Université Concordia University

Intelligent System
Logging and Monitoring
ISyLM Research Lab

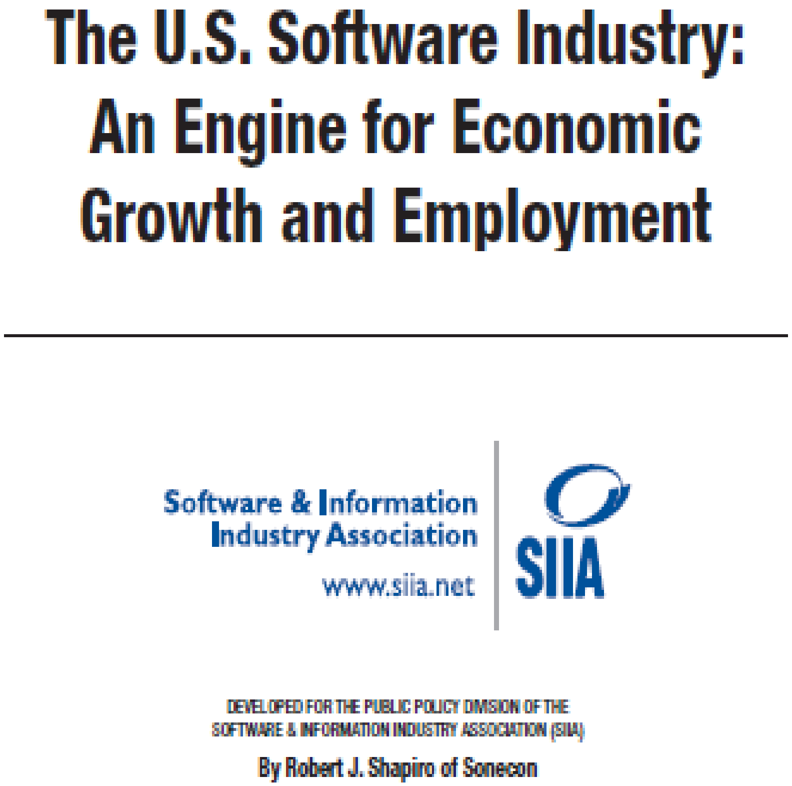
Software Maintenance At Commit-Time

Mathieu Louis Nayrolles
ECE, Concordia University
mathieu.nayrolles@concordia.ca

Concordia, Montréal, QC
June 26, 2018

Motivations

- From 1997 to 2012, software industry production grew from $149 billion to $425 billion.
- The software industry's direct share of U.S. GDP went from 1.7% to 2.6%.
- Software accounted for 12.1% percent of all U.S. labor productivity gains from 1995 to 2004 and 15.4% from 2004 to 2012.



The U.S. Software Industry: An Engine for Economic Growth and Employment

Software & Information Industry Association

www.siia.net

SIIA

DEVELOPED FOR THE PUBLIC POLICY DIVISION OF THE SOFTWARE & INFORMATION INDUSTRY ASSOCIATION (SIIA)

By Robert J. Shapiro of Sonecon

Motivations

- Maintenance of Software Systems can reach up **to 70% of the overall cost.**
- Up to **50%** of the overall maintenance cost can be spent **on identifying and correcting defects.**
- Defects in software cost the U.S. economy **$56 billion annually**.

NIST

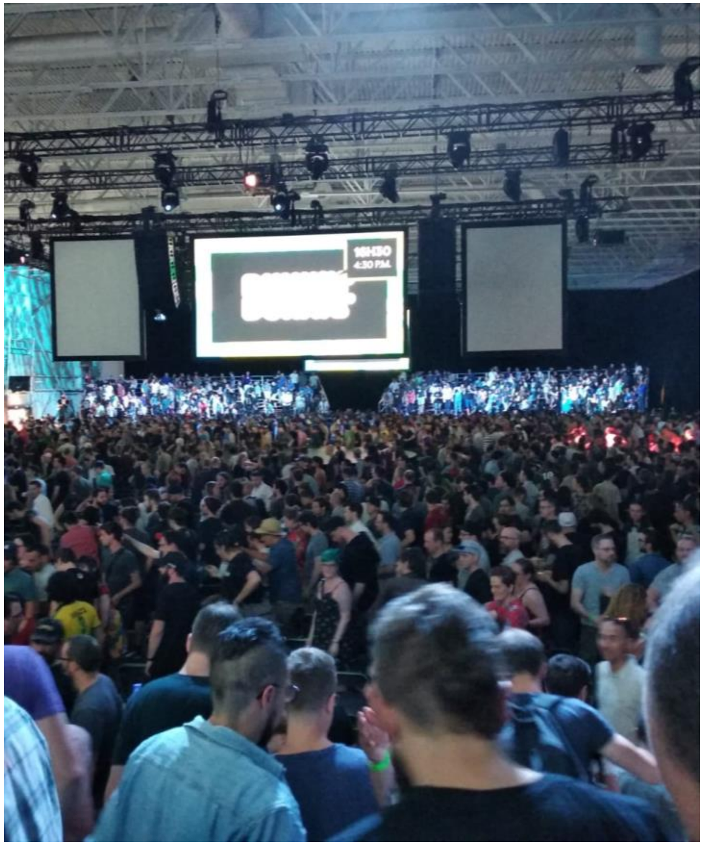
U.S Department of Commerce

Technology Administration

Source: Health, Social and Research, E. 2002. The Economic Impacts of Inadequate Infrastructure for Software Testing

Motivations

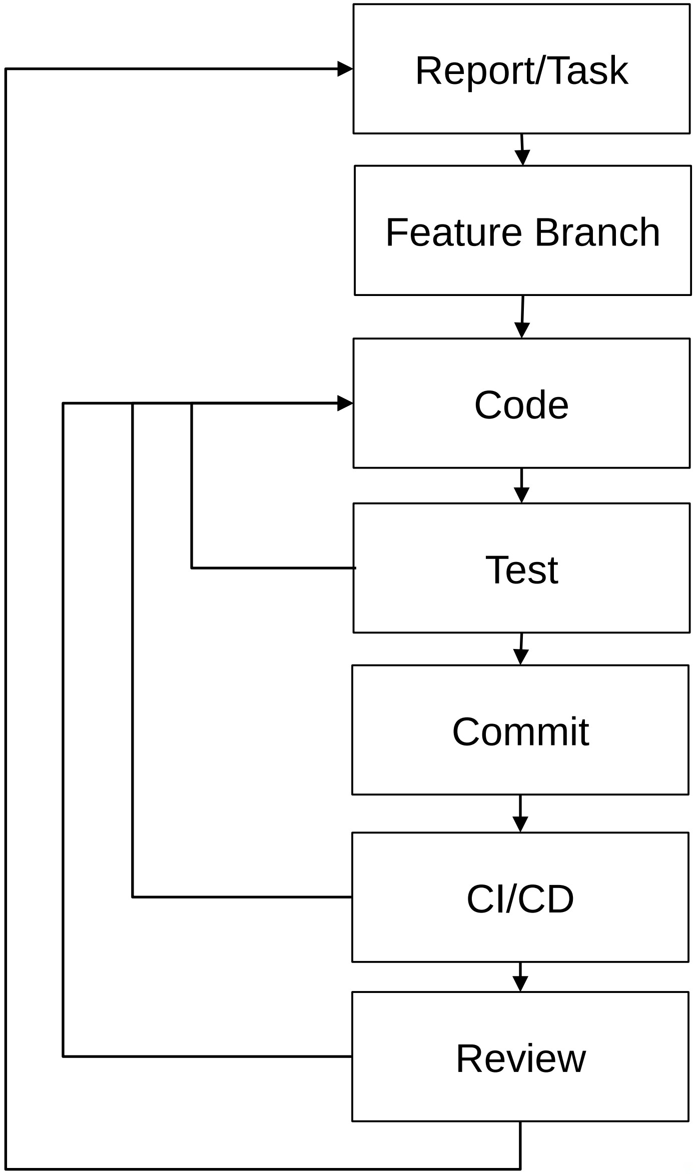
- Offline.
- Little integration with dev.
- No or unclear call to actions.
- Extensive setup.
- False positives.



Source: Ubisoft Montreal Employees during the annual assembly

Classical Workflow & Challenges

- Increased complexity.
- High cost.
- Heavy reliance on people.
- Lack of automated tools.
- Time to market pressure.
- Maintaining quality.



Goal

To empower software developers with intelligent tools that detects defects as they write code, help reproduce on-field crashes and propose fixes without altering their workflows.

