Improving Performance Analysis of Software System Versions Using Change-Based Test Selection

David Georg Reichelt [1] Fabian Scheller [2]

[1]Abteilung Betriebliche Informationsysteme, Universität Leipzig

[2]Institut für Infrastruktur und Ressourcenmanagement, Universität Leipzig

6. November 2015

Basic Idea

- research question: which performance problem classes exist on code level?
- state of the art: little systematic research on performance problem classes on code level
- approach: analyse performance of units tests of versions of a program
  - detect changes
  - detect problems
  - classify problems

Assumptions

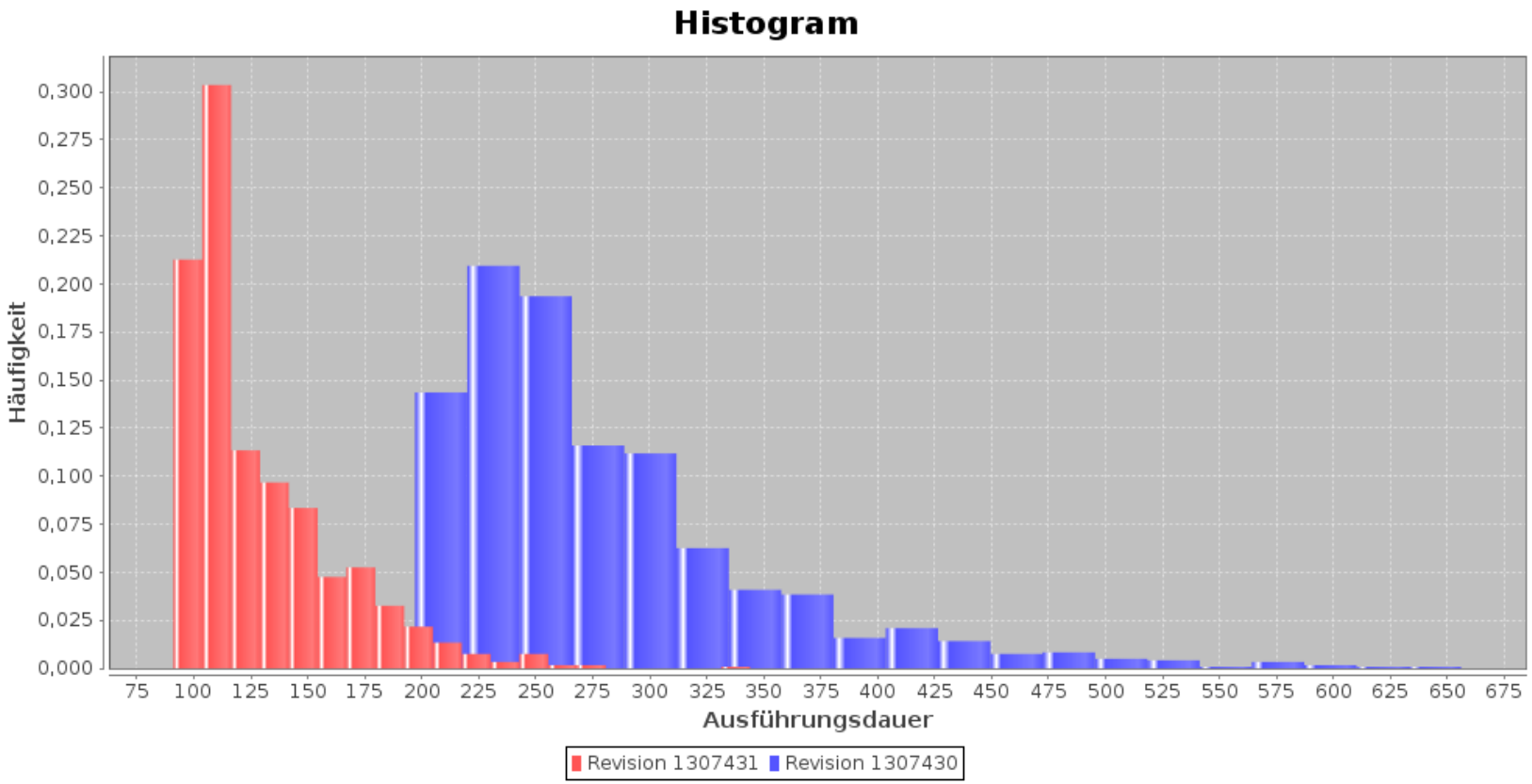
- basic assumptions
  - performance changes during software development
  - performance of unit tests corresponds to performance of program

