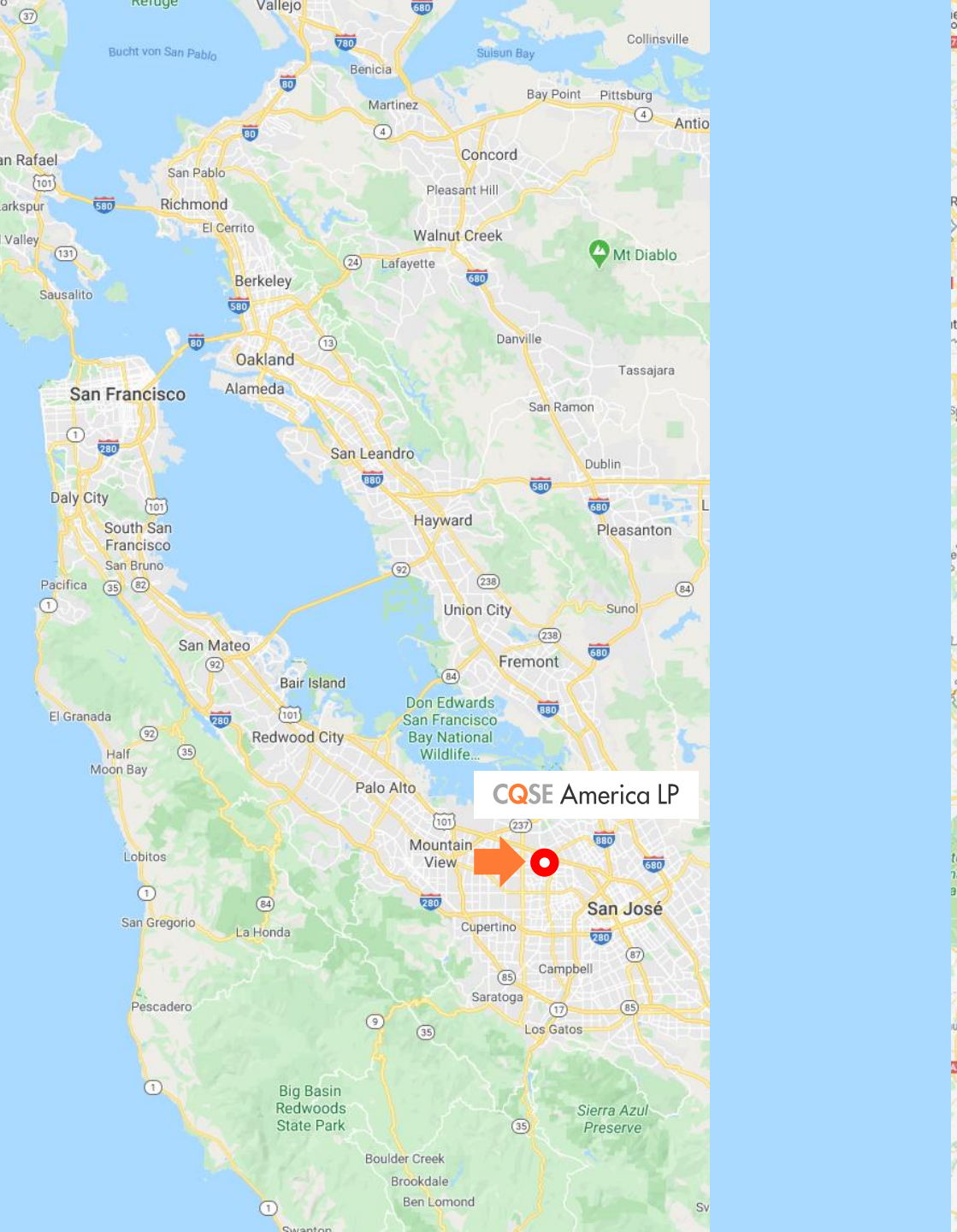
Ein Benchmark zum Formulieren der Erwartungshaltung an Codequalität

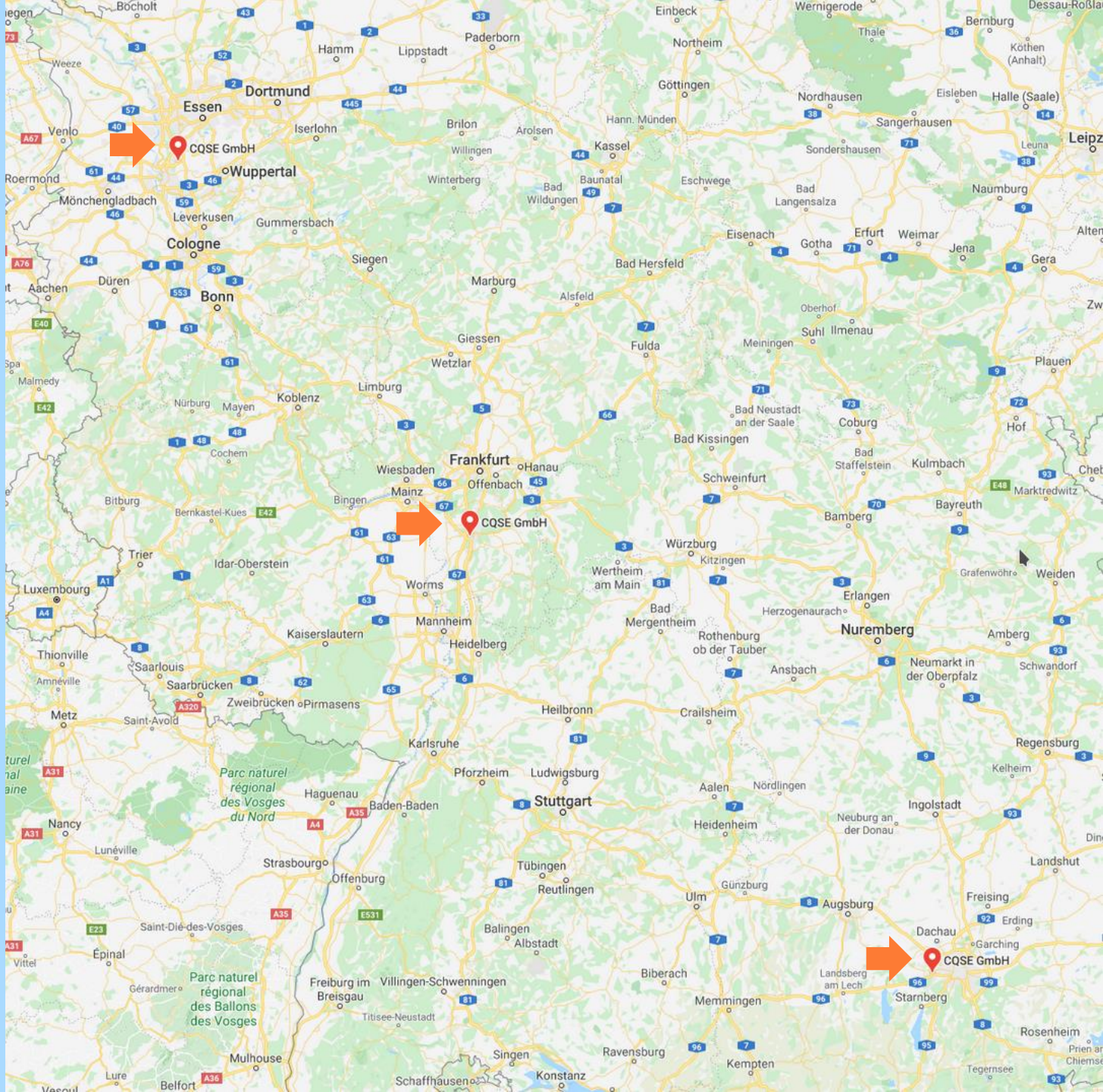
Und bei dir so?

