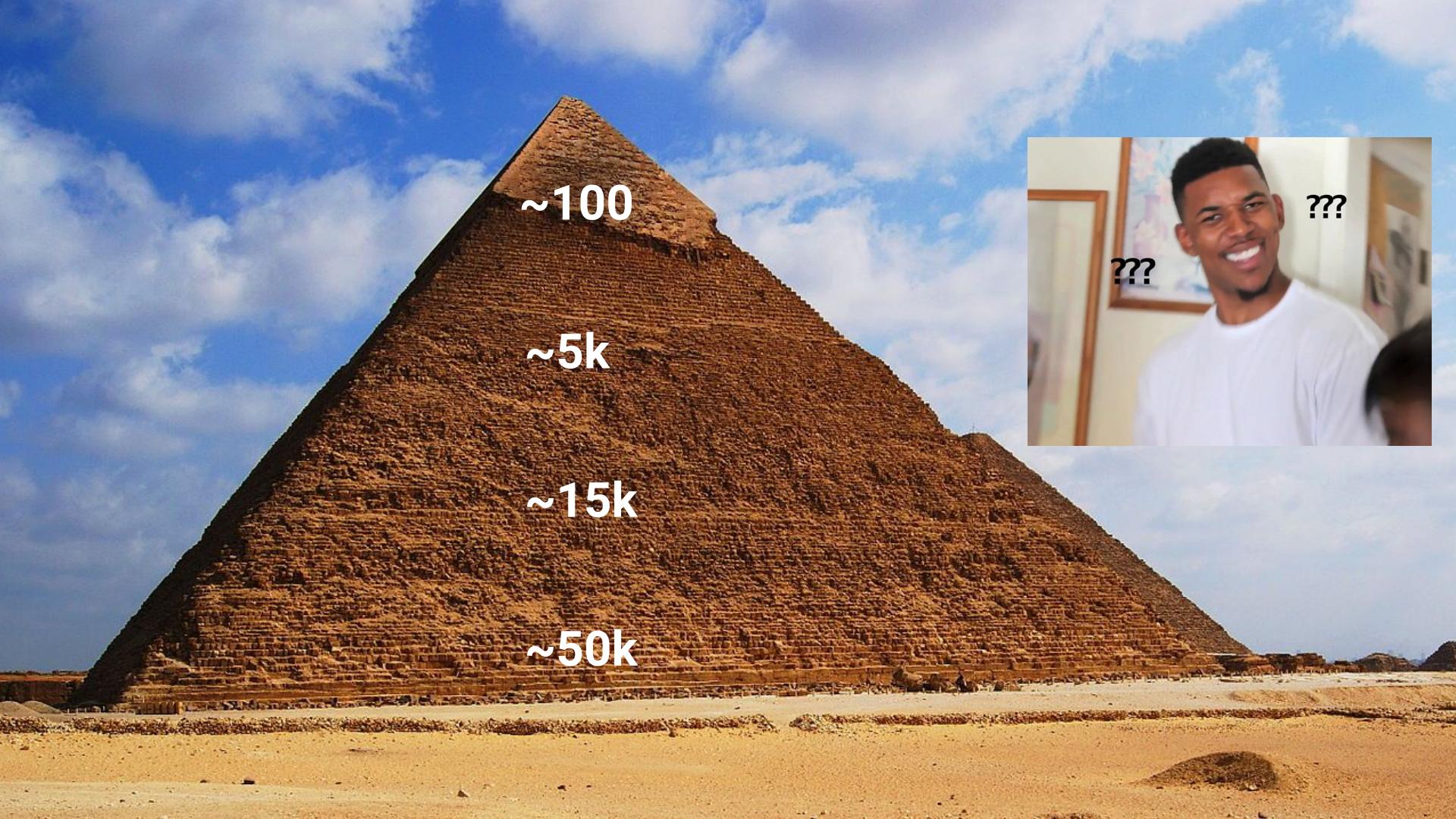


**CAUTION
CONSTRUCTION
ZONE**







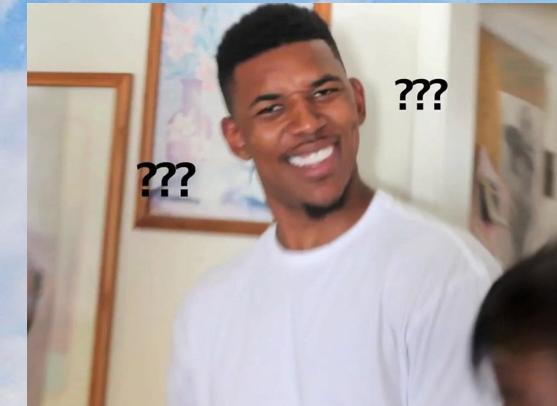


~100

~5k

~15k

~50k



What I'm going to talk about.

1. Risk.
2. Tests.
3. You need, but don't have.
4. You don't need, but have.

What I'm **NOT** going to talk about.

1. Code coverage.
2. What it is.
3. What types there are.
4. What tools there are.
5. All the problems with it.

Standard Code Coverage



Project overview

PACKAGES

Type and press enter to filter packages... Q

default-pkg	86,4%
edu.emory	49,1%
mathcs.backport	0%
java.util	
sun.misc	

Project Clover database Wt lip 1 2014 09:43:03 CEST

Project overview

[Dashboard](#) [Application code](#) [Test code](#) [Test results](#) [Top risks](#) [Quick wins](#) [Coverage tree map](#)

Code coverage

70,4%

215 classes, 10 153 / 14 415 elements

[See more](#)

Test results

100%

1 / 1 tests 0,01 secs

[See more](#)

Code metrics

Branches: 3 784 Statements: 8 465 Methods: 2 166 Classes: 215

Files: 83 Packages: 6 LOC: 33 802 NCLOC: 15 283

Total complexity: 4 384 Complexity density: 0,52

Statements/Method: 3,91 Methods/Class: 10,07 Classes/Package: 35,83

Average method complexity: 2,02

Top 20 project risks

ThreadHelpers CopyOnWriteArrayList.COWSubList Utils

Collections.CheckedMap Collections.CheckedMap.EntrySetView

Arrays TreeMap.ValueSet

ScheduledThreadPoolExecutor.DelayedWorkQueue

Semaphore.NonfairSync TreeMap.BaseEntryIterator

ConcurrentSkipListMap

ReentrantLock.NonfairSync

ReentrantReadWriteLock.ReadLock PriorityQueue.ltr

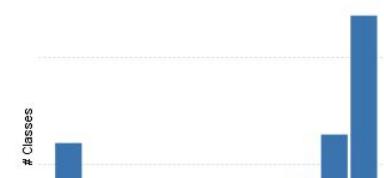
Collections.CheckedMap.EntryView FIFOCondVar

ConcurrentHashMap.HashIterator WaitQueue.WaitNode

Collections.CheckedCollection SynchronousQueue.FifoWaitQueue

[See more](#)

Class Coverage Distribution



Most complex packages

1.	78,5%		edu.emory.mathcs.backport.java.util.concurrent	2431
2.	49,1%		edu.emory.mathcs.backport.java.util	1425
3.	81,2%		edu.emory.mathcs.backport.java.util.concurrent.locks	289
4.	94,2%		edu.emory.mathcs.backport.java.util.concurrent.atomic	152
5.	71,1%		edu.emory.mathcs.backport.java.util.concurrent.helpers	



Atlassian JIRA 1001.0.0-SNAPSHOT (Aggregated)

Project overview

PACKAGES

Type to filter packages...

Test	Test contribution
36	
37	
38	
39	
40	
41	
42	
43	com.atlassian.jira.webtests.ztests.issue.TestWikiRendererXSS.testCodeMacro
44	
45	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssuesNotifications.testBulkDeleteNoti
46	
47	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssuesNotifications.testBulkDeleteSub
48	
49	com.atlassian.jira.webtests.ztests.issue.TestLabelsFormats.testBulkOperations
50	
51	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssuesNotifications.testBulkDeleteNo
52	
53	com.atlassian.jira.webtests.ztests.misc.TestDatabaseSetup.testAll
54	
55	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssuesNotifications.testBulkDeleteSub
56	
57	com.atlassian.jira.webtests.ztests.bulk.TestBulkTransition.testBulkTransitionDuplicateWorkflow
58	
59	com.atlassian.jira.webtests.ztests.issue.TestLabelsFormats.testIssueNav
60	
61	com.atlassian.jira.webtests.ztests.misc.TestDefaultJiraDataFromInstall.testDefaultJiraData
62	
63	com.atlassian.jira.webtests.ztests.bulk.TestBulkMoveIssuesForEnterprise.testBulkMoveIssue
64	
65	com.atlassian.jira.webtests.ztests.bulk.TestBulkMoveWithMultiContexts.testBulkMoveIssueWi
66	
67	3795 if (JiraSystemProperties.getInstance().isVertigoMode()) { 0 return false; }
68	
69	3795 return super.enableUpgradeTasks();

