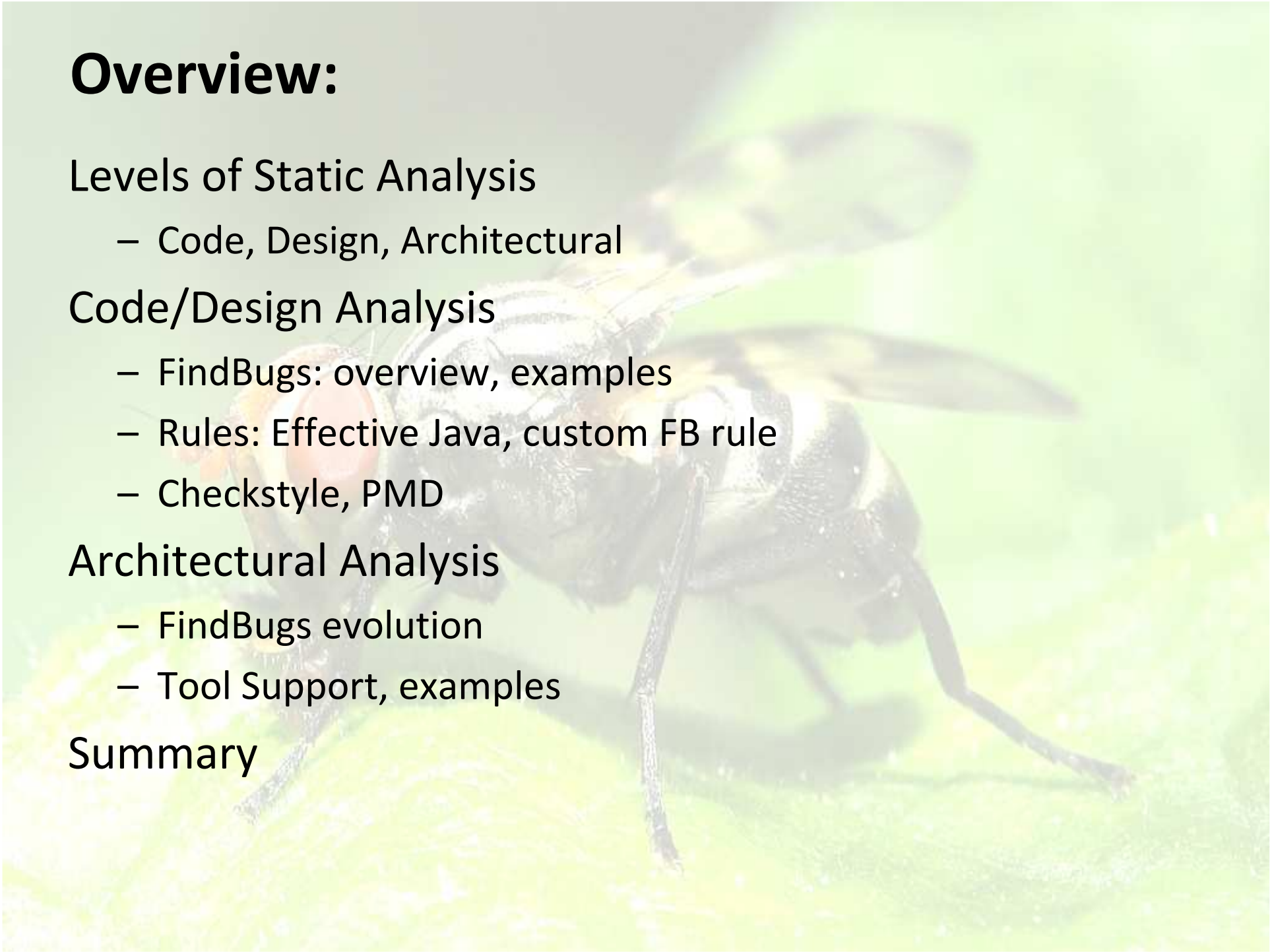
FindBugs

Bernhard Merkle
Research & Development
Software-Engineering
SICK-AG Waldkirch/DE

mailto: Bernhard.Merkle@gmail.com
contact on linkedin.com or xing.com



Overview:

Levels of Static Analysis

- Code, Design, Architectural

Code/Design Analysis

- FindBugs: overview, examples
- Rules: Effective Java, custom FB rule
- Checkstyle, PMD

Architectural Analysis

- FindBugs evolution
- Tool Support, examples

Summary

Possible levels of Static Analysis:

Goal: find, avoid Problems, Increase QA (and measure it)

Micro-Level

- Code
- e.g: =, ==, { },

Macro-Level

- Class-Design
- e.g: by reference, String concat, Exception-Handling

Architecture-Level:

- Layers, Graphs, Subsystems, Compoments, Interfaces
- e.g: Coupling, Dependency, etc…

Tool-support for each level…

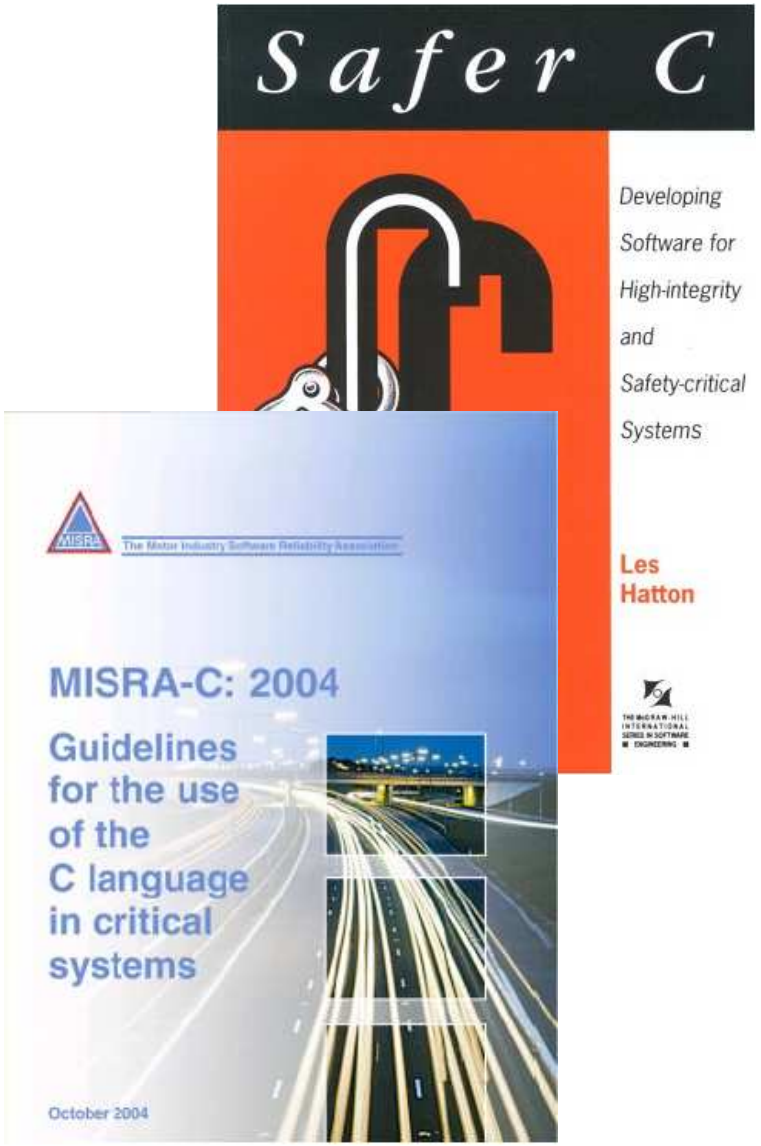
Micro-Level / Code:

Language itself

- Appendix F/ANSI or G/ISO
  - Unspecified behavior
  - Undefined behavior
  - Implementation-defined behavior
  - Locale-specific behavior

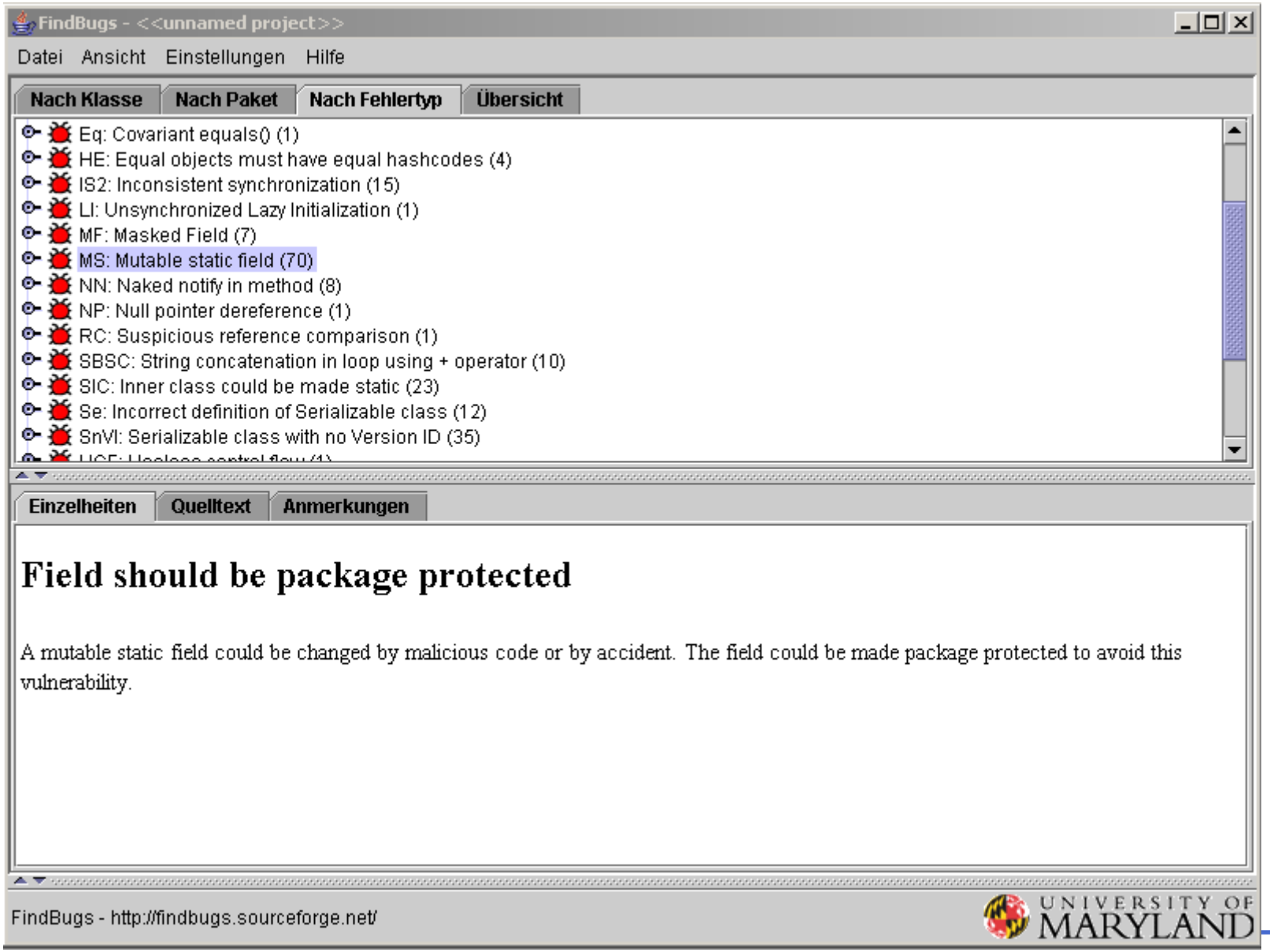
Usage of Language

- Misplacement
- Omission and Addition
- Unexpected behavior
- Complexity

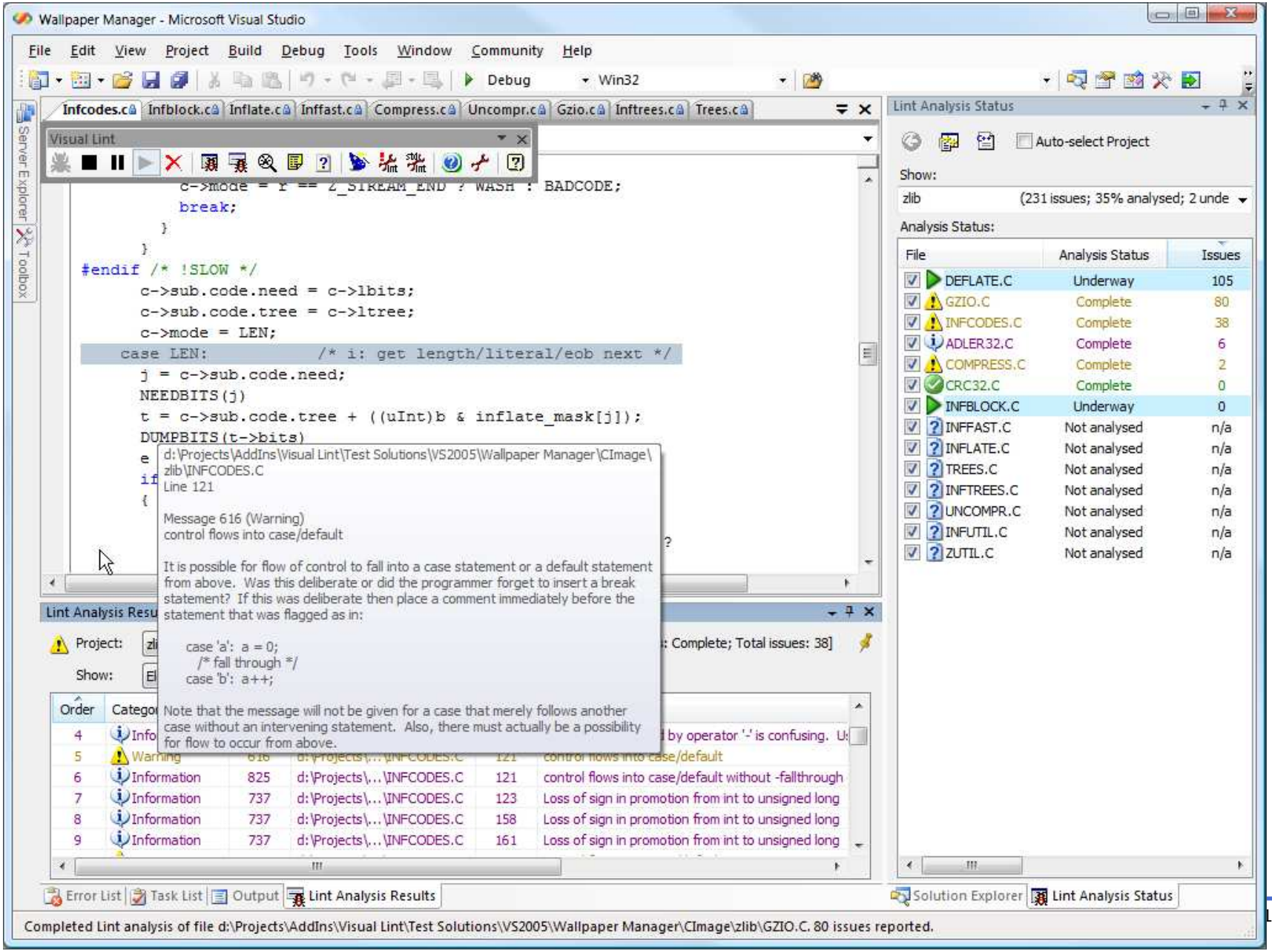


Macro-Level: Design + Coding Guidelines

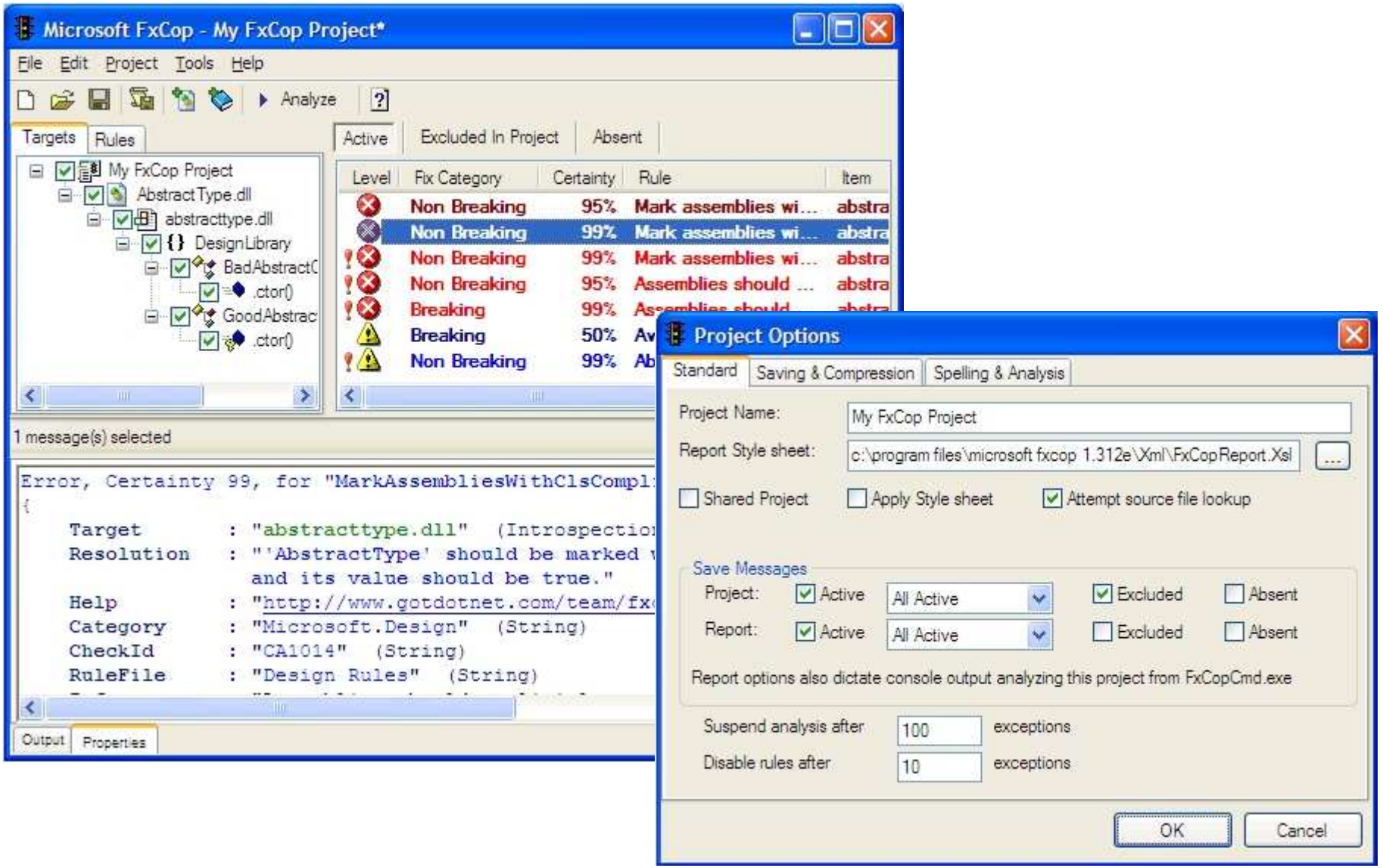
Java: findbugs



C++: VisualLint, Integrations of…



C#: FxCop (Microsoft)



Static Analysis

Analyzes your program without executing it

Doesn't depend on having good test cases

- or even any test cases

Doesn't know what your software is supposed to do

- (In general)
- Looks for violations of reasonable programming
- Shouldn't throw NPE
- Shouldn't allow SQL injection

Not a replacement for testing

- Very good at finding problems on untested paths
- But many defects can't be found with static analysis

Static Analysis

Pros/Cons:

- can help improve code quality
- but lots of improvement still possible and need
- Getting to the point where it really helps

FindBugs as example here:

- Commercial: lint, coverity, QA/C/C++, fortify, CodeTest etc)
- Free: PMD, etc.