CQSE

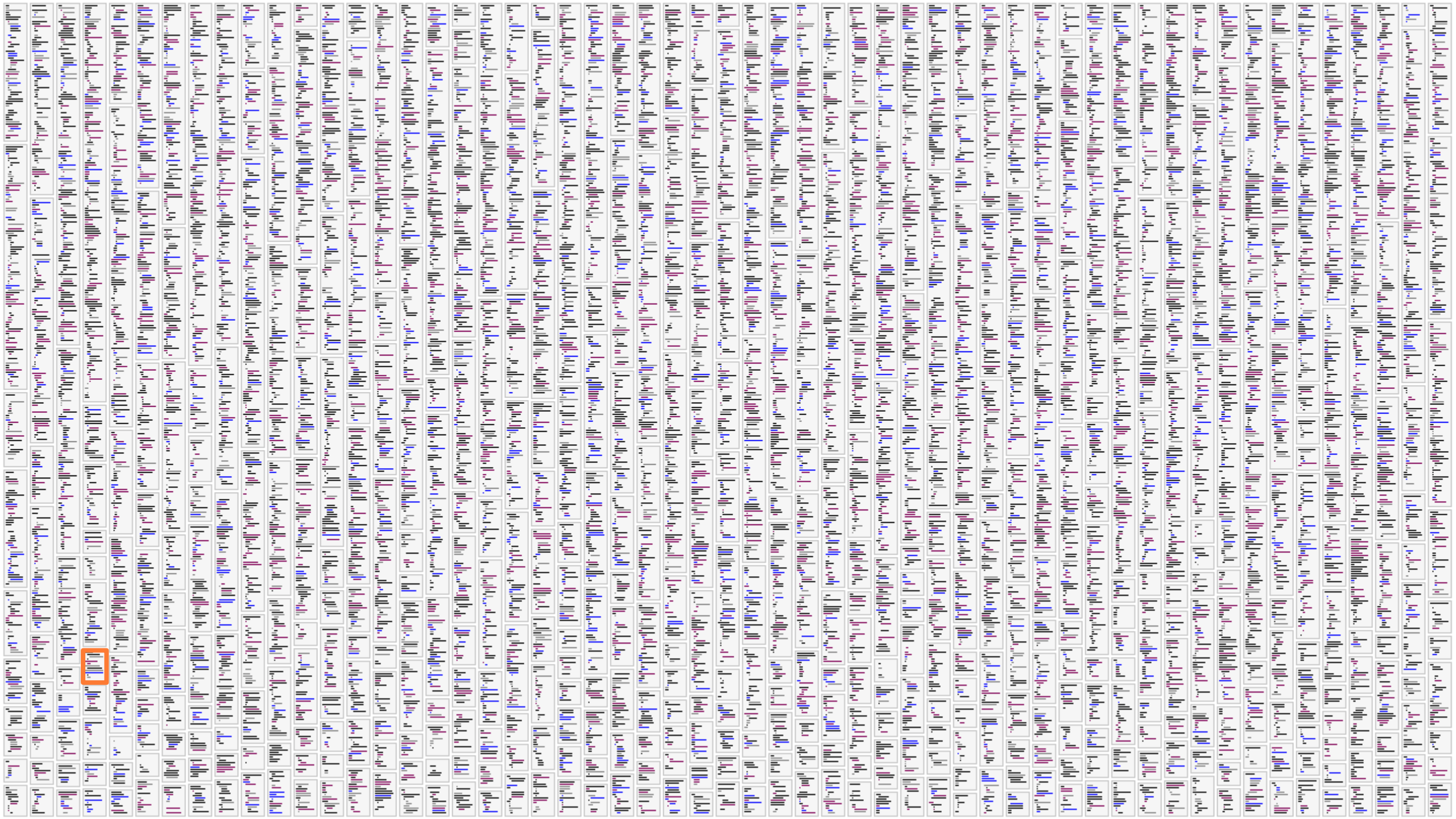
Dr. Nils Göde


Software-Audits
Kontinuierliche Qualitäts- und Testkontrolle
Teamscale
CQSE GmbH
17+ Promotionen in Software Engineering
Eigene Forschung
Enger Kontakt zu Universitäten