$\Rightarrow$ relevant program classes: frameworks or backend components

Example: Apache Commons IO

- performance change between 560660 and 584162
- relevant commits
  - 561233 - Fixing svn locations after TLP move
  - 561417 - Move Commons TLP changes
  - 561491 - Apache Apache
  - 561555 - Apache Apache
  - 561564 - finish long-overdue-to-be-finished removal of documented lang dependency
  - 561628 - Commons TLP: directory [..] has moved to "dist/commons"[..]
  - 567499 - Updated commons parent version to 4.
  - 568574 - Removing the 'Commons ' from the names [..]
  - 568588 - Fixing the user svn url
  - 568979 - Changing name to 'Commons Xxx'
  - **584162 - IO-126 Add facility to specify case sensitivity for prefix and suffix file filter**

Example: Measurement



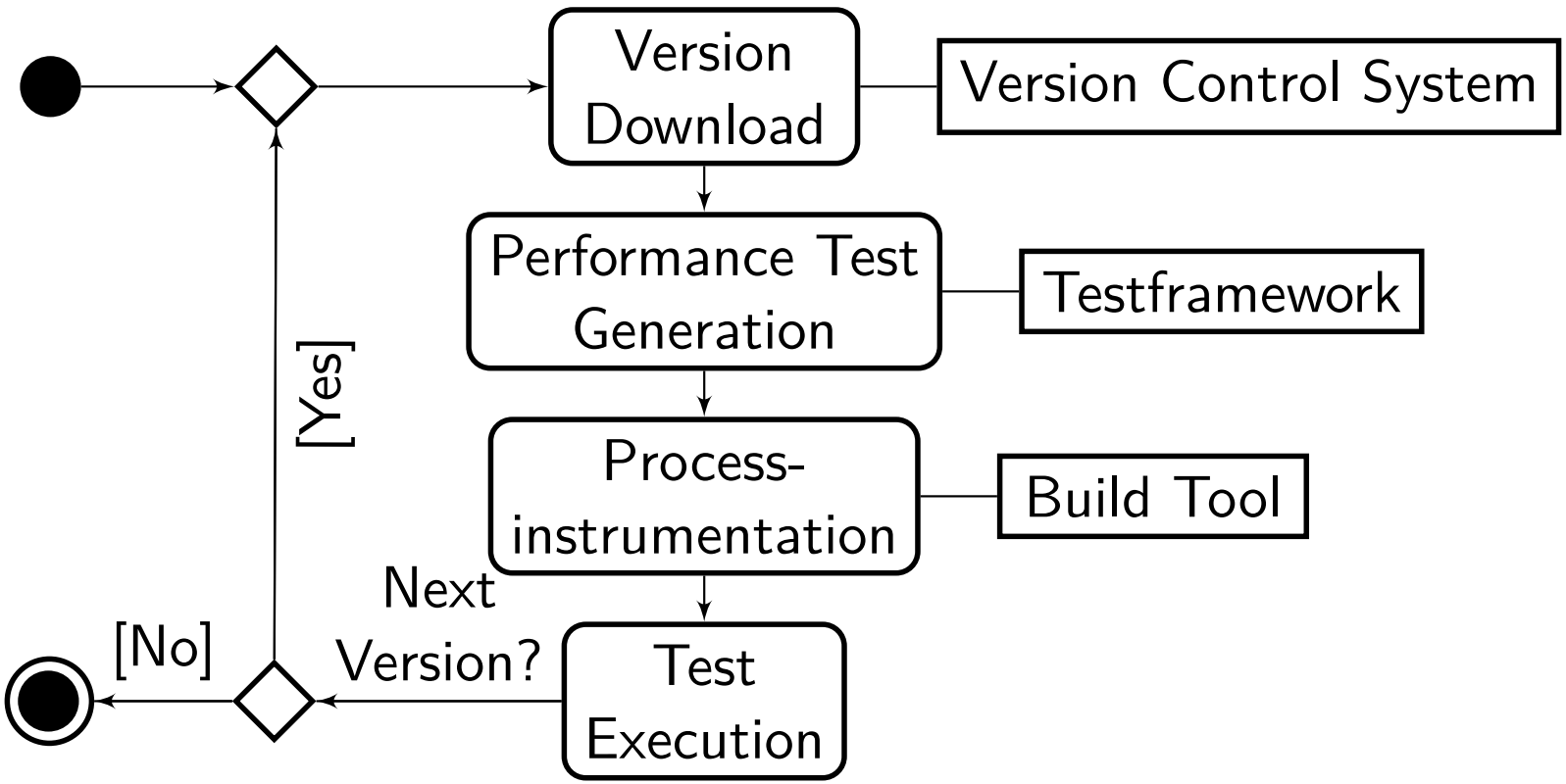
Example: Diff

```
98  --- ../../projekte/commons-io/src/test/java/org/apache/commons/io/output/LockableFileWriterTest.java
99  +++ ../../projekte/commons-io/src/test/java/org/apache/commons/io/output/LockableFileWriterTest.java
100 @@ -19,7 +19,6 @@
101  import java.io.File;
102  import java.io.IOException;
103  import java.io.Writer;
104 -import java.nio.charset.UnsupportedCharsetException;
105
106  import org.apache.commons.io.IOUtils;
107  import org.apache.commons.io.testtools.FileBasedTestCase;
108 @@ -160,12 +159,12 @@
109      }
110
111      //-----------------------------------------------------------------------
112 -    public void testConstructor_File_encoding_badEncoding() throws IOException {
113 +    public void testConstructor_File_encoding_badEncoding() {
114          Writer writer = null;
115          try {
116              writer = new LockableFileWriter(file, "BAD-ENCODE");
117              fail();
118 -        } catch (UnsupportedCharsetException ex) {
119 +        } catch (IOException ex) {
120              // expected
121              assertFalse(file.exists());
122              assertFalse(lockFile.exists());
123
```