Not a stylechecker

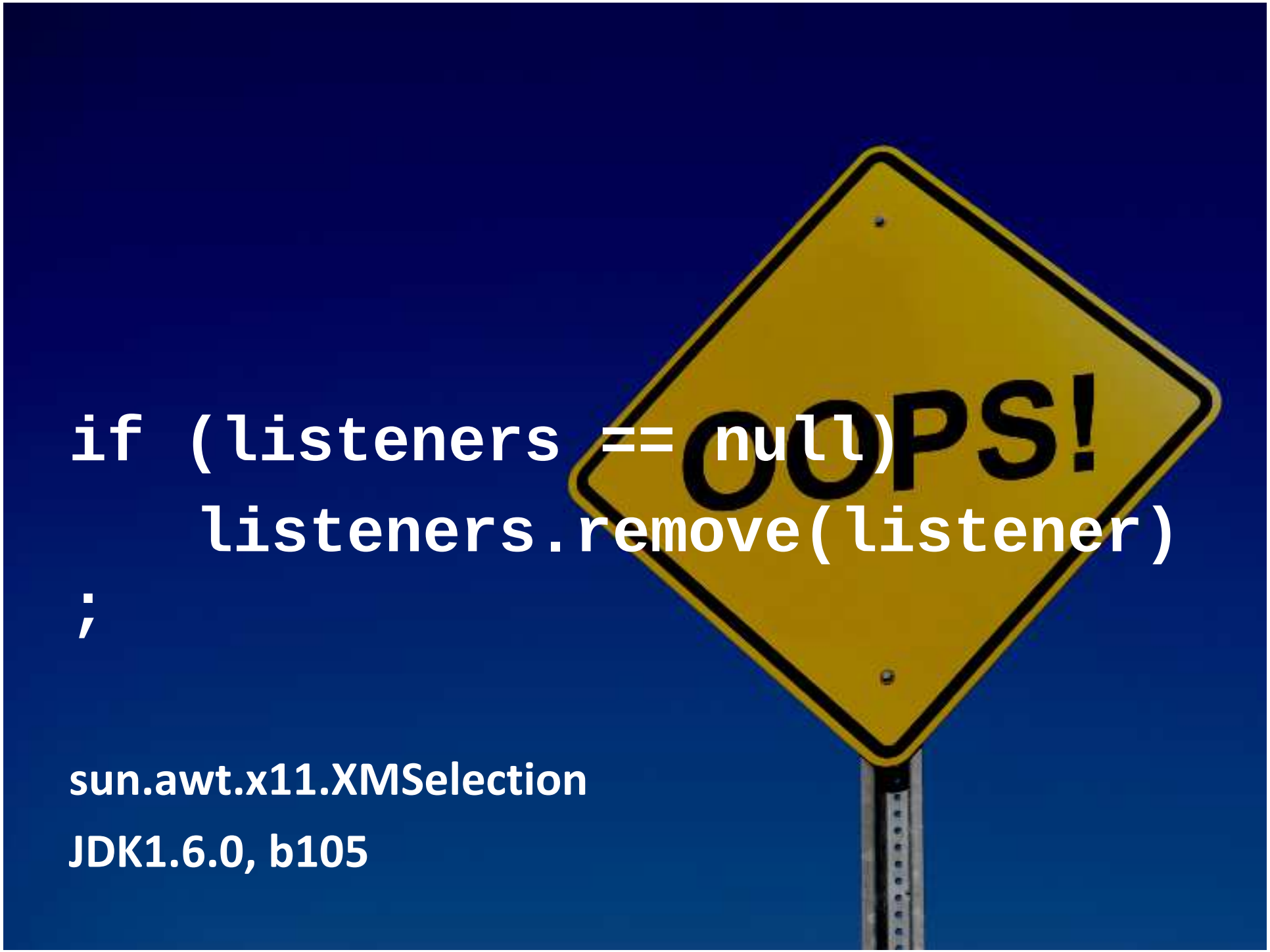
- Checkstyle, IDE, astyle

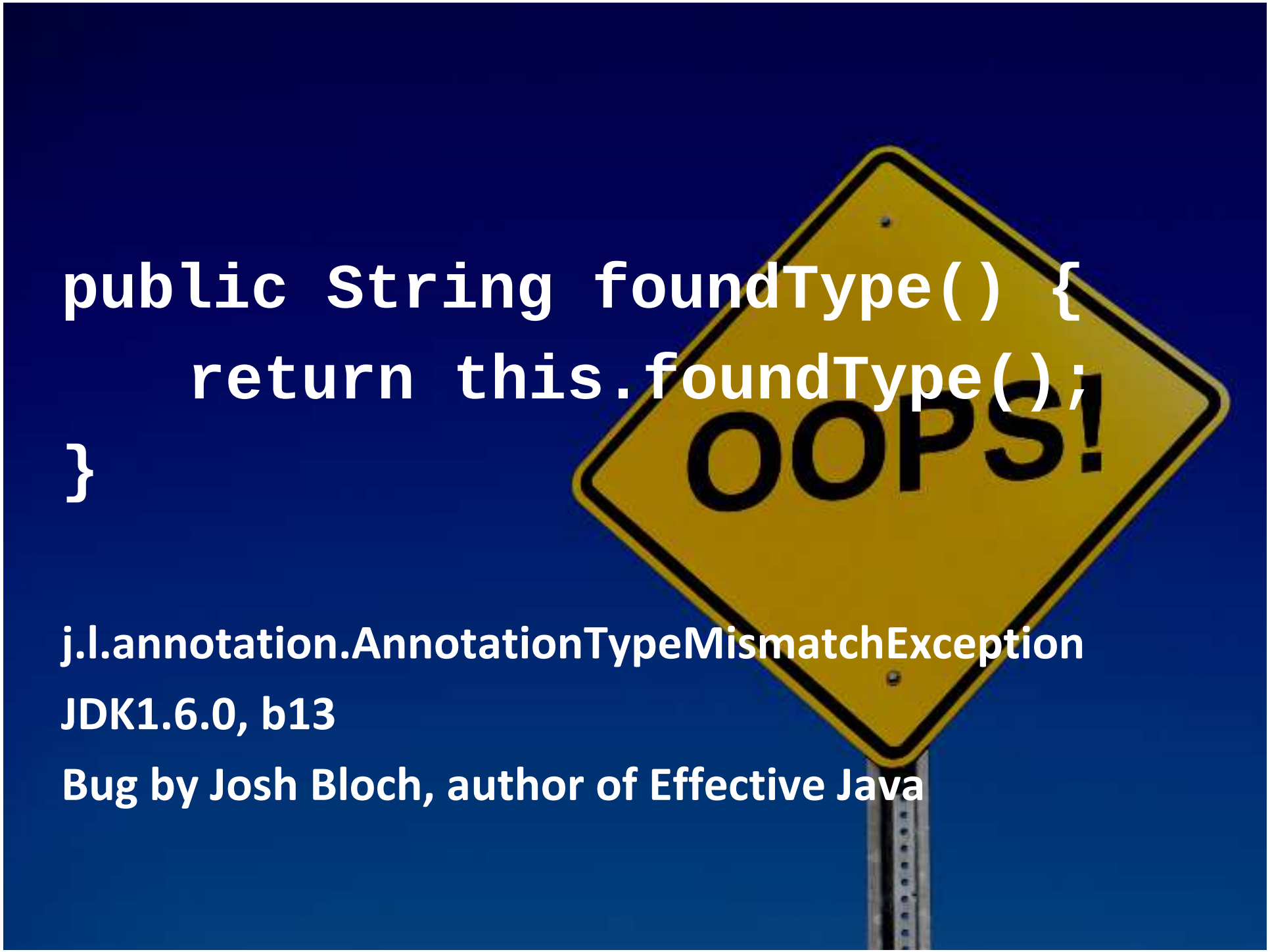
```
if (in == null)
   try {
    in.close();
   …
   }
```

Eclipse 3.0.0. M8

What is lurcking in your code ?

Findbugs is looking for simple, stupid bugs


if (listeners == null)
listeners.remove(listener)
;
OOPS!
sun.awt.x11.XMSelection
JDK1.6.0, b105



j.l.annotation.AnnotationTypeMismatchException

JDK1.6.0, b13

Bug by Josh Bloch, author of Effective Java

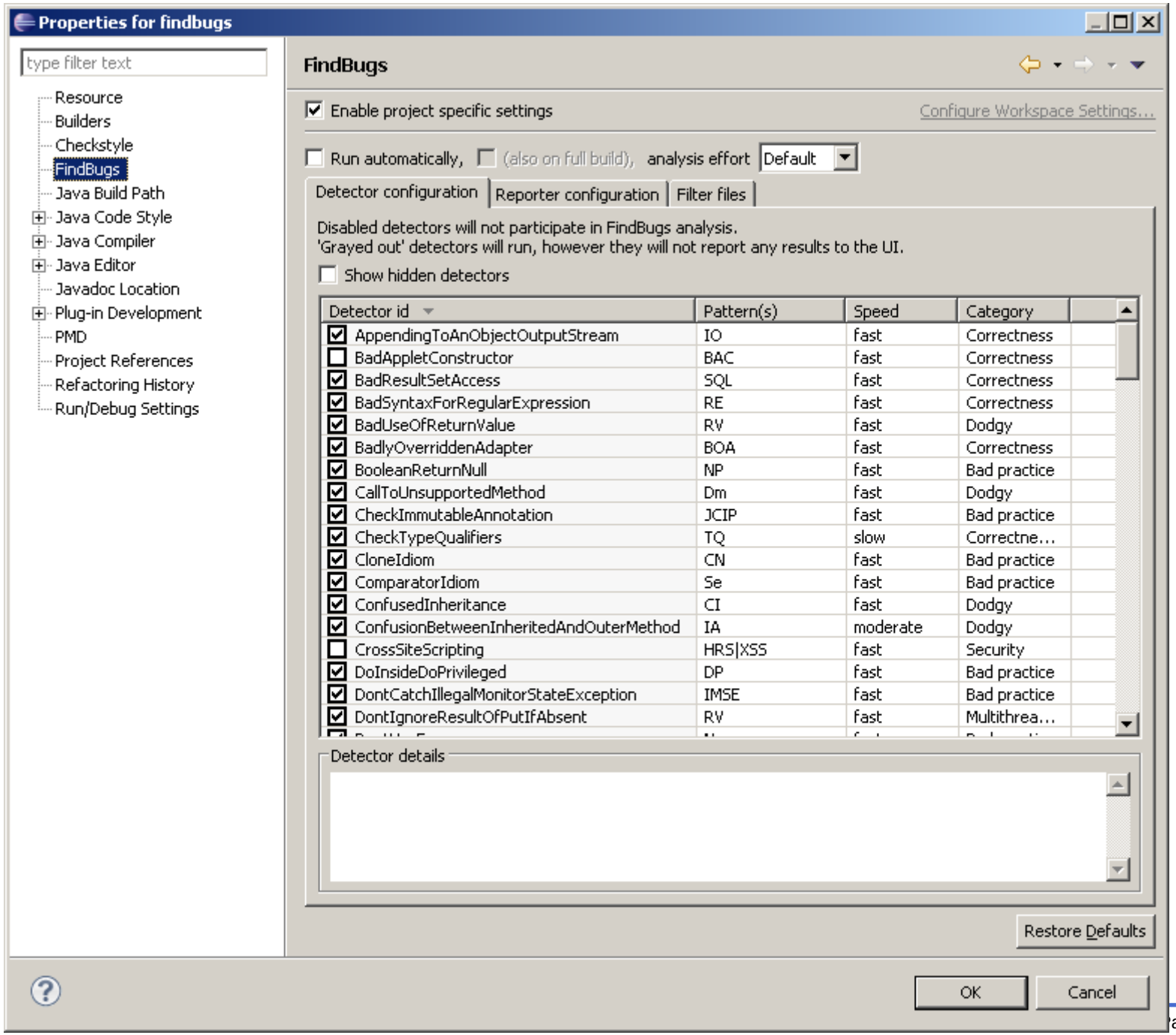
FindBugs:

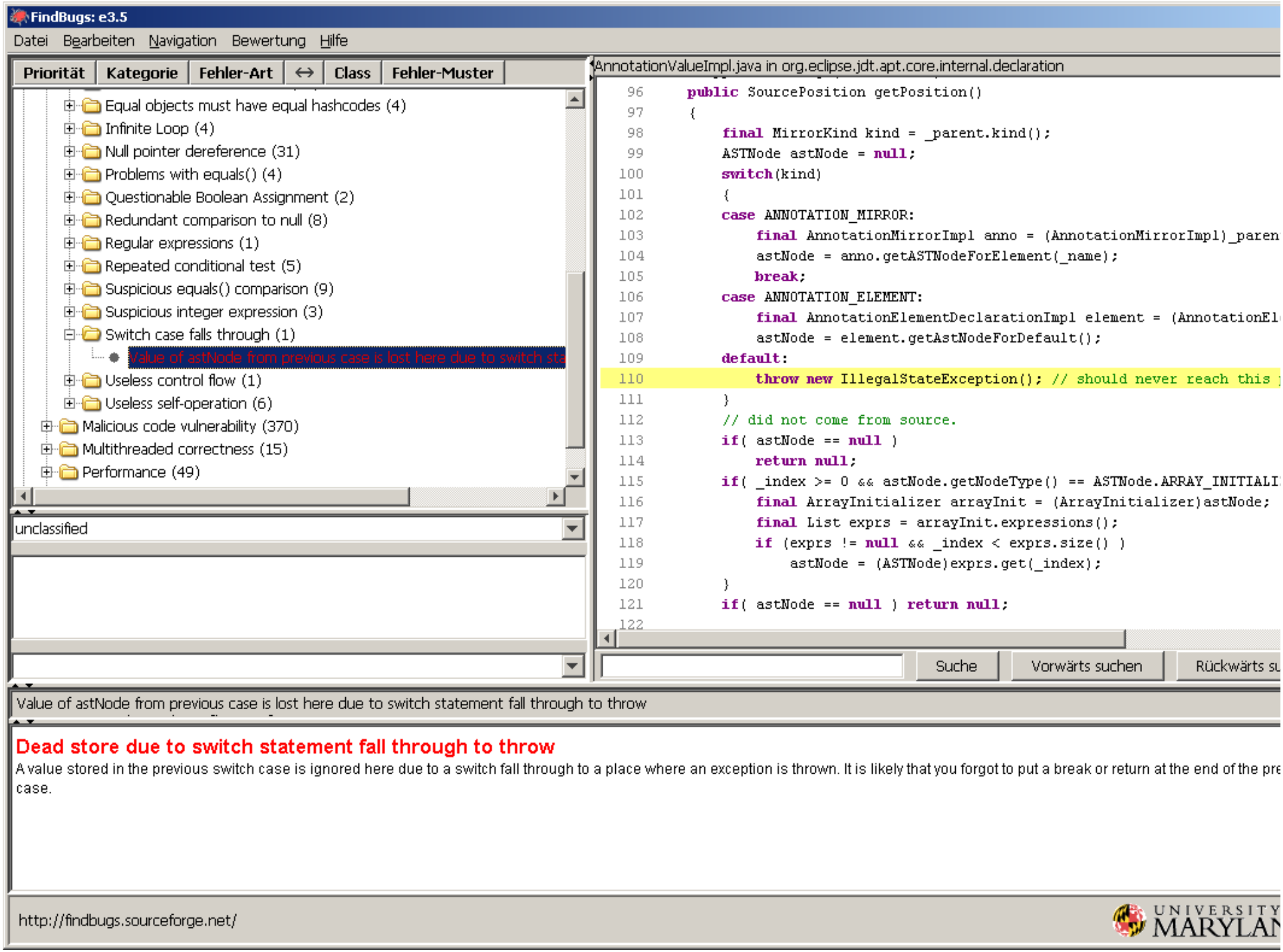
features:

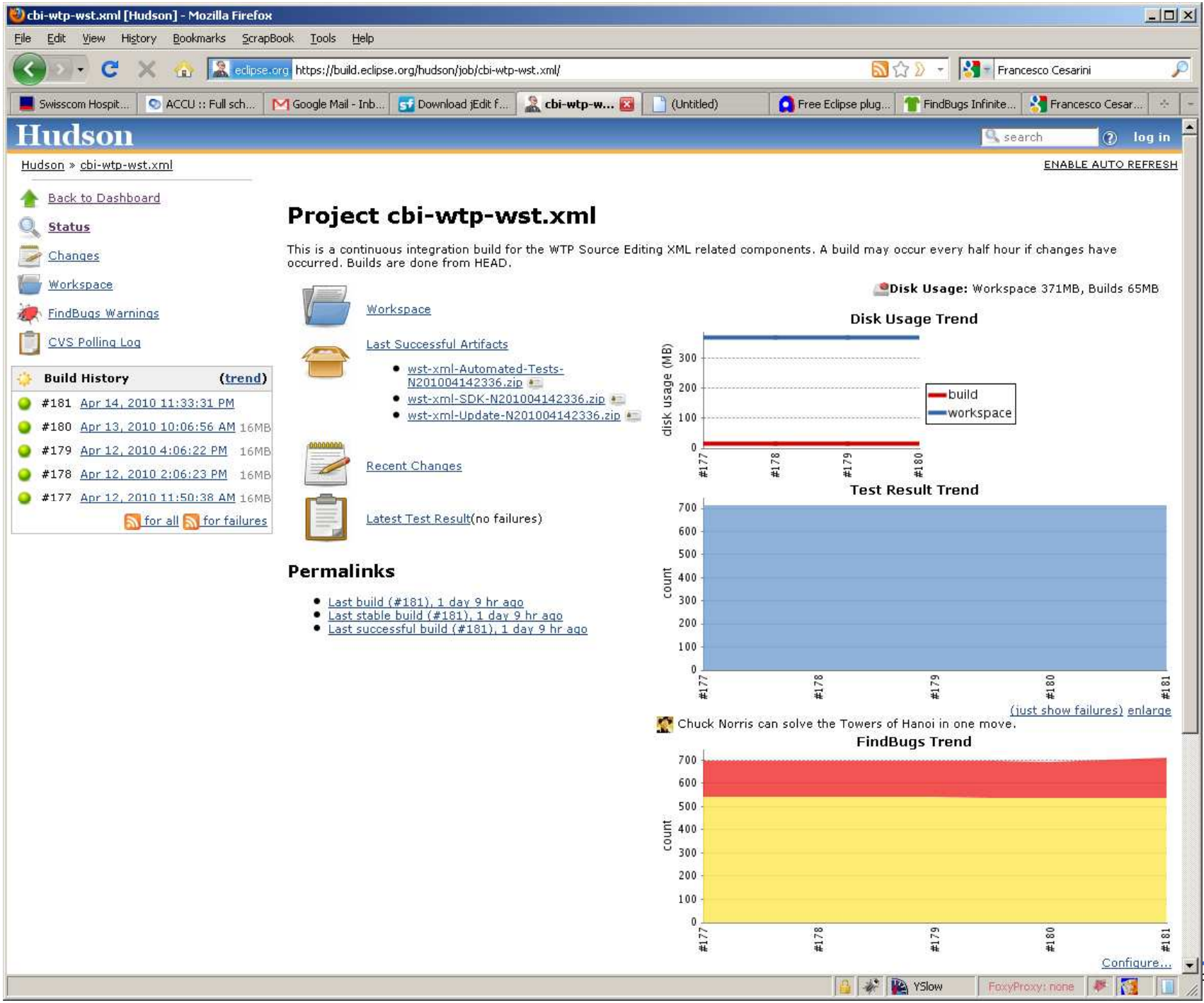
- GUI (swing,eclipse,netbeans)
- Commandline, buildserver
- Bytecode-level static analysis
- XML-based format for saved warnings
  - Handling multiple versions of software
  - Tracking warnings over time

focus developing new analyses, empirical studies, etc.

Large user base: test your idea works in the “real world”


Properties for findbugs
type filter text
Resource
Builders
Checkstyle
FindBugs
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
Plug-in Development
PMD
Project References
Refactoring History
Run/Debug Settings
FindBugs
Enable project specific settings
Configure Workspace Settings...
Run automatically, (also on full build), analysis effort Default
Detector configuration
Reporter configuration
Filter files
Disabled detectors will not participate in FindBugs analysis.
'Grayed out' detectors will run, however they will not report any results to the UI.
Show hidden detectors
Detector id
Pattern(s)
Speed
Category
AppendingToAnObjectOutputStream IO fast Correctness
BadAppletConstructor BAC fast Correctness
BadResultSetAccess SQL fast Correctness
BadSyntaxForRegularExpression RE fast Correctness
BadUseOfReturnValue RV fast Dodgy
BadlyOverriddenAdapter BOA fast Correctness
BooleanReturnNull NP fast Bad practice
CallToUnsupportedMethod Dm fast Dodgy
CheckImmutableAnnotation JCIP fast Bad practice
CheckTypeQualifiers TQ slow Correctne...
CloneIdiom CN fast Bad practice
ComparatorIdiom Se fast Bad practice
ConfusedInheritance CI fast Dodgy
ConfusionBetweenInheritedAndOuterMethod IA moderate Dodgy
CrossSiteScripting HRS|XSS fast Security
DoInsideDoPrivileged DP fast Bad practice
DontCatchIllegalMonitorStateException IMSE fast Bad practice
DontIgnoreResultOfPutIfAbsent RV fast Multithrea...
Detector details
Restore Defaults
OK
Cancel


FindBugs: e3.5
Datei Bearbeiten Navigation Bewertung Hilfe
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
Equal objects must have equal hashcodes (4)
Infinite Loop (4)
Null pointer dereference (31)
Problems with equals() (4)
Questionable Boolean Assignment (2)
Redundant comparison to null (8)
Regular expressions (1)
Repeated conditional test (5)
Suspicious equals() comparison (9)
Suspicious integer expression (3)
Switch case falls through (1)
Value of astNode from previous case is lost here due to switch sta
Useless control flow (1)
Useless self-operation (6)
Malicious code vulnerability (370)
Multithreaded correctness (15)
Performance (49)
unclassified
AnnotationValueImpl.java in org.eclipse.jdt.apt.core.internal.declaration
96 public SourcePosition getPosition()
97 {
98 final MirrorKind kind = _parent.kind();
99 ASTNode astNode = null;
100 switch(kind)
101 {
102 case ANNOTATION_MIRROR:
103 final AnnotationMirrorImpl anno = (AnnotationMirrorImpl)_paren
104 astNode = anno.getASTNodeForElement(_name);
105 break;
106 case ANNOTATION_ELEMENT:
107 final AnnotationElementDeclarationImpl element = (AnnotationEl
108 astNode = element.getAstNodeForDefault();
109 default:
110 throw new IllegalStateException(); // should never reach this
111 }
112 // did not come from source.
113 if(astNode == null)
114 return null;
115 if(_index >= 0 && astNode.getNodeType() == ASTNode.ARRAY_INITIALI
116 final ArrayInitializer arrayInit = (ArrayInitializer)astNode;
117 final List exprs = arrayInit.expressions();
118 if (exprs != null && _index < exprs.size())
119 astNode = (ASTNode)exprs.get(_index);
120 }
121 if(astNode == null) return null;
122
Suche
Vorwärts suchen
Rückwärts su
Value of astNode from previous case is lost here due to switch statement fall through to throw
Dead store due to switch statement fall through to throw
A value stored in the previous switch case is ignored here due to a switch fall through to a place where an exception is thrown. It is likely that you forgot to put a break or return at the end of the pre
case.
http://findbugs.sourceforge.net/
UNIVERSITY
MARYLAN


cbi-wtp-wst.xml [Hudson] - Mozilla Firefox
File Edit View History Bookmarks ScrapBook Tools Help
https://build.eclipse.org/hudson/job/cbi-wtp-wst.xml/
Francesco Cesarini
Hudson
search
log in
Hudson » cbi-wtp-wst.xml
ENABLE AUTO REFRESH
Back to Dashboard
Status
Changes
Workspace
FindBugs Warnings
CVS Polling Log
Build History (trend)
#181 Apr 14, 2010 11:33:31 PM
#180 Apr 13, 2010 10:06:56 AM 16MB
#179 Apr 12, 2010 4:06:22 PM 16MB
#178 Apr 12, 2010 2:06:23 PM 16MB
#177 Apr 12, 2010 11:50:38 AM 16MB
for all for failures
Project cbi-wtp-wst.xml
This is a continuous integration build for the WTP Source Editing XML related components. A build may occur every half hour if changes have occurred. Builds are done from HEAD.
Workspace
Last Successful Artifacts
wst-xml-Automated-Tests-N201004142336.zip
wst-xml-SDK-N201004142336.zip
wst-xml-Update-N201004142336.zip
Recent Changes
Latest Test Result(no failures)
Permalinks
Last build (#181), 1 day 9 hr ago
Last stable build (#181), 1 day 9 hr ago
Last successful build (#181), 1 day 9 hr ago
Disk Usage: Workspace 371MB, Builds 65MB
Disk Usage Trend
disk usage (MB)
build
workspace
Test Result Trend
count
(just show failures) enlarge
Chuck Norris can solve the Towers of Hanoi in one move.
FindBugs Trend
count
Configure...
YSlow
FoxyProxy: none

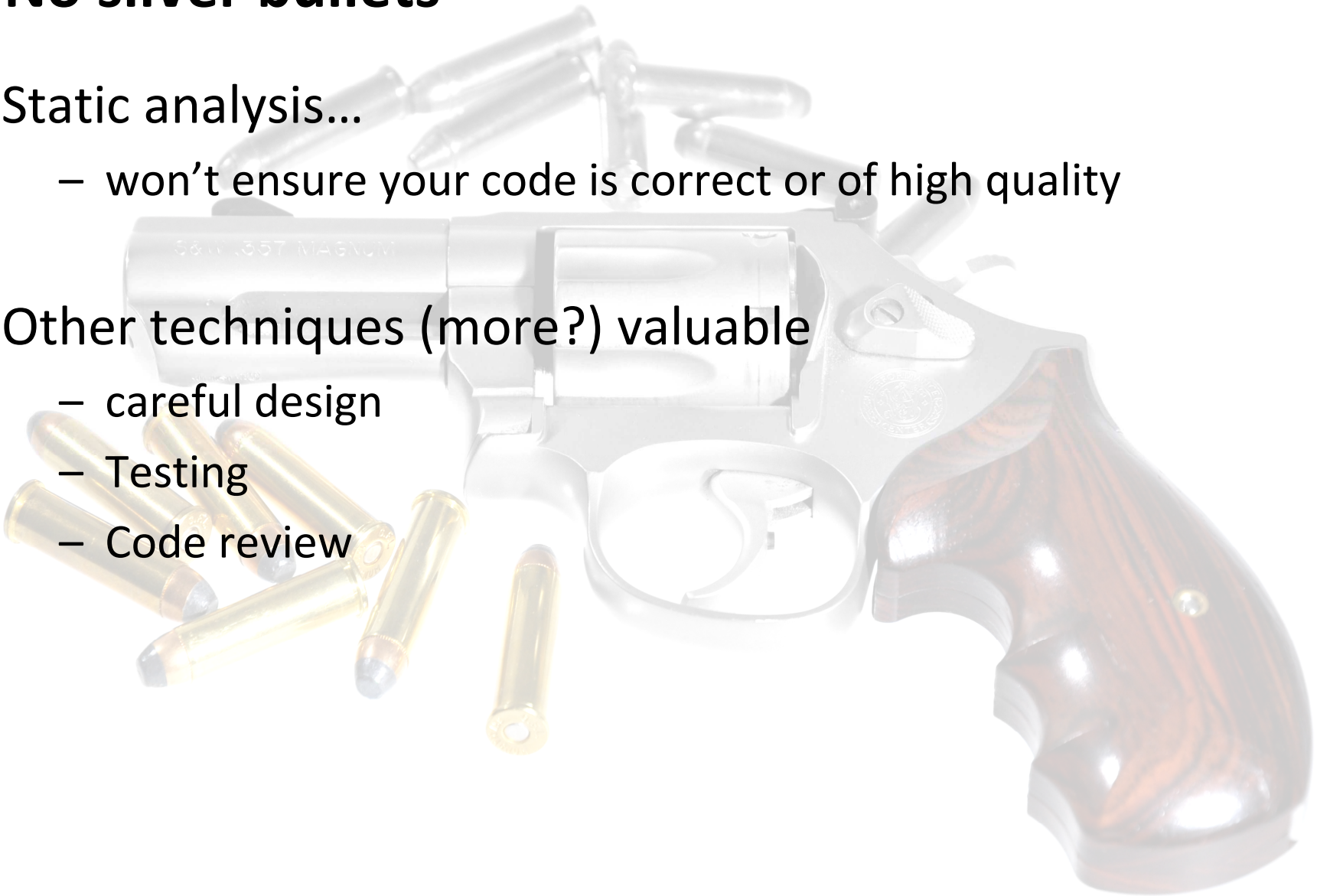
No silver bullets

Static analysis…

- won’t ensure your code is correct or of high quality

Other techniques (more?) valuable

- careful design
- Testing
- Code review




Skim the cream
Find more/faster issues comp. testing
Should be part of your development proc
Integrate in buildserver
Established: look a new+high prio issues