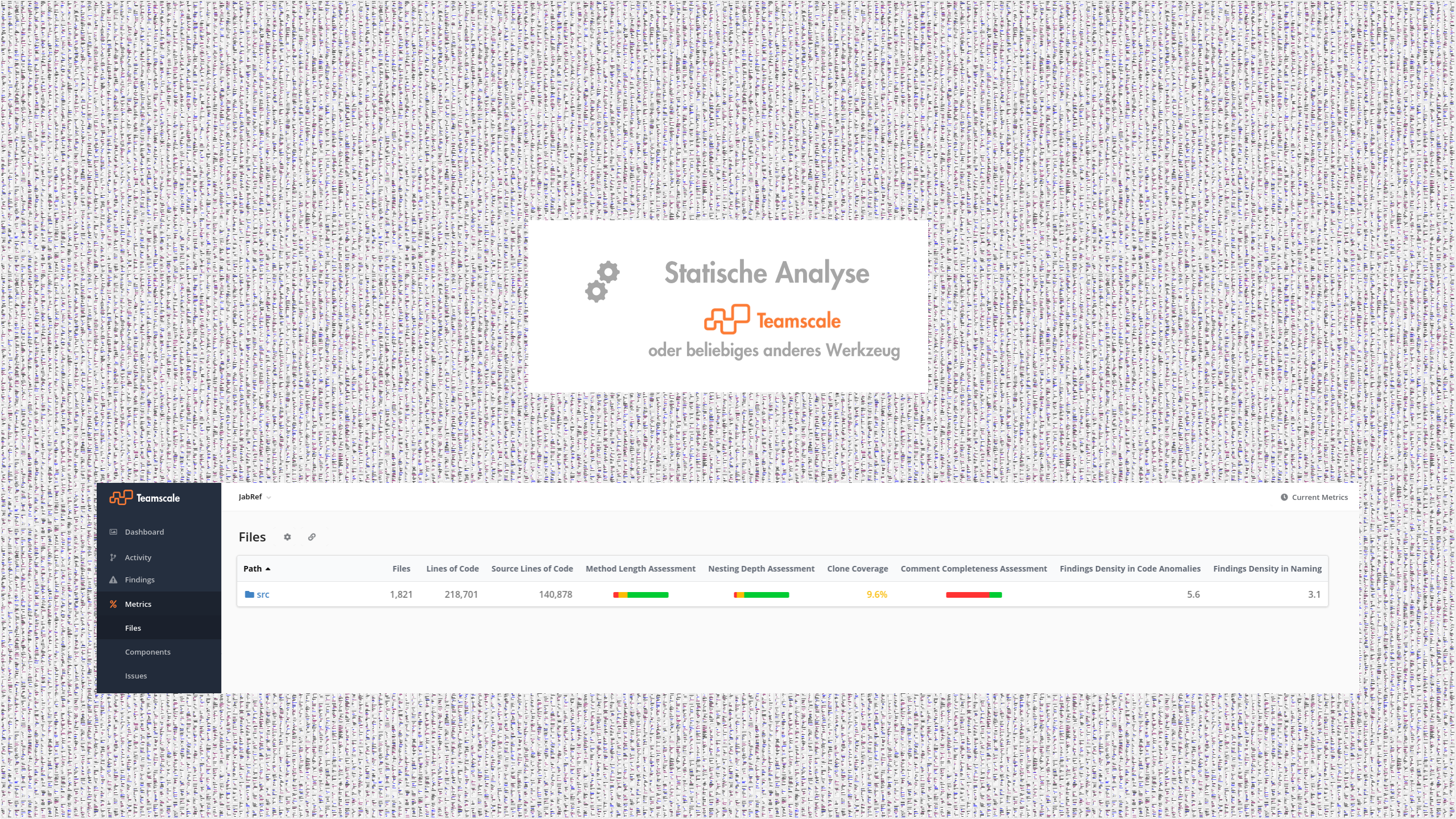

CQSE America LP
Mountain View
San José
Palo Alto
Redwood City
San Francisco
Oakland
Berkeley
Richmond
Concord
Walnut Creek
Pleasanton
Hayward
Fremont
Union City
San Mateo
Daly City
South San Francisco
Mt Diablo
Don Edwards San Francisco Bay National Wildlife...
Big Basin Redwoods State Park
Sierra Azul Preserve


CQSE GmbH
CQSE GmbH
CQSE GmbH
Dortmund
Essen
Cologne
Bonn
Wuppertal
Frankfurt
Stuttgart
Nuremberg
Mannheim
Karlsruhe
Augsburg
Ulm
Parc naturel régional des Vosges du Nord
Parc naturel régional des Ballons des Vosges

```java
import org.apache.commons.cli.ParseException;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/**
 * JabRef's main class to process command line options and to start the UI
 */
public class JabRefMain extends Application {

    private static final Logger LOGGER = LoggerFactory.getLogger(JabRefMain.class);

    private static String[] arguments;

    public static void main(String[] args) {
        arguments = args;
        launch(arguments);
    }

    @Override
    public void start(Stage mainStage) {
        try {
            // Fail on unsupported Java versions
            ensureCorrectJavaVersion();
            FallbackExceptionHandler.installExceptionHandler();

            // Init preferences
            final JabRefPreferences preferences = JabRefPreferences.getInstance();
            Globals.prefs = preferences;
            // Perform migrations
            PreferencesMigrations.runMigrations();

            configureProxy(preferences.getProxyPreferences());

            Globals.startBackgroundTasks();

            applyPreferences(preferences);
```




218.701 LOC
Qualität ?
Technische Schulden ?
Wartbarkeit ?
Zukunftssicherheit ?
Risiken ?
Handlungsbedarf ?


Statische Analyse
Teamscale
oder beliebiges anderes Werkzeug
Teamscale
Dashboard
Activity
Findings
Metrics
Files
Components
Issues
JabRef
Current Metrics
Files
Path
Files
Lines of Code
Source Lines of Code
Method Length Assessment
Nesting Depth Assessment
Clone Coverage
Comment Completeness Assessment
Findings Density in Code Anomalies
Findings Density in Naming
src
1,821
218,701
140,878
9.6%
5.6
3.1

	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			**9,6** %	**22,8** %	**5,6**	**3,1**

Methodenlänge | Schachtelung | Klonüberdeckung | Kommentierung | Codeanomalien pro 1.000 SLOC | Naming-Verstöße pro 1.000 SLOC

JabRef			9,6 %	22,8 %	5,6	3,1

	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1

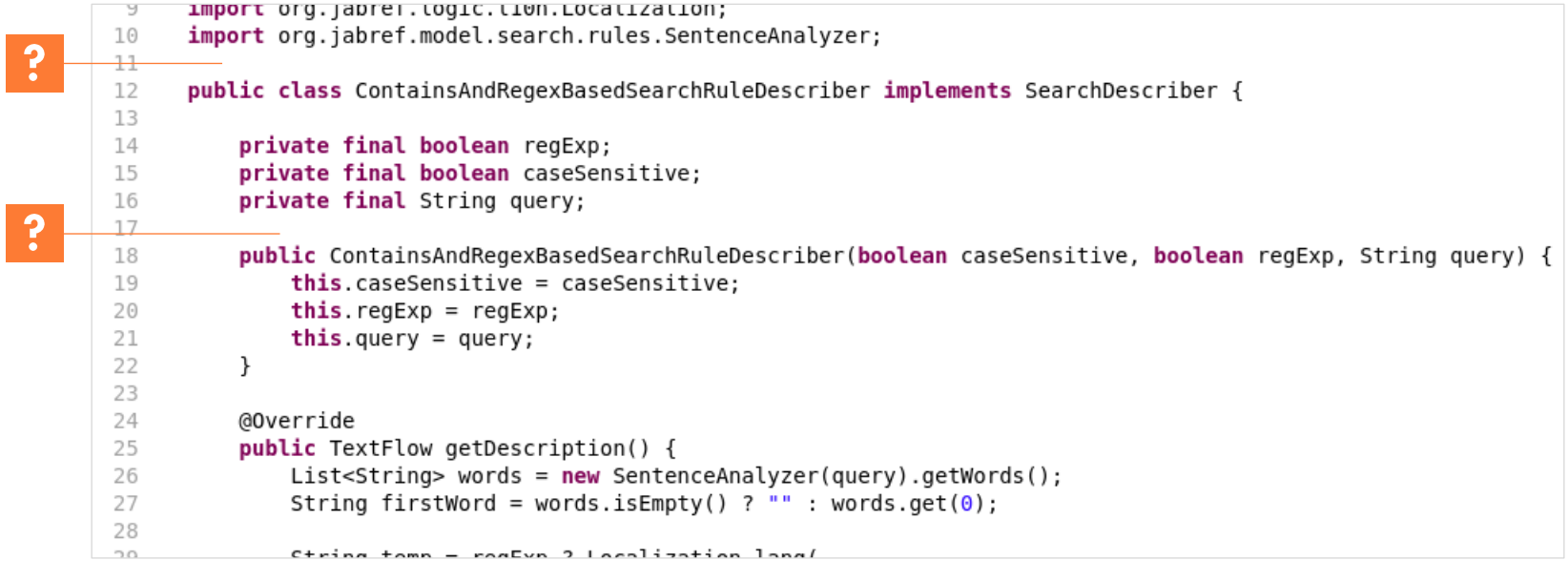
```
170
171     for (int k = 0; k < tokens.length; k++) {
172         String token = tokens[k];
173         if (abbreviateThatIsSingleLetter) {
174             String[] dashes = token.split("-");
175
176             token = Arrays.asList(dashes).stream().map(BibtexNameFormatter::getFirstCharOfString)
177                     .collect(Collectors.joining(".-"));
178         }
179
180         // Output token
181         sb.append(token);
182
183         if (k < (tokens.length - 1)) {
184             // Output Intertoken String
185             if (interToken == null) {
186                 if (abbreviateThatIsSingleLetter) {
187                     sb.append('.');
188                 }
189                 // No clue what this means (What the hell are tokens anyway???
190                 // if (lex_class[name_sep_char[cur_token]] = sep_char) then
191                 //     append ex buf char and check (name sep char[cur token])
```