Custom Code Coverage

covered class

covered line

type of test

test method/case

*	class_name	method_name	line_number	test_type	test_suite	test_case
1	com.atlassian.jira.issue.attachment.s...	start		62	unit	com.atlassian.jira.issue.attachment.store.media.sync.TestJir...
2	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
3	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.issue.TestWikiRendererXSS
4	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
5	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.issue.TestLabelsFormats
6	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
7	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.misc.TestDatabaseSetup
8	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
9	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.bulk.TestBulkTransition
10	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.issue.TestLabelsFormats

covered method

test class/suite

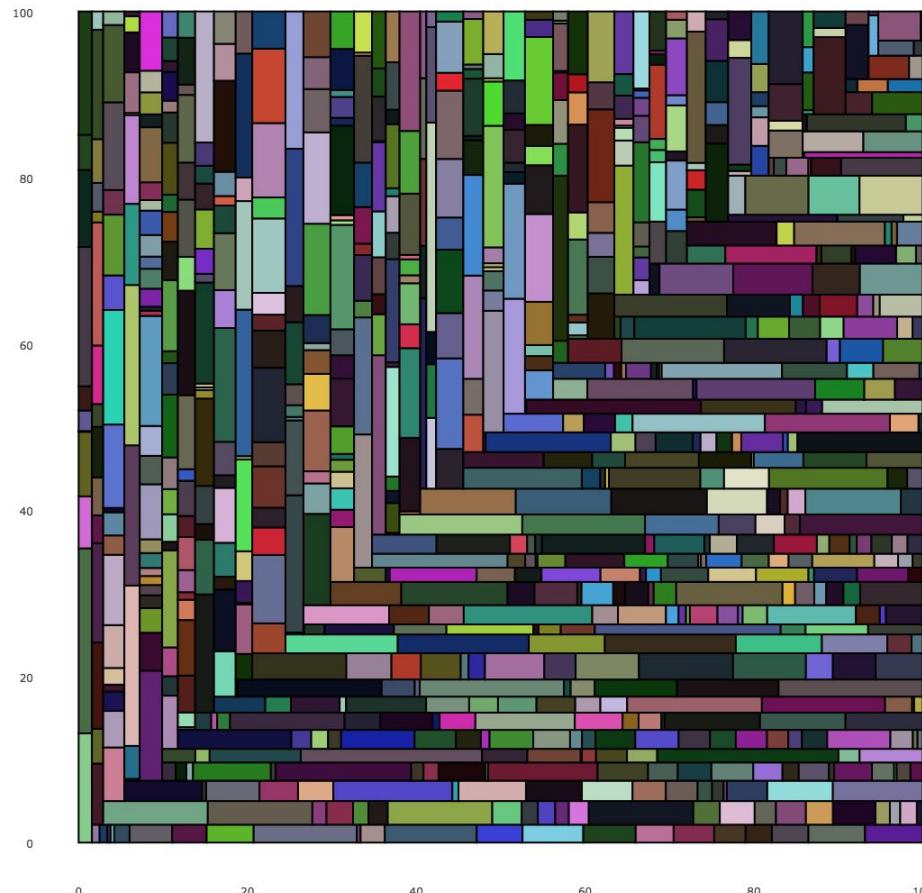
$80\% < X < 99\%$





missing coverage

Risk based code coverage analysis





Risk Based Coverage

```
17 string sInput;
18 int iLength, iN;
19 double dblTemp;
20 bool again = true;
21
22 while (again) {
23     iN = -1;
24     again = false;
25     getline(cin, sInput);
26     system("cls");
27     stringstream(sInput) >> dblTemp;
28     iLength = sInput.length();
29     if (iLength < 4) {
30         again = true;
31         continue;
32     } else if (sInput[iLength - 3] != '.') {
33         again = true;
34         continue;
35     } while (++iN < iLength) {
36         if (isdigit(sInput[iN])) {
37             continue;
38         } else if (iN == (iLength - 3)) {
39             continue;
40         }
41     }
42 }
```

80% * 500k lines

20% * 500k lines

100 000
lines of code without a
single test



probability

×

severity

frequency of unfortunate

✗

lost value



```
$ git log --since='last 24 months' --oneline --follow
bb7bdd4 JVC-347 Commited all Tenant API's mentioned in requirement in jira ticket. Added to
f996644 DEBT-7: Code functioning with --base class issue fix details.
d504ec6 DEBT-7: Code functioning with --base class issue fix details.
a7f23eb DEBT-7: Code functioning with --base class issue fix details.
a34460a DEBT-7: Code functioning with --base class issue fix details.
eee9a2f [MON-37] review changes with class about removing one additional method
f0195f4 [MON-37] Implementing new mail service, new control and new database
2983145 JDEV-31166 Create new server data database to store information for every user class
e51f765 JDEV-31166
5517573 JDEV-31166
97466e8 JDEV-31166
```

```
$ git log --since='last 24 months' --oneline --follow  
bb7bdd4 JVC-347  
f996644 DEBT-7:  
d504ec6 DEBT-7:  
a7f23eb DEBT-7:  
a34460a DEBT-7:  
eee9a2f [MON-37]  
f0195f4 [MON-37]  
2983145 JDEV-31166  
e51f765 JDEV-31166  
5517573 JDEV-31166  
97466e8 JDEV-31166
```

Tests in Space / TIS-8
Faulty low-pressure oxidizer turbopump

Edit Comment Assign More Start Progress Done Start Review Votes: 0 Watchers: 1 Stop watching this issue

Description
Faulty low-pressure oxidizer turbopump

Test Coverage

- Check the durability of the high-pressure fuel | Smoke | Regression | TURBOPUMP | APPROVED | 2h 0m
- Check the stability of the low-pressure oxidizer turbopump | Functional | IGNITER | APPROVED | 3h 0m
- Ensure the axial-flow pump is enabled | Smoke | VALVES | APPROVED | 1h 30m

Test Runs

TIS-8 - Faulty low-pre | NOT EXECUTED | 10/02/2015 | 0% | 3 | 1

Impacted Test Results

- Check the durability of the high-pressure fuel turbopump (with steps) | FAIL | 10/02/2015 13:44

Activity



N bugs / M days



{



```
"name": "customer bugs",
"tracker": "jac",
"project": "JRA",
"issue_type": "Bug",
"score": 50.0
```

},

{



```
"name": "blocking pipeline
"tracker": "jdog",
"project": "JDEV",
"issue_type": "Bug",
"priority": "Blocker",
"score": 100.0
```

},

{



```
"name": "non blocking pipe
"tracker": "jdog",
"project": "JDEV",
"issue_type": "Bug",
"score": 50.0
```