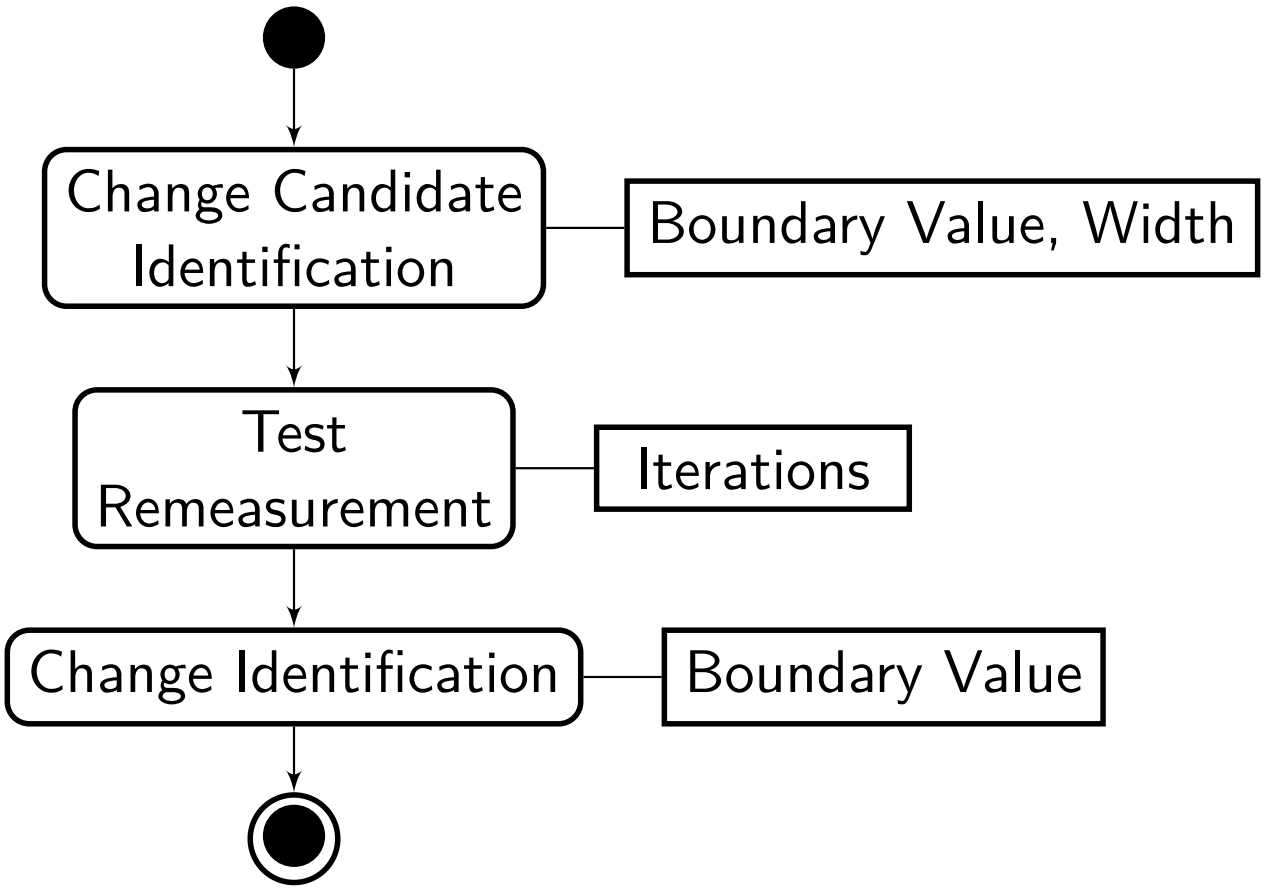
Steps

- main problem: performance measurements are instable
  - background processes
  - just-in-time compilation
  - ..
- approach: measurement and refined measurement
  - measurement of performance for all testcases in all versions
  - identification of performance changes
  - identification of performance problems