	Methodenlänge	Schachtelung	**Klonüberdeckung**	Kommentierung	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1

```
125     }
126
127     private static void addResourceFile(String name, String resource, ZipOutpu
128         ZipEntry zipEntry = new ZipEntry(name);
129         out.putNextEntry(zipEntry);
130         OpenDocumentSpreadsheetCreator.addFromResource(resource, out);
131         out.closeEntry();
132     }
133
134     private static void addFromResource(String resource, OutputStream out) {
135         URL url = OpenDocumentSpreadsheetCreator.class.getResource(resource);
136         try (InputStream in = url.openStream()) {
137             byte[] buffer = new byte[256];
138             synchronized (out) {
139                 while (true) {
140                     int bytesRead = in.read(buffer);
141                     if (bytesRead == -1) {
142                         break;
143                     }
144                     out.write(buffer, 0, bytesRead);
145                 }
146             }
147         } catch (IOException e) {
148             LOGGER.warn("Cannot get resource", e);
149         }
150     }
151 }
152
```

```
105     }
106
107     private static void addResourceFile(String name, String resource, ZipOutpu
108         ZipEntry zipEntry = new ZipEntry(name);
109         out.putNextEntry(zipEntry);
110         OpenOfficeDocumentCreator.addFromResource(resource, out);
111         out.closeEntry();
112     }
113
114     private static void addFromResource(String resource, OutputStream out) {
115         URL url = OpenOfficeDocumentCreator.class.getResource(resource);
116         try (InputStream in = url.openStream()) {
117             byte[] buffer = new byte[256];
118             synchronized (out) {
119                 while (true) {
120                     int bytesRead = in.read(buffer);
121                     if (bytesRead == -1) {
122                         break;
123                     }
124                     out.write(buffer, 0, bytesRead);
125                 }
126             }
127         } catch (IOException e) {
128             LOGGER.warn("Cannot get resource", e);
129         }
130     }
131 }
132
```

	Methodenlänge	Schachtelung	Klonüberdeckung	**Kommentierung**	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1



	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	**Codeanomalien** pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1

```
438         List<String> names;
439         if (style.isSortByPosition()) {
440             // We need to sort the reference marks according to their
441             names = sortedReferenceMarks;
442         } else if (style.isNumberEntries()) {
443             // We need to sort the reference marks according to the so
444             // entries:
445             SortedMap<BibEntry, BibDatabase> newMap = new TreeMap<>(en
446             for (Map.Entry<BibEntry, BibDatabase> bibtexEntryBibtexDat
447                 newMap.put(bibtexEntryBibtexDatabaseEntry.getKey(), bi
448             }
449             entries = newMap;
450             // Rebuild the list of cited keys according to the sort or
451             cited.clear();
452             for (BibEntry entry : entries.keySet()) {
453                 cited.add(entry.getCiteKeyOptional().orElse(null));
454             }
455             names = Arrays.asList(xReferenceMarks.getElementNames());
456         } else {
457             names = sortedReferenceMarks;
458         }
459
460         // Remove all reference marks that don't look like JabRef cita
```

```
324          * Sets the title of the main window.
325          */
326         public void setWindowTitle() {
327             BasePanel panel = getCurrentBasePanel();
328
329             // no database open
330             if (panel == null) {
331                 // setTitle(FRAME_TITLE);
332                 return;
333             }
334
335             String mode = panel.getBibDatabaseContext().getMode().getFormattedName();
336             String modeInfo = String.format(" (%s)", Localization.lang("%0 mode", mode));
337             boolean isAutosaveEnabled = Globals.prefs.getBoolean(JabRefPreferences.LOCAL_A
338
339             if (panel.getBibDatabaseContext().getLocation() == DatabaseLocation.LOCAL) {
340                 String changeFlag = panel.isModified() && !isAutosaveEnabled ? "*" : "";
341                 String databaseFile = panel.getBibDatabaseContext()
342                                            .getDatabasePath()
343                                            .map(Path::toString)
344                                            .orElse(Localization.lang("untitled"));
345                 // setTitle(FRAME_TITLE + " - " + databaseFile + changeFlag + modeInfo);
346             } else if (panel.getBibDatabaseContext().getLocation() == DatabaseLocation.SHA
```

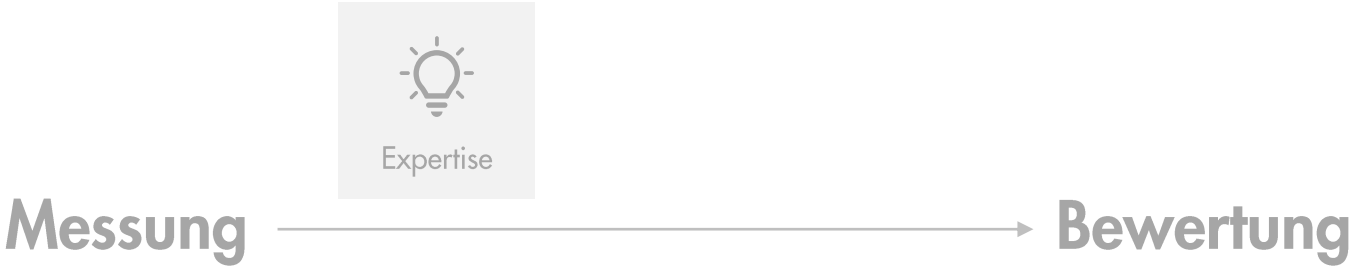
	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	Codeanomalien pro 1.000 SLOC	**Naming-Verstöße** pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1

```
 9  public class UndoableModifySubtree extends AbstractUndoableJabRefEdit {
10
11      /**
12       * A backup of the groups before the modification
13       */
14      private final GroupTreeNode m_groupRoot;
15
16      private final GroupTreeNode m_subtreeBackup;
17
18      /**
19       * The path to the global groups root node
20       */
21      private final List<Integer> m_subtreeRootPath;
22
23      /**
24       * This holds the new subtree (the root's modified children) to allow redo.
25       */
```

	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1

Handlungsbedarf ?

Messung ≠ Bewertung !


Expertise
Messung
Bewertung

The first rule of functions is that they should be small. The second rule of functions is that *they should be smaller than that.* […] **Functions should hardly ever be 20 lines long**.