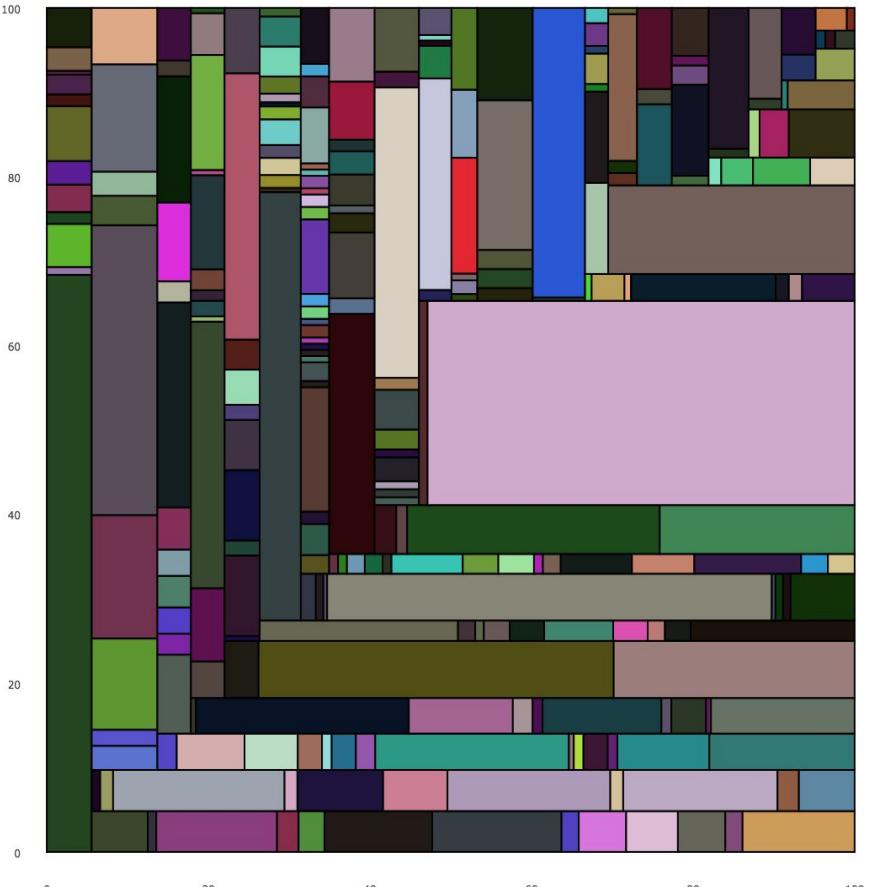


Risk based code coverage analysis

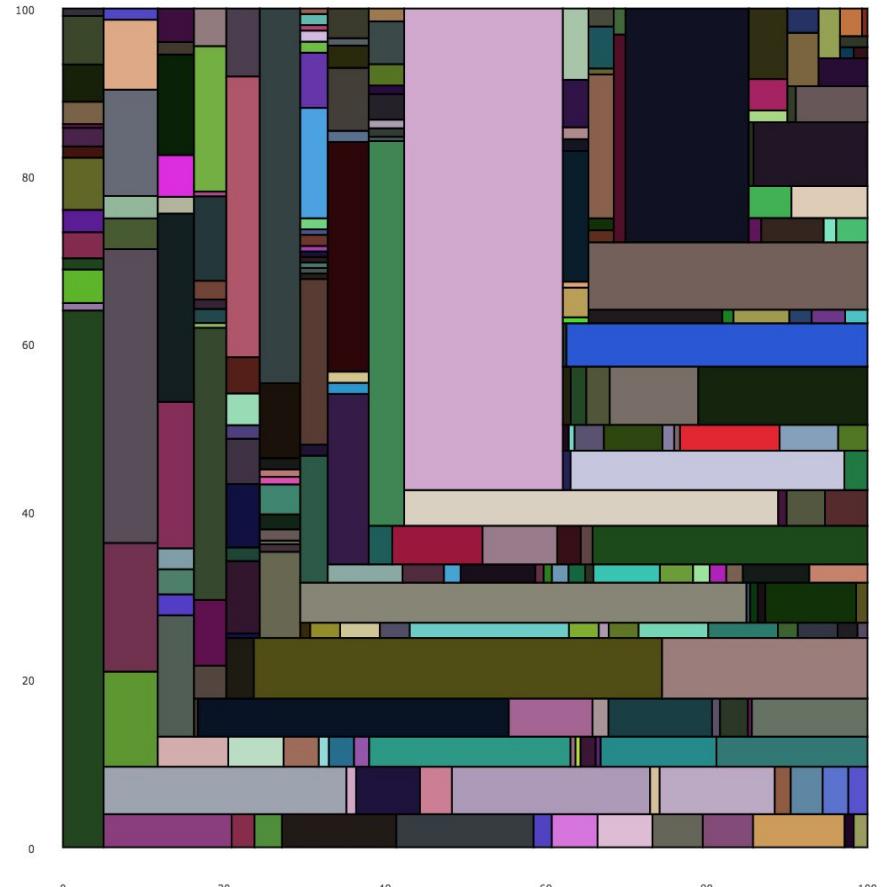


N bug points / M days

commit frequency



bug score



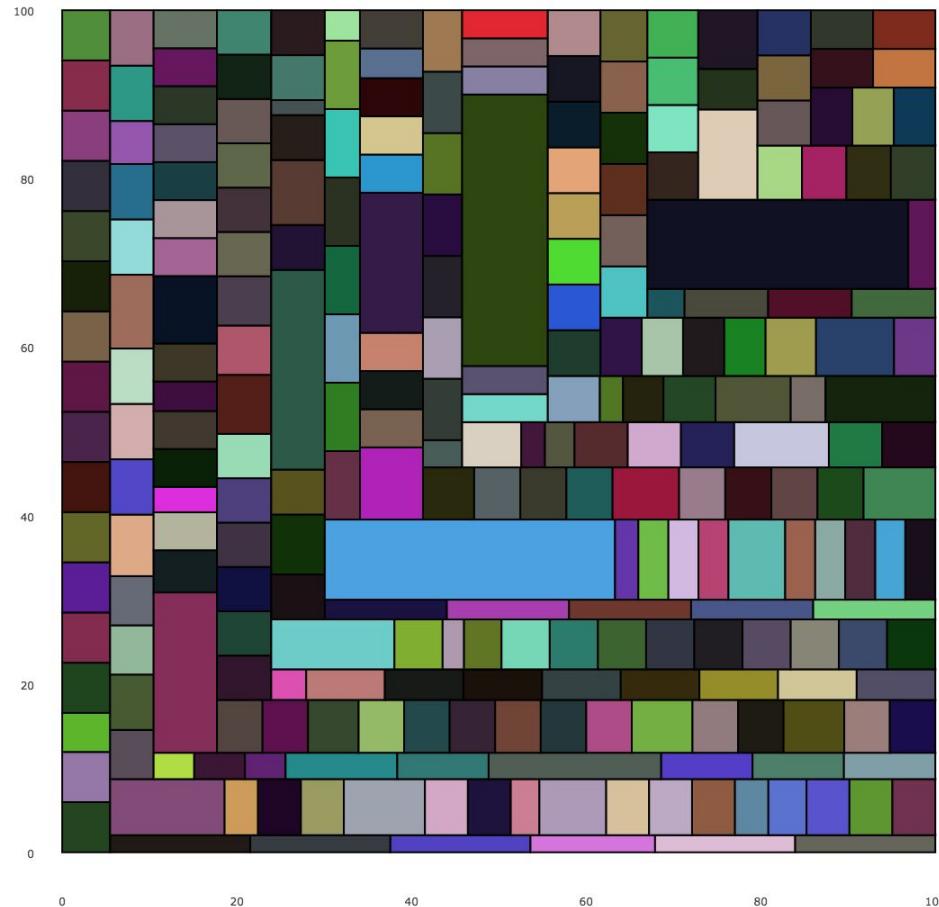
$$\text{BP} = \text{BS} / \text{CF}$$

BP bug probability

CF commit frequency

BS bug score

bug probability





the
SUMMIT

SUMMIT

SUM



I SEE dev/pies

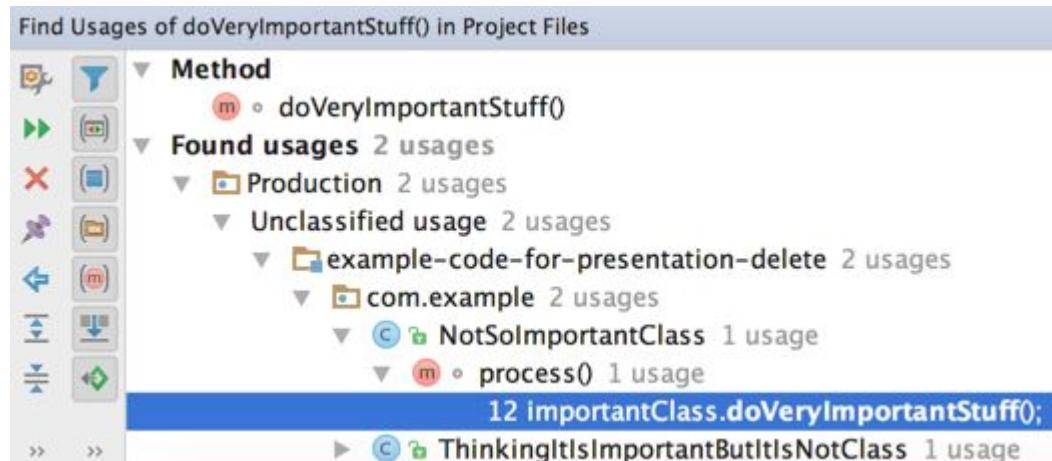
INTECH
P.S. REPORT

IN THE EVENT OF
3 STEPS TO SURVIVAL
1. Assistance
2. Termination

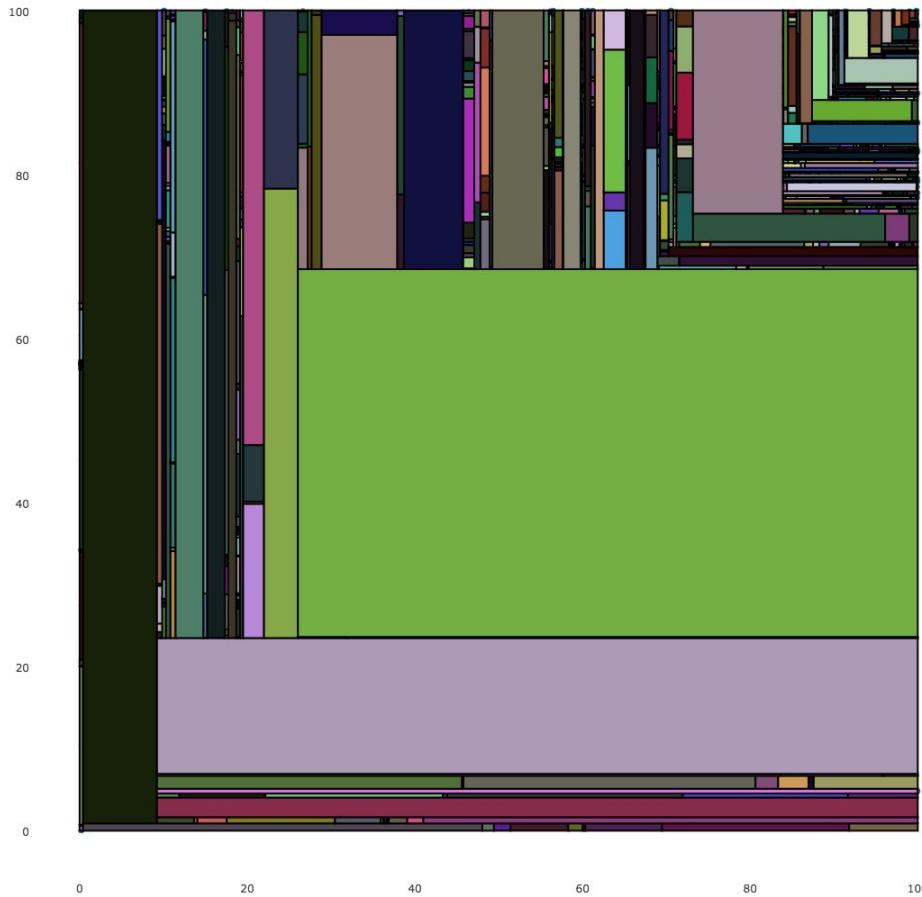
3
4

```
class VeryImportantClass {  
    void doVeryImportantStuff() {  
        System.out.println("doing very important stuff");  
    }  
}  
  
public class NotSoImportantClass {  
    private final VeryImportantClass importantClass;  
  
    public NotSoImportantClass(VeryImportantClass importantClass) {  
        this.importantClass = importantClass;  
    }  
  
    void process() {  
        importantClass.doVeryImportantStuff();  
    }  
}
```

```
class VeryImportantClass {  
    void doVeryImportantStuff() {  
        System.out.println("doing very important stuff");  
    }  
}
```



