Research and Education with Eclipse · Poster Reception and BOF at ECOOP, Darmstadt, July 22, 2003

Automated Debugging in Eclipse

(at not even the touch of a button)

Andreas Zeller

Lehrstuhl Softwaretechnik
Universität des Saarlandes, Saarbrücken

A True Story

Mozilla: Netscape's open source web browser

Developed by zillions of volunteers

Mozilla bug #24735, reported by *anantk@yahoo.com*:

```
Ok the following operations cause mozilla to crash
consistently on my machine

 -> Start mozilla
 -> Go to bugzilla.mozilla.org
 -> Select search for bug
 -> Print to file setting the bottom and right margins to
    .50 (I use the file /var/tmp/netscape.ps)
 -> Once it's done printing do the exact same thing again on
    the same file (/var/tmp/netscape.ps)
 -> This causes the browser to crash with a segfault
```

Why does Mozilla crash?

We want to determine the *cause* of the Mozilla crash:

> *The* ***cause*** *of any event ("****effect****") is a preceding event without which the effect would not have occurred.*
>
> — Microsoft Encarta

To prove causality, we must show experimentally that

1. the effect occurs when the cause occurs
2. the effect does *not* occur when the cause does *not* occur

In our case, the *effect* is Mozilla crashing.
The *cause* must be something *variable* – e.g. the HTML input.

Our Issue: Simple Causes

A cause alone does not suffice – the cause must be *simple*, too:

- Simple test case ⇒ *simple program state*
- Simple test case ⇒ *general representative*

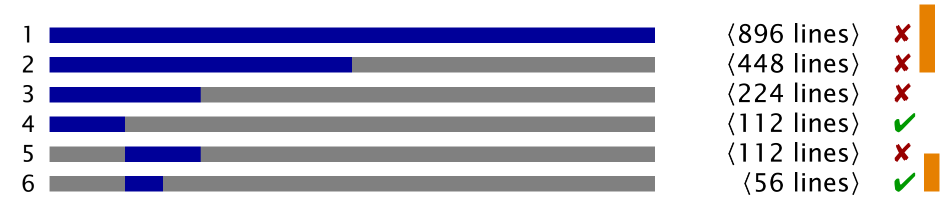
Mozilla BugAThon – Volunteers *simplify test cases*:

Pledges	Reward
5 bugs	invitation to the Gecko *launch party*
10 bugs	the invitation, plus an attractive *Gecko stuffed animal*
12 bugs	same, but animal *autographed* by the Father of Gecko
15 bugs	the invitation, plus a Gecko *T-shirt*
17 bugs	same, but T-shirt *signed* by the grateful engineer
20 bugs	same, but T-shirt signed by the *whole raptor team*

Can't we automate this?

Simplifying Failure-Inducing Input

Delta Debugging uses an *automated test* to simplify HTML pages—until each character is *relevant for the failure*:



1		⟨896 lines⟩	✘
2		⟨448 lines⟩	✘
3		⟨224 lines⟩	✘
4		⟨112 lines⟩	✔
5		⟨112 lines⟩	✘
6		⟨56 lines⟩	✔
⋮			
57	**<SELECT␣NAME="priority"␣MULTIPLE␣SIZE=7>**	⟨40 characters⟩	✘
58	<SELECT␣NAME="priori**ty"␣MULTIPLE␣SIZE=7>**	⟨20 characters⟩	✔
59	**<SELECT␣NAME="priori**ty"␣MULTIPLE␣SIZE=7>	⟨20 characters⟩	✔
60	<SELECT␣NA**ME="priority"␣MULTIPLE␣SIZE=7>**	⟨30 characters⟩	✔
61	**<SELECT␣NA**ME="priori**ty"␣MULTIPLE␣SIZE=7>**	⟨20 characters⟩	✘
62	**<SELECT␣NA**ME="priority"␣MULTIP**LE␣SIZE=7>**	⟨10 characters⟩	✘
⋮			
75	<**SELECT␣**NAME="priority"␣MULTIPLE␣SIZE=7**>**	⟨8 characters⟩	✔
76	**<**S**ELECT␣**NAME="priority"␣MULTIPLE␣SIZE=7**>**	⟨8 characters⟩	✔
77	**<S**E**LECT␣**NAME="priority"␣MULTIPLE␣SIZE=7**>**	⟨8 characters⟩	✔
⋮			
90	**<SELECT** NAME="priority"␣MULTIPLE␣SIZE=7**>**	⟨8 characters⟩	✘