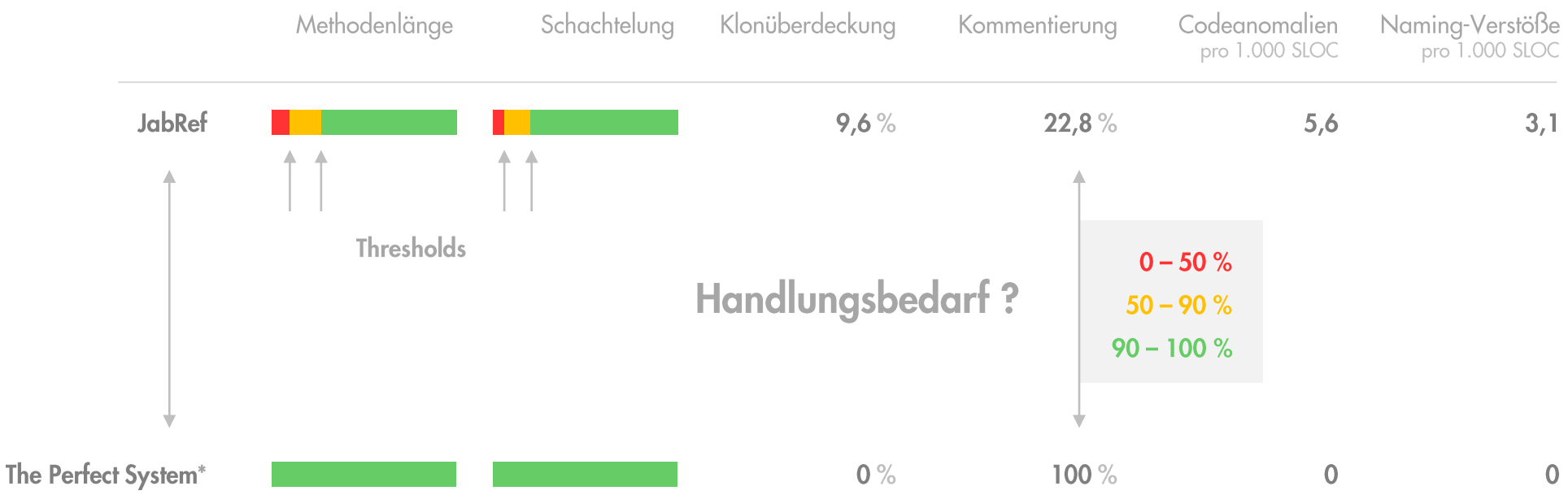
– Robert C. Martin, 2008

Number one in the stink parade is duplicated code. If you see the same code structure in more than one place, you can be sure that your program will be better if you find a way to unify them.

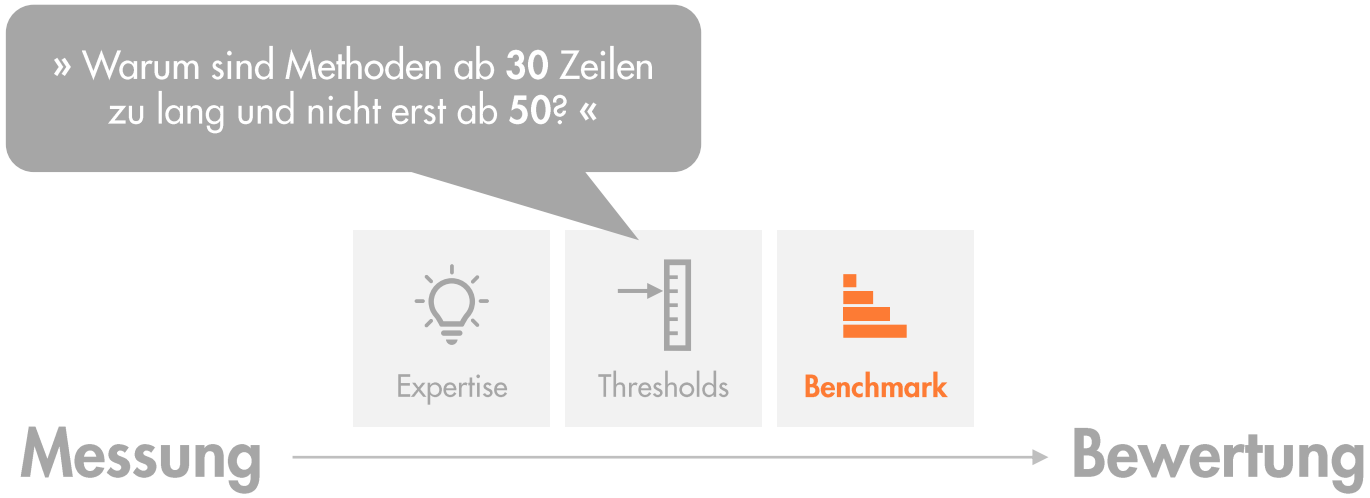
– Martin Fowler, 1999

Naming conventions make programs more understandable by making them easier to read. They can also give information about the function of the identifier […]

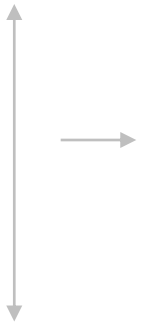
– Java Code Conventions, 1997



* Existiert nicht


» Warum sind Methoden ab 30 Zeilen zu lang und nicht erst ab 50? «
Expertise
Thresholds
Benchmark
Messung
Bewertung

	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1
The ~~Perfect~~ Average System			? %	? %	?	?



→ Handlungsbedarf

» Ja, aber … «

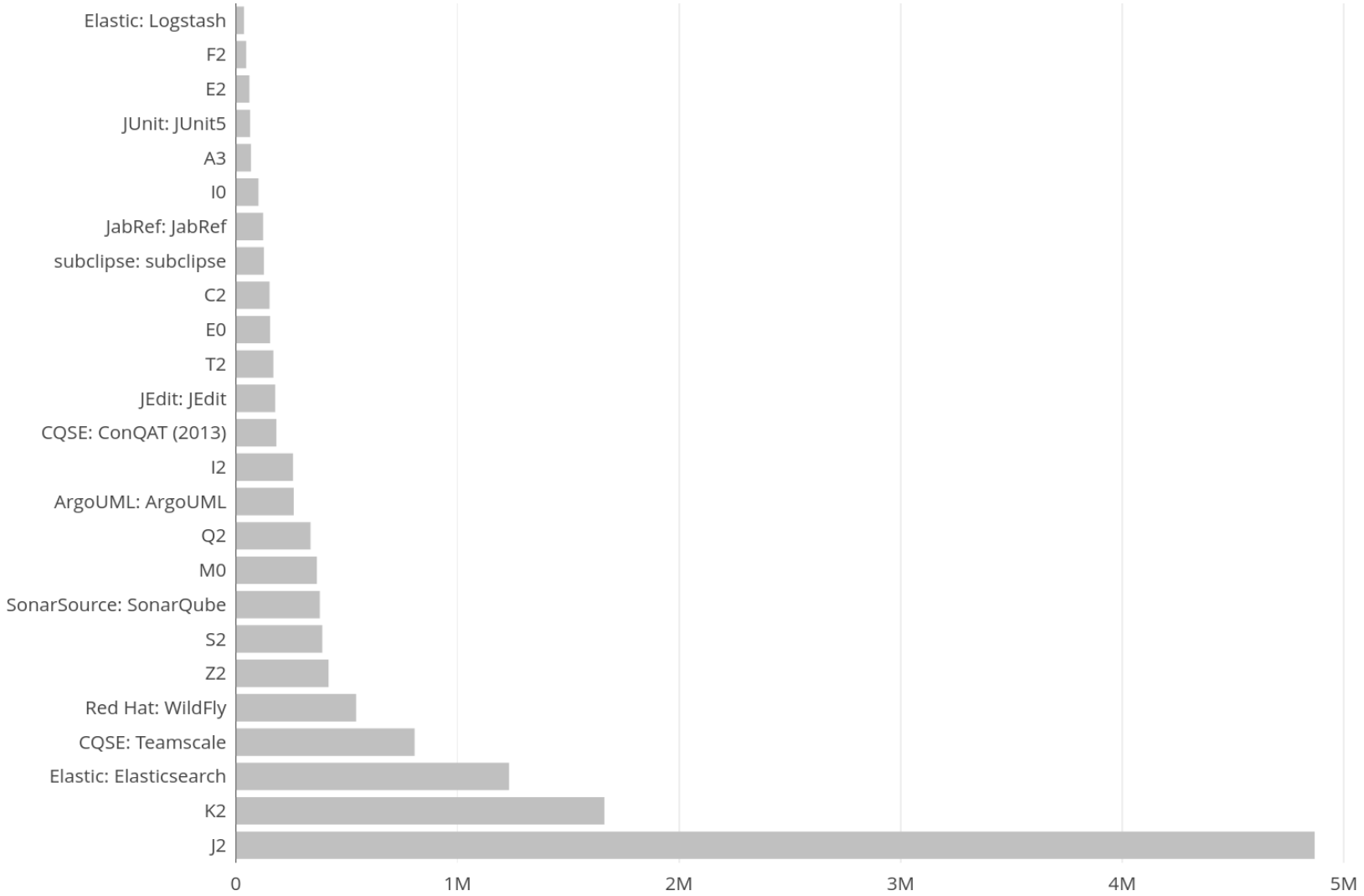
	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
JabRef			9,6 %	22,8 %	5,6	3,1
SonarQube			6,6 %	27,9 %	0,9	0,2