UNIVERSITÉ Concordia UNIVERSITY

ERICSSON

UBISOFT MONTREAL

NSERC CRSNG

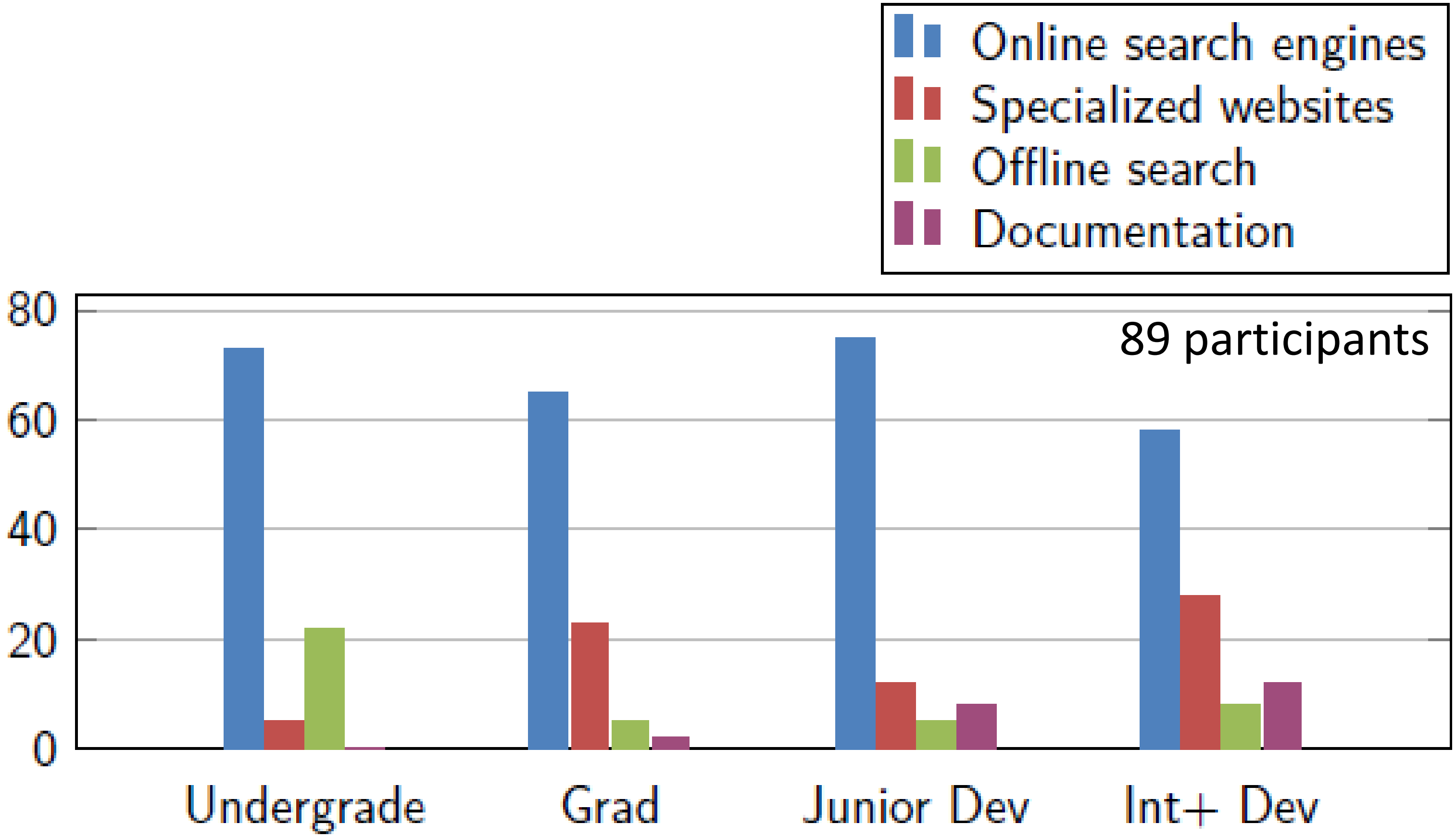
Mitacs

Results

Contributions

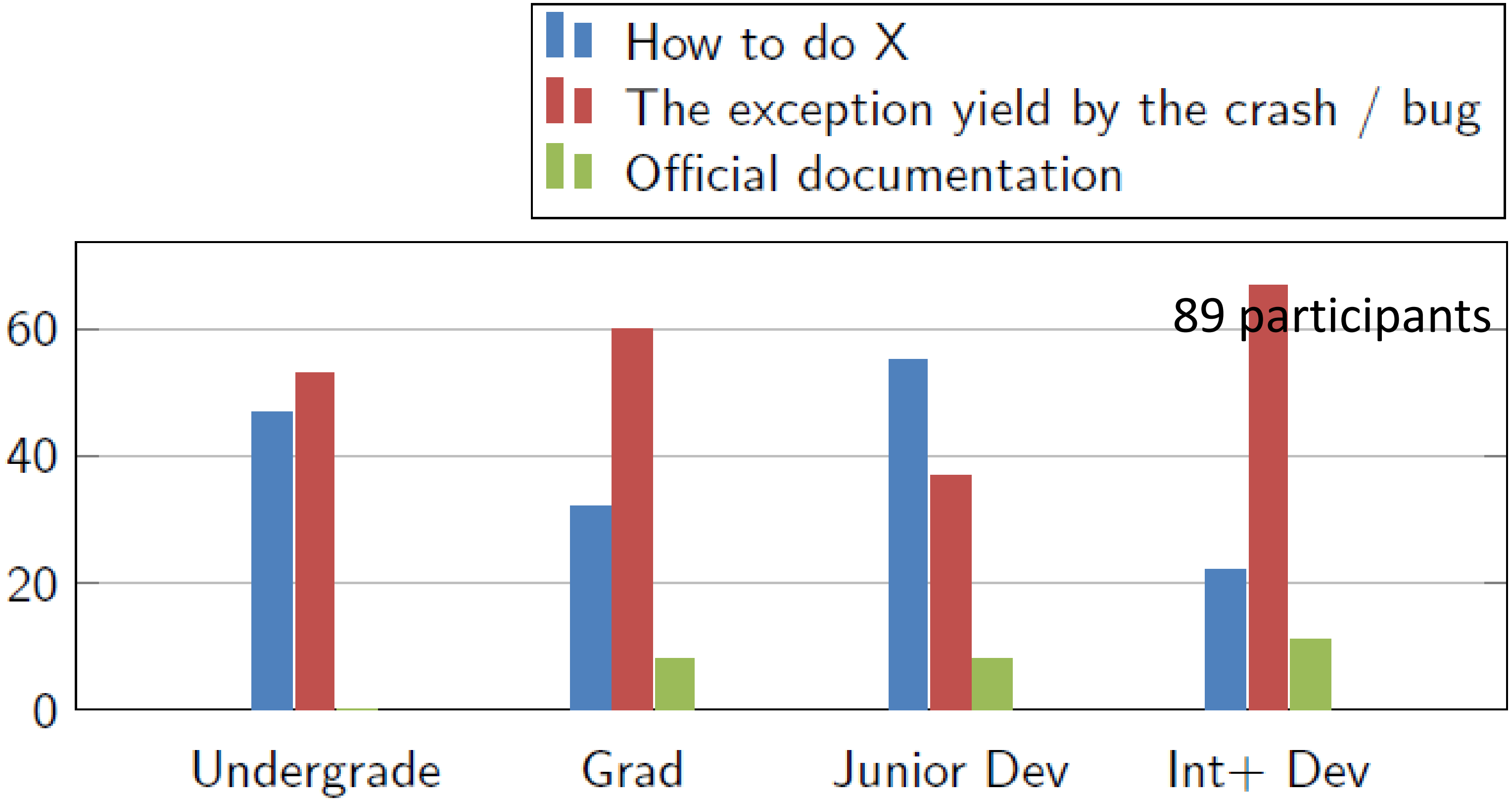
- BUMPER: A Tool for Coping with Natural Language Searches of Millions of Bugs and Fixes.
- PRECINCT: An Approach for Preventing Clone Insertion at Commit-Time.
- BIANCA: Preventing Bug Insertion at Commit-Time Using Dependency Analysis and Clone Detection.
- CLEVER: Combining Code Metrics with Clone Detection for Just-In-Time Fault Prevention and Resolution in Large Industrial Projects.
- JCHARMING: A bug reproduction approach using crash traces and directed model checking.
- Towards a Classification of Bugs to Facilitate Software Maintainability Tasks.

Where do developers look for information when facing an unknown bug/crash?



89 participants

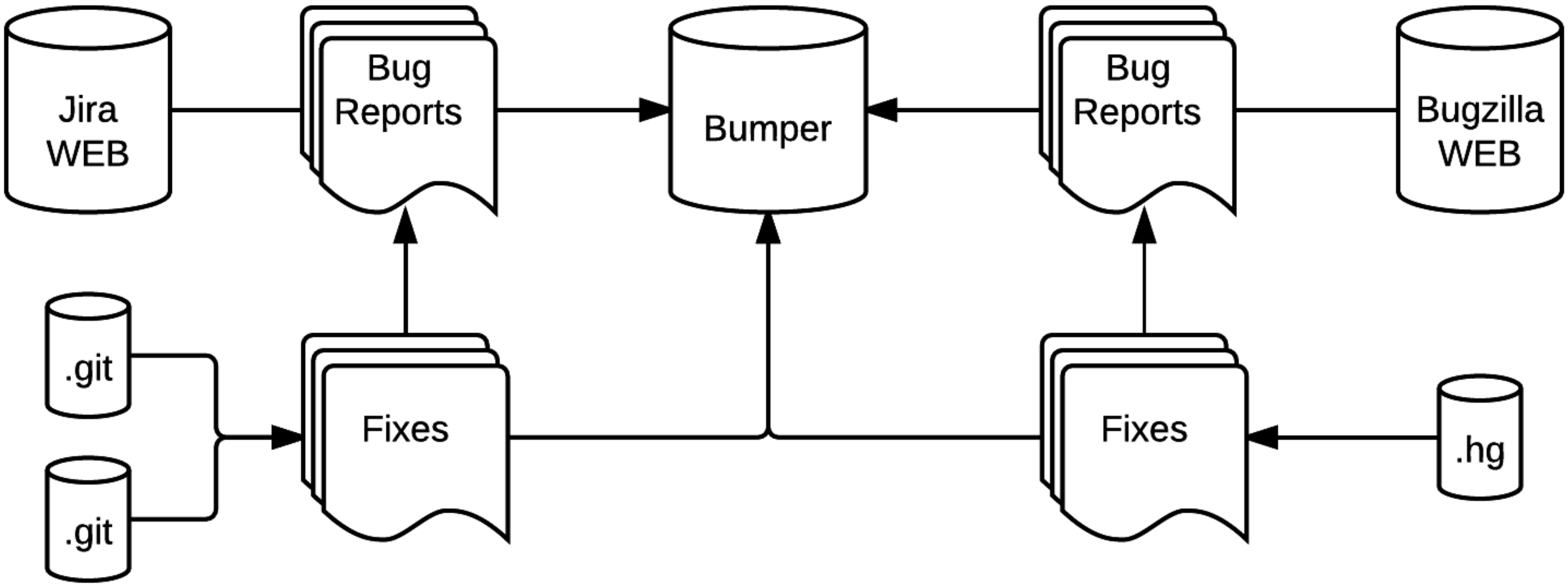
What do developers search for when facing an unknown bug/crash?