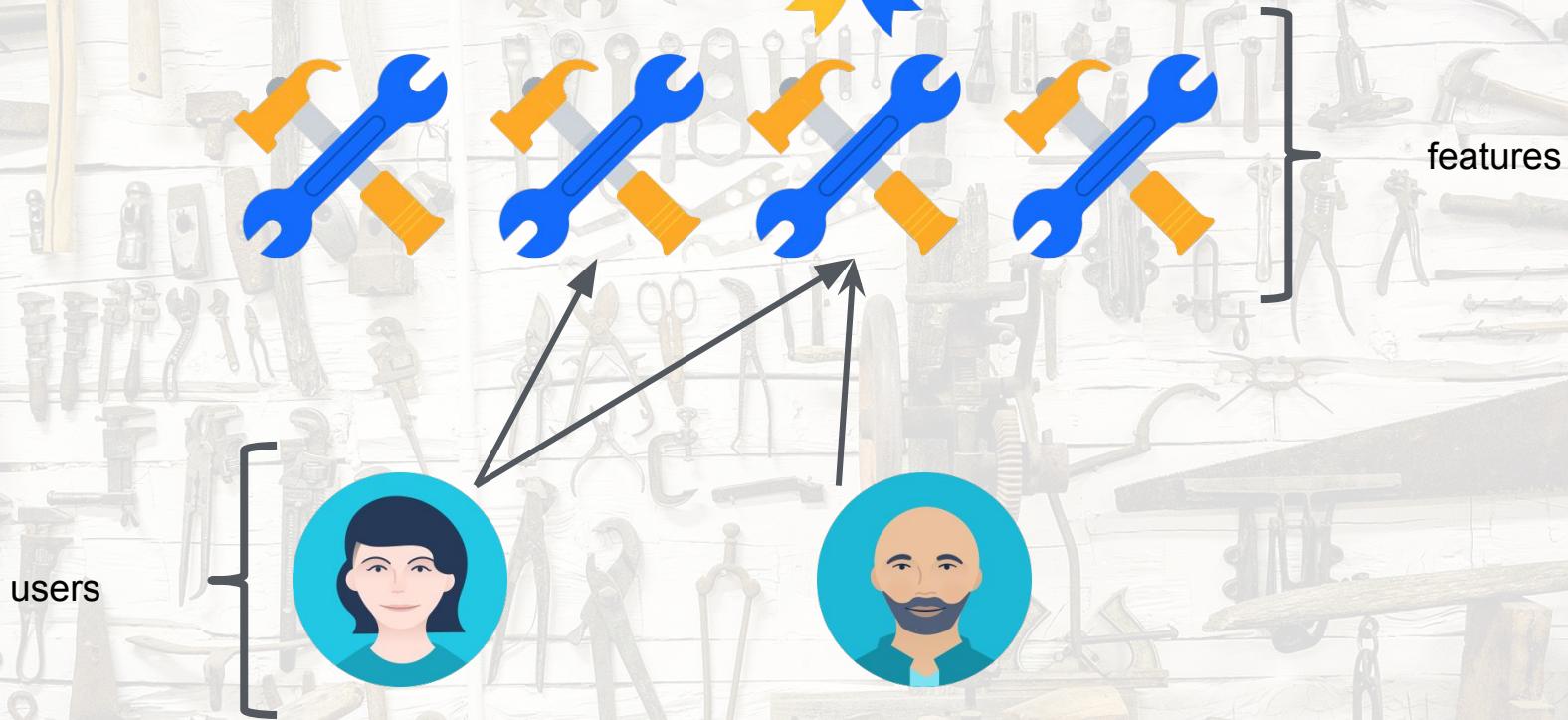
dependent code







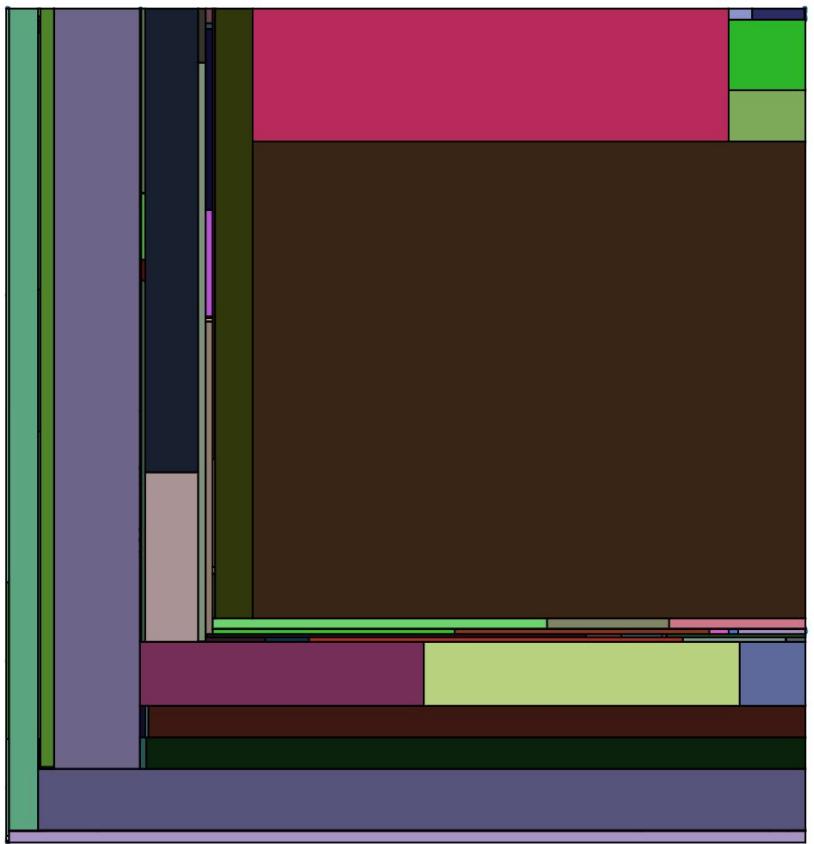
Risk based code coverage analysis



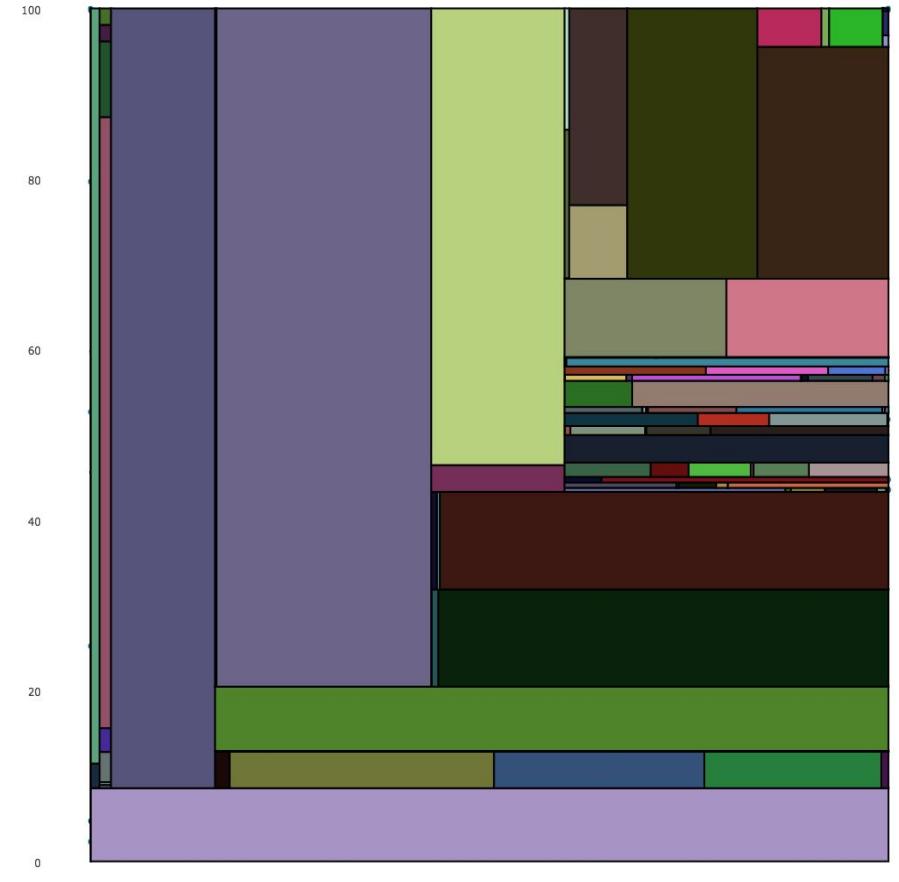
instance1	bob	GET	/resource1
instance1	bob	GET	/resource1
instance2	kate	PUT	/resource2
instance3	eli	DELETE	/resource2
instance4	ann	GET	/resource3
instance2	john	DELETE	/resource5
instance4	ann	GET	/resource3