Bug Categories
Bug Patterns

Bug Categories

- Correctness
- Bad practice
- Dodgy code
- Multithreaded correctness
- Potential performance problems
- Malicious code vulnerability

Bug Patterns

Some big, broad and common patterns

- Dereferencing a null pointer
- An impossible checked cast
- Methods whose return value should not be ignored

Lots of small, specific bug patterns, that together find lots of bugs

- Every Programming Puzzler
- Every chapter in *Effective Java*

Analysis Techniques

Local pattern matching

- **String.toLowerCase()**, don't ignore the return value

Intraprocedural dataflow analysis

- Null pointer, type cast errors

Interprocedural method summaries

- E.g. method always dereferences its parameter

Context sensitive interprocedural analysis

- Interprocedural flow of untrusted data, injection

Categories, ranking, use cases:

customize what you want to look at

audit of a newly written module with 1,000 lines of code

scan 1,000,000 lines of “legacy” code

use cases require different tunings

Infinite recursive loop

Typical bug for CS students

```
/** Construct a WebSpider */
public WebSpider() {
  w = new WebSpider();
}
```

Okay this is (silly) student stuff…

Number of infinite recursive loops

5 infinite recursive loops in JDK

27 across all versions of JDK,

40+ in Google's Java code

??? in Eclipse code

??? in your code

org.netbeans.modules.web.monitor.client.MonitorAction

```
public static void log(String s) {
   log ("MonitorAction::" +s)
}
```

Examples of null pointer bugs

//com.sun.corba.se.impl.naming.cosnaming.NamingContextImpl

```
if (name != null || name.length > 0)
```

//com.sun.xml.internal.ws.wsdl.parser.RuntimeWSDLParser

```
if (part == null | part.equals(""))
```

// sun.awt.x11.ScrollPanePeer

```
if (g != null)
    paintScrollBars(g,colors);
g.dispose();
```

Redundant Check For Null

Checking a value to see if it is null When it can't be null

```
// java.awt.image.LoopupOp, lines 236-247
public final WritableRaster filter(
          Raster src, WritableRaster dst) {
     int dstLength = dst.getNumBands();
     // Create a new destination Raster,
     // if needed
     if (dst == null)
```

Bad Method Invocation

Methods whose return value shouldn't be ignored

- Strings are immutable, so functions like **trim()** and **toLowerCase()** return new String

Lots of specific rules about particular API methods

- Hard to memorize, easy to get wrong

Examples of bad method calls

// com.sun.rowset.CachedRowSetImpl

```
if (type == Types.DECIMAL || type == Types.NUMERIC)
      ((java.math.BigDecimal)x).setScale(scale);
```

// com.sun.xml.internal.txw2.output.XMLWriter

```
try { ... }
catch (IOException e) {
      new SAXException("Server side Exception:" + e);
}
```

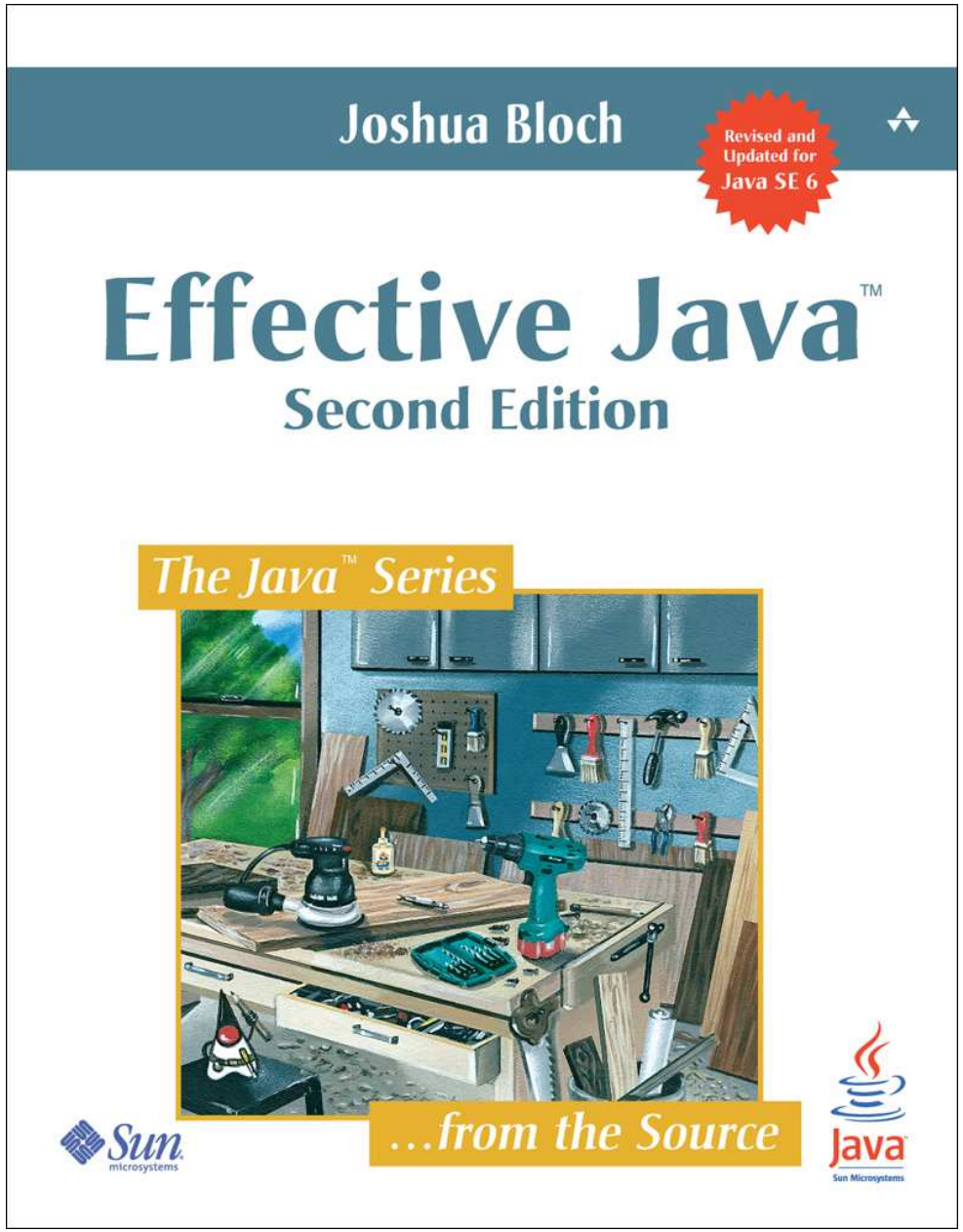
More results:

Glassfish

- 29 ignored return values
- 59 classes with equals() but not hashCode()
- 9 calls to equals will always return false
- 18 statement + 98 branches with NPE
- 10 methods with IRL

Weblogic

- 1166 ignored return values
- 254 classes with equals() but not hashCode()
- 45 calls to equals will always return false
- 35 statement + 98 branches with NPE
- 21 methods with IRL



ACCU

- mentored developer list

- Effective Java
- (Effective C++)
- (Design Patterns)

3 Methods Common to All Objects

- **Item 8: Obey the general contract when overriding equals**
- Item 9: Always override hashCode when you override equals
- Item 10: Always override toString
- Item 11: Override clone judiciously
- Item 12: Consider implementing Comparable

Item 8: Obey contract overriding `equals`

easiest way to avoid problems is:

- not to override **`equals`**

right thing to do if any of the following conditions apply:

- **Each instance of the class is inherently unique (no value)**
  - **`Thread`**
- **don't care whether class provides a "logical equality" test**
  - **`java.util.Random`**
- **A superclass has already overridden equals, && the superclass behavior is appropriate for this class.**
  - `Set` impls inherit `equals` from `AbstractSet` (dito `List`, `Map`)

Item 8: Obey contract overriding `equals`

right thing to do if any of the following conditions apply:

- **class is private or package-private, and you are certain that its equals method will never be invoked**
  - Really certain ?

```
@Override public boolean equals(Object o) {
  // Method is never called
 throw new AssertionError();
}
```

Item 8: Obey contract overriding `equals`

When to override **`Object.equals()`**?

- *logical equality* that differs from mere object identity
- Value classes (Integer, Date, etc)

- must adhere to its general contract for `Object` [JavaSE6]:
  The equals method implements an *equivalence relation.*
  - *Reflexive*: $a \sim a$
  - *Symmetric*: if $a \sim b$ then $b \sim a$
  - *Transitive*: if $a \sim b$ and $b \sim c$ then $a \sim c$.
  - *Consistent*: multiple invocations of x.equals(y) return same value
  - For any non-null reference value x, x.equals(null) must return false.

Item 8: Obey contract overriding `equals`

a recipe for a high-quality **`equals()`** method:

- **Use `==` operator to check if argument is a reference to this object**
  - If so, return **`true`**. Performance optimization
- **Use the `instanceof` operator to check if the argument has the correct type**
  - If not, return **`false`**.
- **Cast the argument to the correct type**
- **For each "significant" field in the class, check** matches
  - For primitive fields, use **`==`** operator
  - For object reference fields, use **`equals()`**
  - For **`float/double`** use **`Float/Double.compare()`**
  - Arrays.equals
- Watch out for **`null`**

```
(field == null ? o.field == null :
  field.equals(o.field))
```

Item 8: Obey contract overriding `equals`

Caveat when overriding **`equals()`** method

- **Always override hashCode when override equals** (Item 9).
- **Don't try to be too clever**
  - If you are overly aggressive in searching for equivalence, it's easy to get into trouble. E.g. aliasing))
- **Don't substitute another type for Object in the equals declaration (override vs. overload)**

```
//MyClass.class
public boolean equals(MyClass o) {
...
}
```

→ Consistent use of the @Override annotation

Test Item 8 with real code

Equal objects must have equal hash codes

- Fail: override equals() but not hashCode() (or other way around)
- Object violating the contract won't work in hash table, maps, set, etc

- Examples: (53 bugs in JDK 1.6.0)
- `java.awt.geom.Area`
- `javax.management.Attribute`

4 Classes and Interfaces

- **Item 13: Minimize the accessibility of classes and members**
- Item 14: In public classes, use accessor methods,not public fields
- Item 15: Minimize mutability
- Item 16: Favor composition over inheritance
- Item 17: Design and document for inheritance or else prohibit it
- Item 18: Prefer interfaces to abstract classes
- Item 19: Use interfaces only to define types
- Item 20: Prefer class hierarchies to tagged classes
- Item 21: Use function objects to represent strategies
- Item 22: Favor static member classes over nonstatic

Item 13: min accessibility of classes/members

well-designed module ?

- *decouples* the modules that comprise a system,
- increases software reuse

Java access modifiers

- For top-level (non-nested) classes and interfaces:
  - **`package-private`**
  - **`public`**
- For members (fields, methods, nested classes, and nested interfaces),
  - **`private`**
  - **`package-private`**
  - **`protected`**
  - **`public`**

Item 13: min accessibility of classes/members

Rule: always use lowest access level

- consider nested If only used by one other class
- private fields leak into API if class implements **`Serializable`**
- protected member is part of exported API, must be supported

Instance fields should never be public

- give up the ability to limit the values that can be stored
- no enforcing of invariants
- no control over time of modification (→ thread unsafe !)
- no switch to a new internal data representation

Item 13: min accessibility of classes/members

public static fields ?

- Same as public instance fields. But make them `final`
- fields should be primitive value or reference to immutable

```
// Potential security hole! Client/content
public static final Thing[] VALUES = { ... };

// Fix ?
private static final Thing[] PRIVATE_VALUES = { ... };

public static final List<Thing> VALUES =
Collections.unmodifiableList(Arrays.asList(PRIVATE_VALUES
  ));

public static final Thing[] values()
{return PRIVATE_VALUES.clone();}
```

When Bad Code Isn't A Bug

When Bad Code Isn't A Bug

```
// com.sun.jndi.dns.DnsName, lines 345-347
if (n instanceof CompositeName) {
   // force ClassCastException
   n = (DnsName) n;
}

// com.sun.corba.se.impl.dynamicany.DynAnyComplexImpl
String expectedMemberName = null;
try {
    expectedMemberName =
       expectedTypeCode.member_name(i);
} catch (BadKind badKind) { // impossible
} catch (Bounds bounds) { // impossible
}
if ( !(expectedMemberName.equals(memberName) ... ))
{
```

Other Categories

Bad practice

Dodgy code

Multithreaded correctness

Performance

Vulnerability to malicious code

Bad Practice

A class that defines an equals method but inherits hashCode from Object

equals method doesn't handle null argument

Serializable class without a serialVersionUID

Exception caught and ignored

Broken out from the correctness category because I never want a developer to yawn when I show them a "correctness" bug

Dodgy code

Dead local store - a value is stored into a local variable, but that value is never used

Use of non-short circuit boolean logic

Switch statement fallthrough

Branch where code on both branches is identical

Multithreaded correctness

Inconsistent synchronization - a lock is held most of the time a field is accessed, but not always

Problems with wait/notify - e.g., call to wait() not in loop

thread unsafe lazy initialization of static field

Performance

Unused field

Invocation of Boolean or Integer constructors

Using hashCode or equals method on a URL

final constant field that could be made static

Loop with quadratic string concatenation

Inner class that could be made static

Vunerability to Malicious code

public static non-final fields

public static final fields that reference mutable objects

Methods that don’t defensively copy mutable arguments before storing them into fields

Methods that don’t defensively copy mutable values stored in fields before returning them

Confusing/bad naming

- Maybe more a “style” issue ?
- Methods with identical names and signatures
  - But different capitalization of names
  - Override in subclass really ? Overload ?
- Method name same as class name

- → resolved by Java5 @Overrides annotation

Bad naming in Eclipse

```
package org.eclipse.jface.dialogs;

public abstract class Dialog extends Window {
   protected Button getOKButton () {
   ….
   }

public class InputDialog extends Dialog {
   protected Button getOkButton() {
   ….
   }
```

Bad naming in BCEL

```
public int hashcode() {
   return basic_type.hashCode();
}
```

Read Return Value ignored

- InputStream.read()
- → read into a byte array, return number of bytes read

- Can be less than number requested
- Programmers sometimes fail to check return value

- Example: of GNU Class path XXX todo

Return Value ingored: e.g. immutable Classes

Operations on immutable objects

```
// Eclipse 3.0.0M8
String name workingCopy.getName();
Name.replace('/', '.');
```

Or example:

- create Thread and do NOT start it
- Create Exception but do NOT throw it (also in eclipse)

Inconsistent Synchronisation

- Common idiom for thread safe classes is to sync on the receiver object (“this”)
- Findbugs checks for field access
  - Find classes, where lock on “this” is not always (only sometimes)
  - Unsnychronized access, if reachable from multiple threads, → race conditions

Example: GNU classpath 0.08 java.util.Vector

```
public int lastIndexOf(Object elem)
{
   return lastIndexOf(elem, elementCount – 1);
}

public synchronized int lastIndexOf(Ojbect e, int
index)
{
   ….
}
```

Ideas:

We do not know the design of the class.
Thread safe of not ?

No STD-Annotation for that
(but there are submissions for that)

So findbugs looks if only 80% are synced.
If yes, 20% may be bugs… ;-)

Writing new FindBugs detectors

Why?:

- You may find bug patterns in your own code
- OK, write a FindBugs detector

How?

- Inspect bytecode…
- Use “ByteCode Outline” Eclipse Plugin

There are many ways to implement a FindBugs detector

- Often, simple techniques (e.g., sequential scan) suffice

Basic Approach Algorithm:

Start with a bug (important!)

Write the simplest possible detector

While (not happy && False positive rate)

Refine: improve analysis FP suppression heuristics

Done

Example Bug

Don’t use String literals for the synchronized blocks!

```
static private final String LOCK = "LOCK";
void someMethod() {
     synchronized(LOCK) {
        ...
     }
}
```

Real world example from Jetty

If other code synchronizes on same String: deadlock

Writing a detector

Add test case

Bytecode:

```
LDC "LOCK"
DUP
ASTORE 1
MONITORENTER
```

Let’s use opcode stack.

Could also look for bytecode sequence.

SynchronizationOnSharedBuiltinConstant

```
public void sawOpcode(int seen) {
if (seen == MONITORENTER) {
   OpcodeStack.Item top = stack.getStackItem(0);

   if (top.getSignature().equals("Ljava/lang/String;")
       && top.getConstant() instanceof String)

           bugReporter.reportBug(new BugInstance(this,
           "DL_SYNCHRONIZATION_ON_SHARED_CONSTANT",
           NORMAL_PRIORITY)
           .addClassAndMethod(this)
           .addString((String)constant)
           .addSourceLine(this));
  }

}
```

Results

Found issue fixed in Jetty-352.

Jetty-352 didn’t fix all occurrences in Jetty (Jetty-362).

Also found occurrences in Eclipse, glassfish, Sun’s JDK, netbeans, nutch, oc4j, weblogic, websphere

Not bad for 20 minutes work

Bytecode frameworks

Fb detectors are based on bytecode

Frameworks:

- BCEL (http://jakarta.apache.org/bcel/); DOM-like API
- ASM (http://asm.objectweb.org/); SAX-like API

Currently BCEL, future is ASM

Types of detectors

implementation techniques:

- Inspecting class/method/field structure
- Micropatterns: simple bytecode patterns
- Stack-based patterns
- Dataflow analysis
- Interprocedural analysis

supported by ready-made base classes
and support infrastructure

More simple rules:

do not require code analysis: e.g.

- classes that override equals() but not hashCode()
- method naming problems
  (e.g., hashcode() instead of hashCode())

More simple rules

Naming.java

```
        return;

    if (mName.equals("equal") && sig.equals("(Ljava/lang/Object;)Z")) {
        bugReporter.reportBug(new BugInstance(this, "NM_BAD_EQUAL", HIGH_P
                .lowerPriorityIfDeprecated());
        return;
    }
    if (mName.equals("hashcode") && sig.equals("()I")) {
        bugReporter.reportBug(new BugInstance(this, "NM_LCASE_HASHCODE", H
                .lowerPriorityIfDeprecated());
        return;
    }
    if (mName.equals("tostring") && sig.equals("()Ljava/lang/String;")) {
        bugReporter.reportBug(new BugInstance(this, "NM_LCASE_TOSTRING", H
                .lowerPriorityIfDeprecated());
        return;
    }

    if (obj.isPrivate() || obj.isStatic() || mName.equals("<init>"))
        return;
```

Stack-based patterns

Micropatterns where the values on the operand stack are significant.

Example:

- As seen : look for monitorenter on constant String value

Typical implementation strategy:

- Inquire about values on operand stack
- Warn when suspicious instruction sequence/stack values seen

Dataflow analysis

Use intraprocedural dataflow analysis to infer (probable) facts within methods.

- You may need to dust off your copy of the Dragon book.

Examples:

- Find dereferences of null values
- Find field accesses not consistently protected by a lock

Interprocedural analysis

Summarize method behavior, and use that summary at each call site

Examples:

- Method parameters that are unconditionally dereferenced
- Return values that are always nonnull
- Methods that always throw an exception

Caveat

```
[...]
> I'm running FindBugs 1.3.9 on a large project via both the Eclipse
> plug-in and Ant.
>
> Using the default Eclipse plug-in settings, I get 189 bugs.
>
> Using the Ant task I get 149.
>
> Missing from the Ant report are:
> * Dead store to local variable
> * Nullcheck of a value previously dereferenced
> * Redundant nullcheck
> * Useless control flow to next line
> * Method ignores exceptional return value
[...]
> Can anyone tell me:
> 1) why Ant finds fewer bugs than Eclipse
> 2) how to find via Ant the bugs that Eclipse catches.

--
```

Caveat

Bytecode != Bytecode

Eclipse ejc and JDK javac

New Features of FindBugs

Behaviour Annotations

- Help your friendly static analyzer
- proprietary Annotations in fb and IntelliJ (MS SAL, etc)

- JSR 305: Annotations for Software Defect Detection
- Examples
  - @NonNull
  - @CheckForNull
  - @checkReturnValue
  - @Tainted/@Unteinated/@Detainted
- JSR 308: Annotations on Java Types
  - ArrayList<@Nonnull String> a = ...
  - JDK 7, more flexible

Computing bug history

Dashboard like

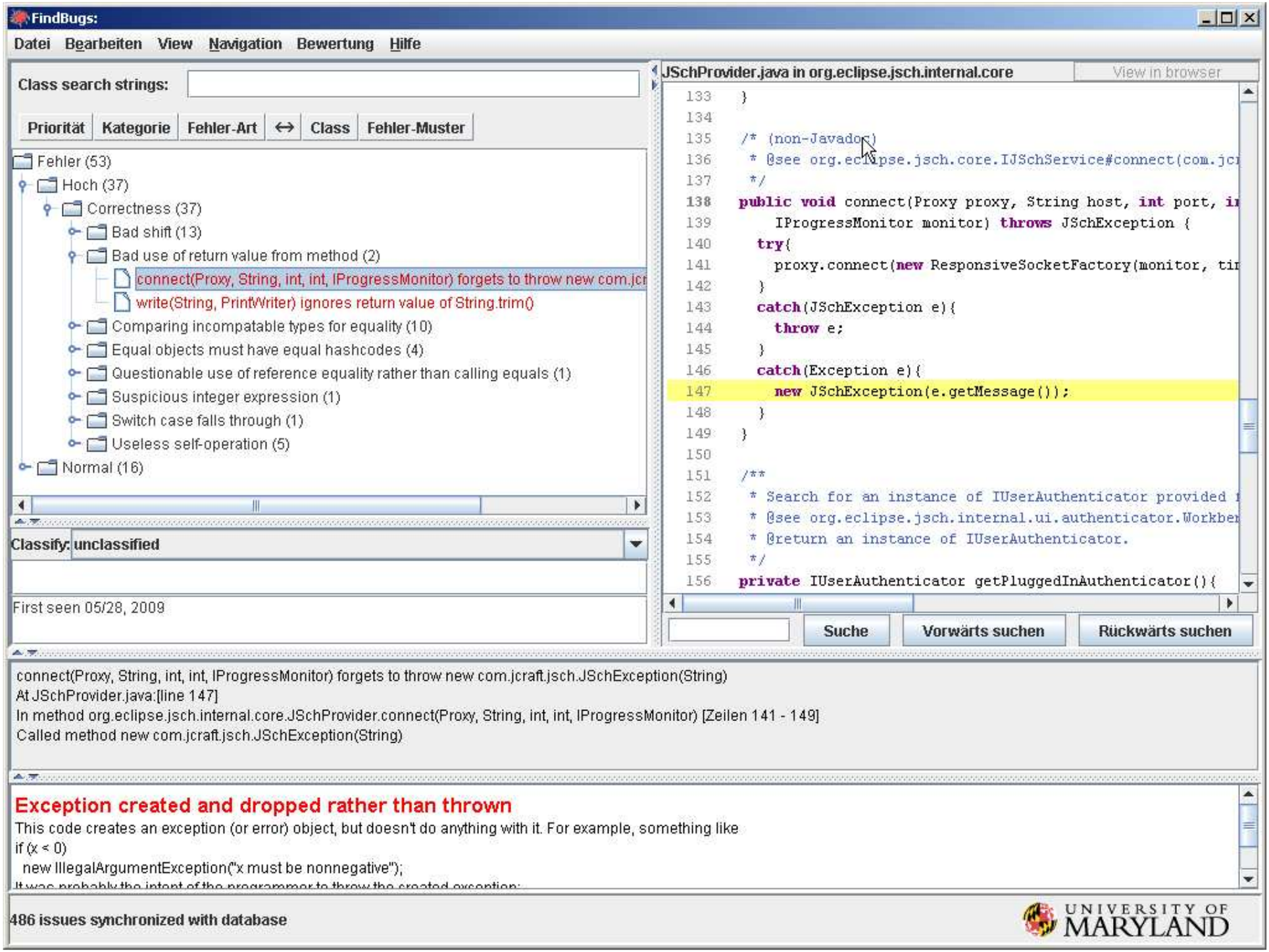
- Keep track when bugs were introduced/resolved
- Historical recorder