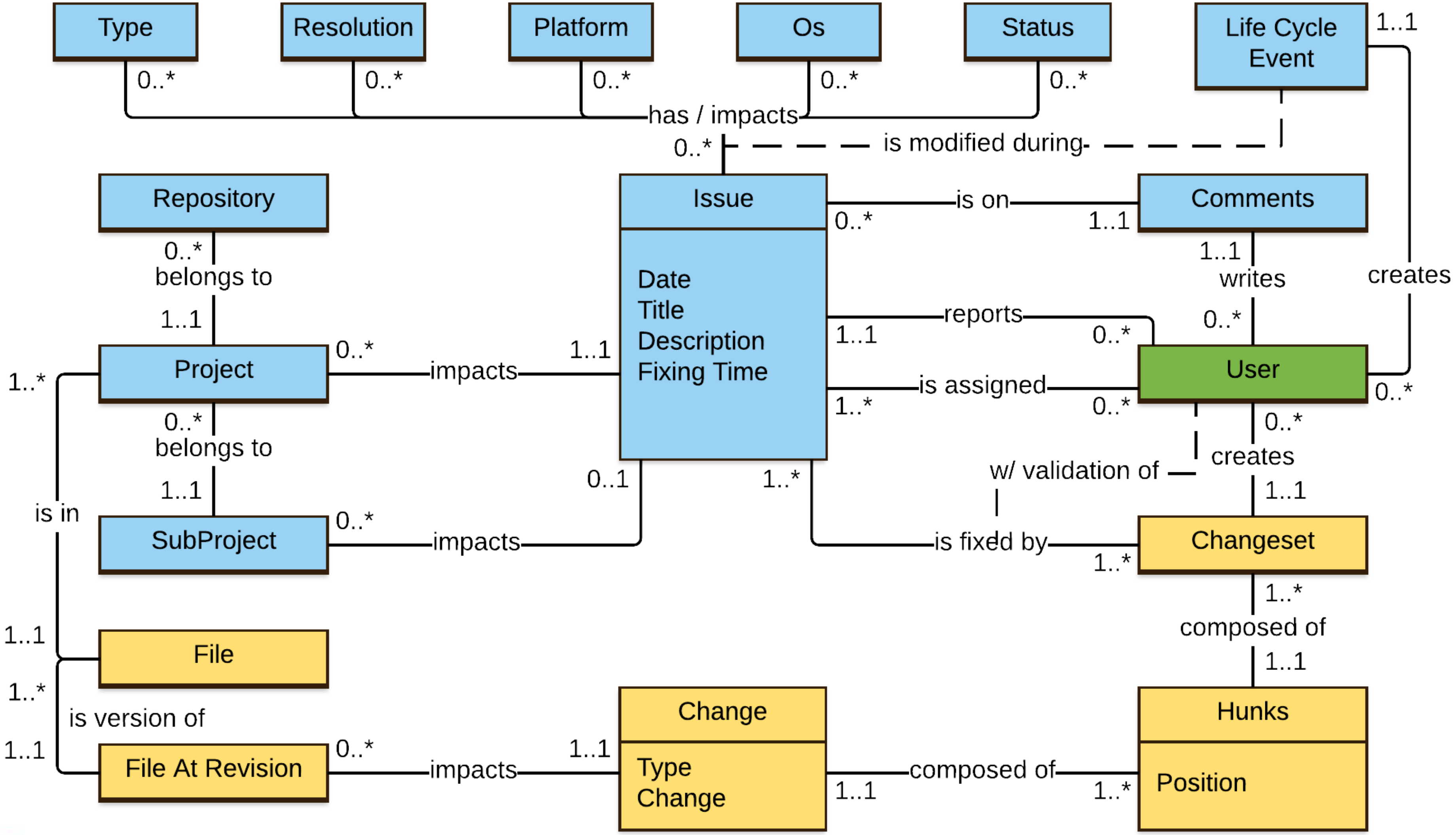
89 participants

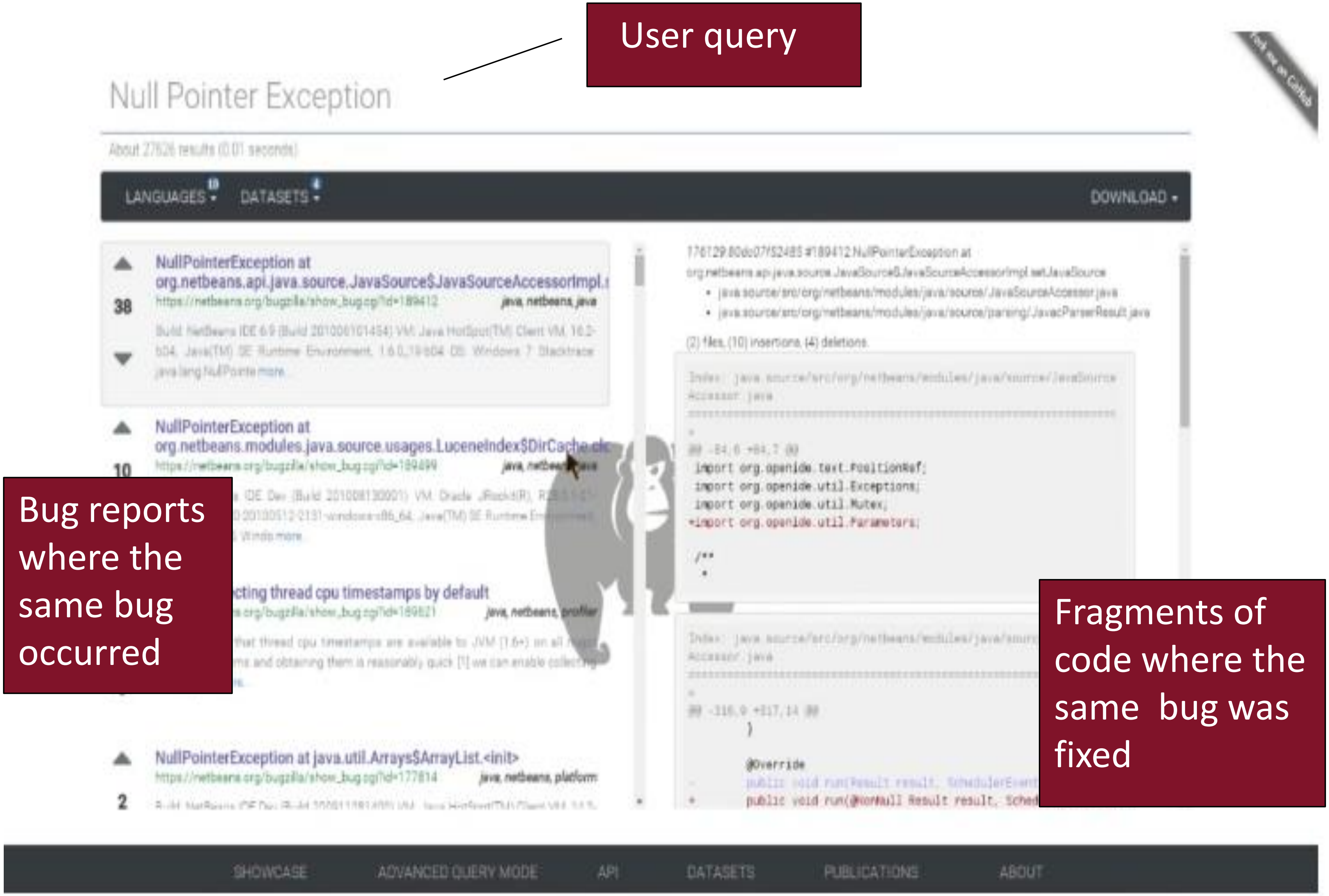
BUMPER

- Aggregate millions of bugs / fixes.



BUMPER


Type
Resolution
Platform
Os
Status
Life Cycle Event
0..*
0..*
0..*
0..*
0..*
1..1
has / impacts
0..*
is modified during
Repository
Issue
Comments
0..*
belongs to
1..1
Date
Title
Description
Fixing Time
is on
0..*
1..1
1..1
writes
0..*
creates
Project
1..*
0..*
impacts
1..1
reports
1..1
0..*
User
0..*
is assigned
1..*
0..*
0..*
belongs to
1..1
creates
1..1
w/ validation of
0..1
1..*
is in
SubProject
0..*
impacts
is fixed by
1..*
Changeset
1..*
composed of
1..1
1..1
File
1..*
is version of
1..1
File At Revision
0..*
impacts
1..1
Change
Type
Change
composed of
1..1
1..*
Hunks
Position



Null Pointer Exception

About 27626 results (0.01 seconds)

38
NullPointerException at org.netbeans.api.java.source.JavaSource$JavaSourceAccessorImpl.s
https://netbeans.org/bugzilla/show_bug.cgi?id=189412 java, netbeans, java
Build: NetBeans IDE 6.9 (Build 201006101454) VM: Java HotSpot(TM) Client VM, 16.2-b04, Java(TM) SE Runtime Environment, 1.6.0_19-b04 OS: Windows 7 Stacktrace: java.lang.NullPointe more

10
NullPointerException at org.netbeans.modules.java.source.usages.LuceneIndex$DirCache.clc
https://netbeans.org/bugzilla/show_bug.cgi?id=189499 java, netbeans, java
IDE Dev (Build 201008130001) VM: Oracle JRockit(R), R28.0.1 ... 20100512-2131-windows-x86_64, Java(TM) SE Runtime Environment ... Windo more

...cting thread cpu timestamps by default
...s.org/bugzilla/show_bug.cgi?id=189521 java, netbeans, profiler
... that thread cpu timestamps are available to JVM (1.6+) on all major ... ms and obtaining them is reasonably quick [1] we can enable collecting ...

2
NullPointerException at java.util.Arrays$ArrayList.<init>
https://netbeans.org/bugzilla/show_bug.cgi?id=177814 java, netbeans, platform

176129.80dc07f52485 #189412 NullPointerException at org.netbeans.api.java.source.JavaSource$JavaSourceAccessorImpl.setJavaSource

- java.source/src/org/netbeans/modules/java/source/JavaSourceAccessor.java
- java.source/src/org/netbeans/modules/java/source/parsing/JavacParserResult.java

(2) files, (10) insertions, (4) deletions.

```
Index: java.source/src/org/netbeans/modules/java/source/JavaSource
Accessor.java
===================================================================
=
@@ -84,6 +84,7 @@
 import org.openide.text.PositionRef;
 import org.openide.util.Exceptions;
 import org.openide.util.Mutex;
+import org.openide.util.Parameters;

 /**
  *
```

```
Index: java.source/src/org/netbeans/modules/java/sourc
Accessor.java
===================================================
=
@@ -116,9 +117,14 @@
         }

         @Override
-        public void run(Result result, SchedulerEvent
+        public void run(@NonNull Result result, Sched
```

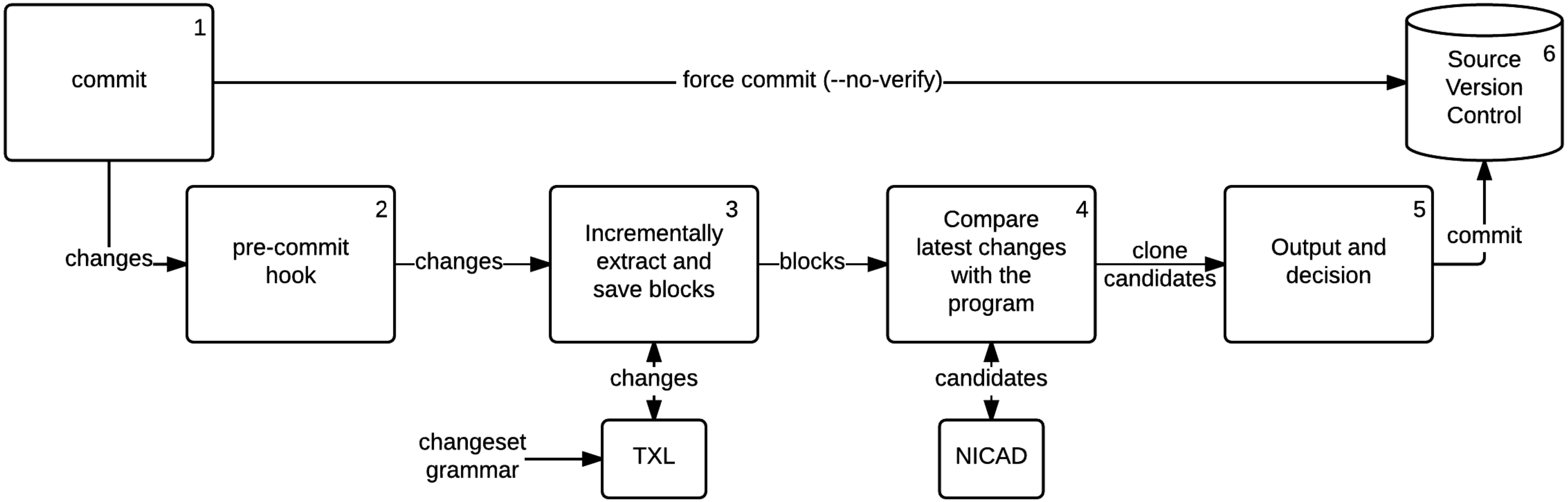
User query

Bug reports where the same bug occurred

Fragments of code where the same bug was fixed

PRECINCT

- Detect Near Miss Clones At Commit-Time



PRECINCT

```
+ using System;

+ public class Test
+ {
+ public static void Main()
+ {
+ Console.WriteLine("Hello Concordia; let's sort...");
+ int[] array = new int[]{1, 5, 6, 2, 3};
+ for(int i=0; i<array.Length; i++){
+    for(int j=0; j<array.Length - 1; j++){
+        if(array[j] > array[j+1]){
+            int tmp = array[j+1];
+            array[j+1] = array[j];
+            array[j] = tmp;
+        }
+    }
+ }
+
+ Console.WriteLine(string.Join(",", array));
+ }
+ }
```

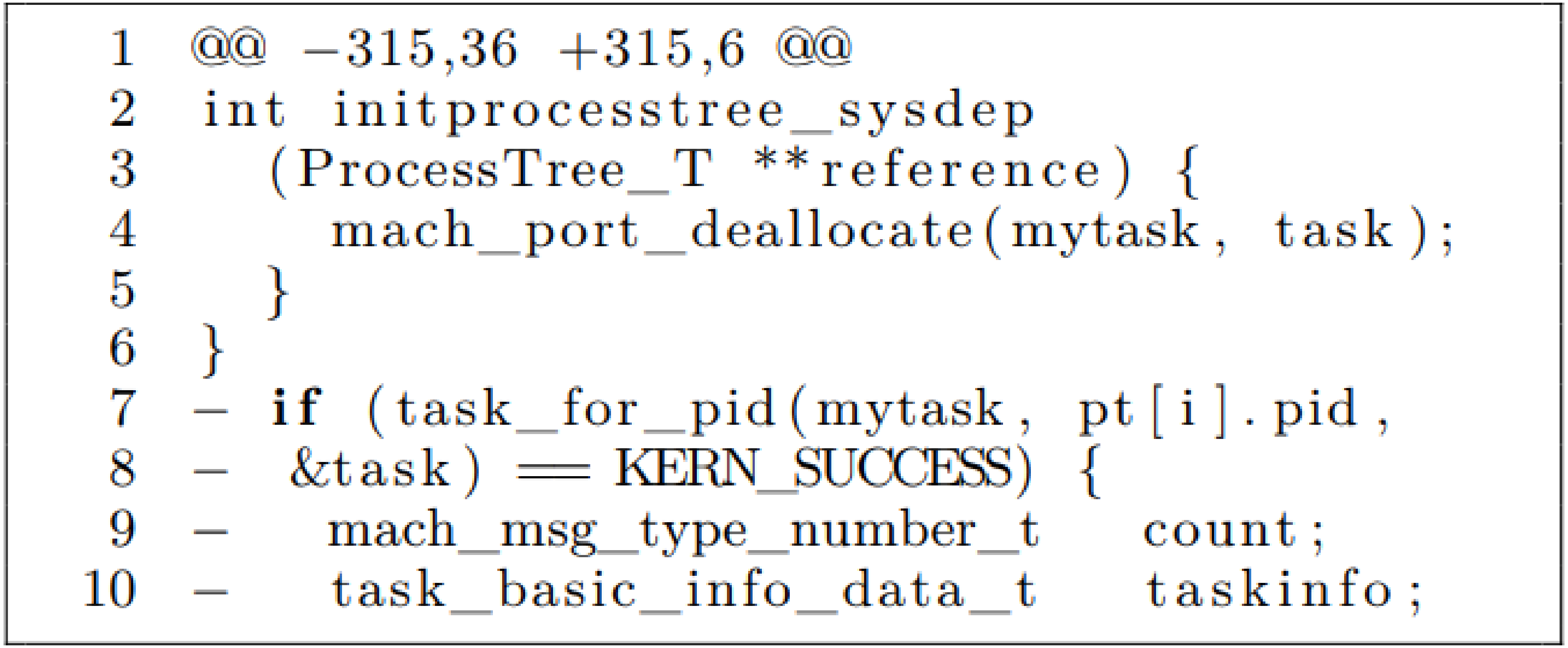
```
A[] = A[]
for(
B=0
B<A.Length
B++
){
for(
C=0
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
}
```

PRECINCT

```
for(
C=A.Length
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

```
for(
C=B
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

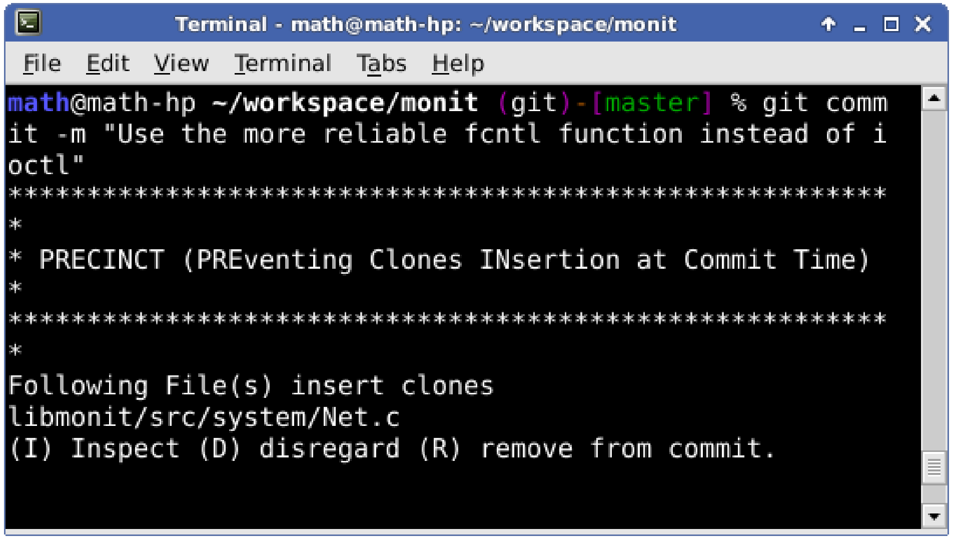
PRECINCT



```
@@ -315,36 +315,6 @@
 int initprocesstree_sysdep
    (ProcessTree_T **reference) {
       mach_port_deallocate(mytask, task);
    }
 }
- if (task_for_pid(mytask, pt[i].pid,
-  &task) == KERN_SUCCESS) {
-     mach_msg_type_number_t   count;
-     task_basic_info_data_t   taskinfo;
```