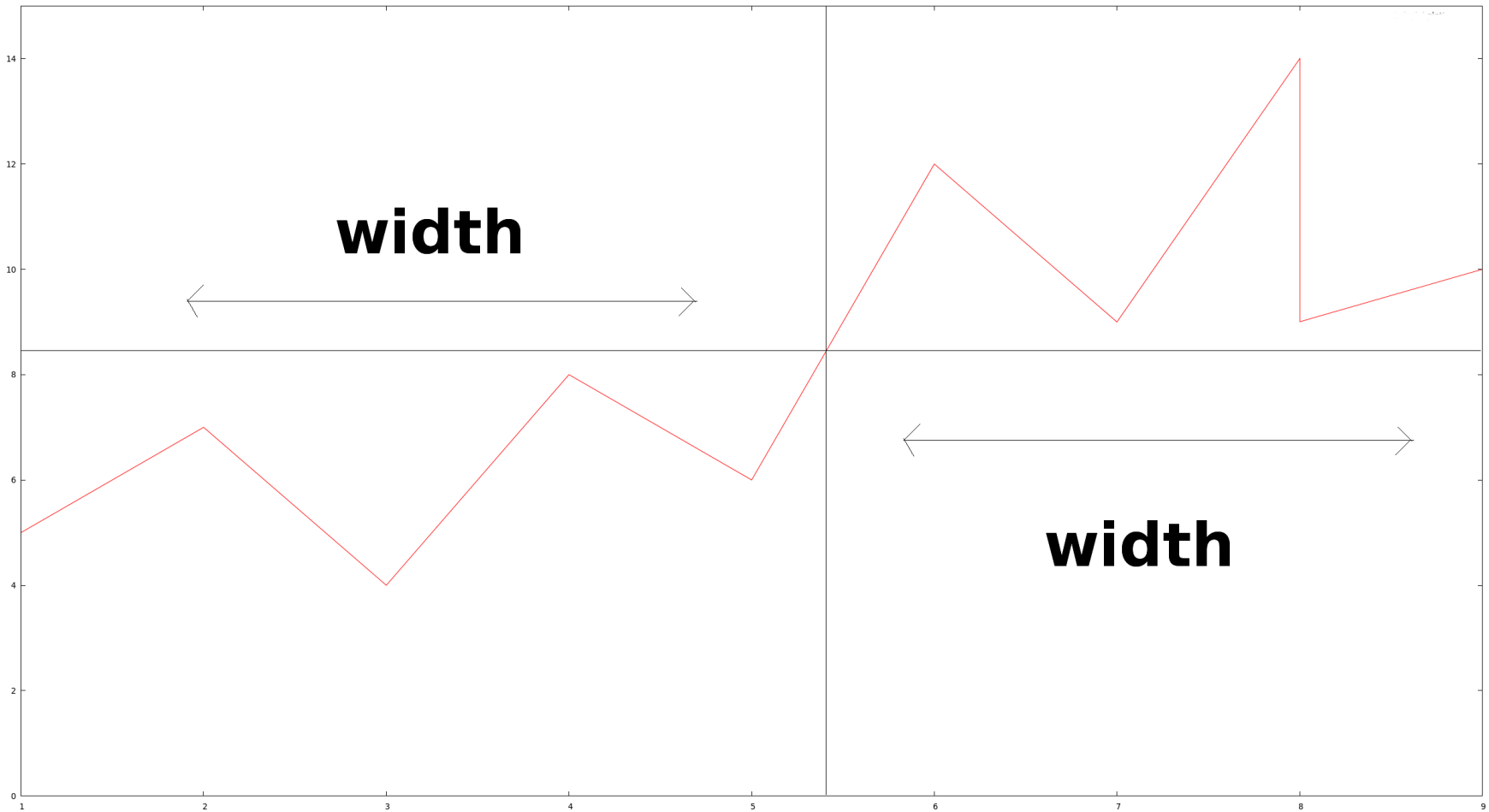
Performance Measurement



Identification of Performance Changes



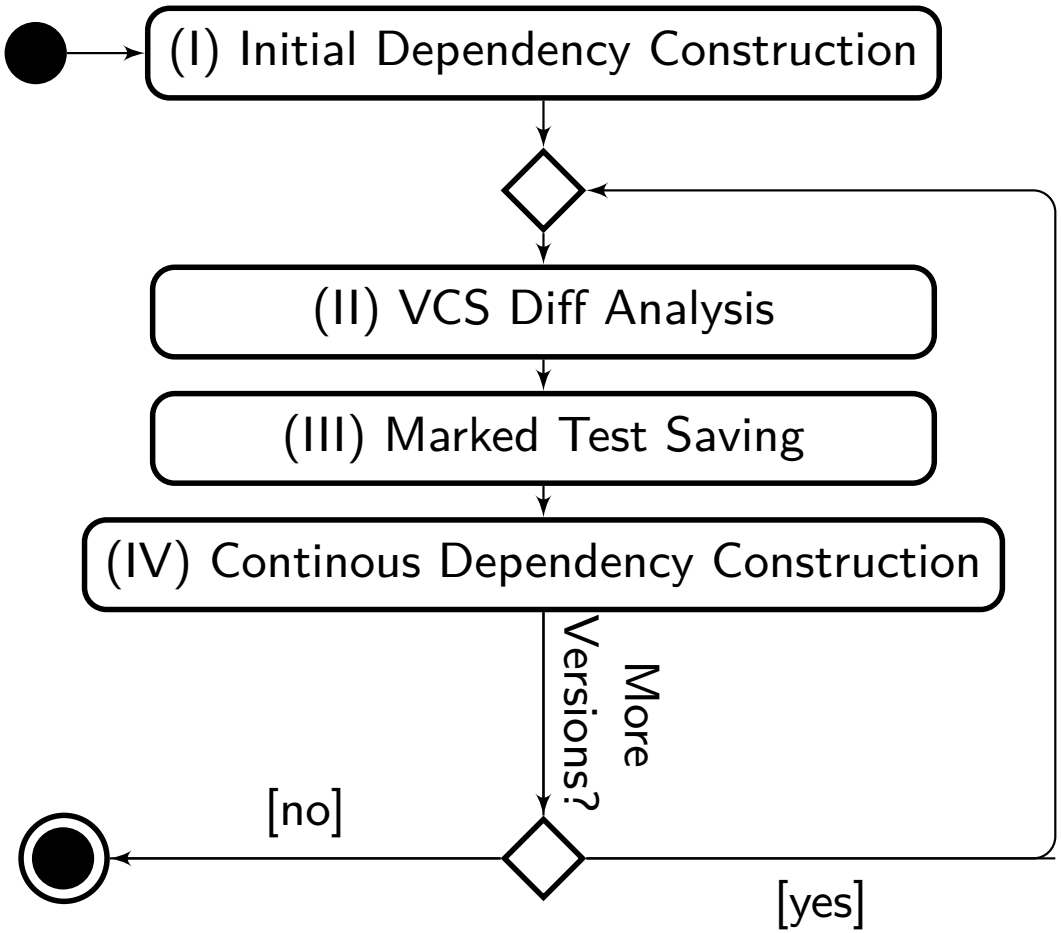
Heuristic



Enhanced PeASS process

- problems of old process
  - many unneccessary measurements
  - heuristics lead to lost performance changes
- enhanced process
  - change-based test selection
    - usage of kieker instrumentation and trace analysis
    - selecting relevant tests for re-run
  - measurement for selected unit tests
  - identification of performance problems

Change-based Test Selection



Enhanced PeASS process

- find first performance change in Apache Commons IO with 4000 iterations each
  - reduced test executions: 428 tests, 8 h 4 min
  - all test executions: 6320 tests, $\sim$ 119 h
- counts of normal and reduced tests for chosen Apache Commons projects

Project	Normal Tests	Reduced Tests
Commons BCEL	100521	12109
Commons BeanUtils	97178	8261
Commons Collections	44058	842
Commons IO	659190	77259
Commons Jelly	298	13

Related Work

- green mining (Hindle et al., 2014)
- search of performance errors in bug tracker (Jin et al., 2012, Nistor et al., 2013)