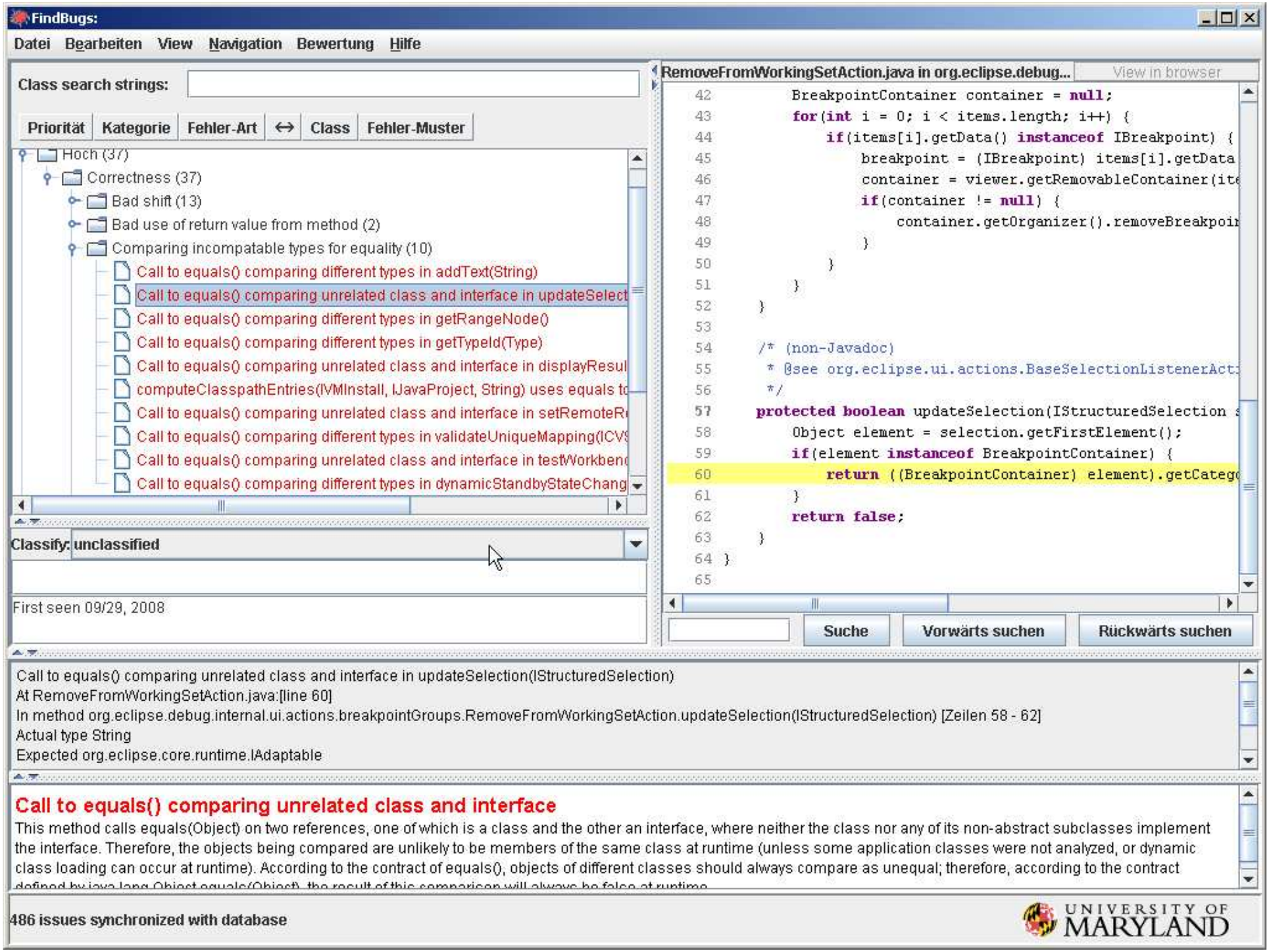
Cloud version

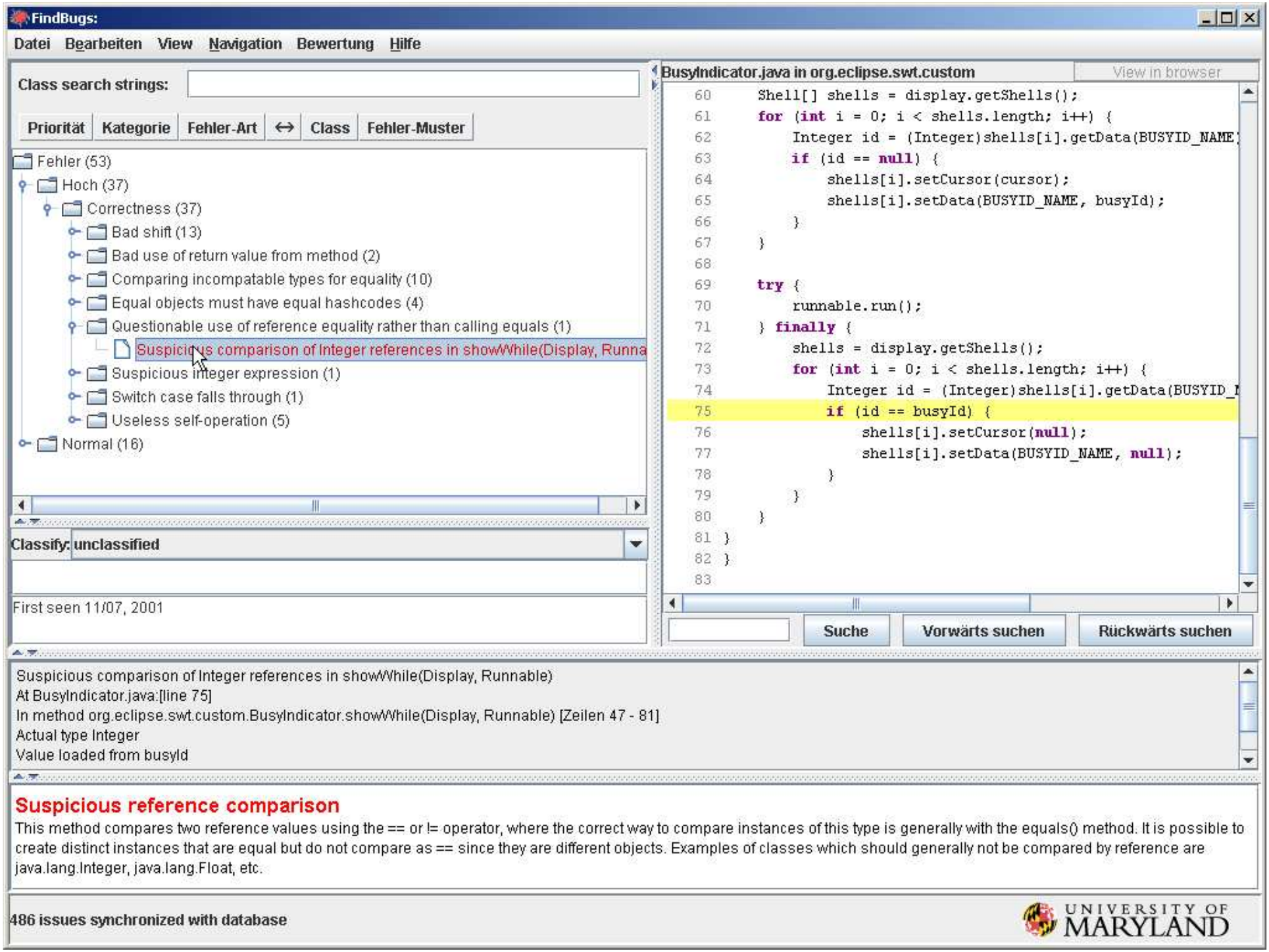
- Show this ☺

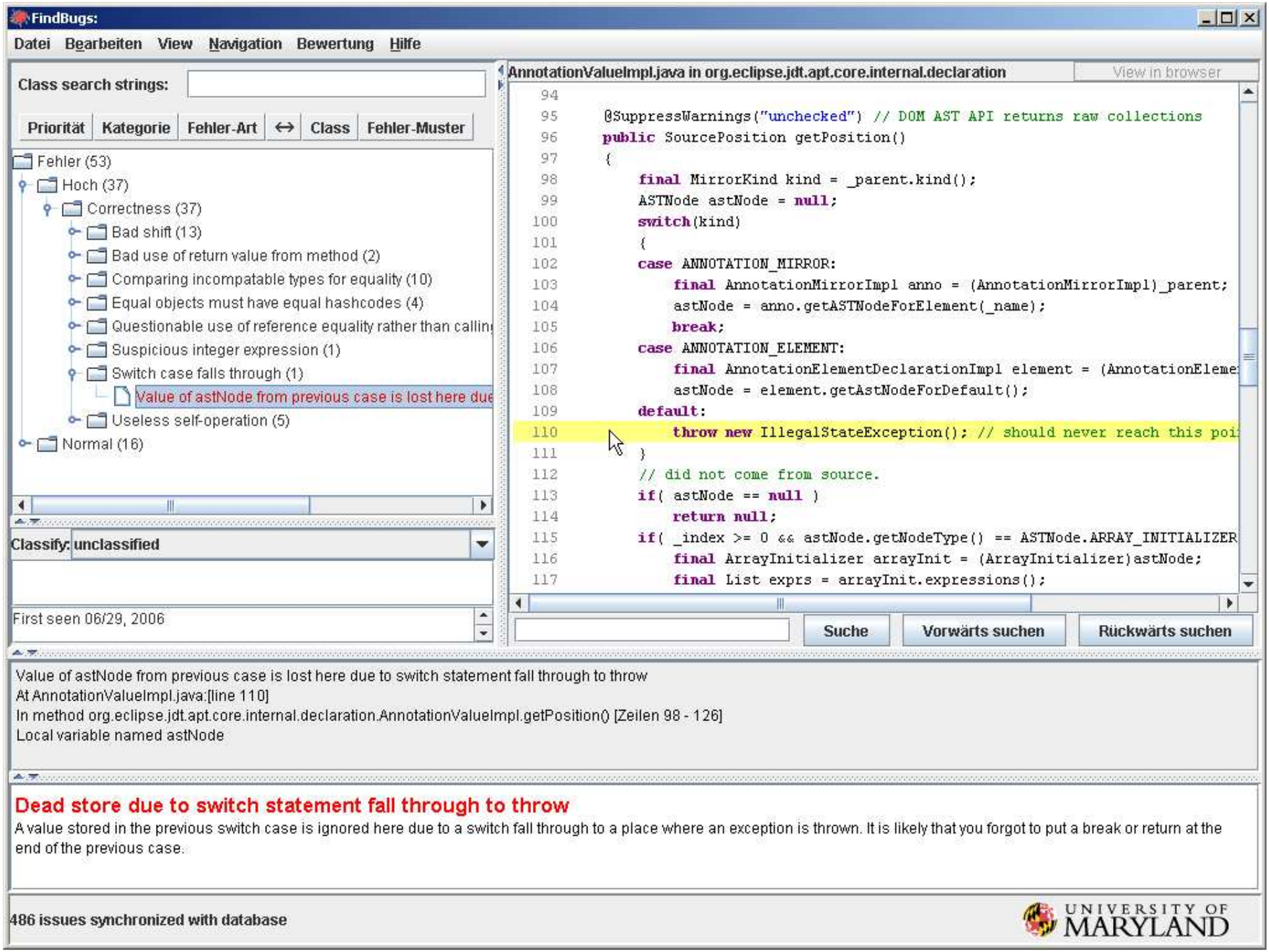
Roadmap ?

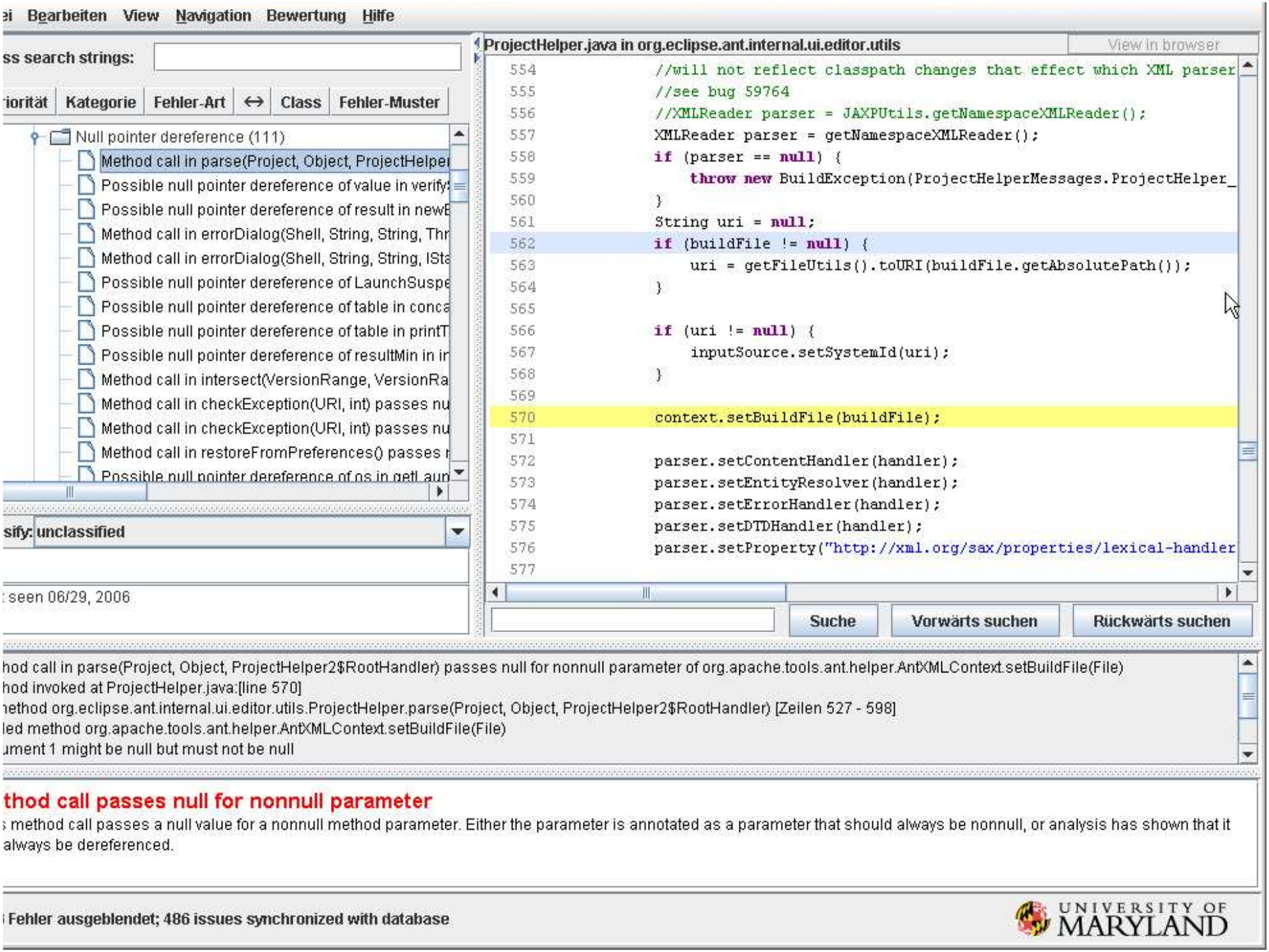
- Refactor findbugs itself (see next chapter)
- Client API, multicore analysis, cleanup BCEL, etc


FindBugs:
Datei Bearbeiten View Navigation Bewertung Hilfe
Class search strings:
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
Fehler (53)
Hoch (37)
Correctness (37)
Bad shift (13)
Bad use of return value from method (2)
connect(Proxy, String, int, int, IProgressMonitor) forgets to throw new com.jcr
write(String, PrintWriter) ignores return value of String.trim()
Comparing incompatable types for equality (10)
Equal objects must have equal hashcodes (4)
Questionable use of reference equality rather than calling equals (1)
Suspicious integer expression (1)
Switch case falls through (1)
Useless self-operation (5)
Normal (16)
Classify: unclassified
First seen 05/28, 2009
JSchProvider.java in org.eclipse.jsch.internal.core
View in browser
133 }
134
135 /* (non-Javadoc)
136 * @see org.eclipse.jsch.core.IJSchService#connect(com.jcr
137 */
138 public void connect(Proxy proxy, String host, int port, in
139 IProgressMonitor monitor) throws JSchException {
140 try{
141 proxy.connect(new ResponsiveSocketFactory(monitor, tim
142 }
143 catch(JSchException e){
144 throw e;
145 }
146 catch(Exception e){
147 new JSchException(e.getMessage());
148 }
149 }
150
151 /**
152 * Search for an instance of IUserAuthenticator provided b
153 * @see org.eclipse.jsch.internal.ui.authenticator.Workbe
154 * @return an instance of IUserAuthenticator.
155 */
156 private IUserAuthenticator getPluggedInAuthenticator(){
Suche
Vorwärts suchen
Rückwärts suchen
connect(Proxy, String, int, int, IProgressMonitor) forgets to throw new com.jcraft.jsch.JSchException(String)
At JSchProvider.java:[line 147]
In method org.eclipse.jsch.internal.core.JSchProvider.connect(Proxy, String, int, int, IProgressMonitor) [Zeilen 141 - 149]
Called method new com.jcraft.jsch.JSchException(String)
Exception created and dropped rather than thrown
This code creates an exception (or error) object, but doesn't do anything with it. For example, something like
if (x < 0)
new IllegalArgumentException("x must be nonnegative");
It was probably the intent of the programmer to throw the created exception;
486 issues synchronized with database
UNIVERSITY OF MARYLAND

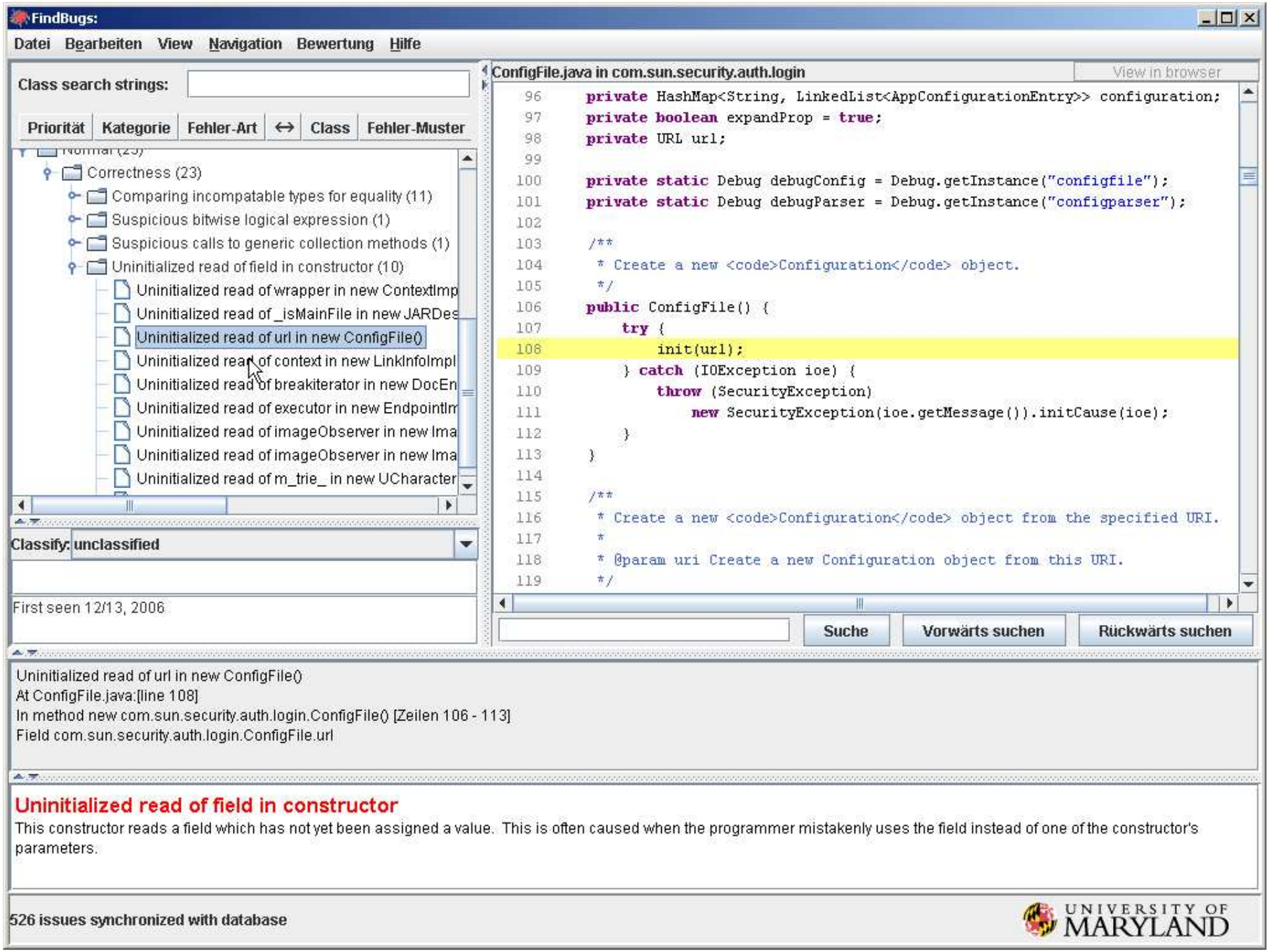

FindBugs:
Datei Bearbeiten View Navigation Bewertung Hilfe
Class search strings:
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
Hoch (37)
Correctness (37)
Bad shift (13)
Bad use of return value from method (2)
Comparing incompatable types for equality (10)
Call to equals() comparing different types in addText(String)
Call to equals() comparing unrelated class and interface in updateSelect
Call to equals() comparing different types in getRangeNode()
Call to equals() comparing different types in getTypeId(Type)
Call to equals() comparing unrelated class and interface in displayResul
computeClasspathEntries(IVMInstall, IJavaProject, String) uses equals t
Call to equals() comparing unrelated class and interface in setRemoteR
Call to equals() comparing different types in validateUniqueMapping(ICV
Call to equals() comparing unrelated class and interface in testWorkben
Call to equals() comparing different types in dynamicStandbyStateChang
Classify: unclassified
First seen 09/29, 2008
RemoveFromWorkingSetAction.java in org.eclipse.debug...
View in browser
42 BreakpointContainer container = null;
43 for(int i = 0; i < items.length; i++) {
44 if(items[i].getData() instanceof IBreakpoint) {
45 breakpoint = (IBreakpoint) items[i].getData
46 container = viewer.getRemovableContainer(ite
47 if(container != null) {
48 container.getOrganizer().removeBreakpoi
49 }
50 }
51 }
52 }
53
54 /* (non-Javadoc)
55 * @see org.eclipse.ui.actions.BaseSelectionListenerAct
56 */
57 protected boolean updateSelection(IStructuredSelection
58 Object element = selection.getFirstElement();
59 if(element instanceof BreakpointContainer) {
60 return ((BreakpointContainer) element).getCateg
61 }
62 return false;
63 }
64 }
65
Suche
Vorwärts suchen
Rückwärts suchen
Call to equals() comparing unrelated class and interface in updateSelection(IStructuredSelection)
At RemoveFromWorkingSetAction.java:[line 60]
In method org.eclipse.debug.internal.ui.actions.breakpointGroups.RemoveFromWorkingSetAction.updateSelection(IStructuredSelection) [Zeilen 58 - 62]
Actual type String
Expected org.eclipse.core.runtime.IAdaptable
Call to equals() comparing unrelated class and interface
This method calls equals(Object) on two references, one of which is a class and the other an interface, where neither the class nor any of its non-abstract subclasses implement the interface. Therefore, the objects being compared are unlikely to be members of the same class at runtime (unless some application classes were not analyzed, or dynamic class loading can occur at runtime). According to the contract of equals(), objects of different classes should always compare as unequal; therefore, according to the contract
486 issues synchronized with database
UNIVERSITY OF MARYLAND