PRECINCT

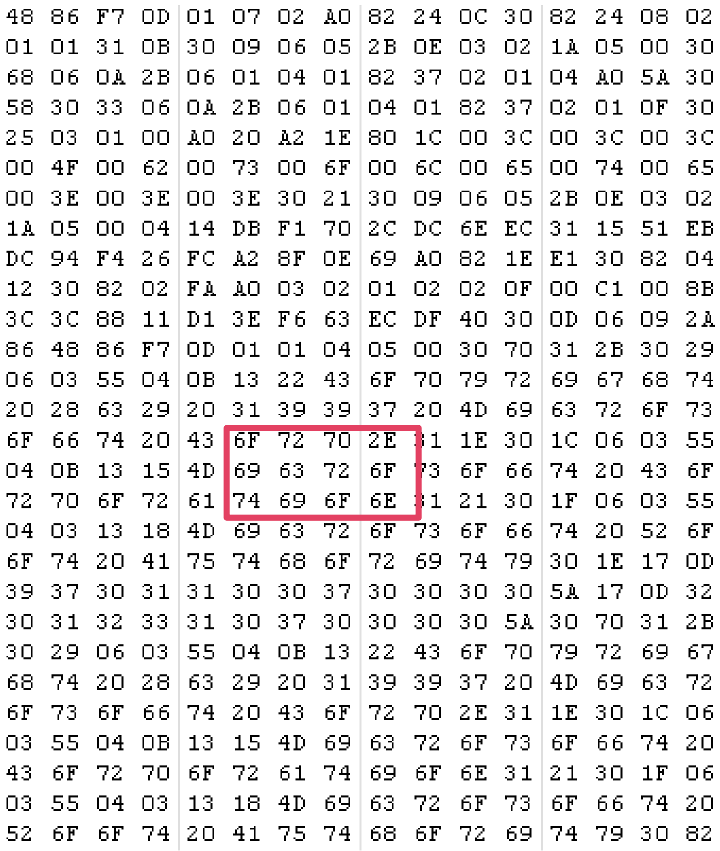
- Tested on 3 OSS
- 97.7% Precision
- 100% Recall
- 98.8% F1-Measure
- 6839 Clones
- 2.5x faster
- Workflow Compliant



```
math@math-hp ~/workspace/monit (git)-[master] % git comm
it -m "Use the more reliable fcntl function instead of i
octl"
********************************************************
*
* PRECINCT (PREventing Clones INsertion at Commit Time)
*
********************************************************
*
Following File(s) insert clones
libmonit/src/system/Net.c
(I) Inspect (D) disregard (R) remove from commit.
```

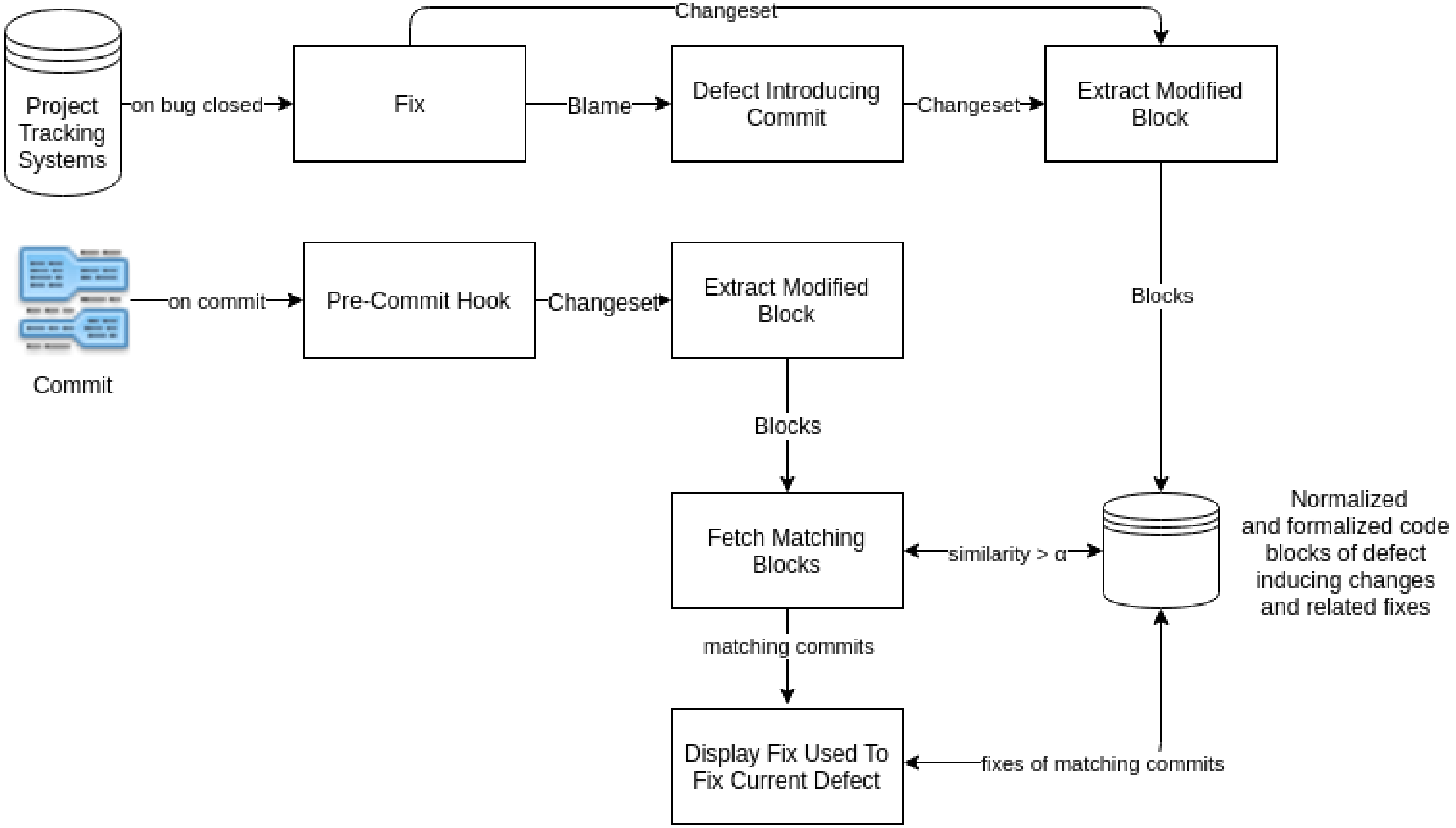
BIANCA: Preventing Bug Insertion at Commit-Time Using Clone Detection

- BIANCA learns known defects by mining BUMPER-indexed systems.
- It intercepts developer’s code and compares it to signatures of known defects.
- If a match exists, a flag is raised and a fix is proposed.



```
48 86 F7 0D | 01 07 02 A0 | 82 24 0C 30 | 82 24 08 02
01 01 31 0B | 30 09 06 05 | 2B 0E 03 02 | 1A 05 00 30
68 06 0A 2B | 06 01 04 01 | 82 37 02 01 | 04 A0 5A 30
58 30 33 06 | 0A 2B 06 01 | 04 01 82 37 | 02 01 0F 30
25 03 01 00 | A0 20 A2 1E | 80 1C 00 3C | 00 3C 00 3C
00 4F 00 62 | 00 73 00 6F | 00 6C 00 65 | 00 74 00 65
00 3E 00 3E | 00 3E 30 21 | 30 09 06 05 | 2B 0E 03 02
1A 05 00 04 | 14 DB F1 70 | 2C DC 6E EC | 31 15 51 EB
DC 94 F4 26 | FC A2 8F 0E | 69 A0 82 1E | E1 30 82 04
12 30 82 02 | FA A0 03 02 | 01 02 02 0F | 00 C1 00 8B
3C 3C 88 11 | D1 3E F6 63 | EC DF 40 30 | 0D 06 09 2A
86 48 86 F7 | 0D 01 01 04 | 05 00 30 70 | 31 2B 30 29
06 03 55 04 | 0B 13 22 43 | 6F 70 79 72 | 69 67 68 74
20 28 63 29 | 20 31 39 39 | 37 20 4D 69 | 63 72 6F 73
6F 66 74 20 | 43 6F 72 70 | 2E 31 1E 30 | 1C 06 03 55
04 0B 13 15 | 4D 69 63 72 | 6F 73 6F 66 | 74 20 43 6F
72 70 6F 72 | 61 74 69 6F | 6E 31 21 30 | 1F 06 03 55
04 03 13 18 | 4D 69 63 72 | 6F 73 6F 66 | 74 20 52 6F
6F 74 20 41 | 75 74 68 6F | 72 69 74 79 | 30 1E 17 0D
39 37 30 31 | 31 30 30 37 | 30 30 30 30 | 5A 17 0D 32
30 31 32 33 | 31 30 37 30 | 30 30 30 5A | 30 70 31 2B
30 29 06 03 | 55 04 0B 13 | 22 43 6F 70 | 79 72 69 67
68 74 20 28 | 63 29 20 31 | 39 39 37 20 | 4D 69 63 72
6F 73 6F 66 | 74 20 43 6F | 72 70 2E 31 | 1E 30 1C 06
03 55 04 0B | 13 15 4D 69 | 63 72 6F 73 | 6F 66 74 20
43 6F 72 70 | 6F 72 61 74 | 69 6F 6E 31 | 21 30 1F 06
03 55 04 03 | 13 18 4D 69 | 63 72 6F 73 | 6F 66 74 20
52 6F 6F 74 | 20 41 75 74 | 68 6F 72 69 | 74 79 30 82
```

BIANCA


Project Tracking Systems

on bug closed

Fix

Blame

Defect Introducing Commit

Changeset

Extract Modified Block

Changeset

Blocks

Commit

on commit

Pre-Commit Hook

Changeset

Extract Modified Block

Blocks

Fetch Matching Blocks

similarity > α

Normalized and formalized code blocks of defect inducing changes and related fixes

matching commits

Display Fix Used To Fix Current Defect

fixes of matching commits

BIANCA

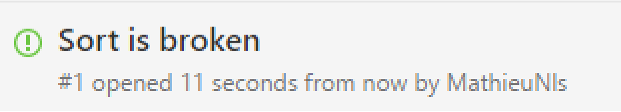
```
+ using System;

+ public class Test
+ {
+ public static void Main()
+ {
+ Console.WriteLine("Hello Concordia; let's sort...");
+ int[] array = new int[]{1, 5, 6, 2, 3};
+ for(int i=0; i<array.Length; i++){
+    for(int j=0; j<array.Length - 1; j++){
+        if(array[j] > array[j+1]){
+            int tmp = array[j+1];
+            array[j+1] = array[j];
+            array[j] = tmp;
+        }
+    }
+ }
+
+ Console.WriteLine(string.Join(",", array));
+ }
+ }
```

```
A[] = A[]
for(
B=0
B<A.Length
B++
){
for(
C=0
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
}
```

BIANCA

```
public class Test
{
public static void Main()
{
Console.WriteLine("Hello Concordia; let's sort...");
int[] array = new int[]{1, 5, 6, 2, 3};
for(int i=0; i<array.Length; i++){
-    for(int j=array.Length; j<array.Length - 1; j++){
+                        for(int j=i; j<array.Length - 1; j++){
         if(array[j] > array[j+1]){
           int tmp = array[j+1];
           array[j+1] = array[j];
           array[j] = tmp;
       }
   }
}
Console.WriteLine(string.Join(",", array));
}
}
```



Sort is broken

#1 opened 11 seconds from now by MathieuNls

```
for(
C=A.Length
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

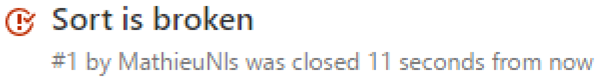
STABLE