requests



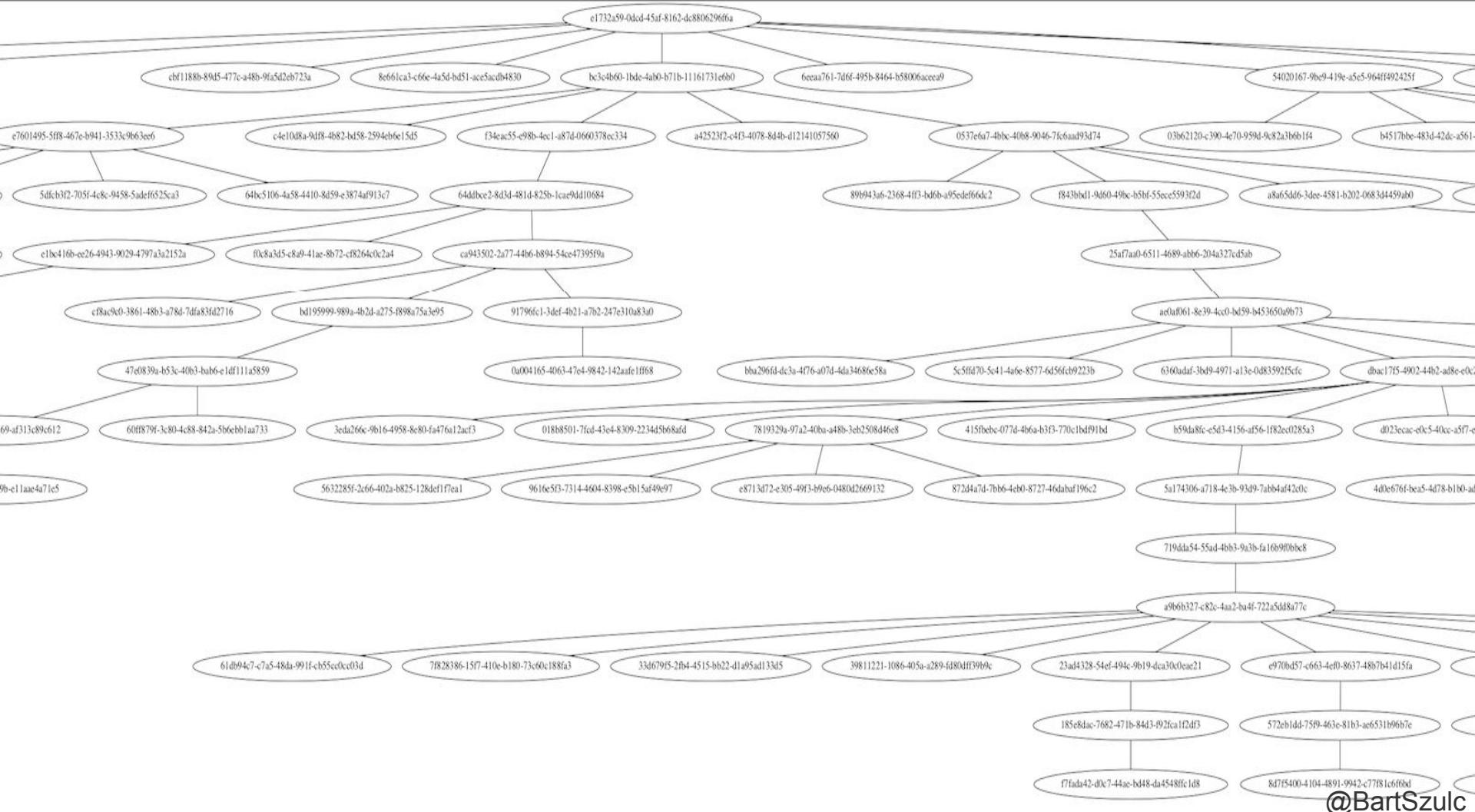
users



GET /feature/process

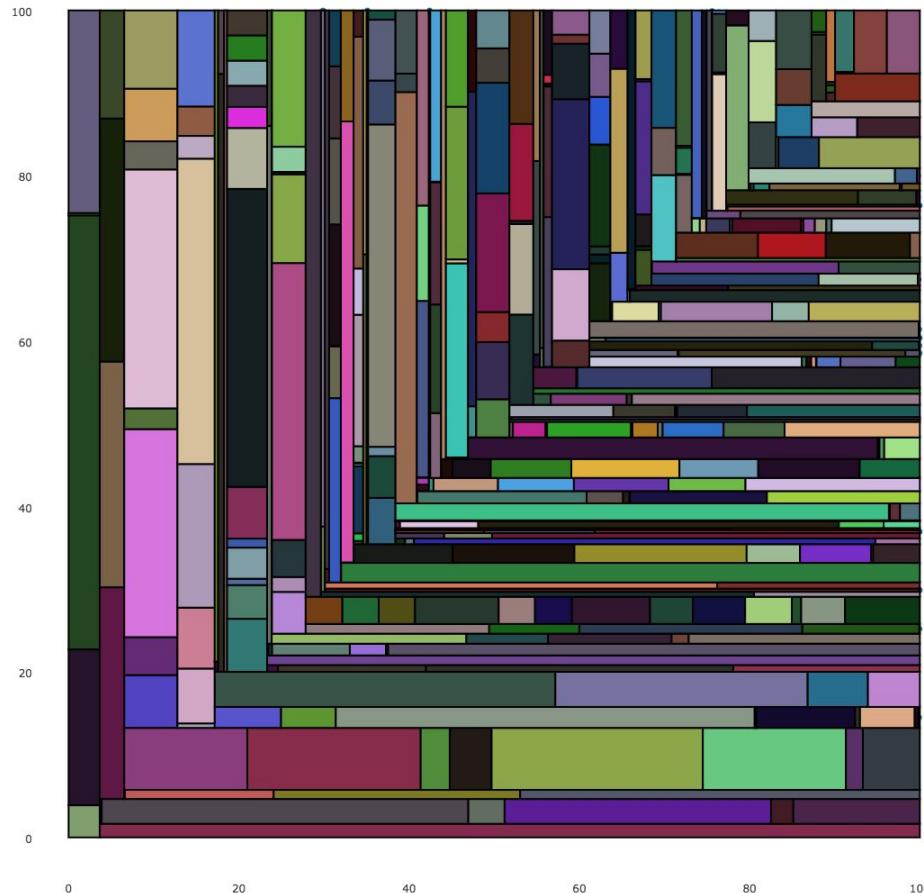
```
@Path("feature")
public class VeryPopularBusinessFeature {

    @GET
    @Path("process")
    public String process() {
        return "Very important processing result";
    }
}
```