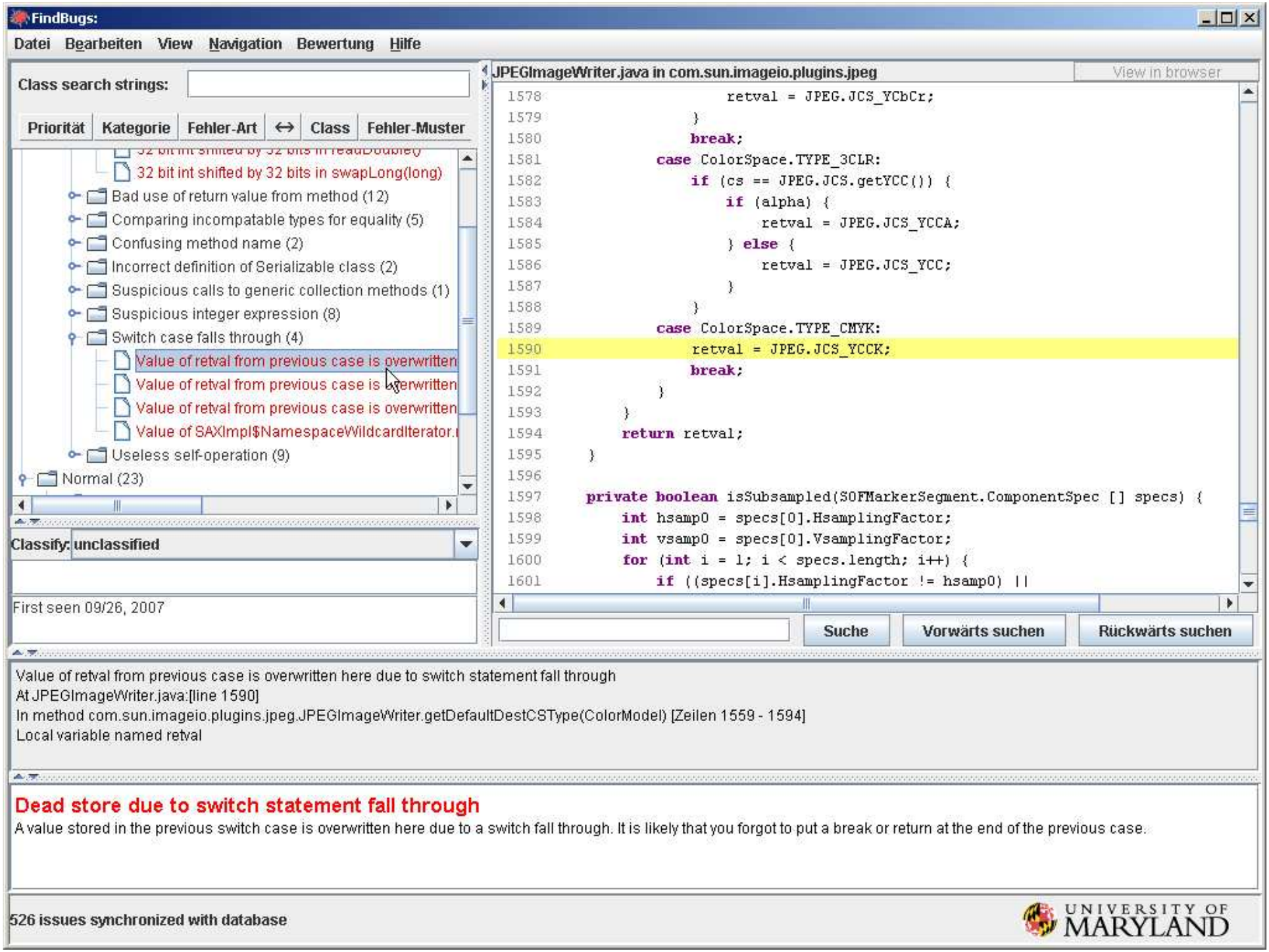

FindBugs:
Datei Bearbeiten View Navigation Bewertung Hilfe
Class search strings:
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
Fehler (53)
Hoch (37)
Correctness (37)
Bad shift (13)
Bad use of return value from method (2)
Comparing incompatable types for equality (10)
Equal objects must have equal hashcodes (4)
Questionable use of reference equality rather than calling equals (1)
Suspicious comparison of Integer references in showWhile(Display, Runna
Suspicious integer expression (1)
Switch case falls through (1)
Useless self-operation (5)
Normal (16)
Classify: unclassified
First seen 11/07, 2001
BusyIndicator.java in org.eclipse.swt.custom
View in browser
60 Shell[] shells = display.getShells();
61 for (int i = 0; i < shells.length; i++) {
62 Integer id = (Integer)shells[i].getData(BUSYID_NAME
63 if (id == null) {
64 shells[i].setCursor(cursor);
65 shells[i].setData(BUSYID_NAME, busyId);
66 }
67 }
68
69 try {
70 runnable.run();
71 } finally {
72 shells = display.getShells();
73 for (int i = 0; i < shells.length; i++) {
74 Integer id = (Integer)shells[i].getData(BUSYID_
75 if (id == busyId) {
76 shells[i].setCursor(null);
77 shells[i].setData(BUSYID_NAME, null);
78 }
79 }
80 }
81 }
82 }
83
Suche
Vorwärts suchen
Rückwärts suchen
Suspicious comparison of Integer references in showWhile(Display, Runnable)
At BusyIndicator.java:[line 75]
In method org.eclipse.swt.custom.BusyIndicator.showWhile(Display, Runnable) [Zeilen 47 - 81]
Actual type Integer
Value loaded from busyId
Suspicious reference comparison
This method compares two reference values using the == or != operator, where the correct way to compare instances of this type is generally with the equals() method. It is possible to create distinct instances that are equal but do not compare as == since they are different objects. Examples of classes which should generally not be compared by reference are java.lang.Integer, java.lang.Float, etc.
486 issues synchronized with database
UNIVERSITY OF MARYLAND

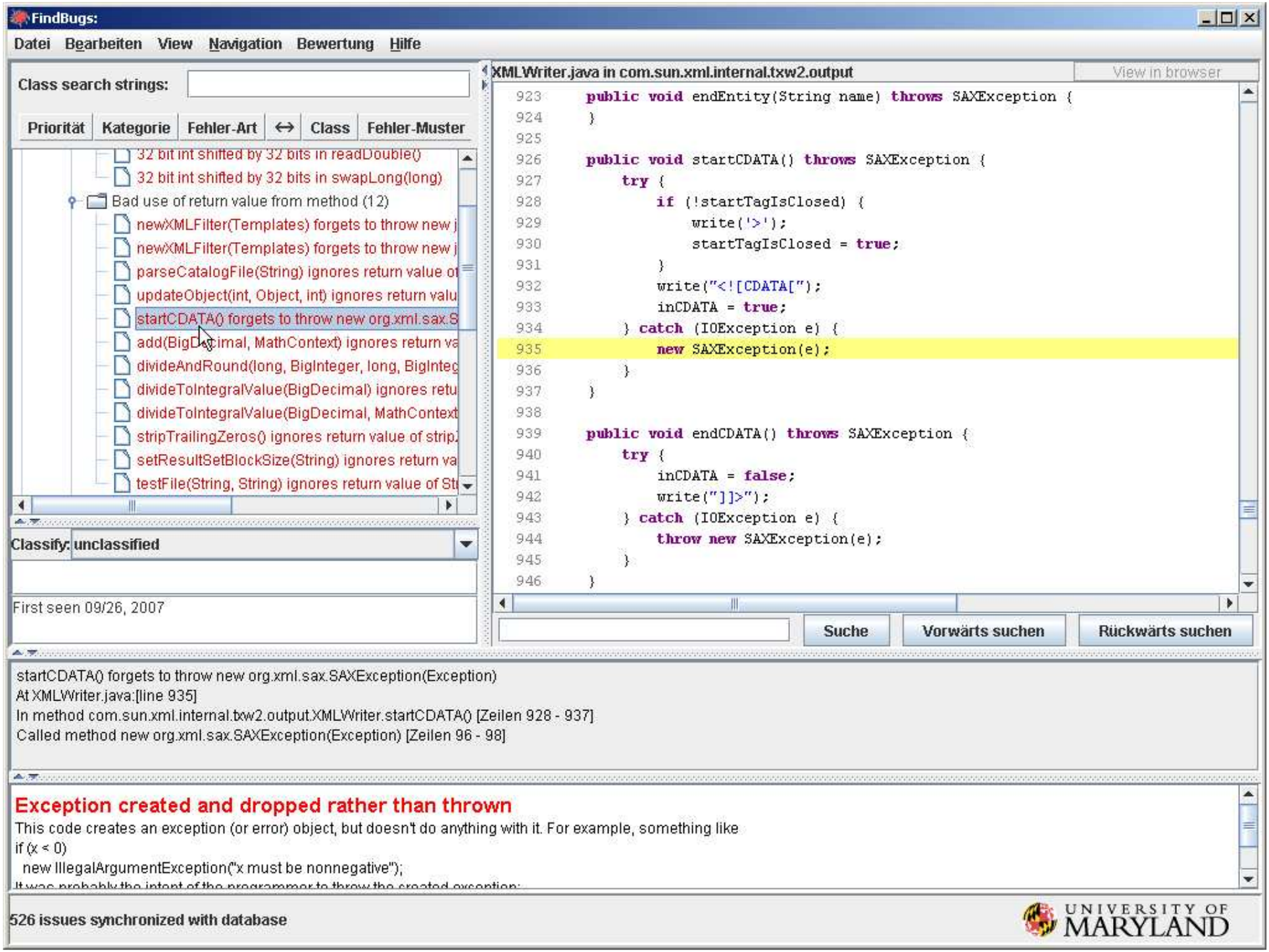

FindBugs:
Datei Bearbeiten View Navigation Bewertung Hilfe
Class search strings:
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
Fehler (53)
Hoch (37)
Correctness (37)
Bad shift (13)
Bad use of return value from method (2)
Comparing incompatable types for equality (10)
Equal objects must have equal hashcodes (4)
Questionable use of reference equality rather than calling
Suspicious integer expression (1)
Switch case falls through (1)
Value of astNode from previous case is lost here due
Useless self-operation (5)
Normal (16)
Classify: unclassified
First seen 06/29, 2006
AnnotationValueImpl.java in org.eclipse.jdt.apt.core.internal.declaration
View in browser
94
95 @SuppressWarnings("unchecked") // DOM AST API returns raw collections
96 public SourcePosition getPosition()
97 {
98 final MirrorKind kind = _parent.kind();
99 ASTNode astNode = null;
100 switch(kind)
101 {
102 case ANNOTATION_MIRROR:
103 final AnnotationMirrorImpl anno = (AnnotationMirrorImpl)_parent;
104 astNode = anno.getASTNodeForElement(_name);
105 break;
106 case ANNOTATION_ELEMENT:
107 final AnnotationElementDeclarationImpl element = (AnnotationEleme
108 astNode = element.getAstNodeForDefault();
109 default:
110 throw new IllegalStateException(); // should never reach this poi
111 }
112 // did not come from source.
113 if(astNode == null)
114 return null;
115 if(_index >= 0 && astNode.getNodeType() == ASTNode.ARRAY_INITIALIZER
116 final ArrayInitializer arrayInit = (ArrayInitializer)astNode;
117 final List exprs = arrayInit.expressions();
Suche
Vorwärts suchen
Rückwärts suchen
Value of astNode from previous case is lost here due to switch statement fall through to throw
At AnnotationValueImpl.java:[line 110]
In method org.eclipse.jdt.apt.core.internal.declaration.AnnotationValueImpl.getPosition() [Zeilen 98 - 126]
Local variable named astNode
Dead store due to switch statement fall through to throw
A value stored in the previous switch case is ignored here due to a switch fall through to a place where an exception is thrown. It is likely that you forgot to put a break or return at the end of the previous case.
486 issues synchronized with database
UNIVERSITY OF MARYLAND

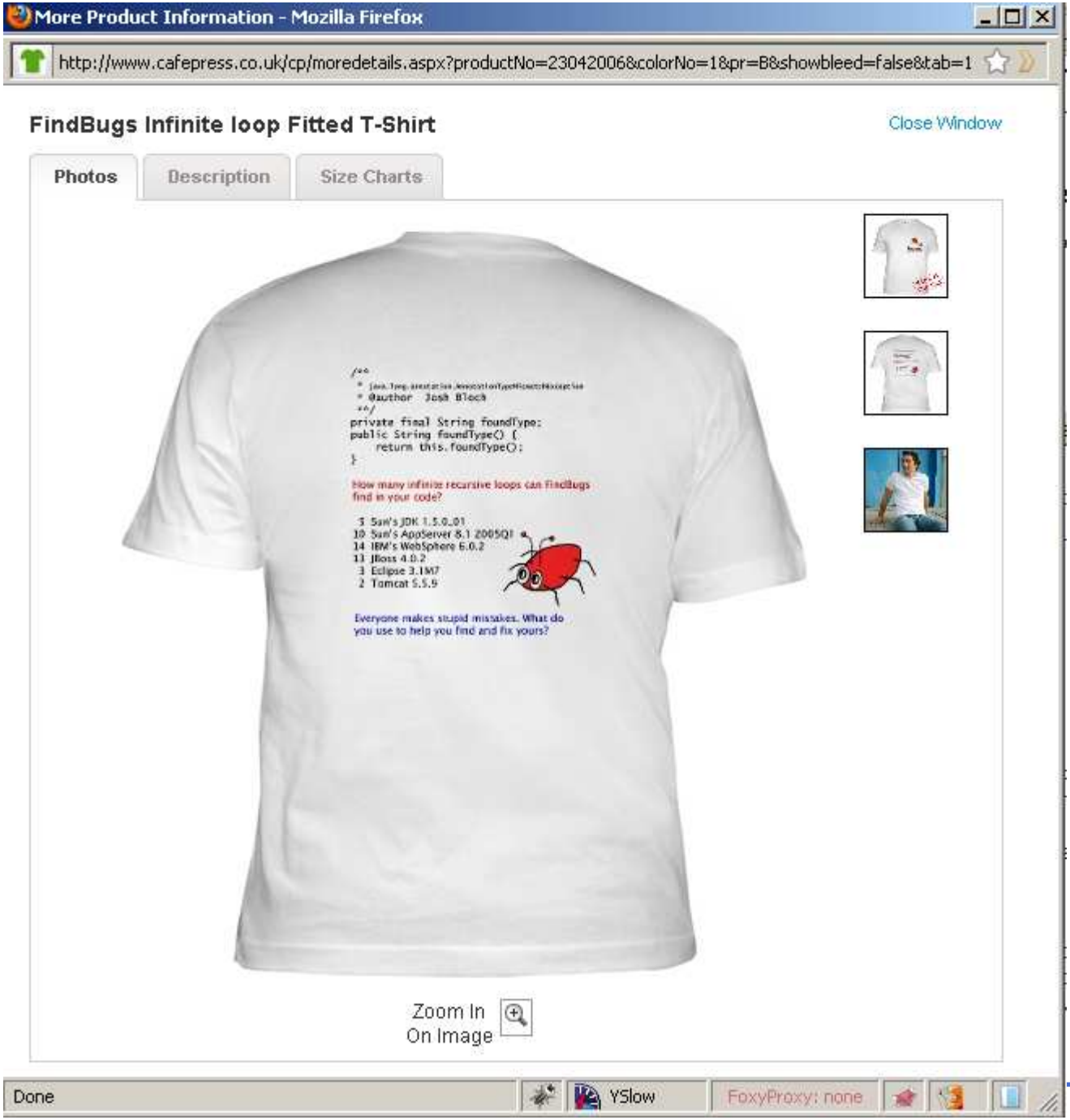

ei Bearbeiten View Navigation Bewertung Hilfe
ss search strings:
iorität Kategorie Fehler-Art ↔ Class Fehler-Muster
Null pointer dereference (111)
Method call in parse(Project, Object, ProjectHelper
Possible null pointer dereference of value in verify
Possible null pointer dereference of result in newB
Method call in errorDialog(Shell, String, String, Thr
Method call in errorDialog(Shell, String, String, ISta
Possible null pointer dereference of LaunchSuspe
Possible null pointer dereference of table in conca
Possible null pointer dereference of table in printT
Possible null pointer dereference of resultMin in ir
Method call in intersect(VersionRange, VersionRa
Method call in checkException(URI, int) passes nu
Method call in checkException(URI, int) passes nu
Method call in restoreFromPreferences() passes r
Possible null pointer dereference of os in getLaun
sify: unclassified
seen 06/29, 2006
ProjectHelper.java in org.eclipse.ant.internal.ui.editor.utils
View in browser
554 //will not reflect classpath changes that effect which XML parser
555 //see bug 59764
556 //XMLReader parser = JAXPUtils.getNamespaceXMLReader();
557 XMLReader parser = getNamespaceXMLReader();
558 if (parser == null) {
559 throw new BuildException(ProjectHelperMessages.ProjectHelper_
560 }
561 String uri = null;
562 if (buildFile != null) {
563 uri = getFileUtils().toURI(buildFile.getAbsolutePath());
564 }
565
566 if (uri != null) {
567 inputSource.setSystemId(uri);
568 }
569
570 context.setBuildFile(buildFile);
571
572 parser.setContentHandler(handler);
573 parser.setEntityResolver(handler);
574 parser.setErrorHandler(handler);
575 parser.setDTDHandler(handler);
576 parser.setProperty("http://xml.org/sax/properties/lexical-handler
577
Suche
Vorwärts suchen
Rückwärts suchen
hod call in parse(Project, Object, ProjectHelper2$RootHandler) passes null for nonnull parameter of org.apache.tools.ant.helper.AntXMLContext.setBuildFile(File)
hod invoked at ProjectHelper.java:[line 570]
ethod org.eclipse.ant.internal.ui.editor.utils.ProjectHelper.parse(Project, Object, ProjectHelper2$RootHandler) [Zeilen 527 - 598]
led method org.apache.tools.ant.helper.AntXMLContext.setBuildFile(File)
ument 1 might be null but must not be null
thod call passes null for nonnull parameter
s method call passes a null value for a nonnull method parameter. Either the parameter is annotated as a parameter that should always be nonnull, or analysis has shown that it always be dereferenced.
Fehler ausgeblendet; 486 issues synchronized with database
UNIVERSITY OF MARYLAND

Findbugs, JDK


FindBugs:
Datei Bearbeiten View Navigation Bewertung Hilfe
Class search strings:
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
Correctness (23)
Comparing incompatable types for equality (11)
Suspicious bitwise logical expression (1)
Suspicious calls to generic collection methods (1)
Uninitialized read of field in constructor (10)
Uninitialized read of wrapper in new ContextImp
Uninitialized read of _isMainFile in new JARDes
Uninitialized read of url in new ConfigFile()
Uninitialized read of context in new LinkInfoImpl
Uninitialized read of breakiterator in new DocEn
Uninitialized read of executor in new EndpointIm
Uninitialized read of imageObserver in new Ima
Uninitialized read of imageObserver in new Ima
Uninitialized read of m_trie_ in new UCharacter
Classify: unclassified
First seen 12/13, 2006
ConfigFile.java in com.sun.security.auth.login
View in browser
96 private HashMap<String, LinkedList<AppConfigurationEntry>> configuration;
97 private boolean expandProp = true;
98 private URL url;
99
100 private static Debug debugConfig = Debug.getInstance("configfile");
101 private static Debug debugParser = Debug.getInstance("configparser");
102
103 /**
104 * Create a new <code>Configuration</code> object.
105 */
106 public ConfigFile() {
107 try {
108 init(url);
109 } catch (IOException ioe) {
110 throw (SecurityException)
111 new SecurityException(ioe.getMessage()).initCause(ioe);
112 }
113 }
114
115 /**
116 * Create a new <code>Configuration</code> object from the specified URI.
117 *
118 * @param uri Create a new Configuration object from this URI.
119 */
Suche
Vorwärts suchen
Rückwärts suchen
Uninitialized read of url in new ConfigFile()
At ConfigFile.java:[line 108]
In method new com.sun.security.auth.login.ConfigFile() [Zeilen 106 - 113]
Field com.sun.security.auth.login.ConfigFile.url
Uninitialized read of field in constructor
This constructor reads a field which has not yet been assigned a value. This is often caused when the programmer mistakenly uses the field instead of one of the constructor's parameters.
526 issues synchronized with database
UNIVERSITY OF MARYLAND


FindBugs:
Datei Bearbeiten View Navigation Bewertung Hilfe
Class search strings:
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
32 bit int shifted by 32 bits in swapLong(long)
Bad use of return value from method (12)
Comparing incompatable types for equality (5)
Confusing method name (2)
Incorrect definition of Serializable class (2)
Suspicious calls to generic collection methods (1)
Suspicious integer expression (8)
Switch case falls through (4)
Value of retval from previous case is overwritten
Value of retval from previous case is overwritten
Value of retval from previous case is overwritten
Value of SAXImpl$NamespaceWildcardIterator.
Useless self-operation (9)
Normal (23)
Classify: unclassified
First seen 09/26, 2007
JPEGImageWriter.java in com.sun.imageio.plugins.jpeg
View in browser
1578 retval = JPEG.JCS_YCbCr;
1579 }
1580 break;
1581 case ColorSpace.TYPE_3CLR:
1582 if (cs == JPEG.JCS.getYCC()) {
1583 if (alpha) {
1584 retval = JPEG.JCS_YCCA;
1585 } else {
1586 retval = JPEG.JCS_YCC;
1587 }
1588 }
1589 case ColorSpace.TYPE_CMYK:
1590 retval = JPEG.JCS_YCCK;
1591 break;
1592 }
1593 }
1594 return retval;
1595 }
1596
1597 private boolean isSubsampled(SOFMarkerSegment.ComponentSpec [] specs) {
1598 int hsamp0 = specs[0].HsamplingFactor;
1599 int vsamp0 = specs[0].VsamplingFactor;
1600 for (int i = 1; i < specs.length; i++) {
1601 if ((specs[i].HsamplingFactor != hsamp0) ||
Suche
Vorwärts suchen
Rückwärts suchen
Value of retval from previous case is overwritten here due to switch statement fall through
At JPEGImageWriter.java:[line 1590]
In method com.sun.imageio.plugins.jpeg.JPEGImageWriter.getDefaultDestCSType(ColorModel) [Zeilen 1559 - 1594]
Local variable named retval
Dead store due to switch statement fall through
A value stored in the previous switch case is overwritten here due to a switch fall through. It is likely that you forgot to put a break or return at the end of the previous case.
526 issues synchronized with database
UNIVERSITY OF MARYLAND