Klonüberdeckung pro Datei in JabRef

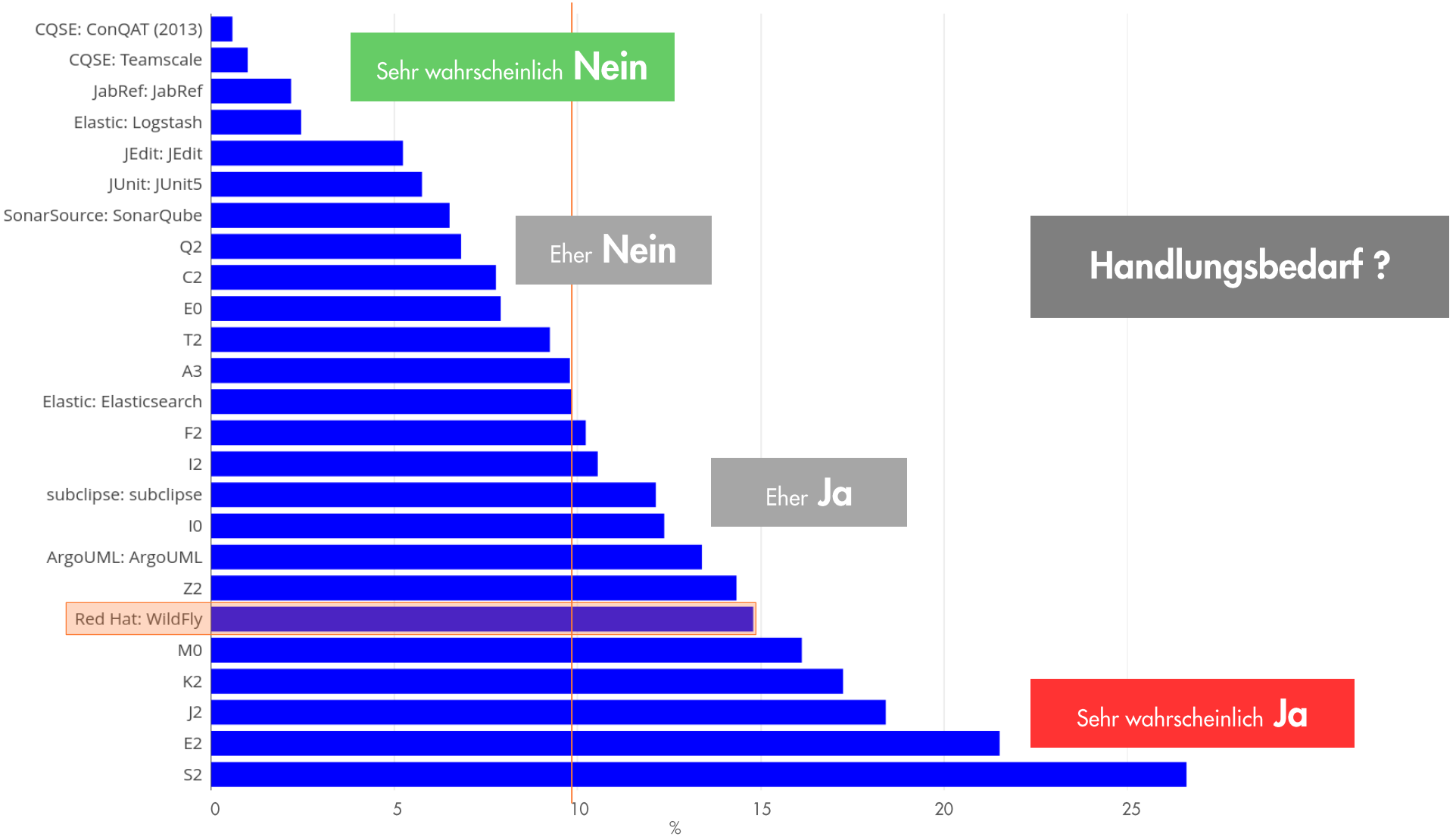
	Methodenlänge	Schachtelung	Klonüberdeckung	Kommentierung	Codeanomalien pro 1.000 SLOC	Naming-Verstöße pro 1.000 SLOC
~~JabRef~~			~~9,6 %~~	~~22,8 %~~	~~5,6~~	~~3,1~~
JabRef			**2,2 %**	**24,8 %**	**7,9**	**1,2**
SonarQube			**6,6 %**	**27,9 %**	**0,9**	**0,2**