@BartSzulc

dependant users



$$\beta(p,q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}$$

$$\sum_{x=1}^{\infty} P(x) = \Gamma(y) = \int_0^{\infty} e^{-x} x^{y-1} dx$$

$$P(x) = \pi x f(x) dx$$

$$\Gamma(y) = \int_1^{\infty} e^{-xt} t^{y-1} dt$$

$$\sigma_x = \sqrt{\sum_{i=1}^n (x_i - \langle x \rangle)^2}$$

$$R = BP * (DM + DU)$$

R risk

BP bug probability

DM dependent methods

DU dependent unique users

$$R = CF(1+0.5BP) * (0.3DM + 0.7DU)$$

R risk

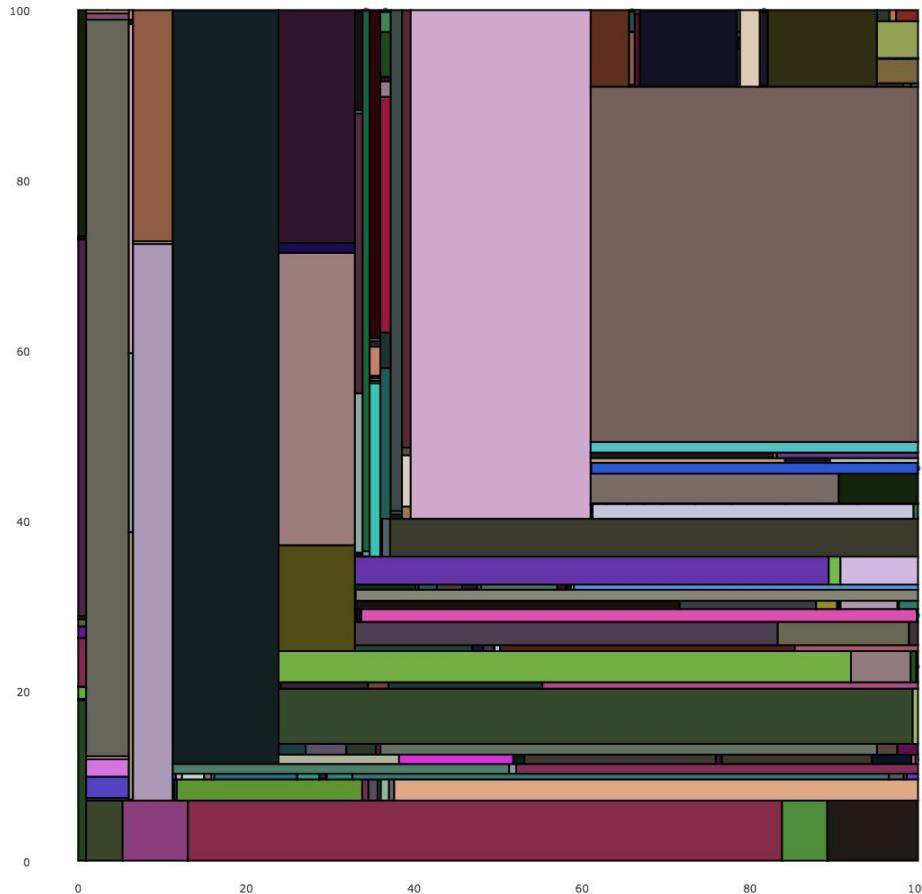
CF commit frequency

BP bug probability

DM dependent methods

DU dependent unique users

risk



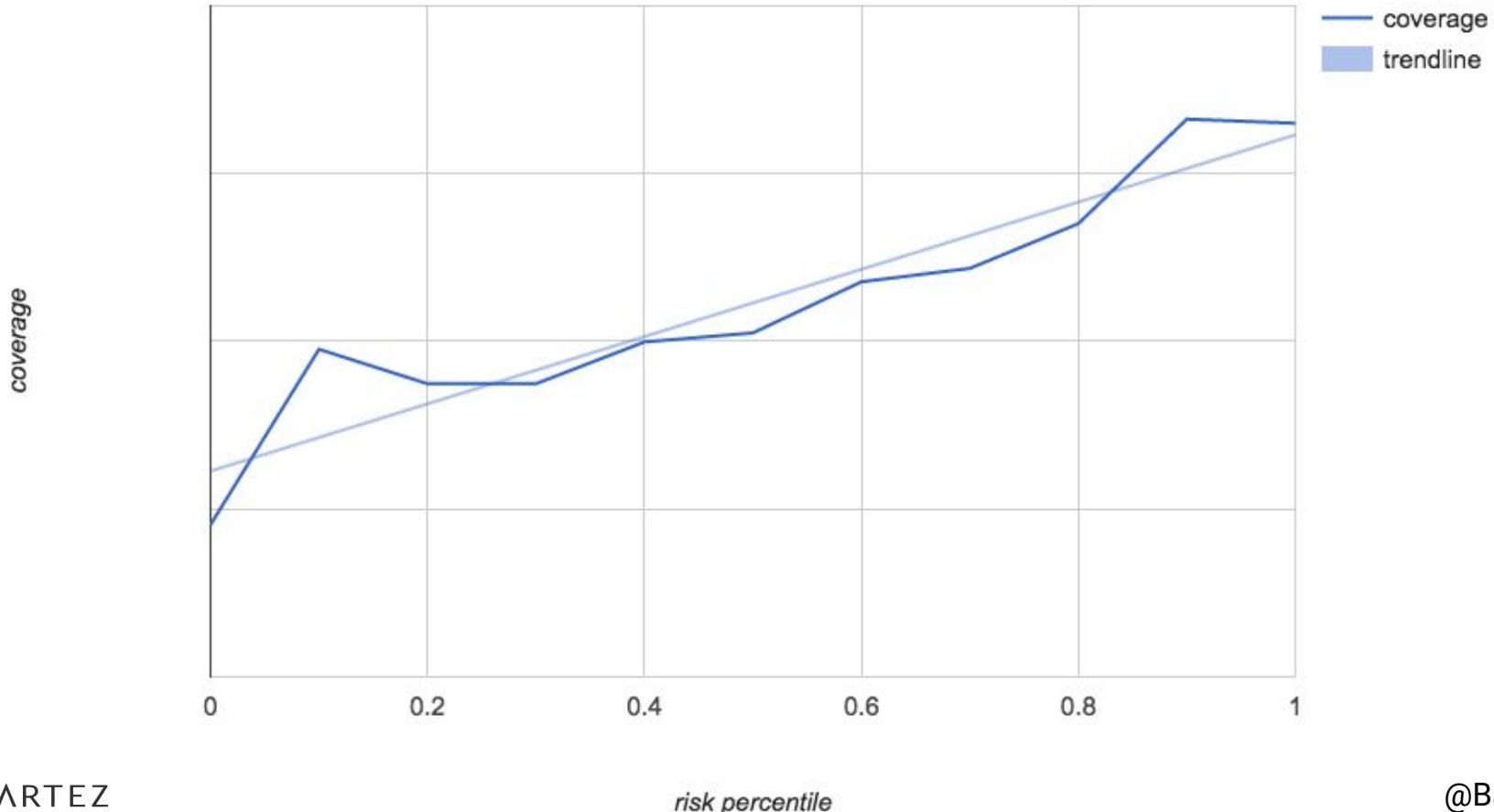
$$c = f(r)$$

$$\lim_{r \rightarrow 1} f(r) = 1$$

$$\lim_{r \rightarrow 0} f(r) = 0$$

bart's test coverage efficiency

bart's test coverage efficiency





**Are there high risk
methods with 0%
statement coverage?**

**Yes! ~10 methods in
risk's 95th percentile**

**Are there high risk
methods without 100%
statement coverage?**

**Yes! ~50 methods in
risk's 95th percentile**

Are there high risk
methods without at least
one *integration test*?

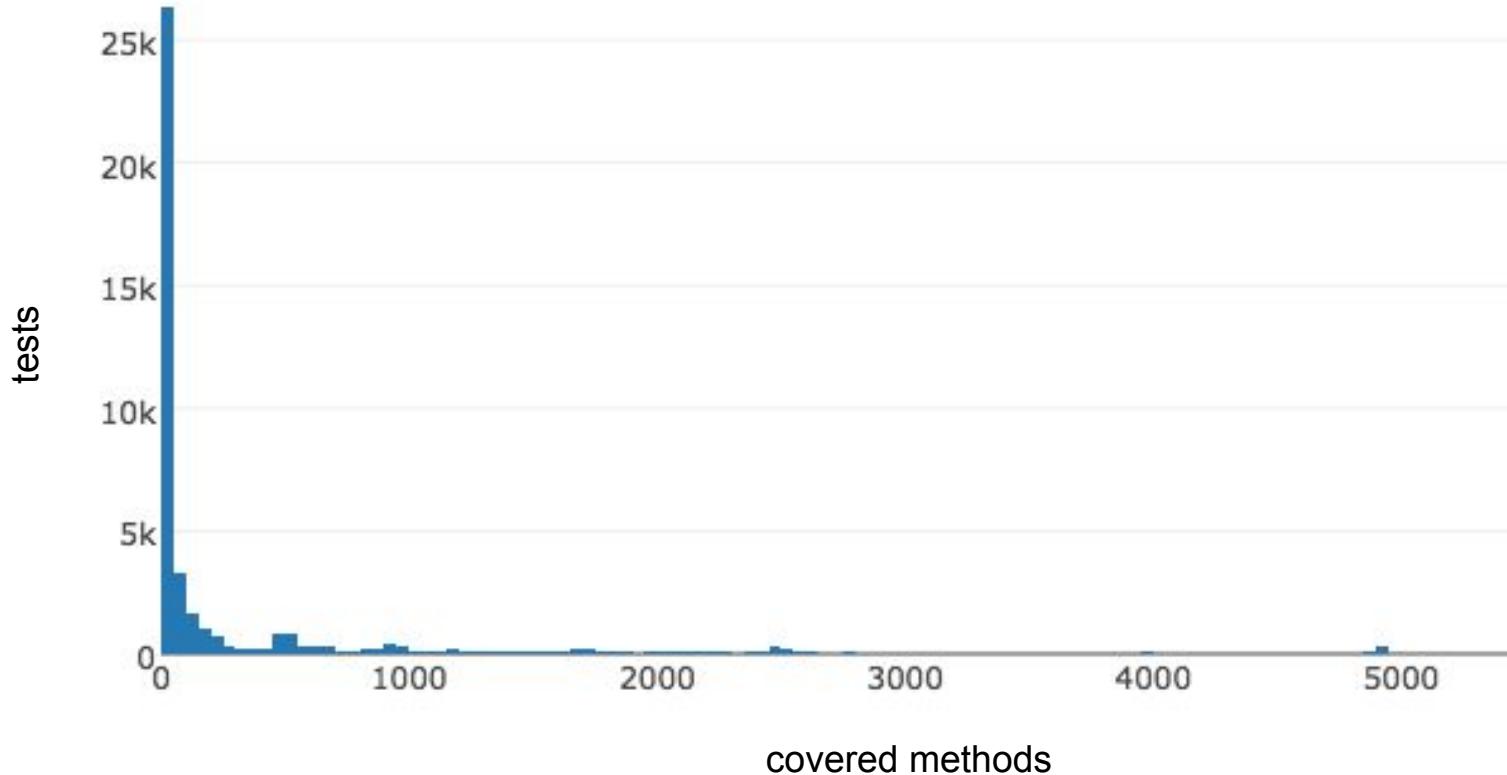
**Yes! ~200 methods in
risk's 95th percentile**