FindBugs:
Datei Bearbeiten View Navigation Bewertung Hilfe
Class search strings:
Priorität Kategorie Fehler-Art ↔ Class Fehler-Muster
32 bit int shifted by 32 bits in readDouble()
32 bit int shifted by 32 bits in swapLong(long)
Bad use of return value from method (12)
newXMLFilter(Templates) forgets to throw new j
newXMLFilter(Templates) forgets to throw new j
parseCatalogFile(String) ignores return value of
updateObject(int, Object, int) ignores return valu
startCDATA() forgets to throw new org.xml.sax.S
add(BigDecimal, MathContext) ignores return va
divideAndRound(long, BigInteger, long, BigInteg
divideToIntegralValue(BigDecimal) ignores retu
divideToIntegralValue(BigDecimal, MathContext
stripTrailingZeros() ignores return value of strip
setResultSetBlockSize(String) ignores return va
testFile(String, String) ignores return value of St
Classify: unclassified
First seen 09/26, 2007
XMLWriter.java in com.sun.xml.internal.txw2.output
View in browser
923 public void endEntity(String name) throws SAXException {
924 }
925
926 public void startCDATA() throws SAXException {
927 try {
928 if (!startTagIsClosed) {
929 write('>');
930 startTagIsClosed = true;
931 }
932 write("<![CDATA[");
933 inCDATA = true;
934 } catch (IOException e) {
935 new SAXException(e);
936 }
937 }
938
939 public void endCDATA() throws SAXException {
940 try {
941 inCDATA = false;
942 write("]]>");
943 } catch (IOException e) {
944 throw new SAXException(e);
945 }
946 }
Suche
Vorwärts suchen
Rückwärts suchen
startCDATA() forgets to throw new org.xml.sax.SAXException(Exception)
At XMLWriter.java:[line 935]
In method com.sun.xml.internal.txw2.output.XMLWriter.startCDATA() [Zeilen 928 - 937]
Called method new org.xml.sax.SAXException(Exception) [Zeilen 96 - 98]
Exception created and dropped rather than thrown
This code creates an exception (or error) object, but doesn't do anything with it. For example, something like
if (x < 0)
new IllegalArgumentException("x must be nonnegative");
526 issues synchronized with database
UNIVERSITY OF MARYLAND


More Product Information - Mozilla Firefox
http://www.cafepress.co.uk/cp/moredetails.aspx?productNo=23042006&colorNo=1&pr=B&showbleed=false&tab=1
FindBugs Infinite loop Fitted T-Shirt
Close Window
Photos
Description
Size Charts
/**
* @author Josh Bloch
**/
private final String foundType;
public String foundType() {
return this.foundType();
}
How many infinite recursive loops can FindBugs find in your code?
3 Sun's JDK 1.5.0_01
10 Sun's AppServer 8.1 2005Q1
14 IBM's WebSphere 6.0.2
11 JBoss 4.0.2
3 Eclipse 3.1M7
2 Tomcat 5.5.9
Everyone makes stupid mistakes. What do you use to help you find and fix yours?
Zoom In On Image
Done
YSlow
FoxyProxy: none

Software-Architecture: Definitions

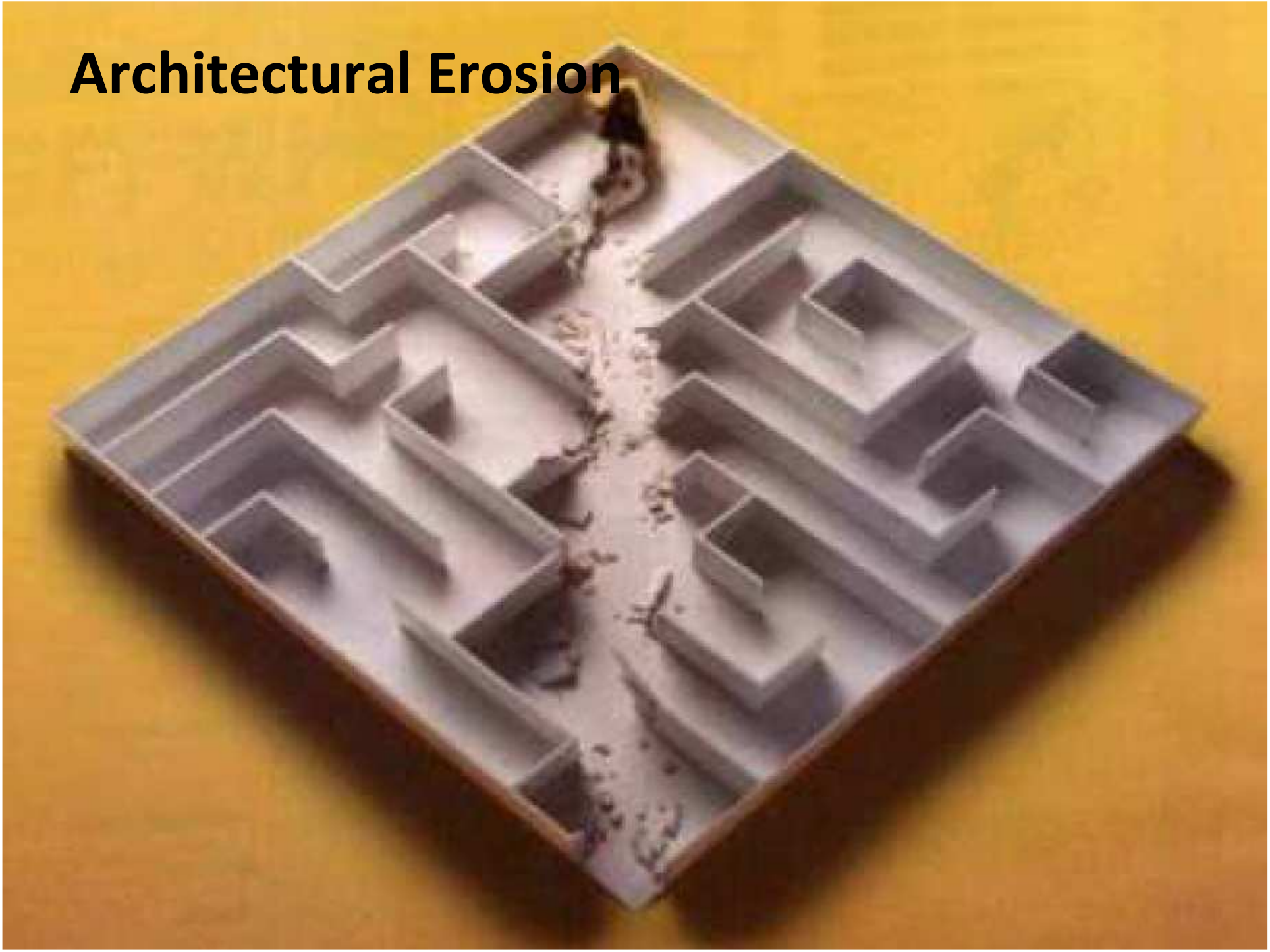
IEEE 1471-2000:

- The fundamental organization of a system,
- embodied in its components,
- their relationship to each other and the environment,
- and the principles governing its design and evolution.

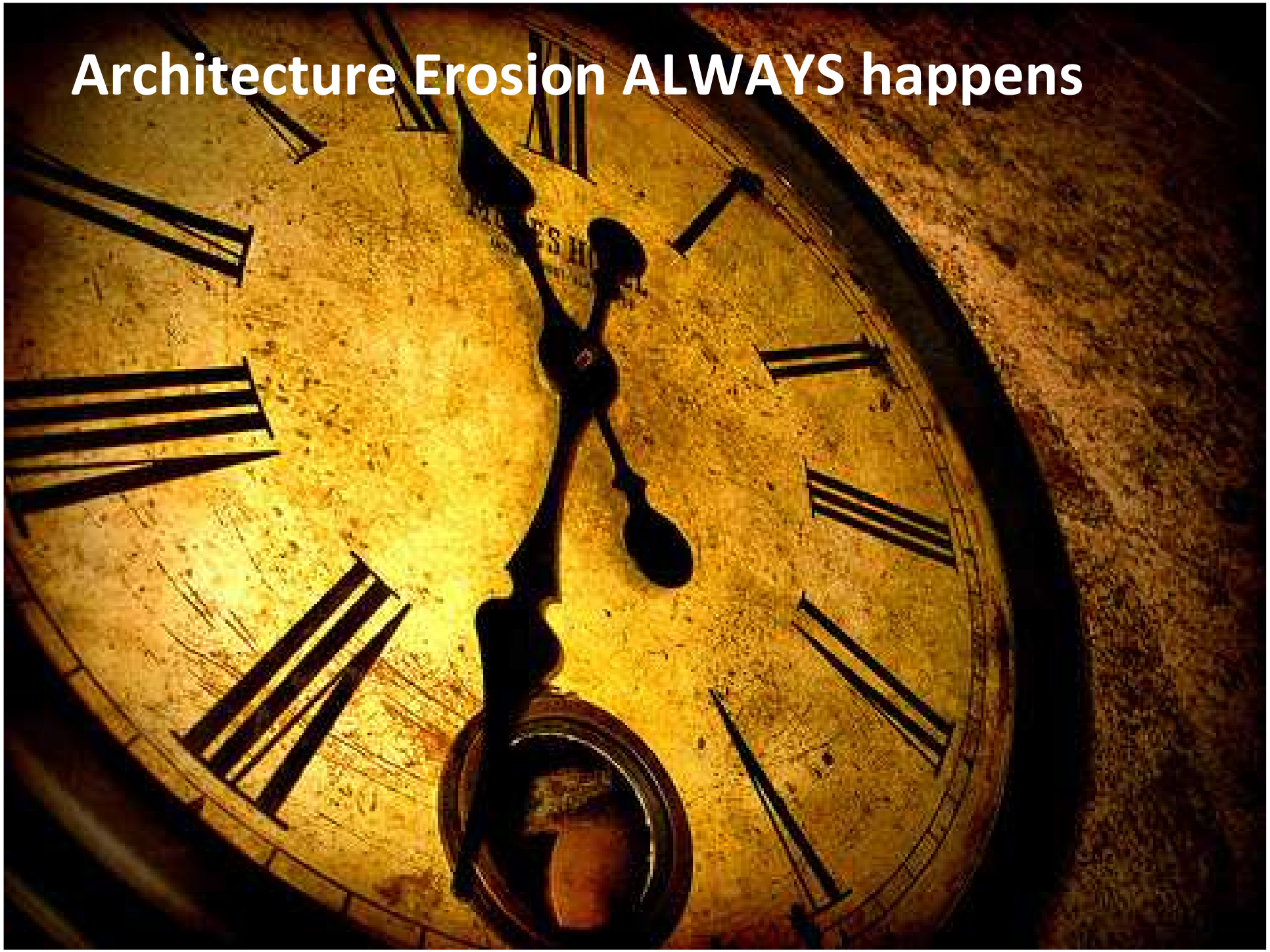
Kruchten: captured in two documents:

- *Software Architecture Document*
- *Software Design Guidelines*

Architectural Erosion



Architecture Erosion ALWAYS happens



Architecture Erosion ALWAYS happens

- **Time pressure** leads to abbreviations,
- **Prototypes** become products
- **Hacks**. Architectural restrictions are ignored
- **Lack of understanding the “should architecture”** leading to extensions and modification which obey only micro- and macro architecture. (e.g. outsourcing)
- **impossible to detect manually** architectural violations in a nontrivilal software system

Architectural Erosion

“Sometimes the developers manage to maintain this purity of design through the initial development and into the first release.

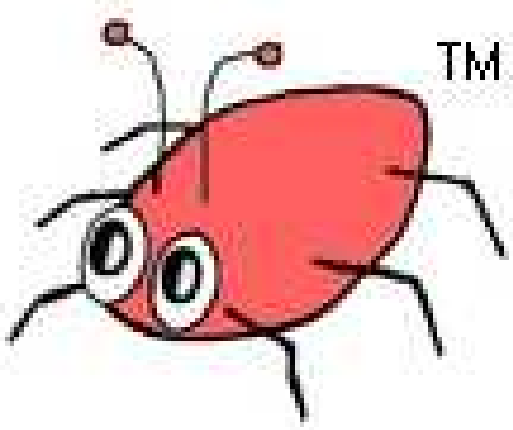
More often something goes wrong.The software starts to rot like a piece of bad meat”.

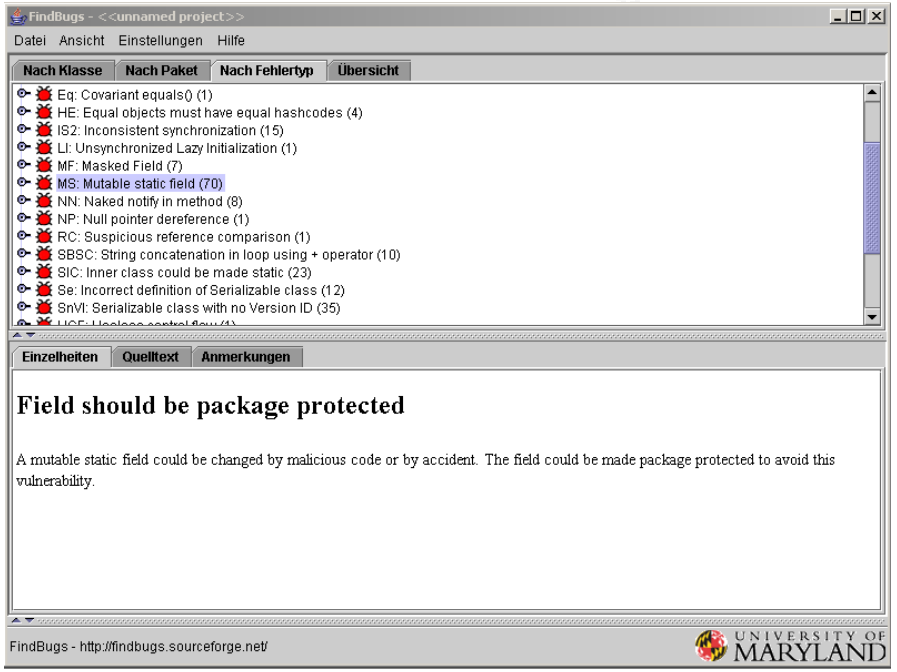
Uncle Bob: “Agile Software Development”

Aim of Architectural Analysis

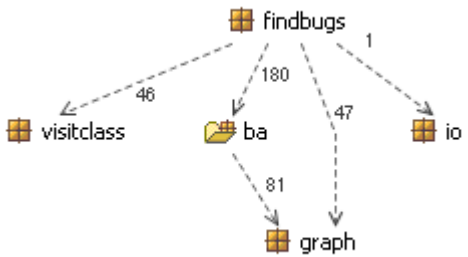
-Findbugs

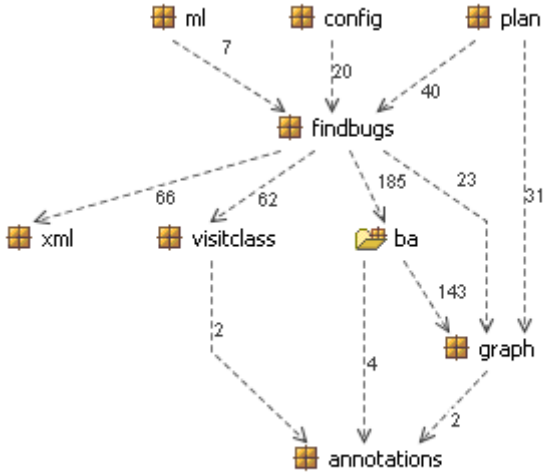
- do codelevel lints care about architectur
- Is there a architecture ?
- Is there a erosion ?
- Do AA-Tool work well ?





Findbugs: the first years

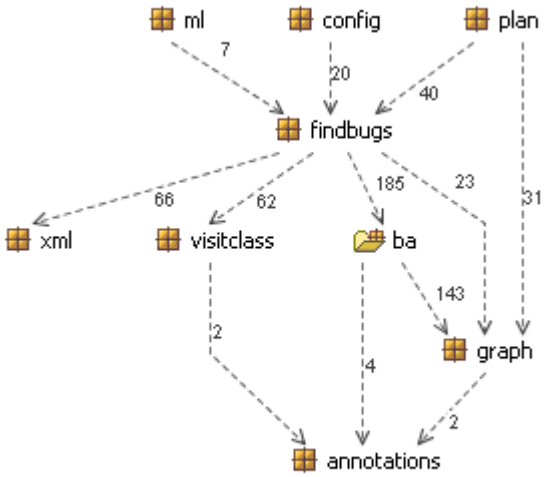




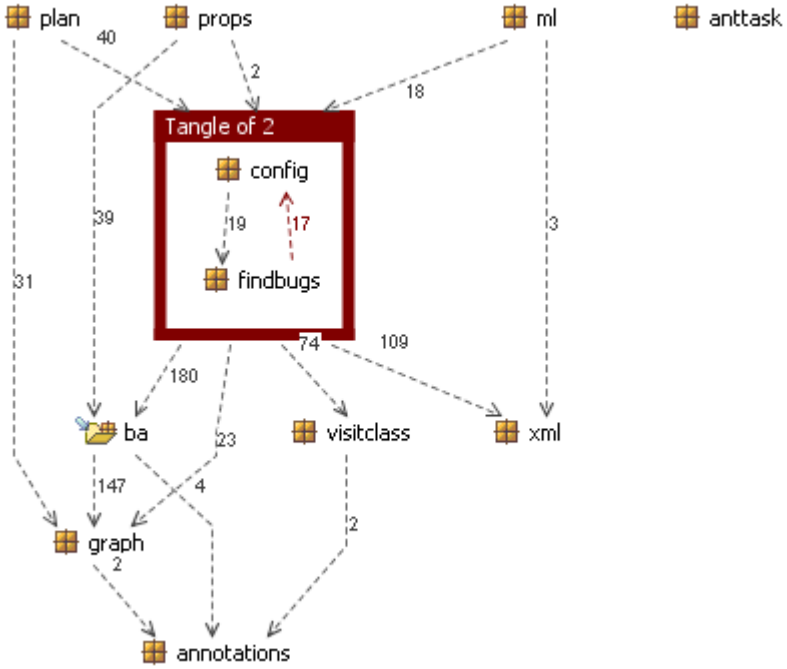
0.7.2
(03/2004)

0.8.6
(10/2004)

Findbugs: the first years

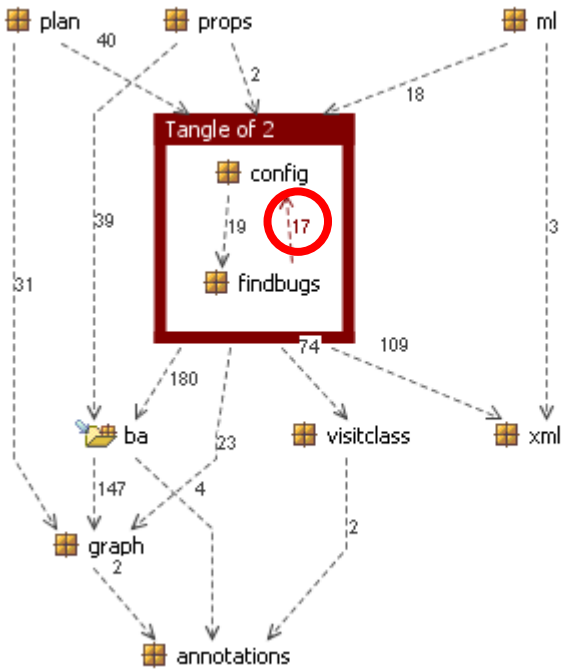


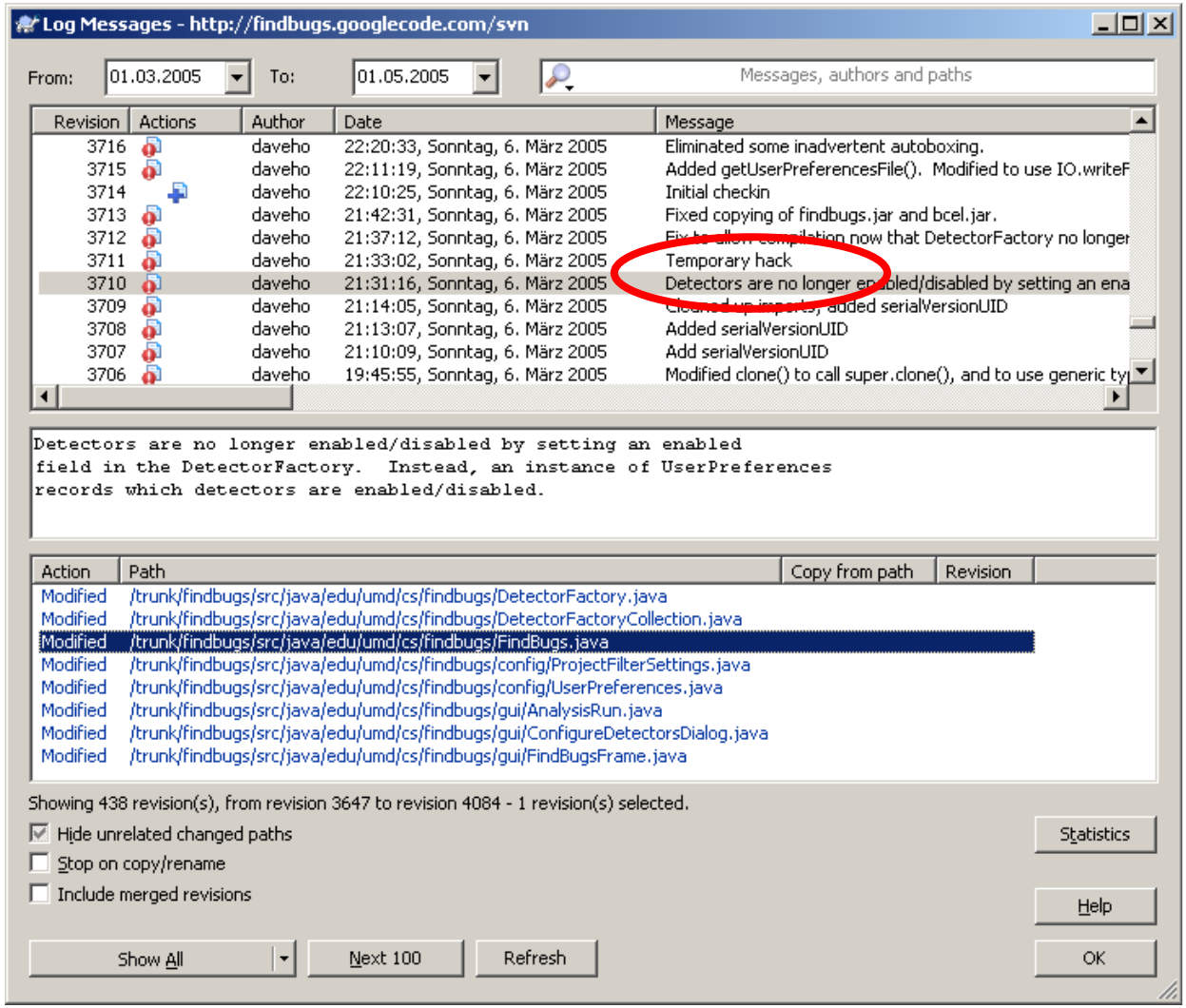
0.8.6
(10/2004)



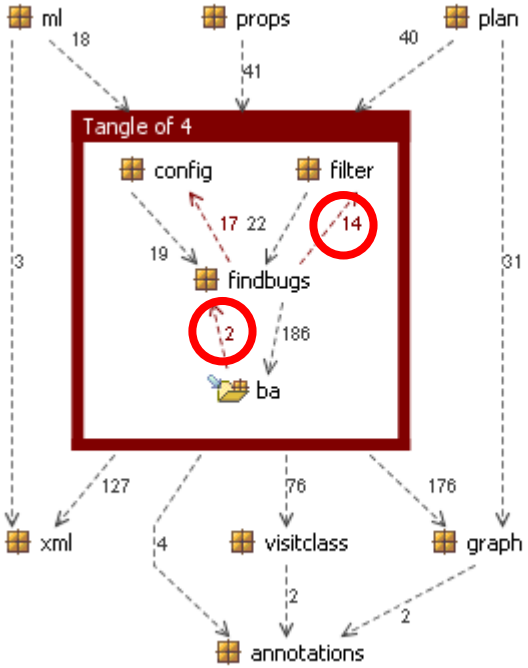
0.8.7
(05/2005)