Simplified bug report: Printing <SELECT> crashes.

Another True Story

Upgrading GDB from 4.16 to 4.17 causes trouble:

```
Date: Fri, 31 Jul 1998 15:11:05 -0500
From: Brian Kahne <bkahne@ibmoto.com>
To: DDD Bug Report Address <bug-ddd@gnu.org>
Subject: Problem with DDD and GDB 4.17

When using DDD with GDB 4.16, the run command correctly
uses any prior command-line arguments, or the value of
"set args".  However, when I switched to GDB 4.17, this
no longer worked:  If I entered a run command in the
console window, the prior command-line options would be
lost. [...]
```

How can we automate this?

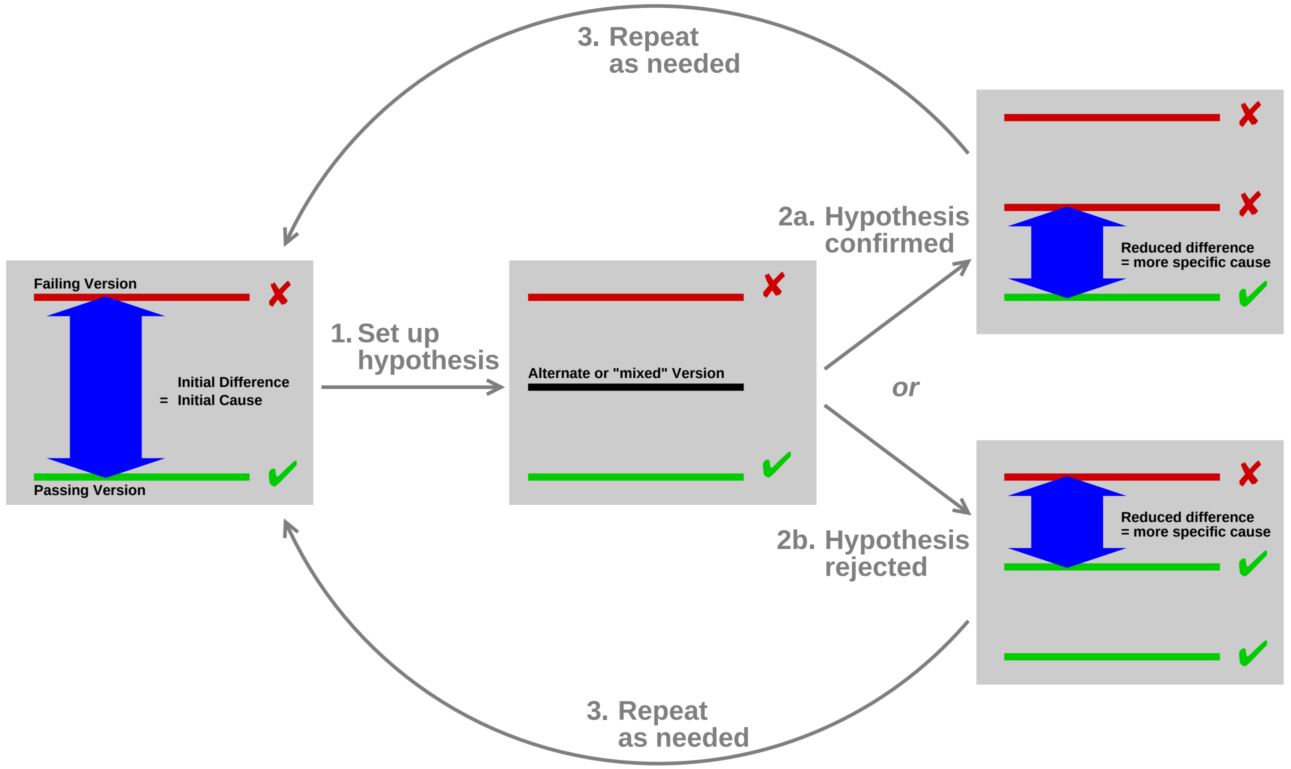
Focus on the Changes

Changes between GDB 4.16 and GDB 4.17:

```
$ diff -r gdb-4.16 gdb-4.17
diff -r gdb-4.16/COPYING gdb-4.17/COPYING
5c5
< 675 Mass Ave, Cambridge, MA 02139, USA
---
> 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
282c282
< Appendix: How to Apply These Terms to Your New Programs
---
> How to Apply These Terms to Your New Programs
⋮
```

and so on for a total of *178.200 lines* at *8721 places*.

Isolating Failure-Inducing Changes



The Failure-Inducing Change

This is the failure-inducing change:

```
diff -r gdb-4.16/gdb/infcmd.c gdb-4.17/gdb/infcmd.c
1239c1278
< "Set arguments to give program being debugged when it is started.\n\
---
> "Set argument list to give program being debugged when it is started.\n\
```

What did go wrong?

- DDD issues “`set args`”
- Reply of GDB 4.17 starts with “`Argument list`”
- DDD expects reply starting with “`Arguments`”!

Requires 280 tests or ~2 hours
(but much faster with frequent tests and ordered changes)

State before Eclipse

For our experiments, we had to specify

Versions. One entry for working and failing version.

Tests. Must distinguish ✔ from ✘.

Construction. Must know how to reconstruct after changes.

Execution. Must know how to invoke program.

We used & maintained a single `Makefile` for this.

✔ okay for a prototype

✘ unbearable for end users—hence never released

No good alternative in sight—*until Eclipse.*

Why Eclipse?

Eclipse provides *one common environment* for

Versions. Eclipse tracks all versions (CVS or local history).

Tests. Eclipse supports automated tests (aka JUnit).