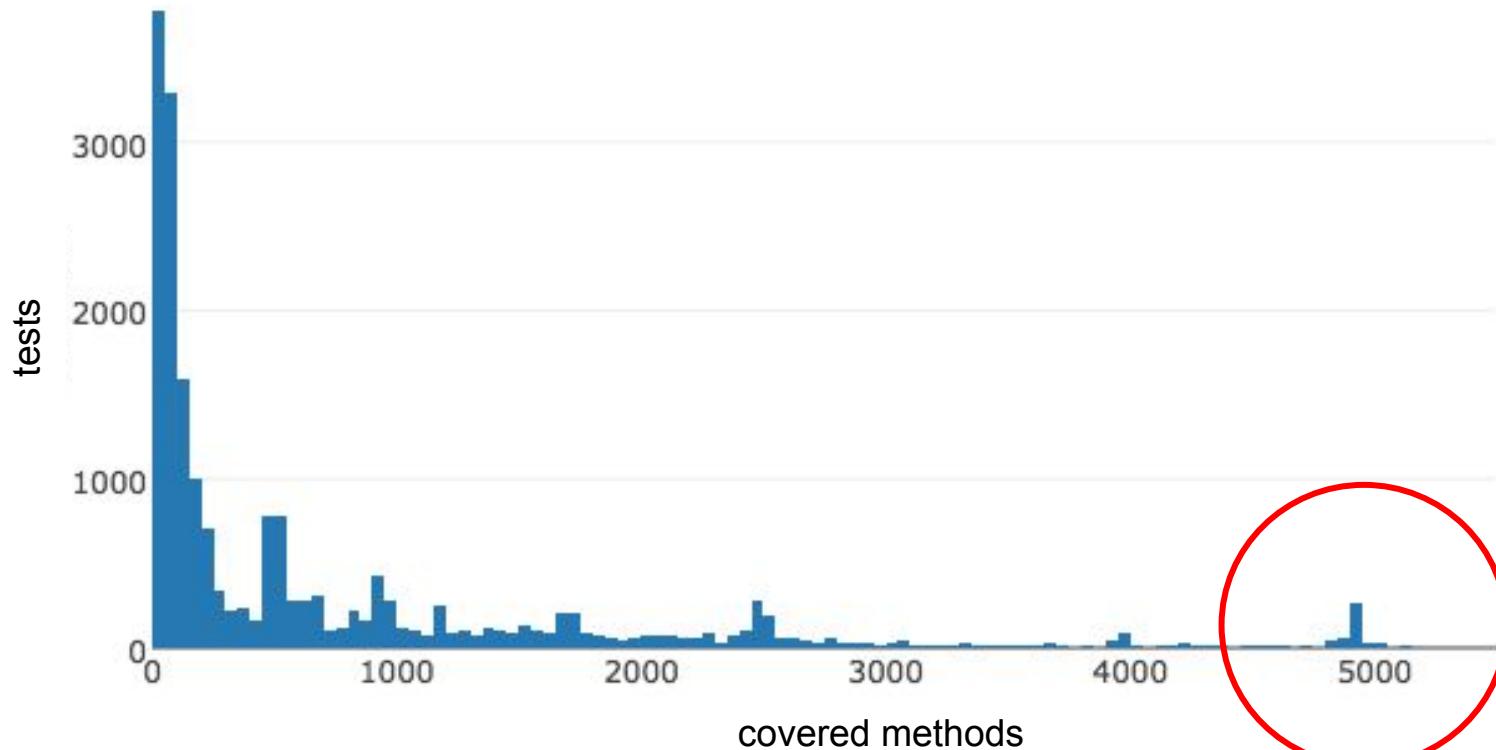


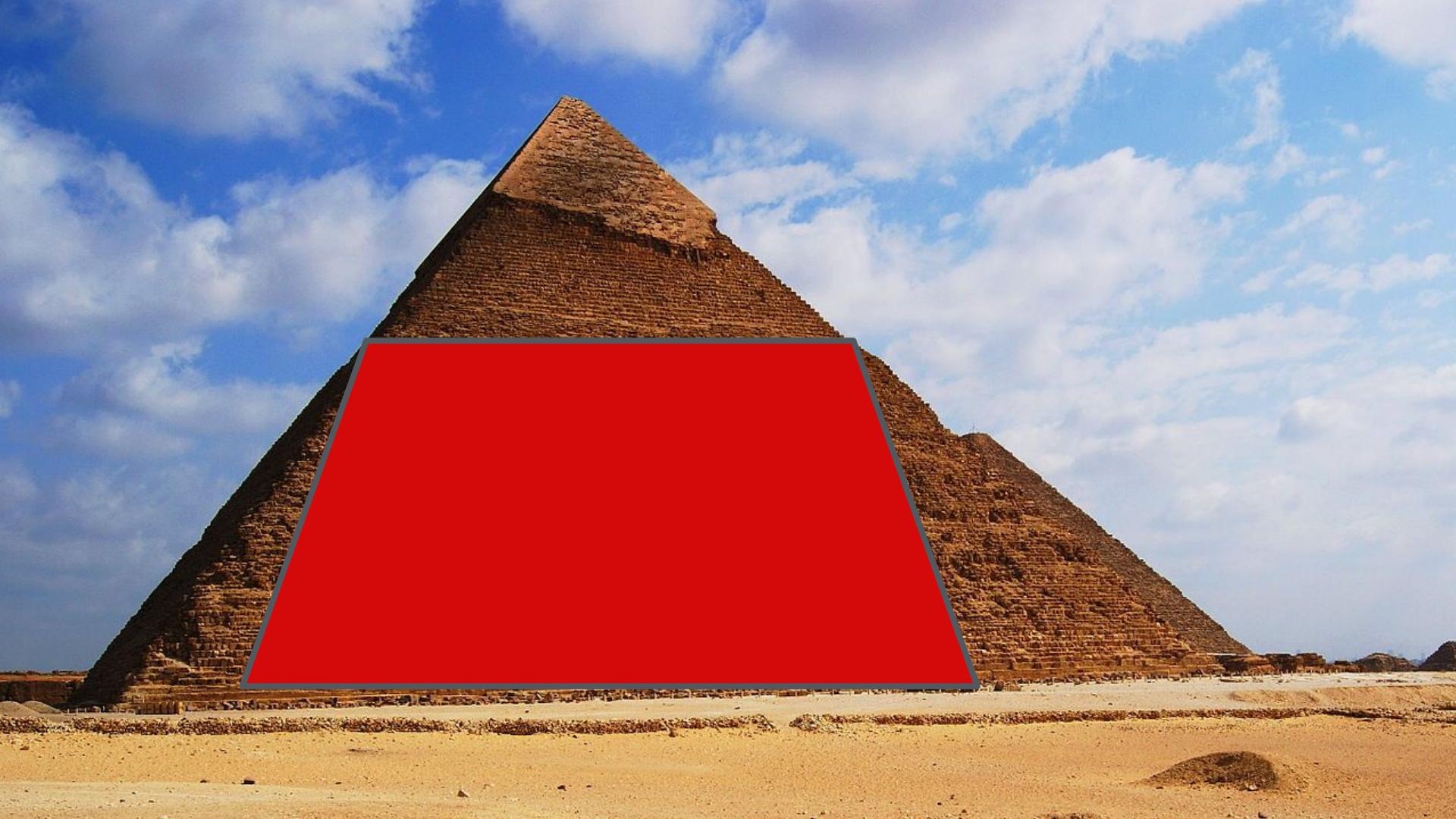
Thoughts, out of nowhere.

1. Dead code. Unused by remaining code.
2. Dead features. Unused by users.
3. Dead tests. Checking dead code and/or dead features.
4. Improvements in test execution. Run only relevant tests.
5. Better understanding of quality of tests.

How good are your tests?







Do I need to run all tests?

covered class

covered line

type of test

test method/case

*	class_name	method_name	line_number	test_type	test_suite	test_case
1	com.atlassian.jira.issue.attachment.s...	start		62	unit	com.atlassian.jira.issue.attachment.store.media.sync.TestJir...
2	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
3	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.issue.TestWikiRendererXSS
4	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
5	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.issue.TestLabelsFormats
6	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
7	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.misc.TestDatabaseSetup
8	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.email.TestBulkDeleteIssue...
9	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.bulk.TestBulkTransition
10	com.atlassian.jira.issue.attachment.s...	start		62	func	com.atlassian.jira.webtests.ztests.issue.TestLabelsFormats

covered method

test class/suite

`/rest/api/1.0/projects/{projectKey}/repos/{repositorySlug}/commits/{commitId}/diff`

RESOURCE-WIDE TEMPLATE PARAMETERS

parameter	value	description
<code>commitId</code>	<code>string</code>	

METHODS

GET

`/REST/API/1.0/PROJECTS/{PROJECTKEY}/REPOS/{REPOSITORYSLUG}/COMMITS/{COMMITID}/DIFF?`
CONTEXTLINES&SINCE&SRCPATH&WHITESPACE&WITHCOMMENTS

This API can also be invoked via a [user-centric URL](#) when addressing repositories in personal projects.
 Retrieve the diff between two provided revisions.