```
for(
C=B
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

BUGGY

BIANCA

```
public class Test
{
public static void Main()
{
Console.WriteLine("Hello Ubi; let's sort...");
int[] array = new int[]{1, 5, 6, 2, 3};
for(int i=0; i<array.Length; i++){
-    for(int j=i; j<array.Length - 1; j++){
+    for(int j=array.Length; j<array.Length - 1;
j++){
        if(array[j] > array[j+1]){
          int tmp = array[j+1];
          array[j+1] = array[j];
          array[j] = tmp;
       }
   }
}
Console.WriteLine(string.Join(",", array));
}
}
```



Sort is broken

#1 by MathieuNls was closed 11 seconds from now

```
for(
C=0
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

BUGGY

```
for(
C=A.Length
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

STABLE

BIANCA

```
for(
C=0
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

STABLE

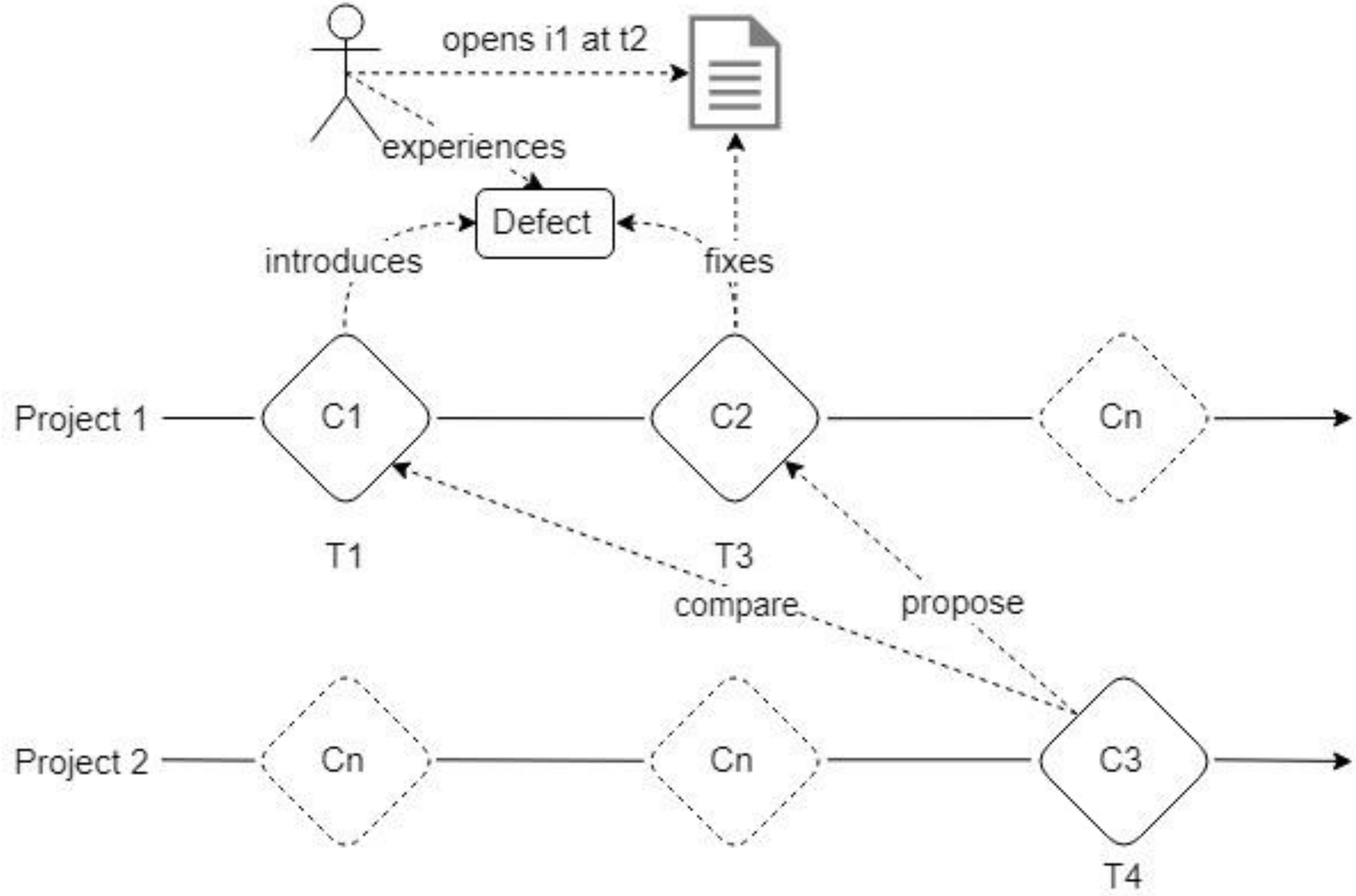
```
for(
C=A.Length
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

BUGGY

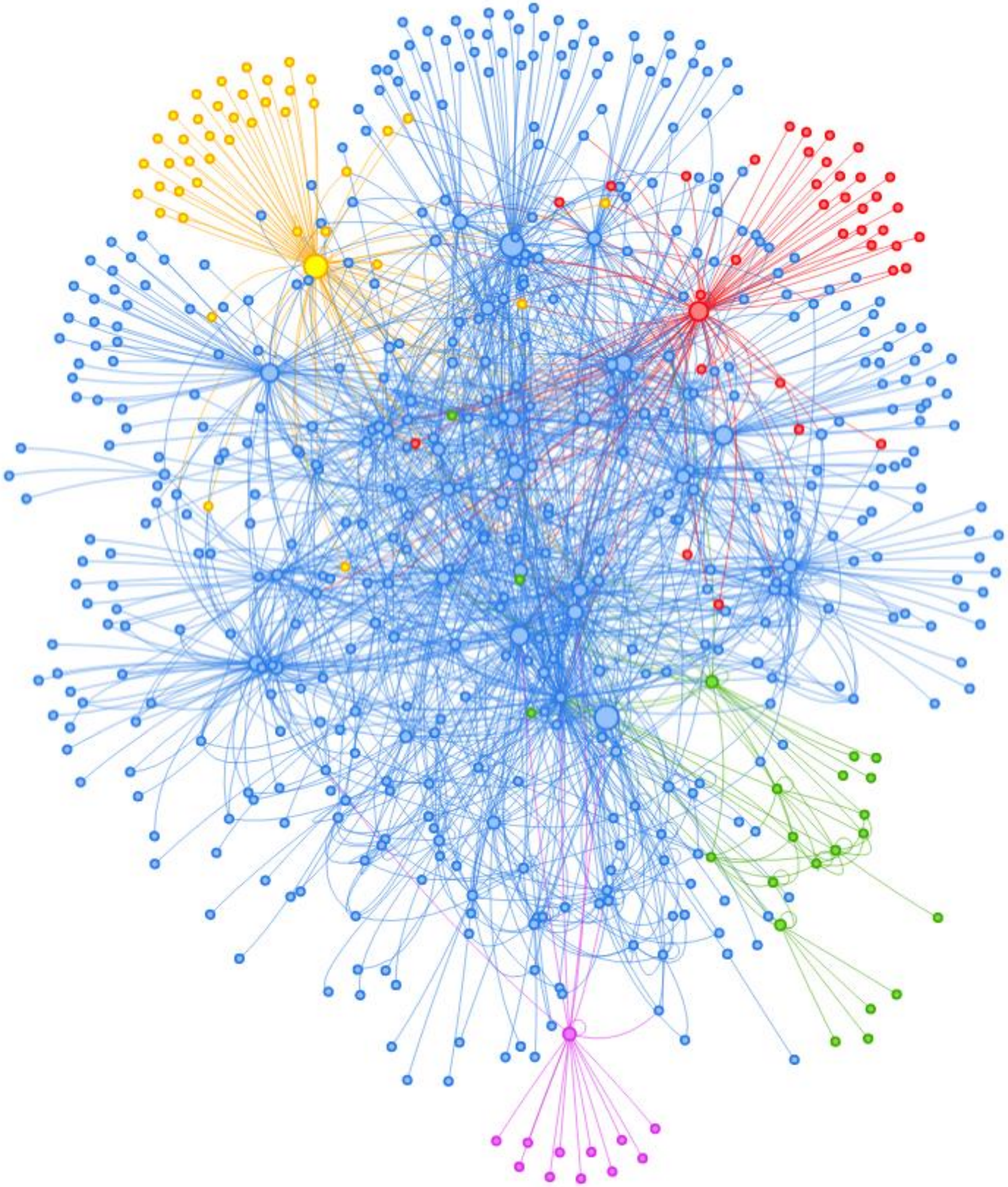
```
for(
C=B
C<A.Length
C++
){
if(
A[C] >
A[C+1]
){
D = A[C+1]
A[C+1] = A[C]
A[C] = D
}
}
```

STABLE

BIANCA

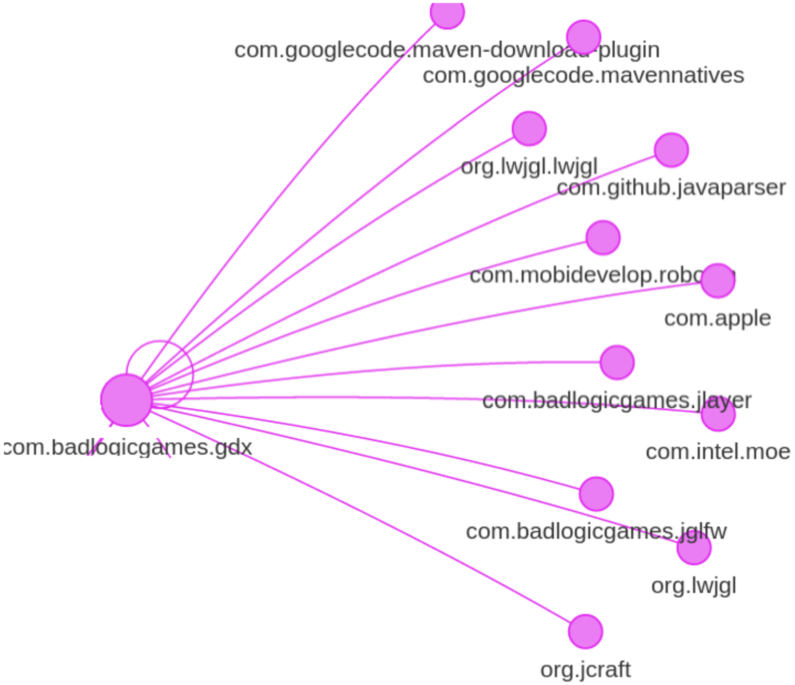

opens i1 at t2
experiences
Defect
introduces
fixes
Project 1
C1
C2
Cn
T1
T3
compare
propose
Project 2
Cn
Cn
C3
T4

BIANCA



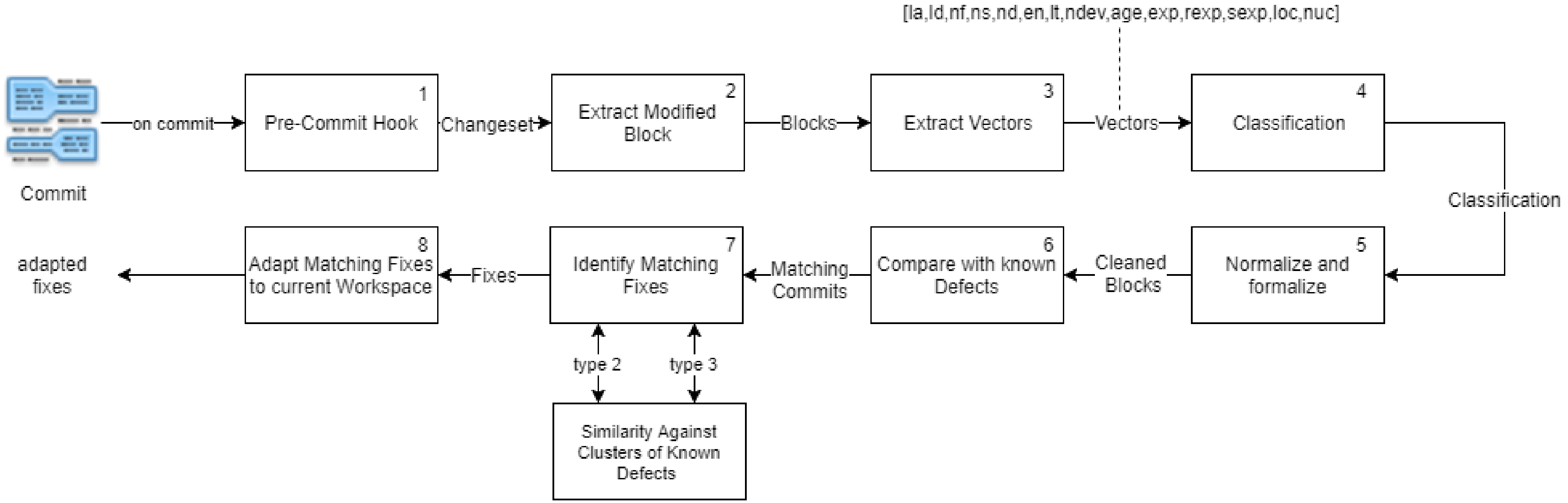
BIANCA

- Tested on 42 OSS
- 90.75% Precision
- 37.15% Recall
- 52.72% F1-Measure
- 41,225 defects
- 8.6% self-fixes
- 78% valid fix proposition
- Workflow Compliant