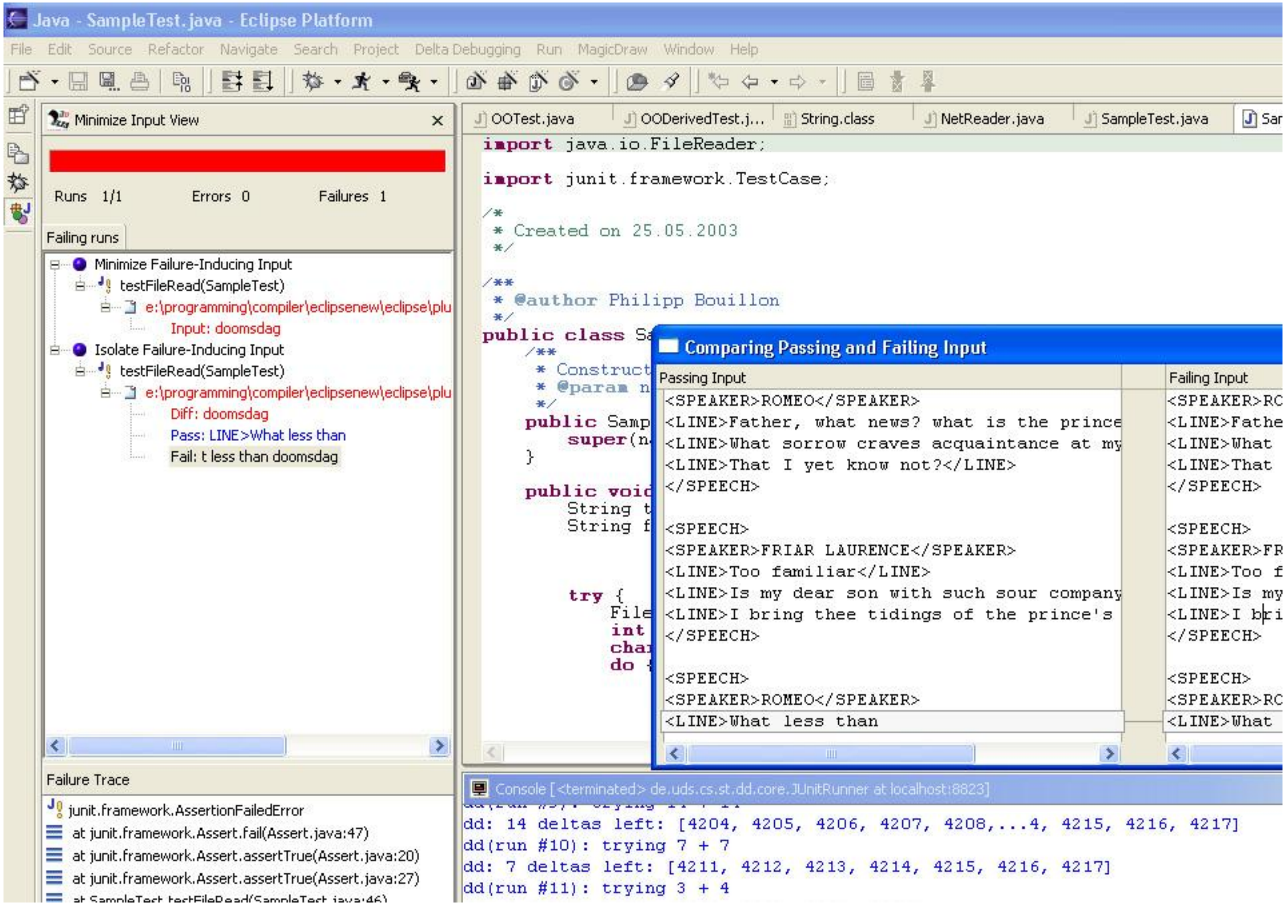
Construction. Eclipse knows how to construct a program.

Execution. Eclipse knows how to invoke a program (via JUnit).

Plus more benefits:

- ✔ Students love it!
- ✔ Several plug-ins for analysis, testing, ...
- ✔ You don't have to edit `Makefiles` or likewise—
  you need not even click a button!

Failure-Inducing Input



Java - SampleTest.java - Eclipse Platform

Minimize Input View

Runs 1/1   Errors 0   Failures 1

Failing runs
- Minimize Failure-Inducing Input
  - testFileRead(SampleTest)
    - e:\programming\compiler\eclipsenew\eclipse\plu
      - Input: doomsdag
- Isolate Failure-Inducing Input
  - testFileRead(SampleTest)
    - e:\programming\compiler\eclipsenew\eclipse\plu
      - Diff: doomsdag
      - Pass: LINE>What less than
      - Fail: t less than doomsdag

Failure Trace

```
junit.framework.AssertionFailedError
at junit.framework.Assert.fail(Assert.java:47)
at junit.framework.Assert.assertTrue(Assert.java:20)
at junit.framework.Assert.assertTrue(Assert.java:27)
at SampleTest.testFileRead(SampleTest.java:46)
```

```
import java.io.FileReader;

import junit.framework.TestCase;

/*
 * Created on 25.05.2003
 */

/**
 * @author Philipp Bouillon
 */
public class Sa
    /**
     * Construct
     * @param n
     */
    public Samp
        super(n
    }

    public voic
        String t
        String f


        try {
            File
            int
            char
            do {
```