Note: This resource is currently *not paged*. The server will internally apply a hard cap to the streamed lines, and it is not possible to request subsequent pages if that cap is exceeded. In the event that the cap is reached, the diff will be cut short and one or more `truncated` flags will be set to `true` on the segments, hunks and diffs substructures in the returned JSON response.

The authenticated user must have **REPO_READ** permission for the specified repository to call this resource.

tia

[Edit](#)[Export ▾](#)

branch name

Last 30 days

issue/*

[Hide Filters](#)

affected branches

140

total saved tests

average saved tests

average efficiency

median saved tests

median efficiency

326,709**2,334****0.43****1,342****0.25**

total saved time

average saved time [s]

average efficiency

median saved time [s]

median efficiency

2,377,231**16,980****0.39****8,550****0.20**

number of tests

branch_name ▾

issue/JSDUM-692-revert

issue/none-fix-agile-startup-for-JSD-tests

standard_tests ▾

5419.000000

tia_tests ▾

10.000000

saved_tests ▾

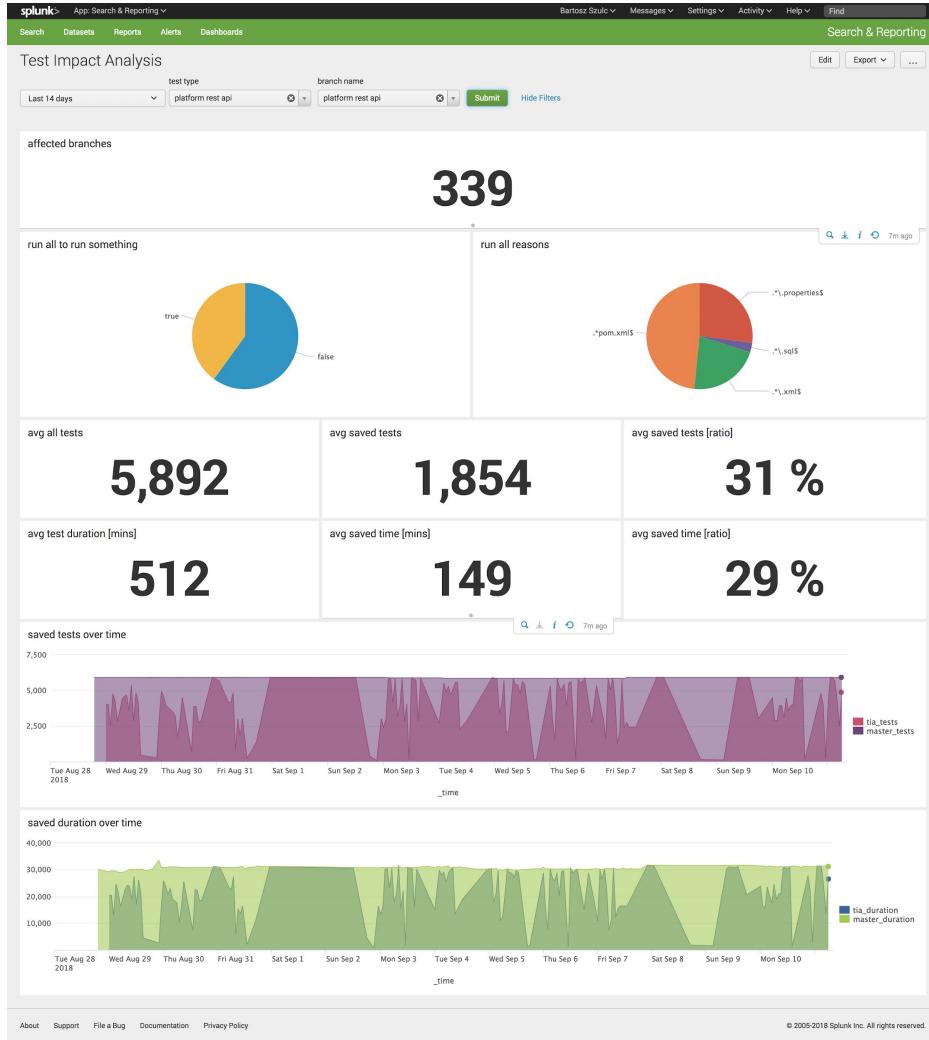
5409.000000

tia_efficien

5350.000000

10.000000

5340.000000





Lessons learnt.

1. Test coverage alone is not very useful.
2. Understanding business and technical perspective helps derive conclusions.
3. Less test coverage isn't necessarily a bad thing.
4. In fact, in some cases, less test coverage might be good.
5. Coverage enables test impact analysis.
6. My brain hurts after statistics... and probably so does yours.



Thank you!

Image credits

https://commons.wikimedia.org/wiki/File:2012-08-30_pano_gdansk_sm2.jpg (Creative Commons 3.0)
<http://www.publicdomainpictures.net/view-image.php?image=27064&picture=construction-zone> (Public Domain)
<https://www.flickr.com/photos/skuds/3781599677> (Creative Commons 2.0)
<https://pixabay.com/en/dna-dns-biology-genetic-material-1388696/> (Creative Commons/Public Domain)
https://commons.wikimedia.org/wiki/File:The_pyramid_of_khafre.jpg (Creative Commons 3.0)
<https://pixabay.com/en/source-code-code-programming-c-583537/> (Creative Commons/Public Domain)
<http://popicok.deviantart.com/art/Risky-place-274701619> (Creative Commons)
<https://www.flickr.com/photos/pauliantero/4685925036> (Creative Commons)
https://commons.wikimedia.org/wiki/File:TeamTimeCar.com-BTTF_DeLorean_Time_Machine-OtoGodfrey.com-JMortonPhoto.com-04.jpg (Creative Commons)
<https://commons.wikimedia.org/wiki/File:JIRA.png> (Public Domain)
<https://www.flickr.com/photos/drphotomoto/10410956343> (Creative Commons)
https://commons.wikimedia.org/wiki/File:Insect_collage.png (Creative Commons 3.0)
<https://www.flickr.com/photos/finizio/183975129> (Creative Commons)
https://commons.wikimedia.org/wiki/File:Explosion-155624_icon.svg (Creative Commons)
<http://www.publicdomainpictures.net/view-image.php?image=33211&picture=lady-bug-clip-art> (Public Domain)
https://commons.wikimedia.org/wiki/File:A_view_of_the_yellow_repository_at_The_National_Archives.jpg (Creative Commons)
<https://pixabay.com/en/database-data-storage-information-309919/> (Creative Commons)
<https://www.flickr.com/photos/sworking/8611316142> (Creative Commons)
<https://pixabay.com/en/business-suit-business-man-690048/> (Creative Commons/Public Domain)
https://commons.wikimedia.org/wiki/File:Flickr_-_moses_namkung_-_The_Crowd_For_DMB_1.jpg (Creative Commons 2.0)
https://commons.wikimedia.org/wiki/File:Huge_collection_of_tools_in_a_store_in_Chloride,_a_ghost_town_in_New_Mexico,_USA_-_July_2013.jpg (Creative Commons 2.0)
<https://commons.wikimedia.org/wiki/File:Tools.svg> (Creative Commons)
https://commons.wikimedia.org/wiki/File:User_icon_2.svg (Public Domain)
https://en.wikipedia.org/wiki/File:User_icon_1.svg (Public Domain)
http://i.ytimg.com/vi/z0DP_nhc0tU/hqdefault.jpg (Citation)
<https://www.pexels.com/photo/numbers-time-watch-white-1778/> (Creative Commons)
https://commons.wikimedia.org/wiki/File:WH_Situation_Room_-_many_conversations.jpg (Public Domain)
[https://en.wikipedia.org/wiki/Galaxy@BartSzulc/media/File:NGC_4414_\(NASA-med\).jpg](https://en.wikipedia.org/wiki/Galaxy@BartSzulc/media/File:NGC_4414_(NASA-med).jpg) (Public Domain)
<https://www.flickr.com/photos/torley/14999534034> (Creative Commons)
https://commons.wikimedia.org/wiki/File:US_Navy_050510-N-4309A-114_Hull_Maintenance_Technician_2nd_Class_Carl_Harris_inspects_the_remaining_high_explosive_material_from_a_disrupted_improvised_explosive_device_during_a_training_exercise_in_Bahrain.jpg (Public Domain)
<https://www.flickr.com/photos/amuderrick/46246205> (Creative Commons)
[https://commons.wikimedia.org/wiki/File:Dominator_\(Kings_Dominion\)_13_Descente_finale_\(crop\).jpg](https://commons.wikimedia.org/wiki/File:Dominator_(Kings_Dominion)_13_Descente_finale_(crop).jpg) (Creative Commons)
<https://www.flickr.com/photos/kgrocki/4098972642> (Creative Commons)
https://en.wikipedia.org/wiki/Gda%C5%84sk@BartSzulc/media/File:Calle_Dlugie_Pobrzeze_Gdansk,_Polonia,_2013-05-20,_DD_06.jpg (Creative Commons)
<https://pixabay.com/en/columns-greek-roman-three-ancient-295498/> (Creative Commons)