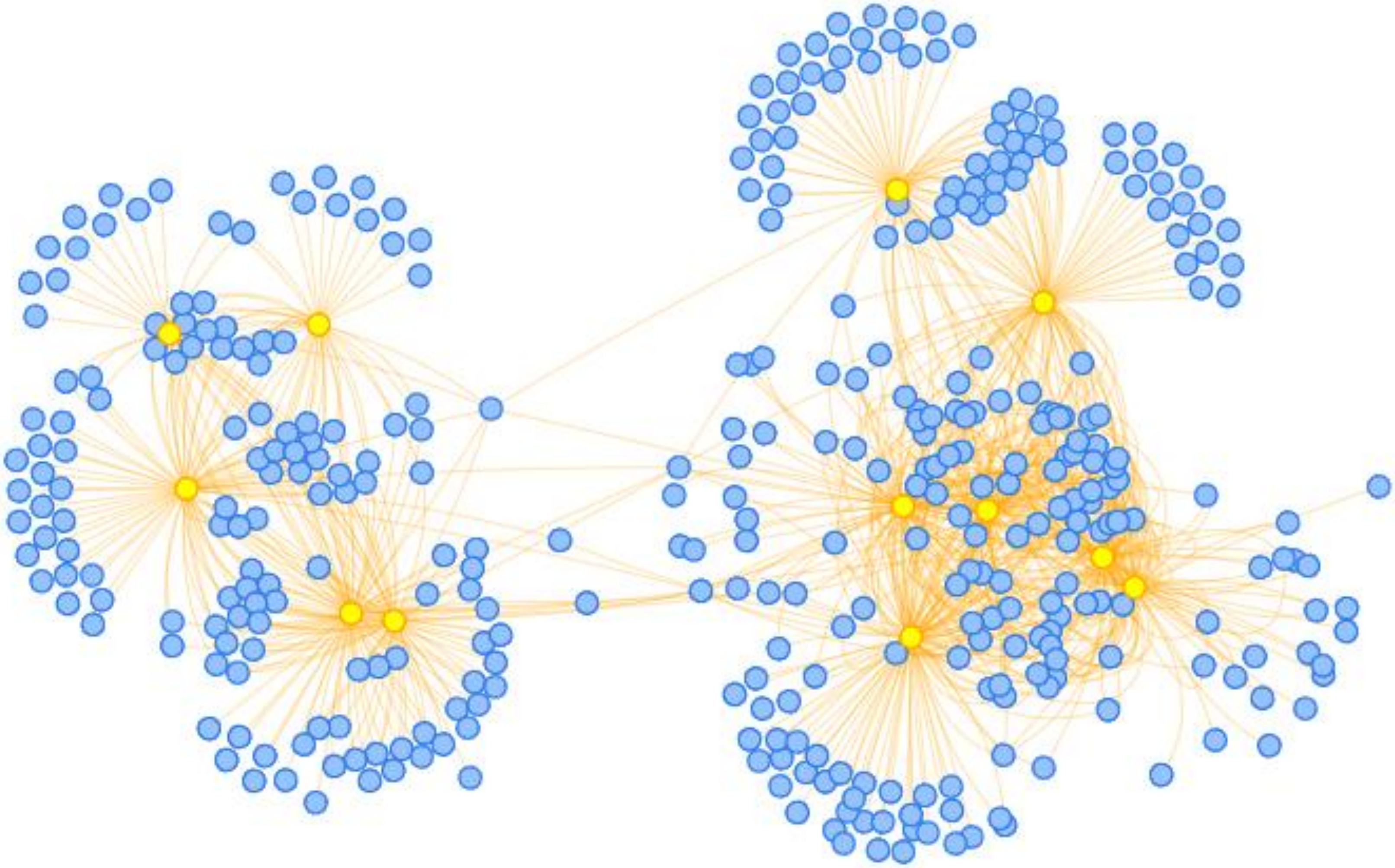


CLEVER

- Detect Bug Introduction At-Commit Time Using Clone Analysis And Code Metrics

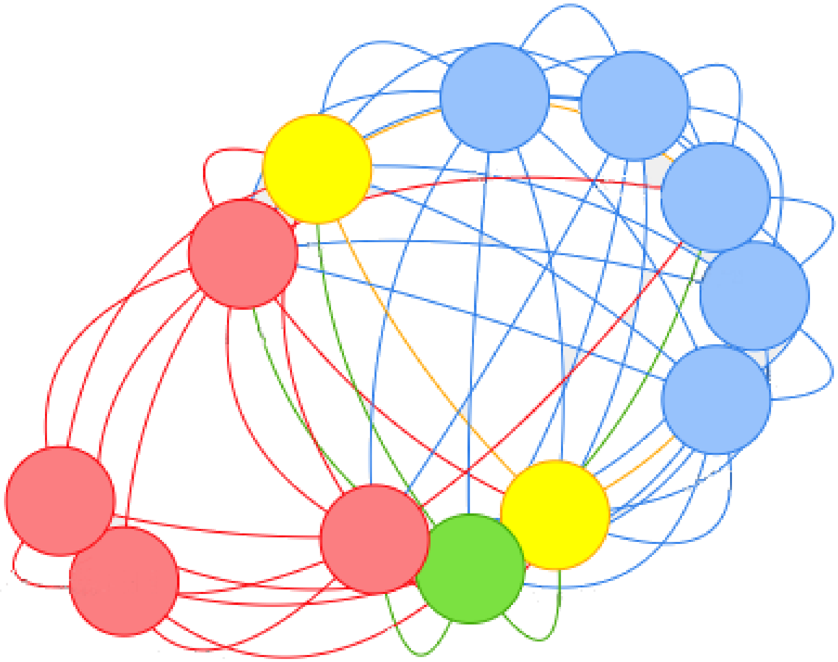


CLEVER



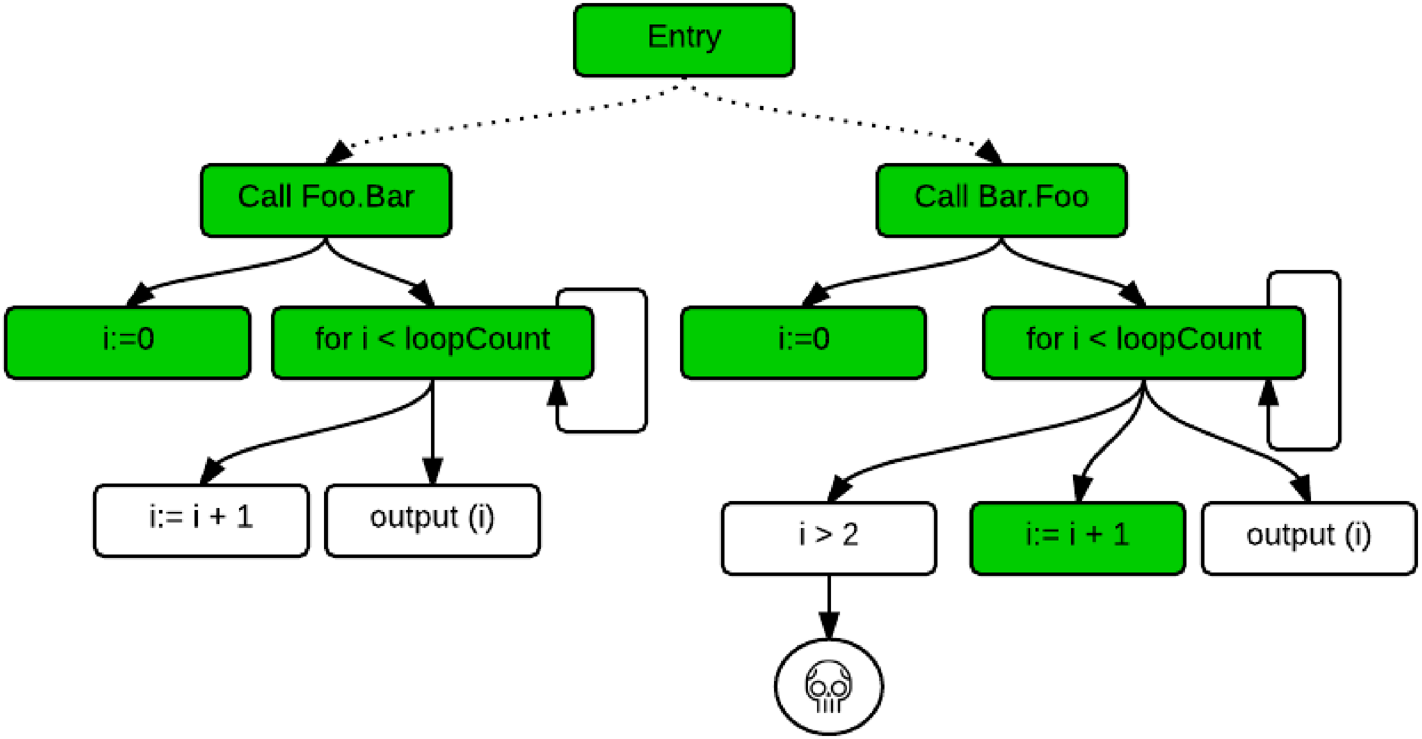
CLEVER

- Tested on 12 CSS
- 79% Precision
- 65% Recall
- 66% valid fix proposition
- Workflow Compliant
- ~2 seconds to ~25 seconds