Comparing Passing and Failing Input

Passing Input

```
<SPEAKER>ROMEO</SPEAKER>
<LINE>Father, what news? what is the prince
<LINE>What sorrow craves acquaintance at my
<LINE>That I yet know not?</LINE>
</SPEECH>

<SPEECH>
<SPEAKER>FRIAR LAURENCE</SPEAKER>
<LINE>Too familiar</LINE>
<LINE>Is my dear son with such sour company
<LINE>I bring thee tidings of the prince's
</SPEECH>

<SPEECH>
<SPEAKER>ROMEO</SPEAKER>
<LINE>What less than
```

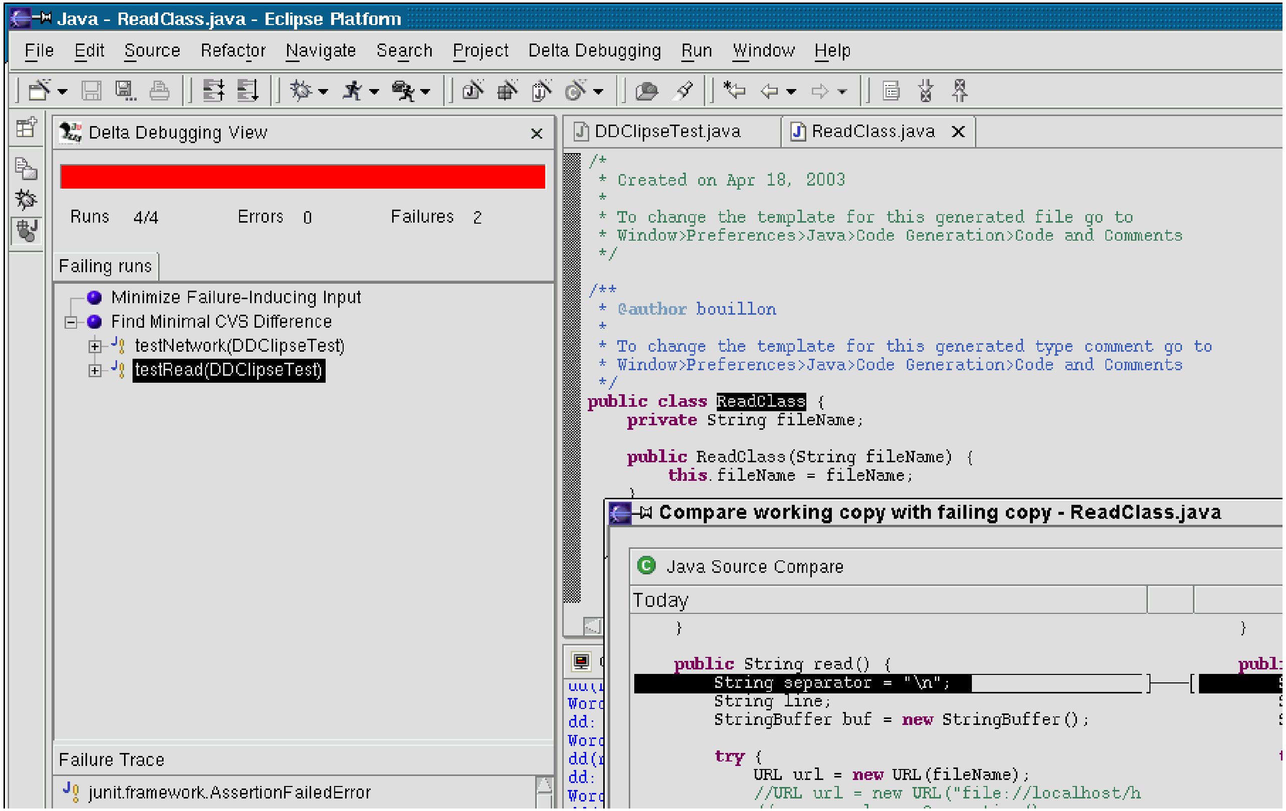
Failing Input

```
<SPEAKER>RO
<LINE>Fathe
<LINE>What
<LINE>That
</SPEECH>

<SPEECH>
<SPEAKER>FR
<LINE>Too f
<LINE>Is my
<LINE>I bri
</SPEECH>

<SPEECH>
<SPEAKER>RO
<LINE>What
```