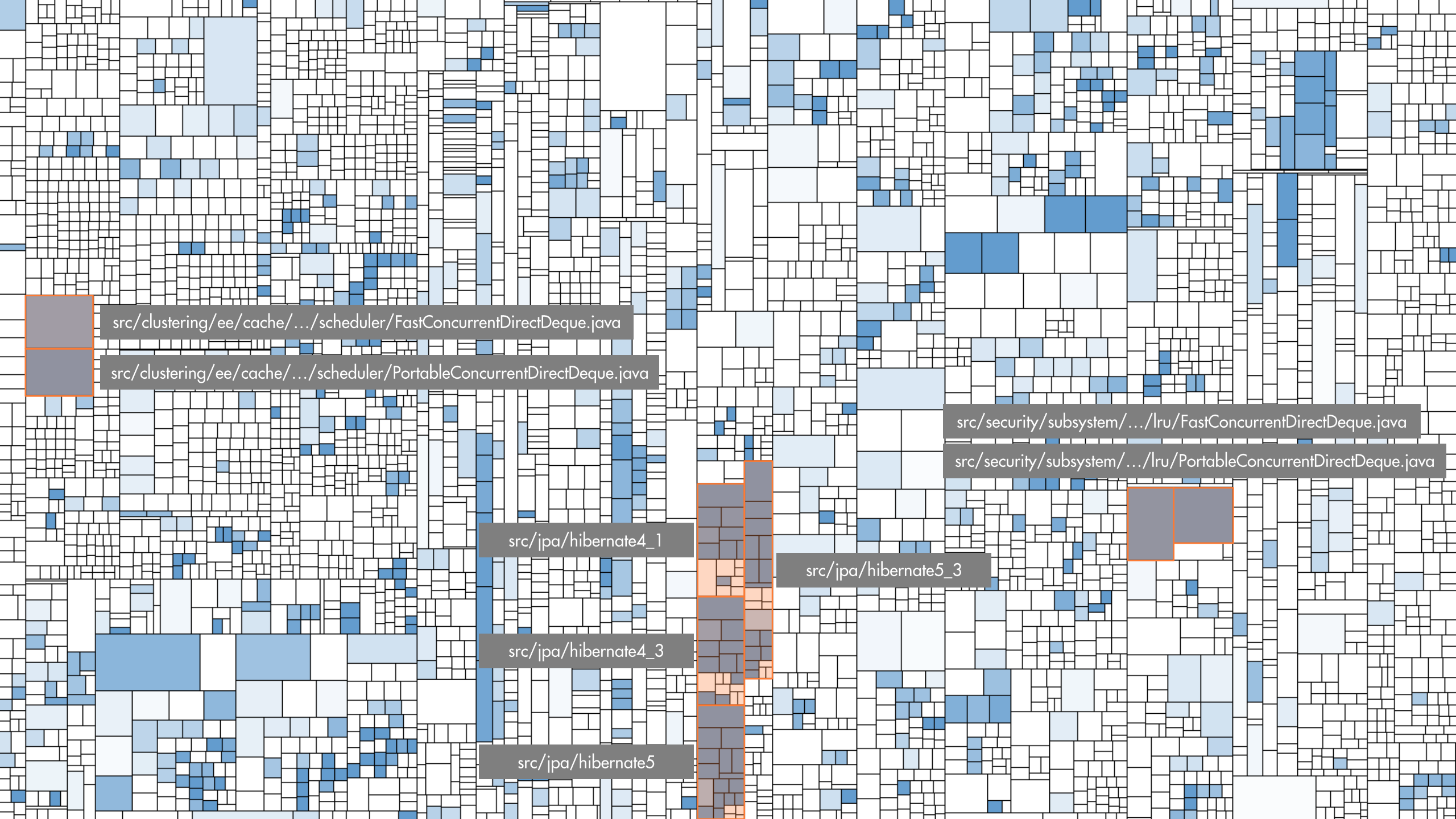
CQSE-Benchmark

Lines of Code



Klonüberdeckung




src/clustering/ee/cache/…/scheduler/FastConcurrentDirectDeque.java
src/clustering/ee/cache/…/scheduler/PortableConcurrentDirectDeque.java
src/security/subsystem/…/lru/FastConcurrentDirectDeque.java
src/security/subsystem/…/lru/PortableConcurrentDirectDeque.java
src/jpa/hibernate4_1
src/jpa/hibernate5_3
src/jpa/hibernate4_3
src/jpa/hibernate5

src/jpa/hibernate4_1

```
    private org.hibernate.stat.Statistics getBaseStatistics(EntityManagerFactory entityMa
        if (entityManagerFactory == null) {
            return null;
        }
        HibernateEntityManagerFactory entityManagerFactoryImpl = (HibernateEntityManagerF
        SessionFactory sessionFactory = entityManagerFactoryImpl.getSessionFactory();
        if (sessionFactory != null) {
            return sessionFactory.getStatistics();
        }
        return null;
    }

    private org.hibernate.stat.EntityStatistics getStatistics(EntityManagerFactory entity
        HibernateEntityManagerFactory entityManagerFactoryImpl = (HibernateEntityManagerF
        SessionFactory sessionFactory = entityManagerFactoryImpl.getSessionFactory();
        if (sessionFactory != null) {
            return sessionFactory.getStatistics().getEntityStatistics(entityName);
        }
        return null;
    }

    private Operation entityDeleteCount = new Operation() {
```

src/jpa/hibernate5

```
    private org.hibernate.stat.Statistics getBaseStatistics(EntityManagerFactory entityMa
        if (entityManagerFactory == null) {
            return null;
        }
        SessionFactory sessionFactory = entityManagerFactory.unwrap(SessionFactory.class)
        if (sessionFactory != null) {
            return sessionFactory.getStatistics();
        }
        return null;
    }

    private org.hibernate.stat.EntityStatistics getStatistics(EntityManagerFactory entity
        if (entityManagerFactory == null) {
            return null;
        }
        SessionFactory sessionFactory = entityManagerFactory.unwrap(SessionFactory.class)
        if (sessionFactory != null) {
            return sessionFactory.getStatistics().getEntityStatistics(pathAddress.getValu
        }
        return null;
    }

    private Operation entityDeleteCount = new Operation() {
```

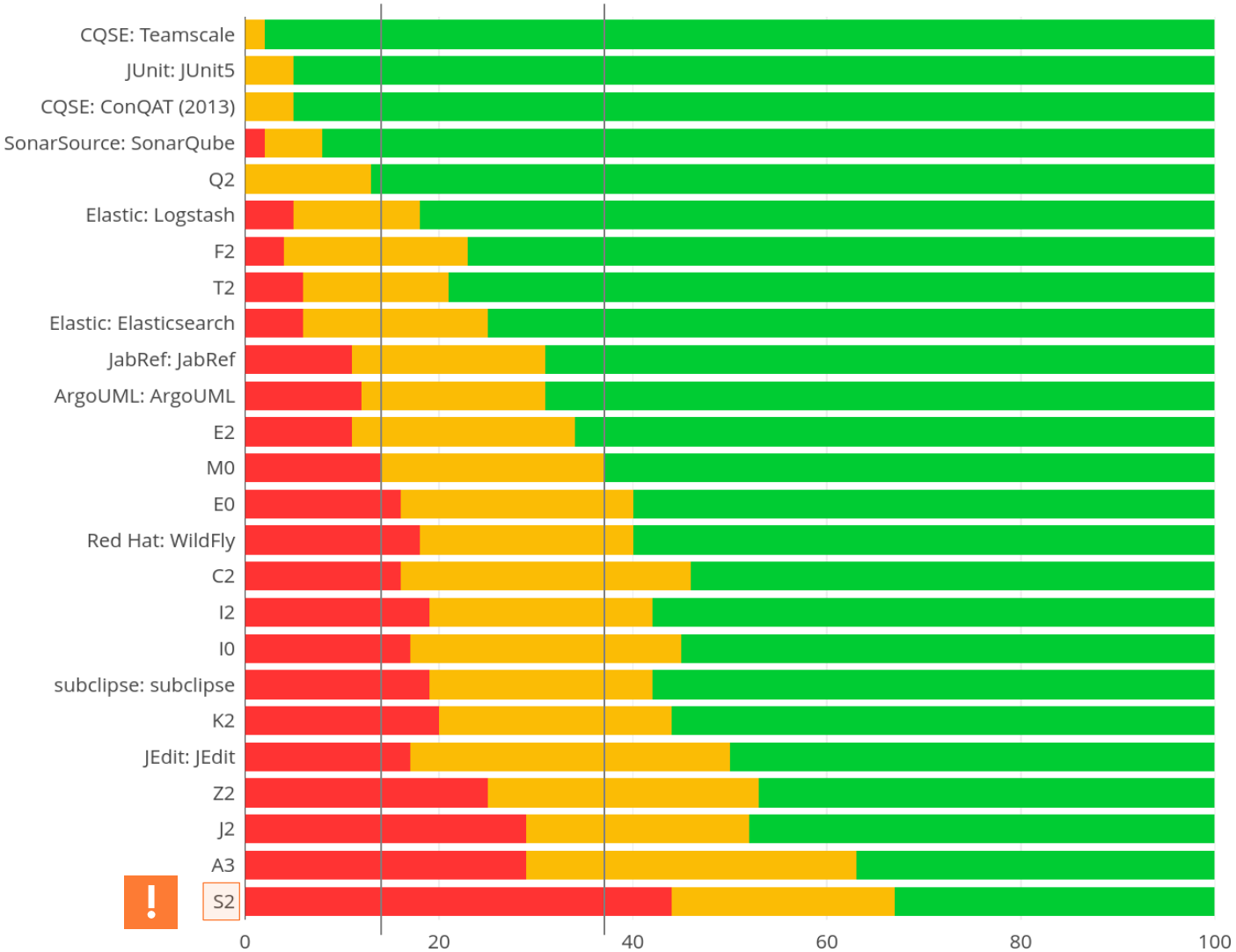
src/jpa/hibernate4_3

```
    private org.hibernate.stat.Statistics getBaseStatistics(EntityManagerFactory entityMa
        if (entityManagerFactory == null) {
            return null;
        }
        HibernateEntityManagerFactory entityManagerFactoryImpl = (HibernateEntityManagerF
        SessionFactory sessionFactory = entityManagerFactoryImpl.getSessionFactory();
        if (sessionFactory != null) {
            return sessionFactory.getStatistics();
        }
        return null;
    }

    private org.hibernate.stat.EntityStatistics getStatistics(EntityManagerFactory entity
        if (entityManagerFactory == null) {
            return null;
        }
        HibernateEntityManagerFactory entityManagerFactoryImpl = (HibernateEntityManagerF
        SessionFactory sessionFactory = entityManagerFactoryImpl.getSessionFactory();
        if (sessionFactory != null) {
            return sessionFactory.getStatistics().getEntityStatistics(entityName);
        }
        return null;
    }

    private Operation entityDeleteCount = new Operation() {
```