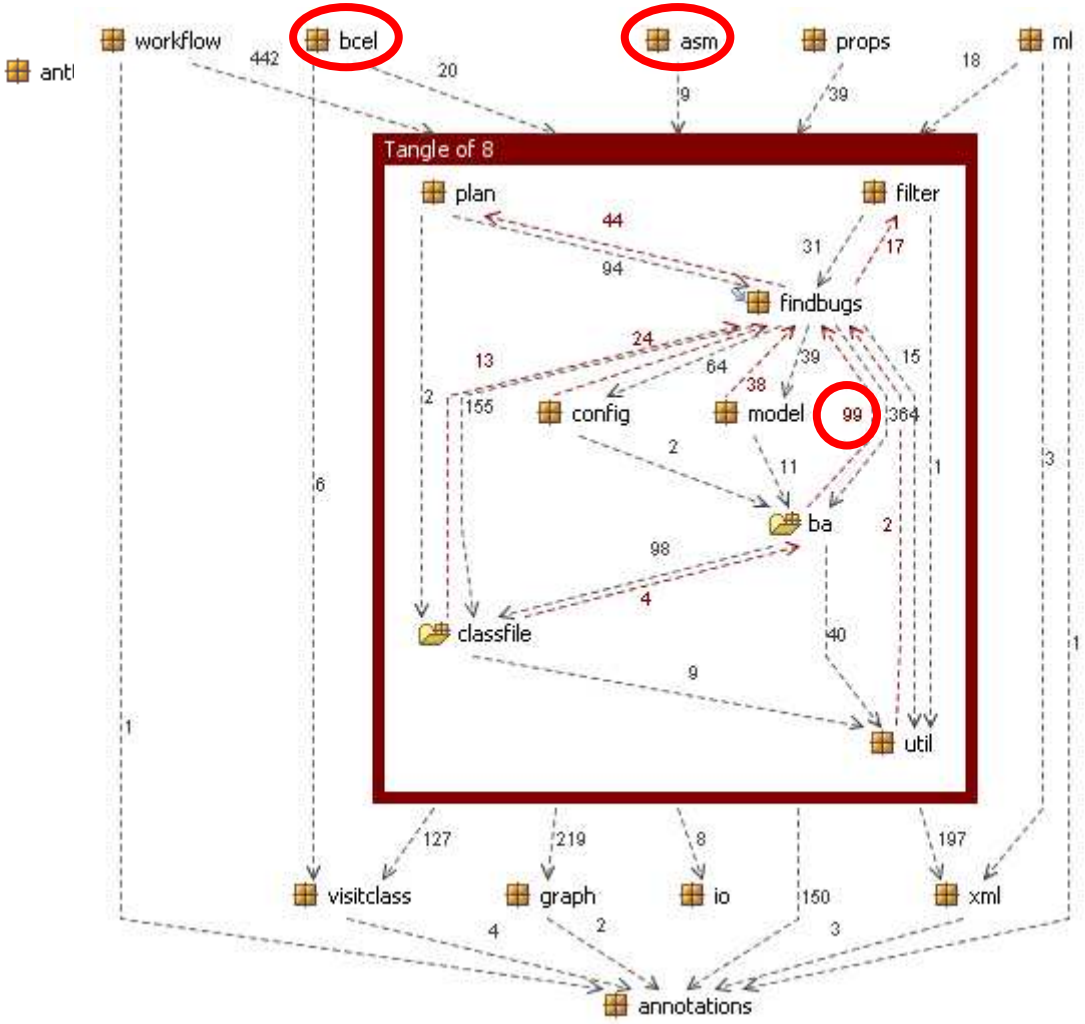
Findbugs 0.8.7: Architectural Analysis





Wellcome crappy architecture…

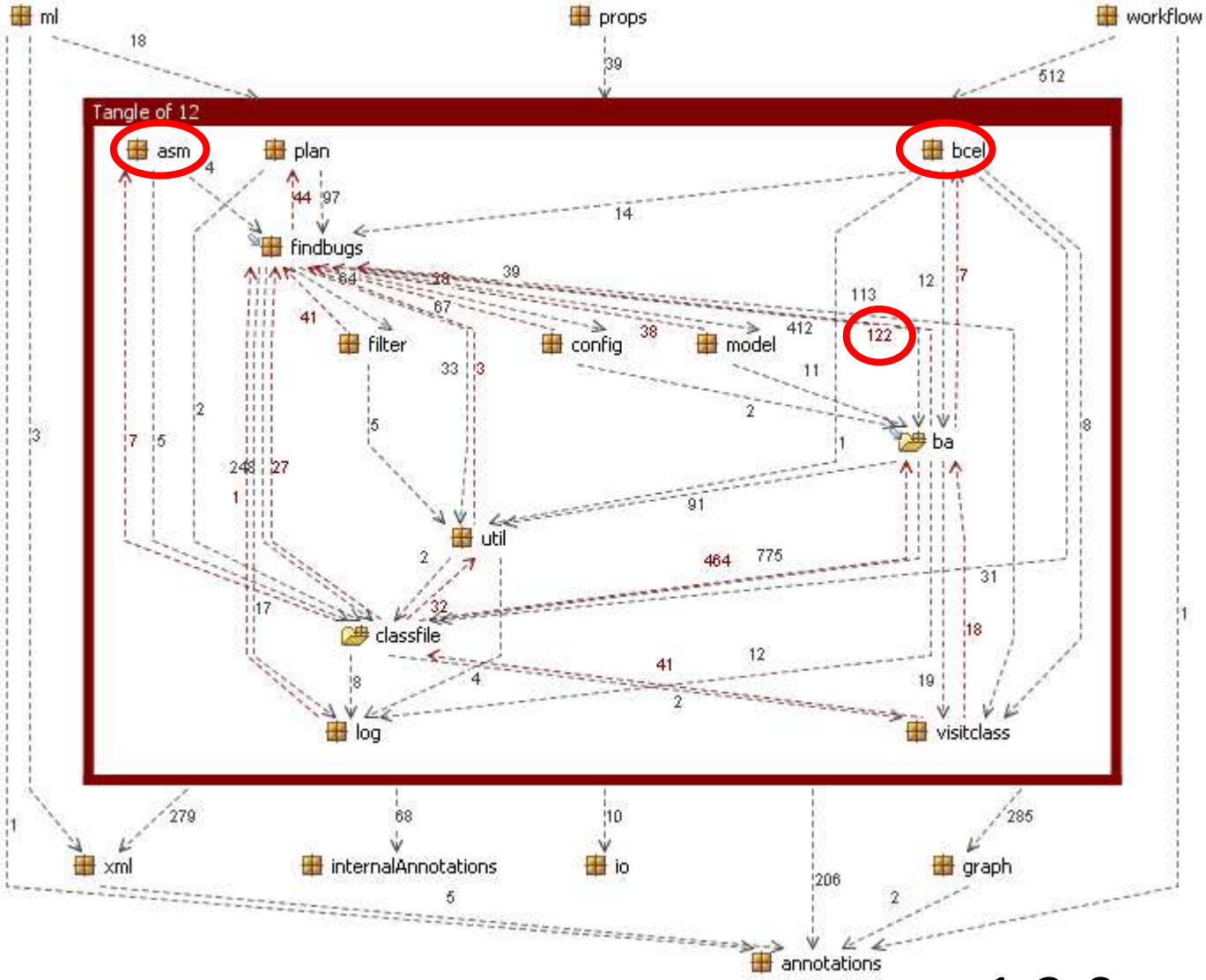




0.8.8
(05/2005)

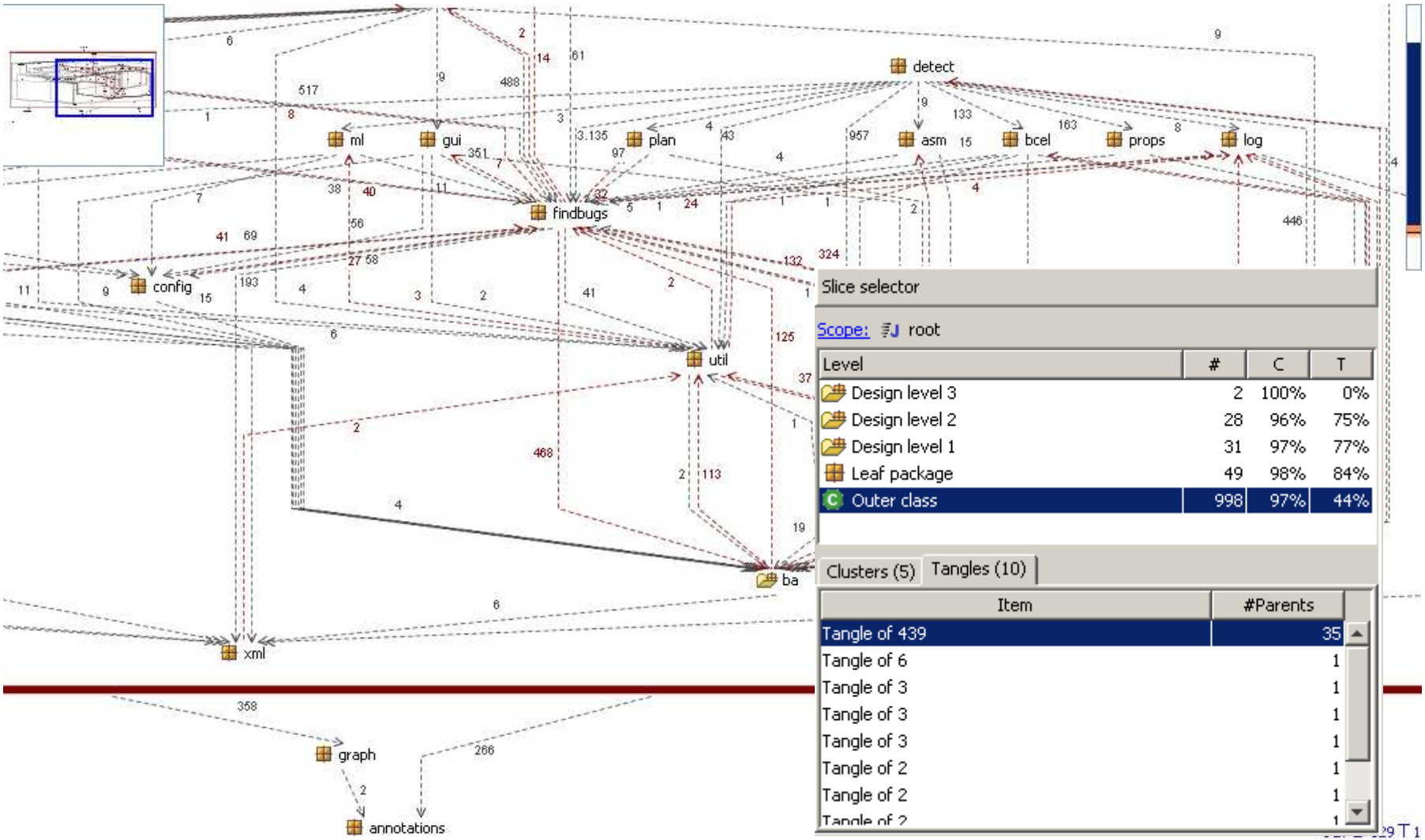
1.0.0
(06/2006)

We can do worse…



1.3.0 (07/2007)

We can do MORE worse…

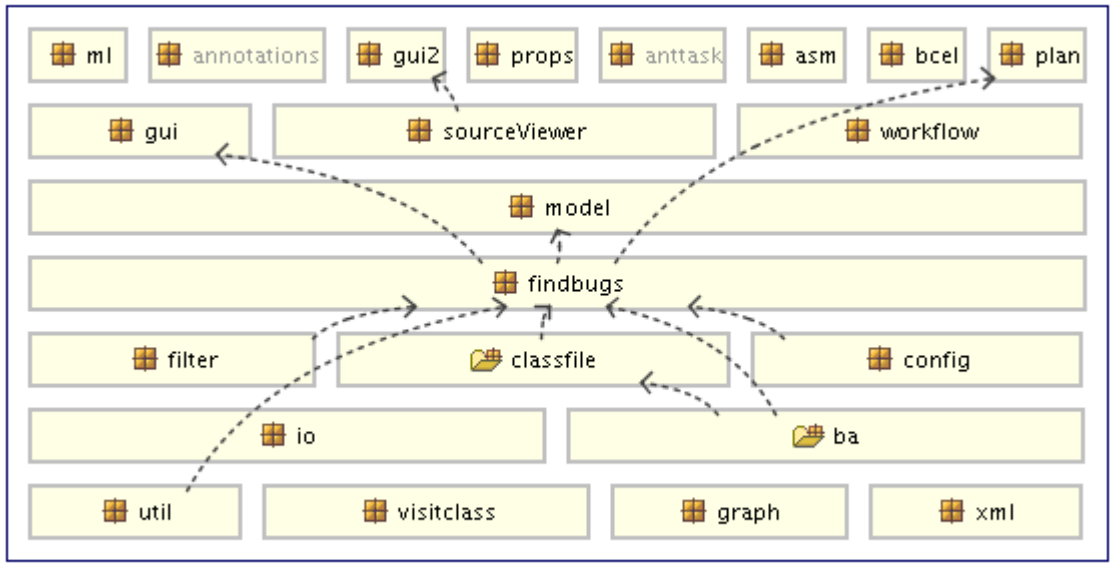



Still managable ?

Findbugs: and the next level of checking ;-)

Findbugs 1.2.1: Architectural Analysis

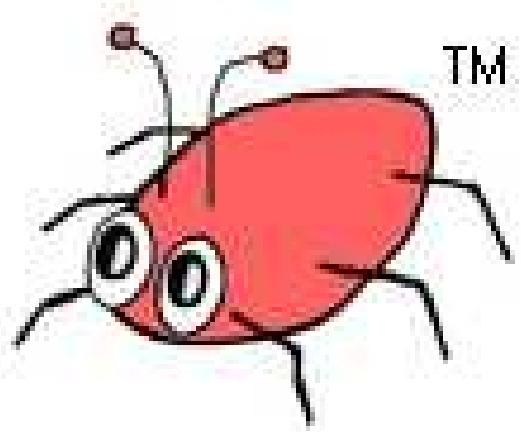
Diagram 1



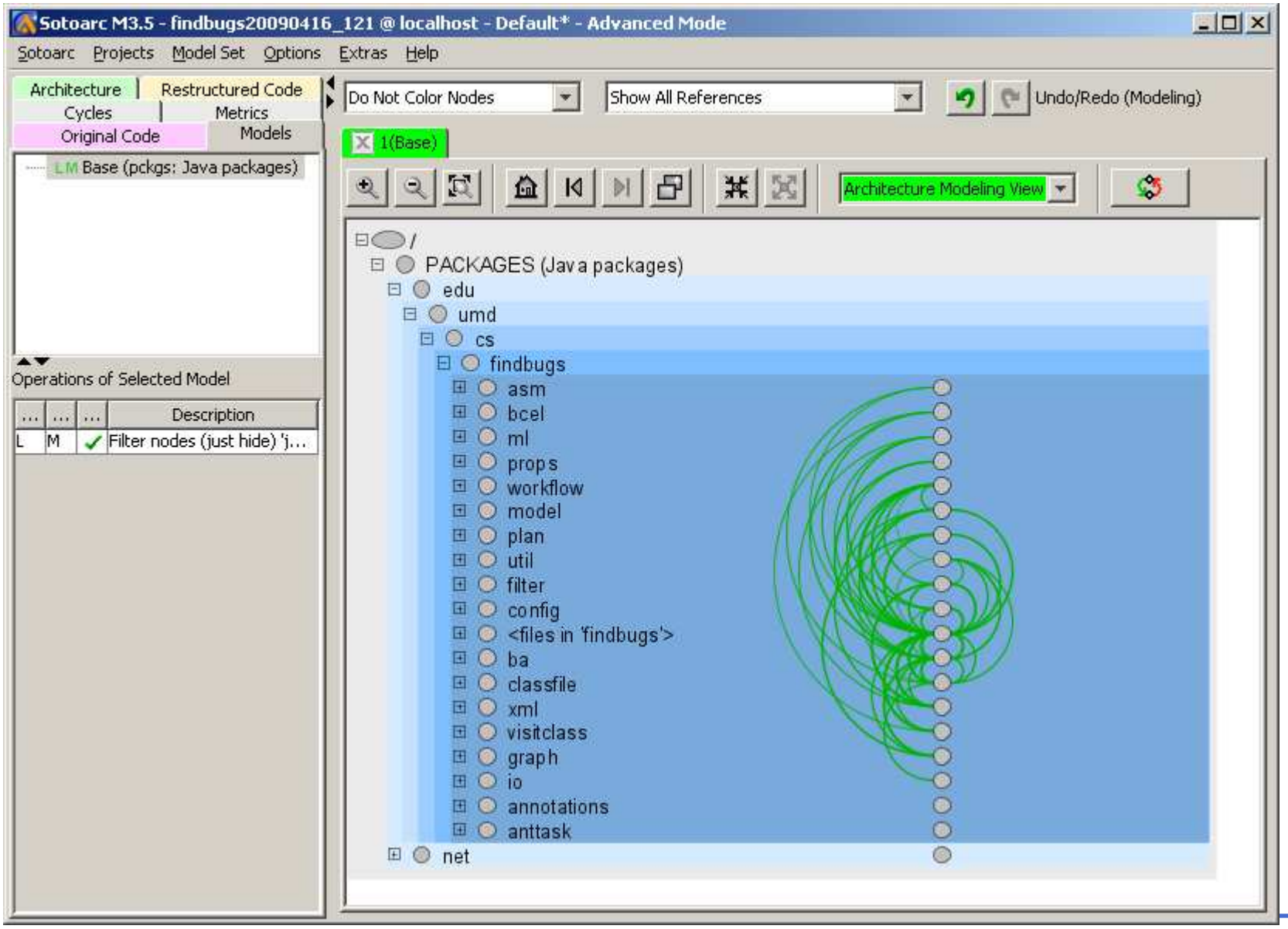
Description: *Subsystem breakout for 'edu.umd.cs.findbugs'*

Violations

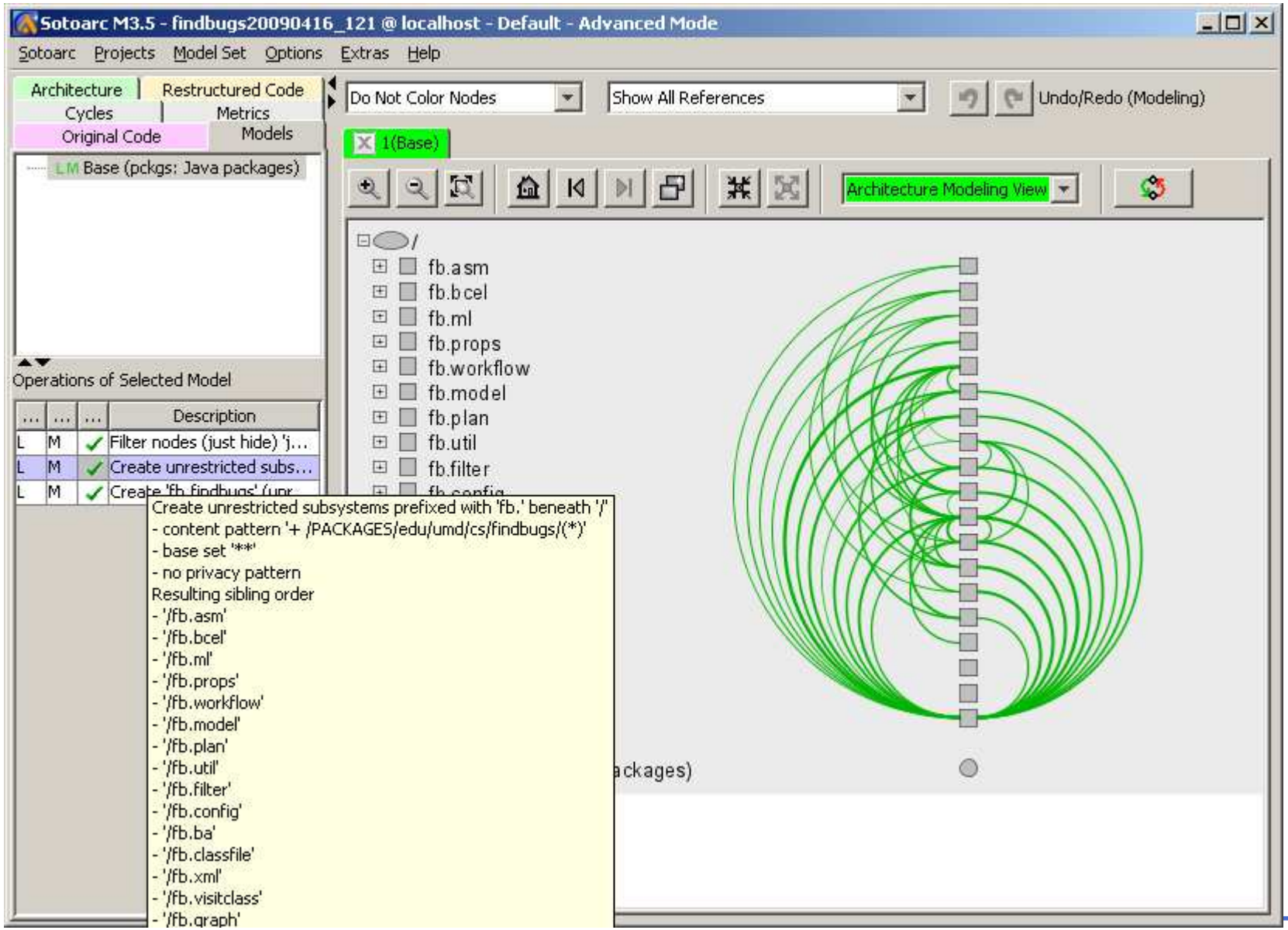
Source	Target	#Violations
sourceViewer	gui2	7
findbugs	plan	37
findbugs	gui	7
findbugs	model	30
filter	findbugs	36
classfile	findbugs	12
config	findbugs	19
ba	findbugs	99
ba	classfile	78
util	findbugs	2
	Total	**327**