- application of performance tests to repositories (Horký et al, 2013) (Heger et al., 2013)
- (performance antipatterns on architecture level (Smith et al, 2003))

Summary

- basic idea: examine development of performance of unit tests during software development
- goal: classification of typical performance problems
- method: change-based test selection
  - reconstruct call hierarchy with kieker
  - analyse VCS diffs
  - only re-run tests which are influenced by change

Future Work

- reusage of change-based test selection
  - usage for load tests
  - usage for performance unit tests
- current work for statistic rigor of PeASS
  - measurement setup: vm runs, warmup, measurement executions
  - statistical evaluation: confidence intervals, boundary values
- implementing root-cause isolation of performance problems

Thanks for your attention! Any questions?

For further questions:

David Georg Reichelt
Abteilung Betriebliche Informationsysteme
Universität Leipzig
reichelt@informatik.uni-leipzig.de

Bibliography

- (Smith et al., 2003) C. U. Smith und L. G. Williams. More new software performance antipatterns: Even more ways to shoot yourself in the foot. In CMG Conference, Seiten 717–725. Citeseer, 2003.
- (Jin et al., 2012) G. Jin, L. Song, X. Shi, J. Scherpelz, and S. Lu. Understanding and detecting real-world performance bugs. ACM SIGPLAN Notices, 47:77–88, 2012.
- (Nistor et al., 2013) A. Nistor, T. Jiang, and L. Tan. Discovering, reporting, and fixing performance bugs. In MSR 2013, pages 237–246. IEEE Press, 2013
- (Horký et al., 2013) V. Horký, F. Haas, J. Kotrc, M. Lacina, and P. Tuma. Performance regression unit testing: a case study. In Computer Performance Engineering, pages 149–163. Springer, 2013.
- (Heger et al., 2013) C. Heger, J. Happe, and R. Farahbod. Automated root cause isolation of performance regressions during software development. In ICPE 13, pages 27–38, New York, USA, 2013. ACM.
- (Hindle et al., 2014) A. Hindle, A. Wilson, K. Rasmussen, E. J. Barlow, J. C. Campbell, and S. Romansky. Greenminer: A hardware based mining software repositories software energy consumption framework. In MSR 2014, pages 12–21, New York, USA, 2014. ACM.