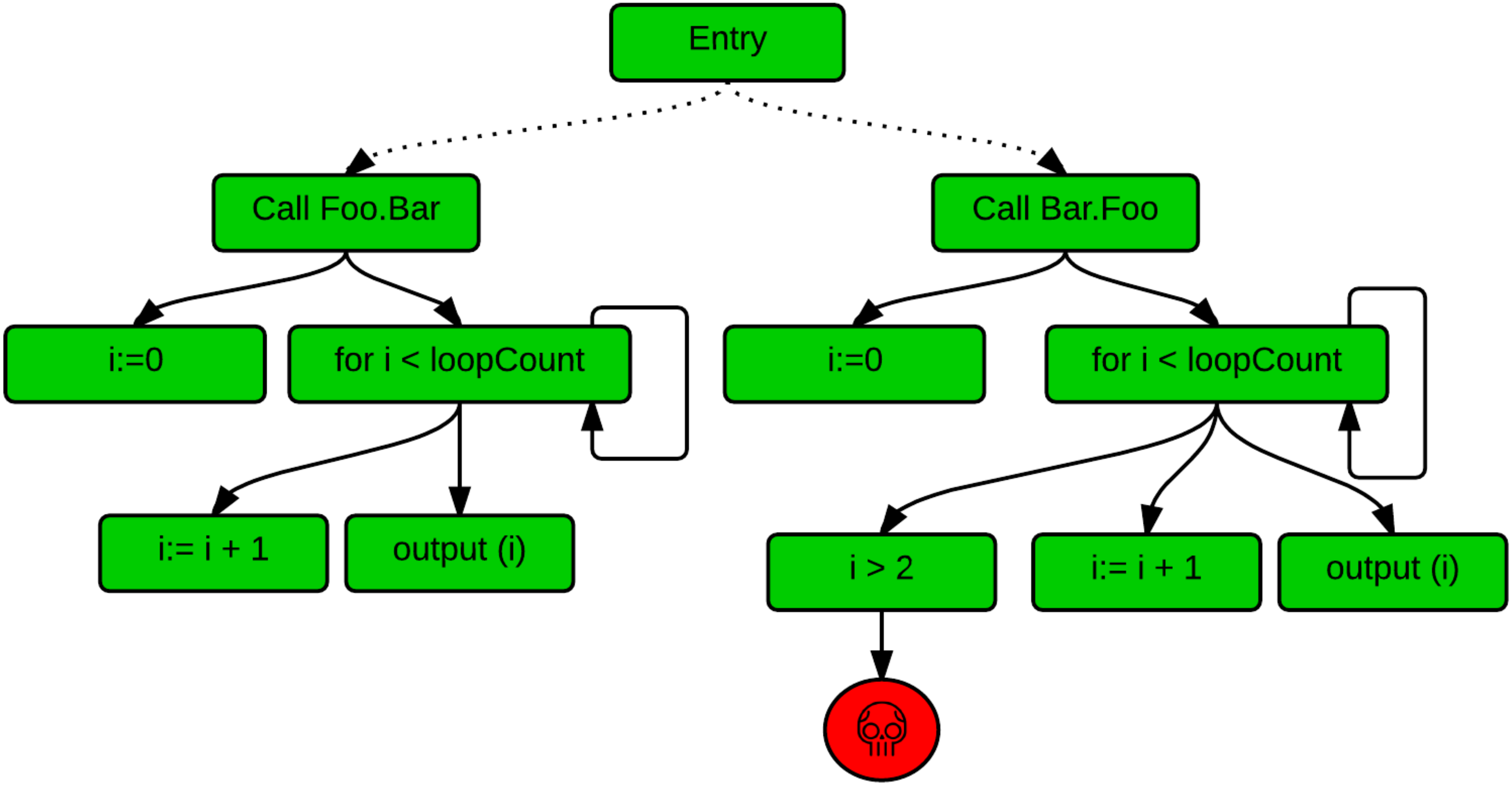


JCHARMING

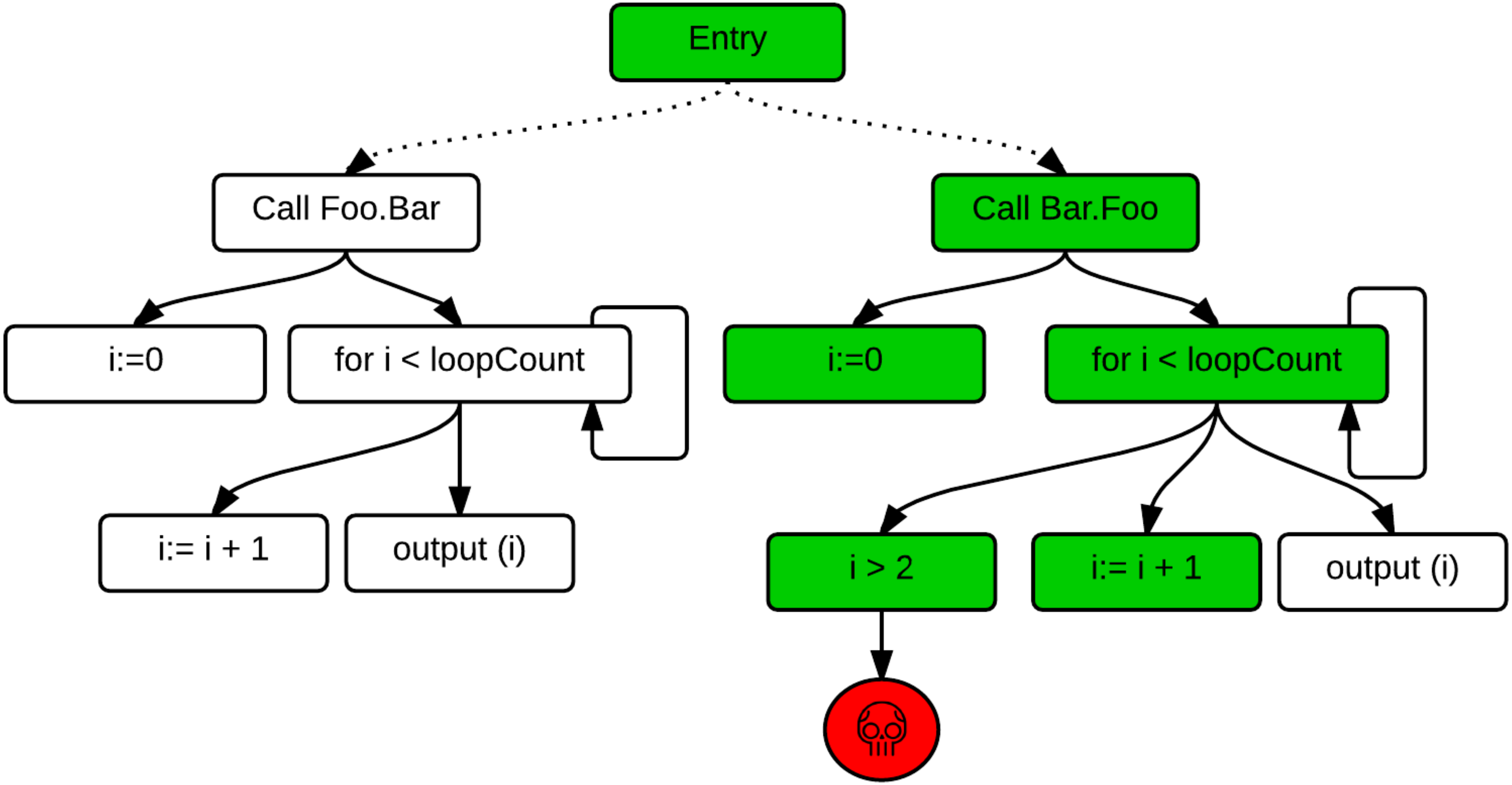
- Reproduce on-field crashes using stack traces and directed model checking



JCHARMING


Entry
Call Foo.Bar
Call Bar.Foo
i:=0
for i < loopCount
i:=0
for i < loopCount
i:= i + 1
output (i)
i > 2
i:= i + 1
output (i)

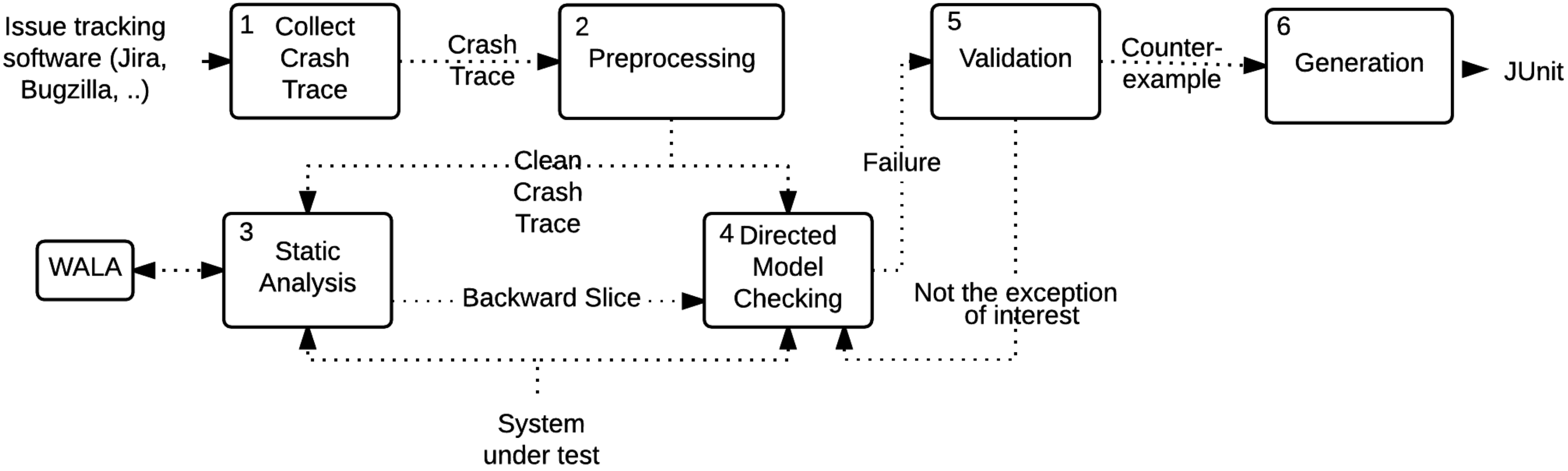
JCHARMING



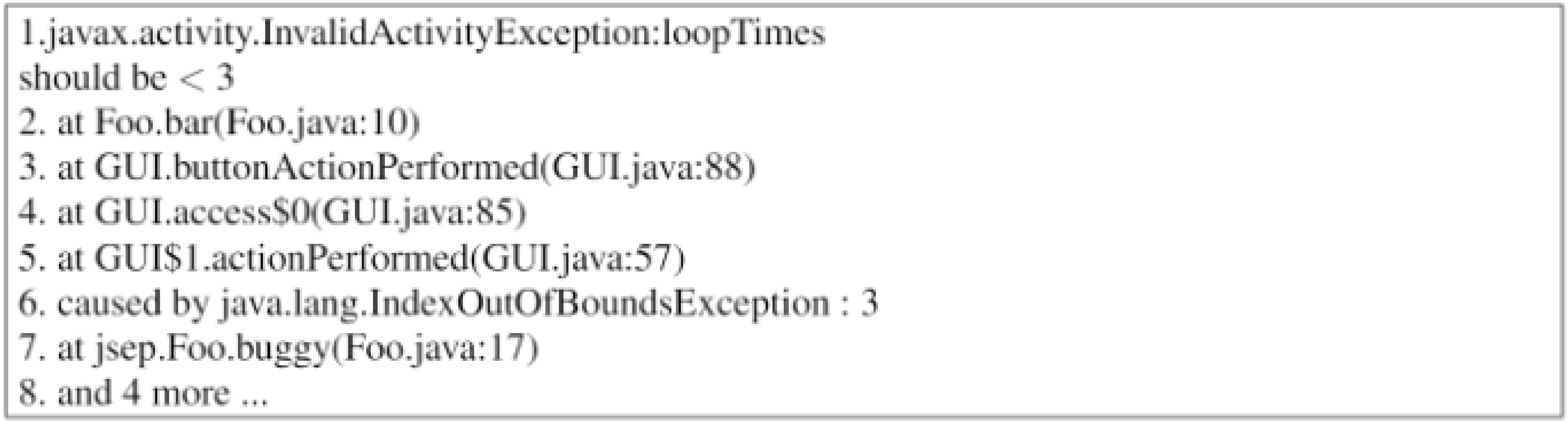
Entry
Call Foo.Bar
Call Bar.Foo
i:=0
for i < loopCount
i:=0
for i < loopCount
i:= i + 1
output (i)
i > 2
i:= i + 1
output (i)

JCHARMING

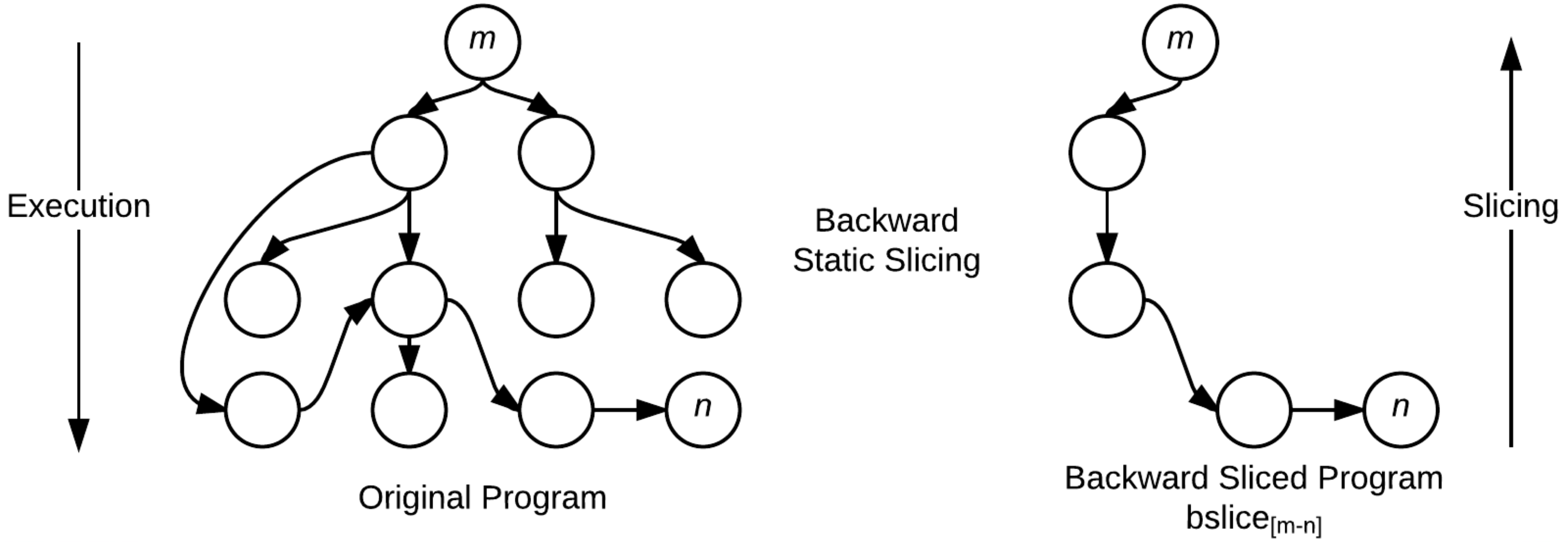
- Reproduce on-field crashes using stack traces and directed model checking



JCHARMING



```
1.javax.activity.InvalidActivityException:loopTimes
should be < 3
2. at Foo.bar(Foo.java:10)
3. at GUI.buttonActionPerformed(GUI.java:88)
4. at GUI.access$0(GUI.java:85)
5. at GUI$1.actionPerformed(GUI.java:57)
6. caused by java.lang.IndexOutOfBoundsException : 3
7. at jsep.Foo.buggy(Foo.java:17)
8. and 4 more ...
```