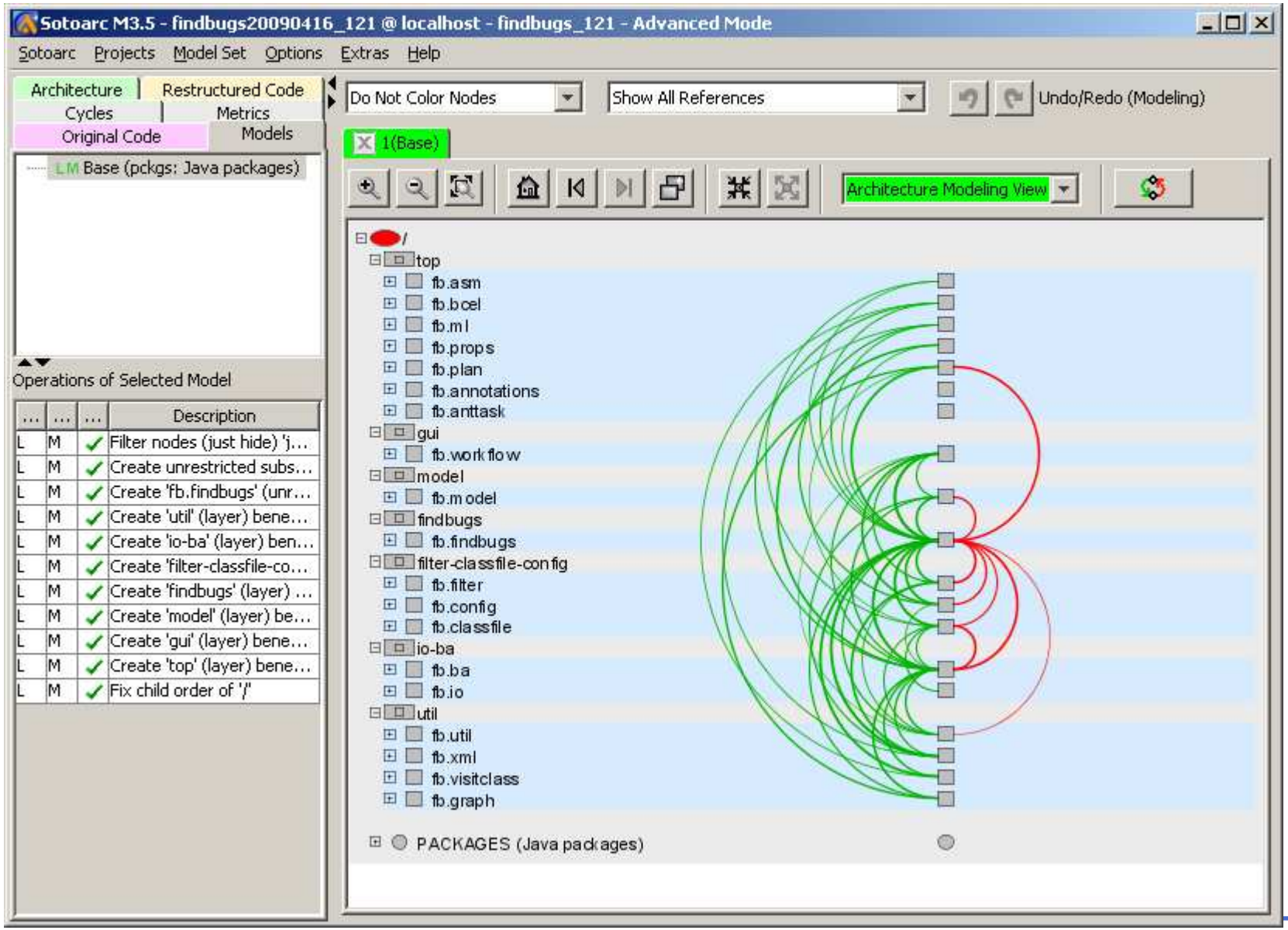
Modeling Subsystems:



Modeling Subsystems:

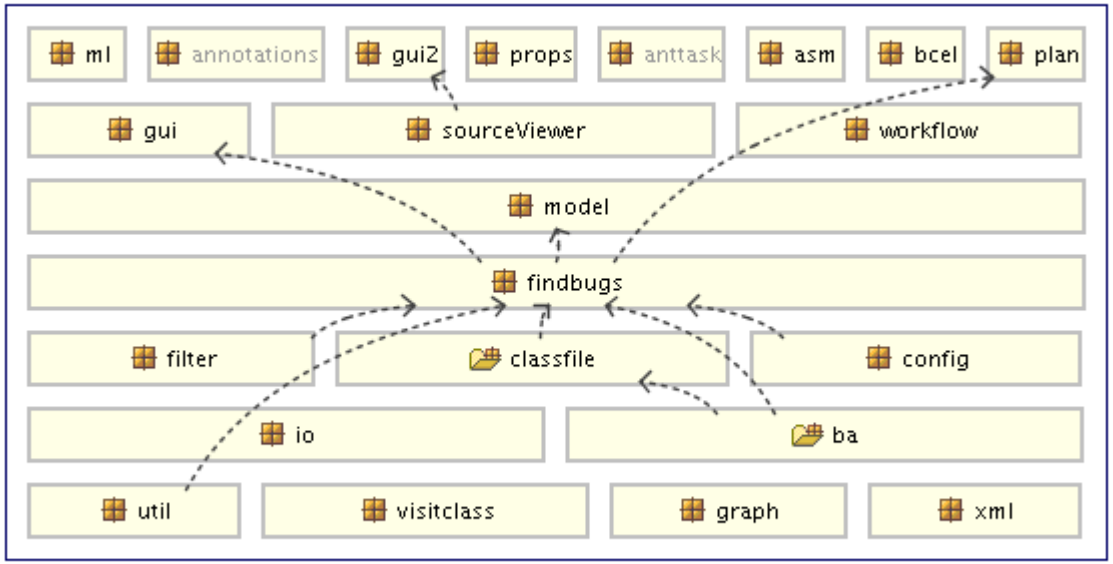


Modeling Subsystems:



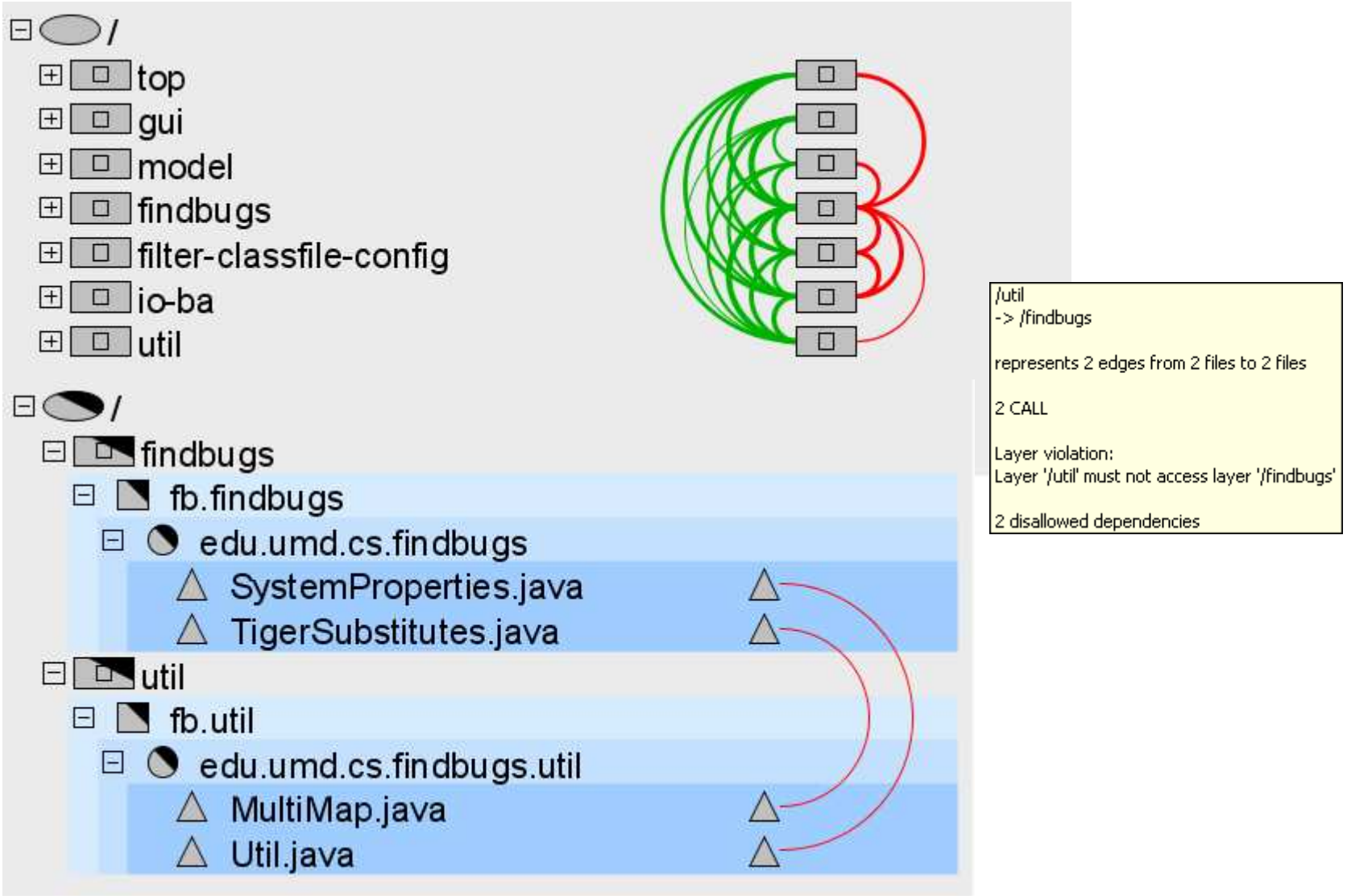
Fixing Architectural Violations

Diagram 1

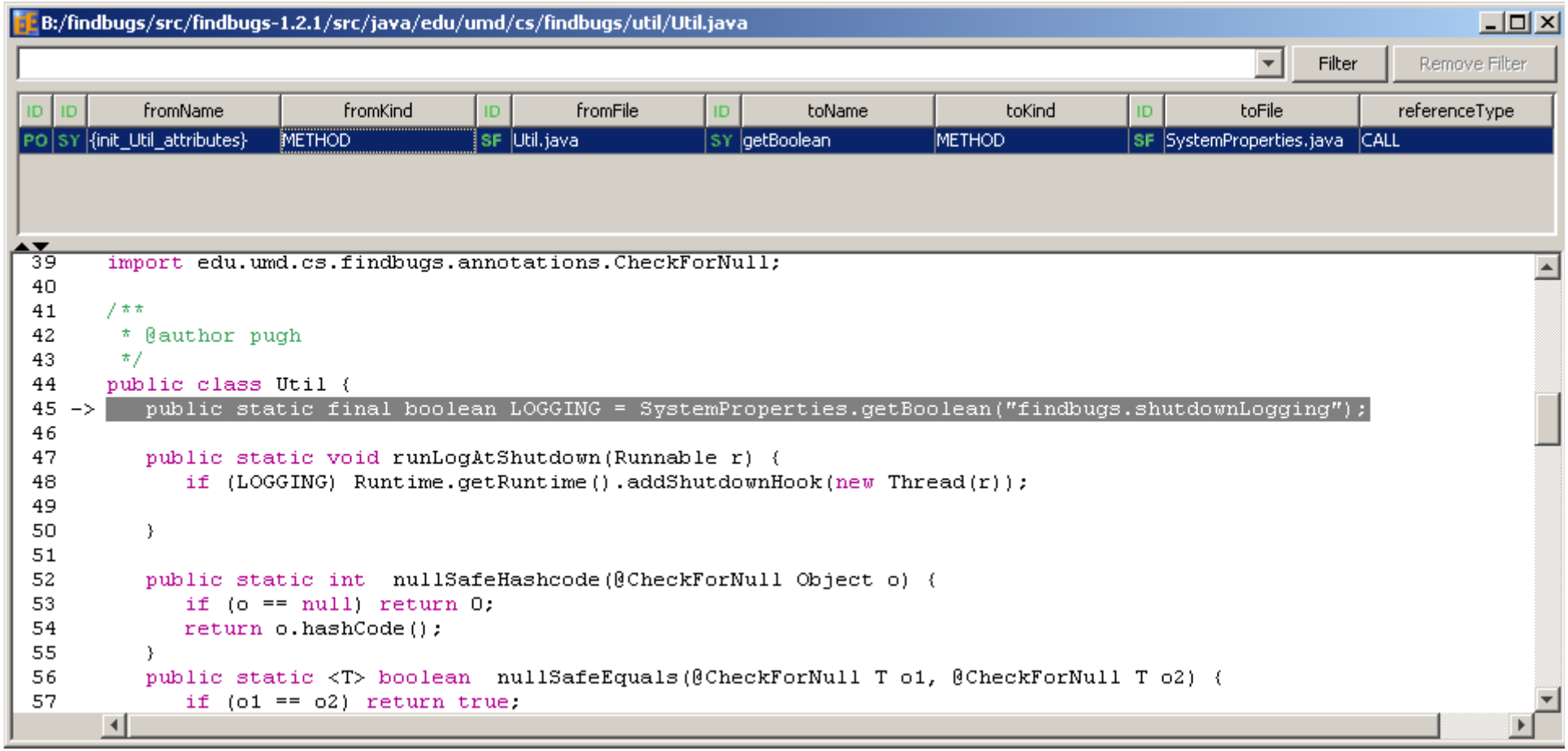


Description: *Subsystem breakout for 'edu.umd.cs.findbugs'*

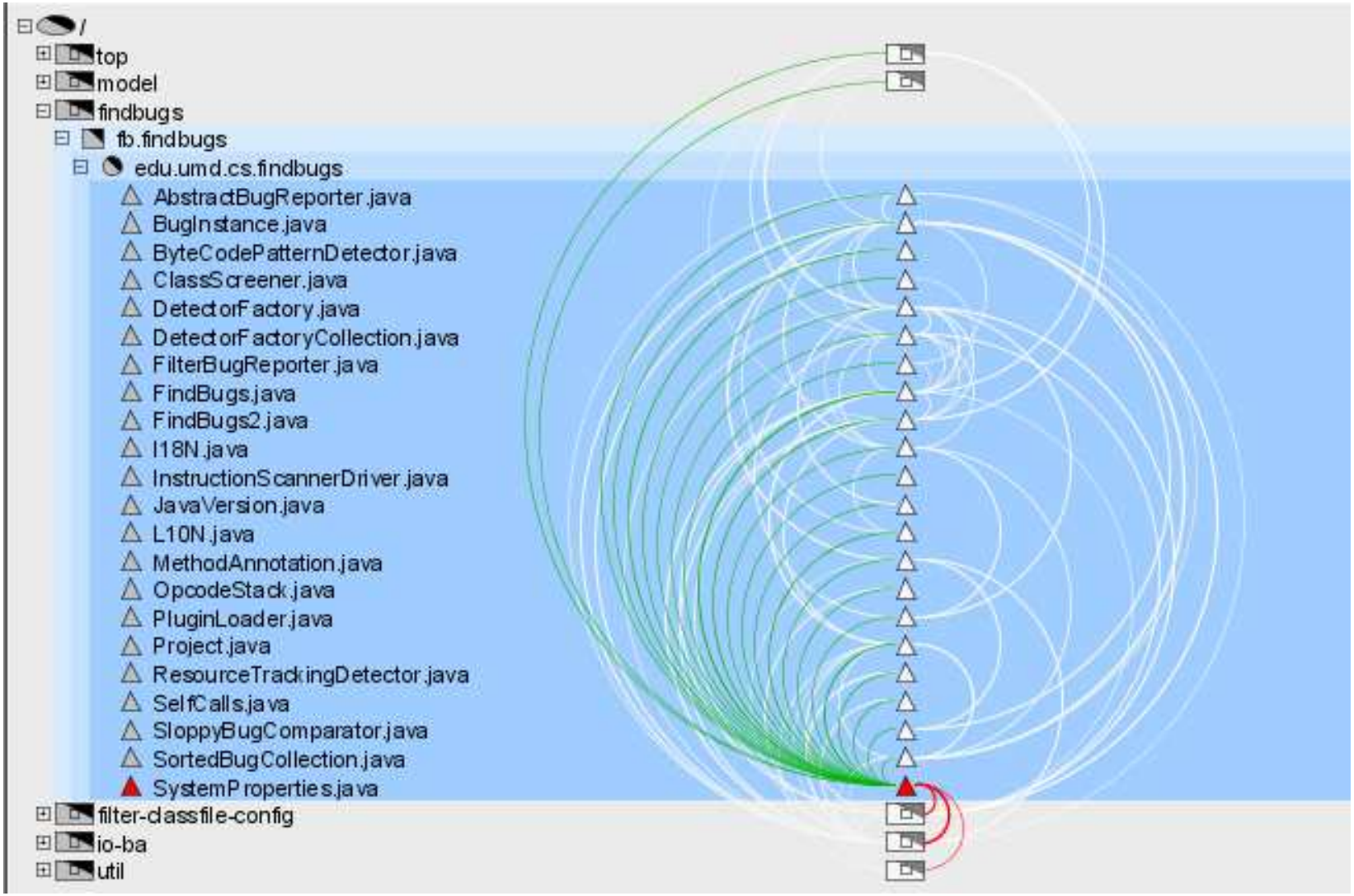
Fixing Architectural Violations



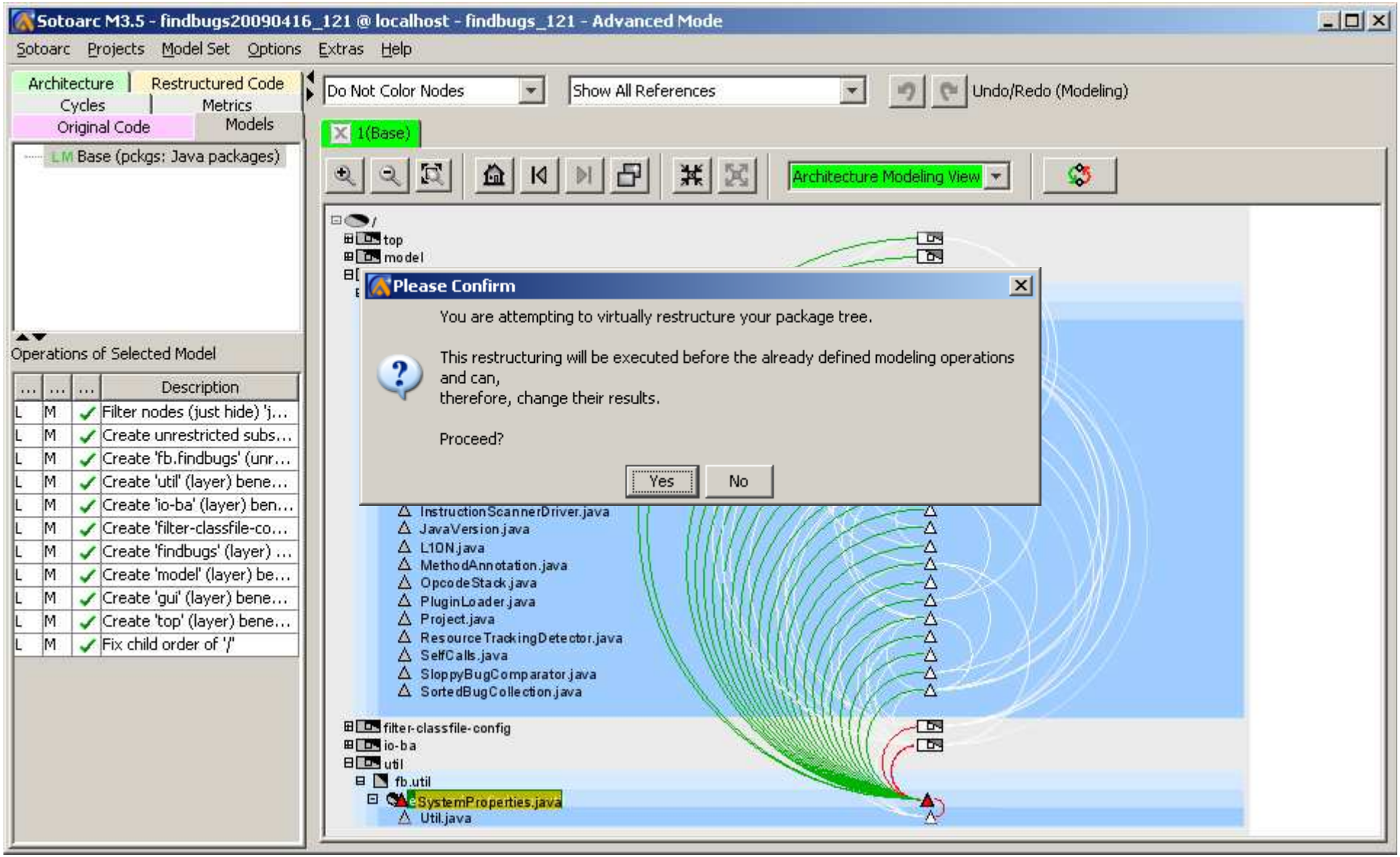
Fixing Architectural Violations



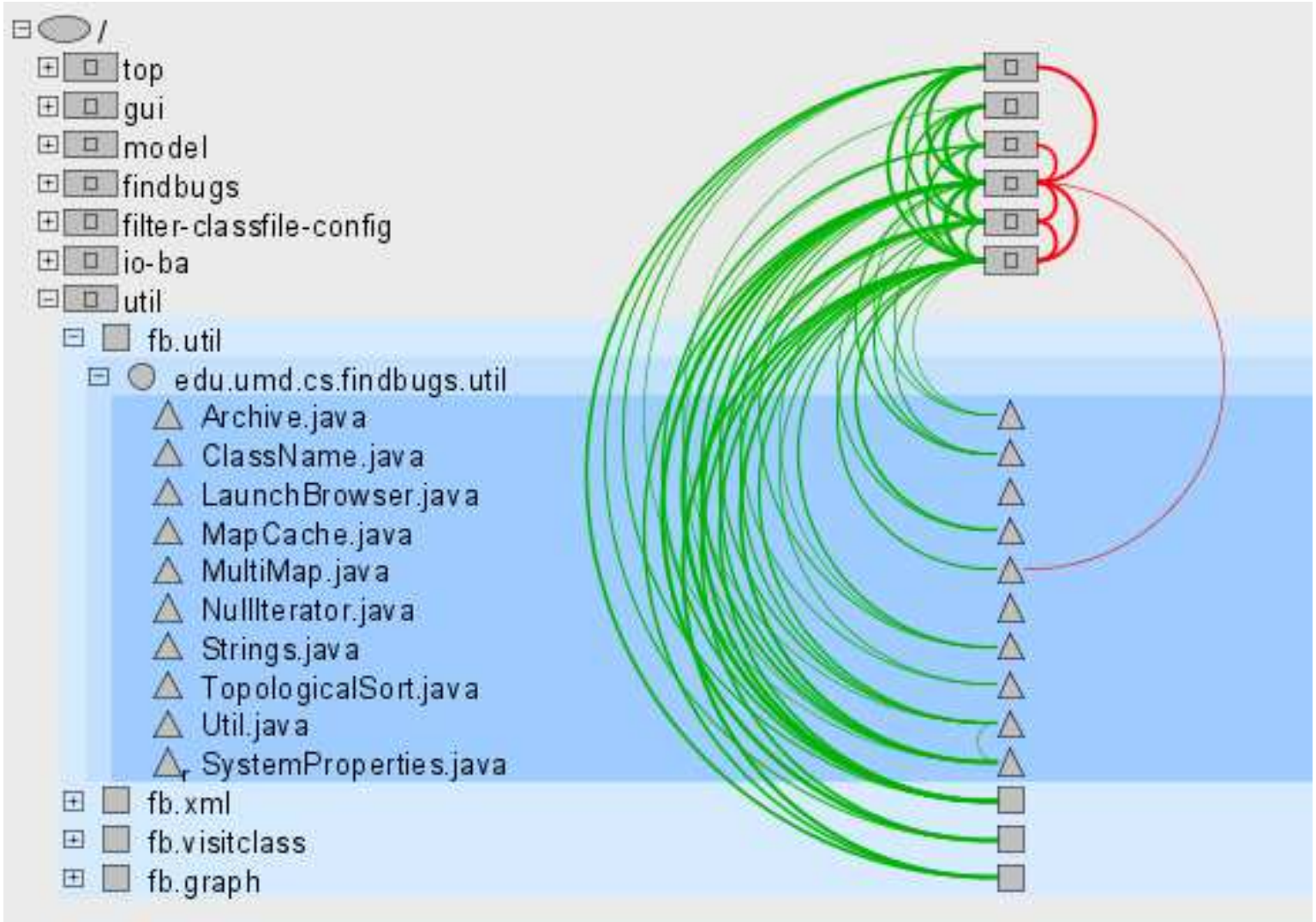
Fixing Architectural Violations



Fixing Architectural Violations



Fixing Architectural Violations



References

URL

- Findbugs
- PMD
- Checkstyle

Books

- Effective Java
- Java Puzzlers
- Other languages: MISRA/C/C++, Effective XYZ

Architecture checking

- ACCU 2009 presentation
- SE-RADIO episode 115

Care about…

Code quality

Architecture

It is really easy !