Console [<terminated> de.uds.cs.st.dd.core.JUnitRunner at localhost:8823]

```
dd: 14 deltas left: [4204, 4205, 4206, 4207, 4208,...4, 4215, 4216, 4217]
dd(run #10): trying 7 + 7
dd: 7 deltas left: [4211, 4212, 4213, 4214, 4215, 4216, 4217]
dd(run #11): trying 3 + 4
```

Failure-Inducing Code Changes


Java - ReadClass.java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Delta Debugging Run Window Help
Delta Debugging View
Failing runs
Failure Trace
DDClipseTest.java
ReadClass.java
Compare working copy with failing copy - ReadClass.java
Java Source Compare
Today

Runs 4/4 Errors 0 Failures 2

- Minimize Failure-Inducing Input
- Find Minimal CVS Difference
  - testNetwork(DDClipseTest)
  - testRead(DDClipseTest)

junit.framework.AssertionFailedError

```
/*
 * Created on Apr 18, 2003
 *
 * To change the template for this generated file go to
 * Window>Preferences>Java>Code Generation>Code and Comments
 */

/**
 * @author bouillon
 *
 * To change the template for this generated type comment go to
 * Window>Preferences>Java>Code Generation>Code and Comments
 */
public class ReadClass {
    private String fileName;

    public ReadClass(String fileName) {
        this.fileName = fileName;
    }
```

```
    }

    public String read() {
        String separator = "\n";
        String line;
        StringBuffer buf = new StringBuffer();

        try {
            URL url = new URL(fileName);
            //URL url = new URL("file://localhost/h
```

Conclusion and Future Work

Finding failure causes automatically is feasible:

- Delta Debugging plugin for *failure-inducing input* available today
- Plugin for *failure-inducing changes* available by October

Advanced diagnoses now conducted on Eclipse:

- Failure-inducing *program states* and *cause-effect chains*
- Failure-inducing and *self-rescuing program code*

Prototype *AskIgor* available as Web service

Integration of plugins underway:

- *Program Analysis* (Soot) to improve diagnosis quality
- *Continuous Testing* (MIT) to test even more frequently

http://www.st.cs.uni-sb.de/dd/

Read More

Automated Debugging. Morgan Kaufmann Publishers, Summer 2004.

Isolating Cause-Effect Chains from Computer Programs. Proc. ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE 2002), Charleston, Nov. 2002.

Isolating Failure-Inducing Thread Schedules. (w/ J.-D. Choi) Proc. ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2002), Rom, July 2002.

Simplifying and Isolating Failure-Inducing Input. (w/ R. Hildebrandt) IEEE Transactions on Software Engineering 28(2), February 2002, pp. 183–200.

Automated Debugging: Are We Close? IEEE Computer, Nov. 2001, pp. 26–31.

Visualizing Memory Graphs. (w/ T. Zimmermann) Proc. of the Dagstuhl Seminar 01211 "‘Software Visualization"’, May 2001. LNCS 2269, pp. 191–204.

Simplifying Failure-Inducing Input. (w/ R. Hildebrandt) Proc. ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2000), Portland, Oregon, August 2000, pp. 135-145.

Yesterday, my program worked. Today, it does not. Why? Proc. ACM SIGSOFT Conference (ESEC/FSE 1999), Toulouse, Sep. 1999, LNCS 1687, pp. 253–267.

http://www.st.cs.uni-sb.de/dd/