Execution

Original Program

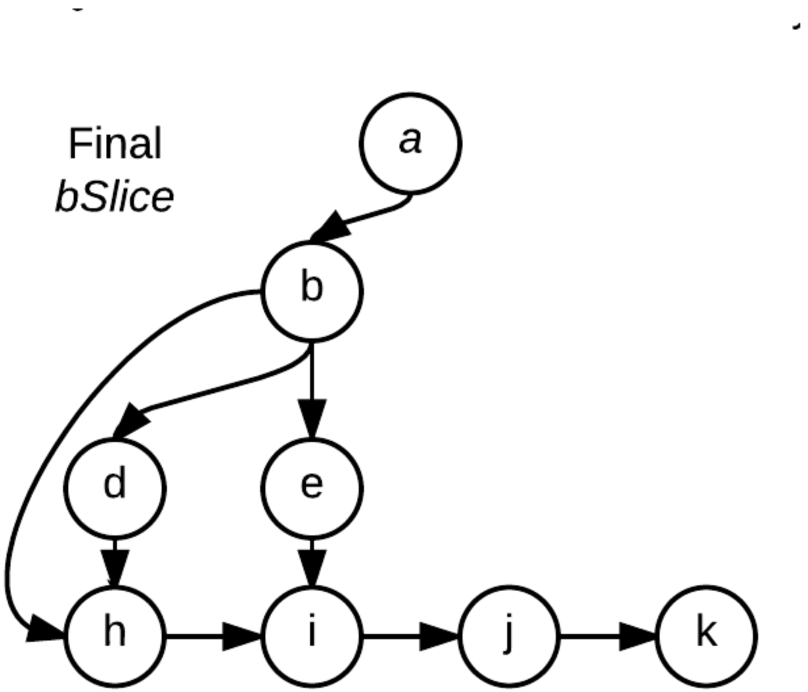
Backward Static Slicing

Slicing

Backward Sliced Program $\text{bslice}_{[m\text{-}n]}$

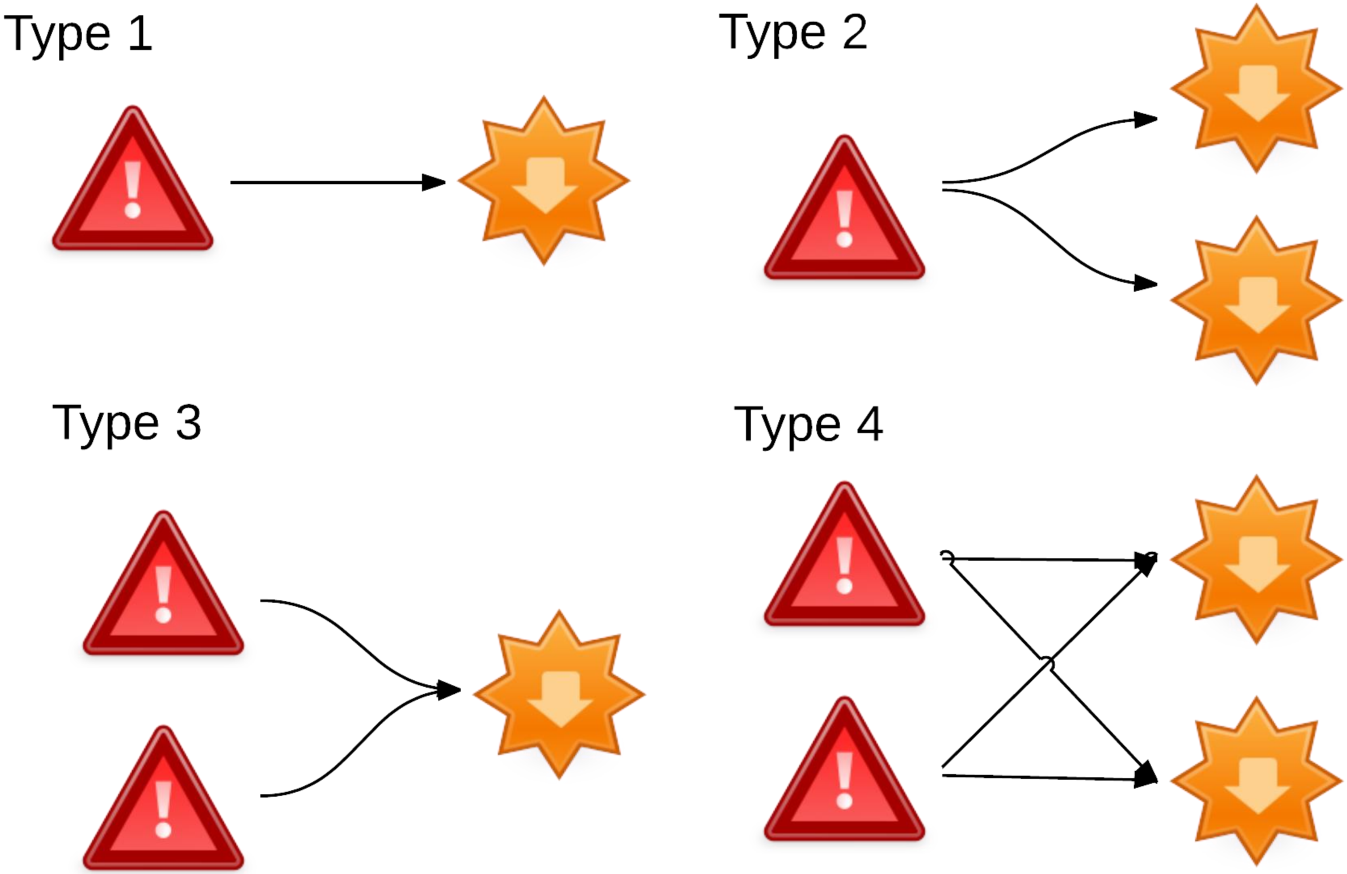
JCHARMING

- Tested on 10 OSS
- 80% success ratio.
- 32 defects.
- Produce unit test that exercise the crash.
- 19 minutes per crash.



Bug Taxonomy

- Categorizing Defects by their fix location and predicting defects type



Type 1

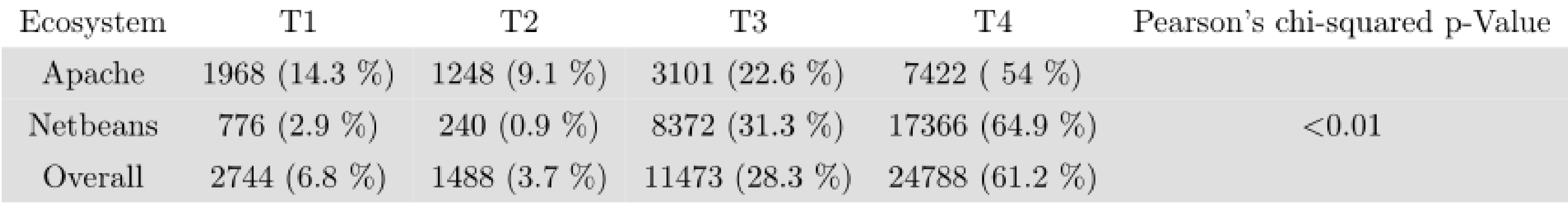
Type 2

Type 3

Type 4

Bug Taxonomy

- Type 4 are predominant
- Type 4 are the most complex
  - duplicate, time to fix, comments, reopening, files changed, severity, changesets, churns



Ecosystem	T1	T2	T3	T4	Pearson's chi-squared p-Value
Apache	1968 (14.3 %)	1248 (9.1 %)	3101 (22.6 %)	7422 (54 %)	<0.01
Netbeans	776 (2.9 %)	240 (0.9 %)	8372 (31.3 %)	17366 (64.9 %)	
Overall	2744 (6.8 %)	1488 (3.7 %)	11473 (28.3 %)	24788 (61.2 %)	

Bug Taxonomy

- Tested on 2 OSE (388 OSS).
- Predict Type 4 defects using SVM on the report words.
- 65.40% precision.
- 94.16% recall.
- 77.19% F1-measure.

Conclusion

- We proposed approaches that fit into the workflow of developers to:
  - Improve coding productivity (BUMPER).
  - Prevent Clone Insertion At Commit-Time (PRECINCT).
  - Prevent Bug Insertion At Commit-Time (BIANCA+CLEVER).
  - Reproduce On-Field Crashes At Report Time (JCHARMING).
  - Propose a new taxonomy of bugs based on their fixes.
- Validated on 455 OSS and CSS.

Publications

- Abdelwahab Hamou-Lhadj, Mathieu Nayrolles: A Project on Software Defect Prevention at Commit-Time: A Success Story of University-Industry Research Collaboration. *(SER&IP 2018, Co-located with the International Conference on Software Engineering 2018).*

- Mathieu Nayrolles, Abdelwahab Hamou-Lhadj: CLEVER: Combining Code Metrics with Clone Detection for Just-In-Time Fault Prevention and Resolution in Large Industrial Projects. *(MSR 2018, Co-located with the International Conference on Software Engineering 2018).*

- Mathieu Nayrolles, Abdelwahab Hamou-Lhadj: Towards a Classification of Bugs to Facilitate Software Maintainability Tasks. *(SQUADE 2018, Co- located with the International Conference on Software Engineering 2018).*

Publications

- Mathieu Nayrolles, Abdelwahab Hamou-Lhadj, Sofiene Tahar, Alf Larsson: A bug reproduction approach based on directed model checking and crash traces. *Journal of Software: Evolution and Process 29(3) (2017).*

- Mathieu Nayrolles, Abdelwahab Hamou-Lhadj: BUMPER: A Tool for Coping with Natural Language Searches of Millions of Bugs and Fixes. *International Conference on Software Analysis, Evolution and Reengineering 2016: 649-652.*

- Mathieu Nayrolles, Abdelwahab Hamou-Lhadj, Sofiene Tahar, Alf Larsson: JCHARMING: A bug reproduction approach using crash traces and directed model checking. *International Conference on Software Analysis, Evolution and Reengineering 2015: 101-110.* ***Best Paper Award***

Publications

- Abdou Maiga, Abdelwahab Hamou-Lhadj, Mathieu Nayrolles, Korosh Koochekian Sabor, Alf Larsson: An empirical study on the handling of crash reports in a large software company: An experience report. *International Conference on Software Maintenance and Evolution 2015: 342-351*

- Mathieu Nayrolles, Eric Beaudry, Naouel Moha, Petko Valtchev, Abdelwahab Hamou-Lhadj: Towards Quality-Driven SOA Systems Refactoring Through Planning. *International Multidisciplinary Conference on e-Technologies 2015: 269-284.*

Publications

THE GLOBE AND MAIL

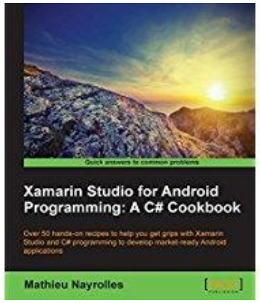
L'actualité

FINANCIAL POST

metro

WIRED

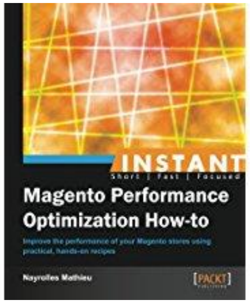
BNN


Quick answers to common problems

Xamarin Studio for Android Programming: A C# Cookbook

Over 50 hands-on recipes to help you get grips with Xamarin Studio and C# programming to develop market-ready Android applications

Mathieu Nayrolles


INSTANT

Short | Fast | Focused

Magento Performance Optimization How-to

Improve the performance of your Magento stores using practical, hands-on recipes

Nayrolles Mathieu

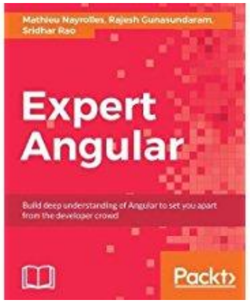
[PACKT]

Magento Site Performance Optimization

Leverage the power of Magento to speed up your website

Mathieu Nayrolles

PACKT open source


Mathieu Nayrolles, Rajesh Gunasundaram, Sridhar Rao

Expert Angular

Build deep understanding of Angular to set you apart from the developer crowd

Packt

Angular 2 Design Patterns and Best Practices

Make the most of Angular 2 by leveraging design patterns and best practices to build stable and high performing apps

Mathieu Nayrolles

PACKT open source

in stal

Mastering Apache Solr

A practical guide to get to grips with Apache Solr

Mathieu Nayrolles

Closing Remarks

- The *truth* is a moving target

- When is the right *time* for JIT Software Maintenance

CONCORDIA.CA