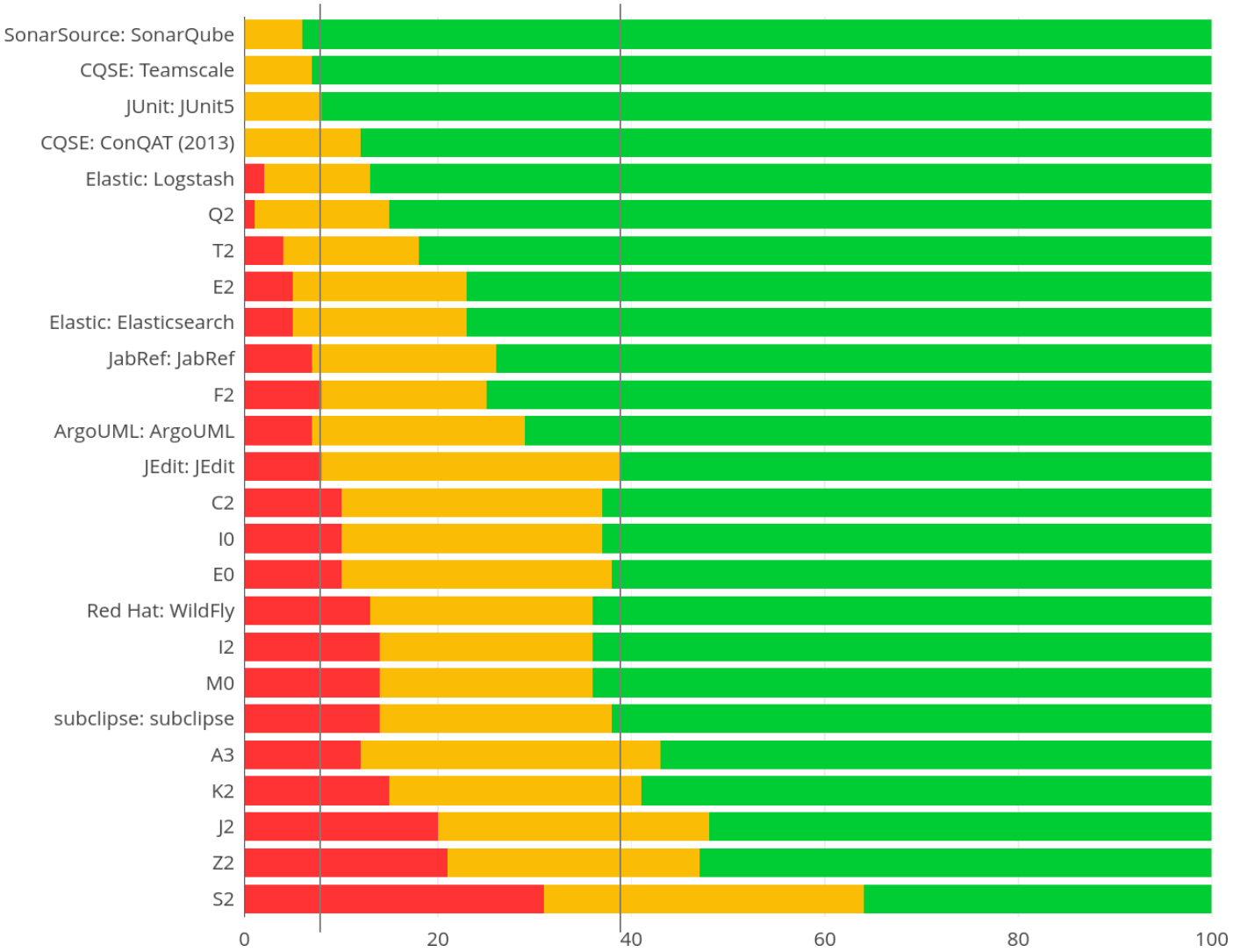
src/jpa/hibernate5_3

```
    private org.hibernate.stat.Statistics getBaseStatistics(EntityManagerFactory entityMa
        if (entityManagerFactory == null) {
            return null;
        }
        SessionFactory sessionFactory = entityManagerFactory.unwrap(SessionFactory.class)
        if (sessionFactory != null) {
            return sessionFactory.getStatistics();
        }
        return null;
    }

    private org.hibernate.stat.EntityStatistics getStatistics(EntityManagerFactory entity
        if (entityManagerFactory == null) {
            return null;
        }
        SessionFactory sessionFactory = entityManagerFactory.unwrap(SessionFactory.class)
        if (sessionFactory != null) {
            return sessionFactory.getStatistics().getEntityStatistics(pathAddress.getValu
        }
        return null;
    }

    private Operation entityDeleteCount = new Operation() {
```

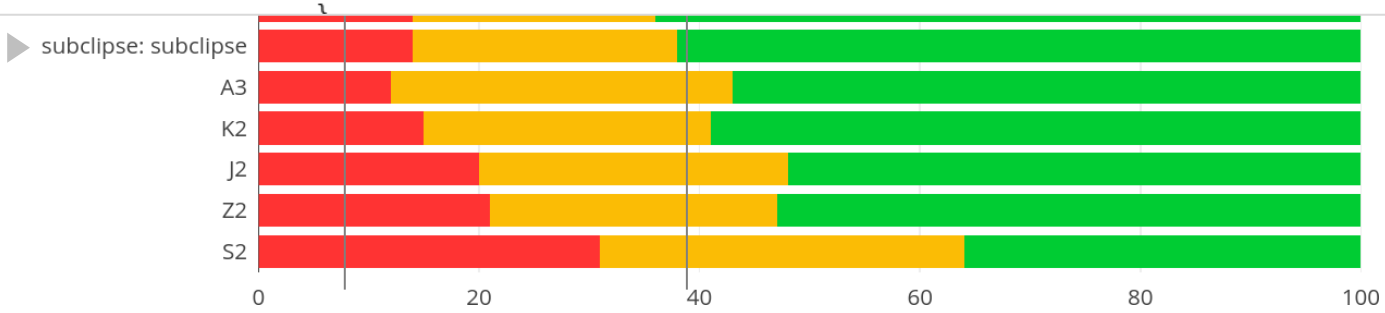
Methodenlänge



Schachtelungstiefe

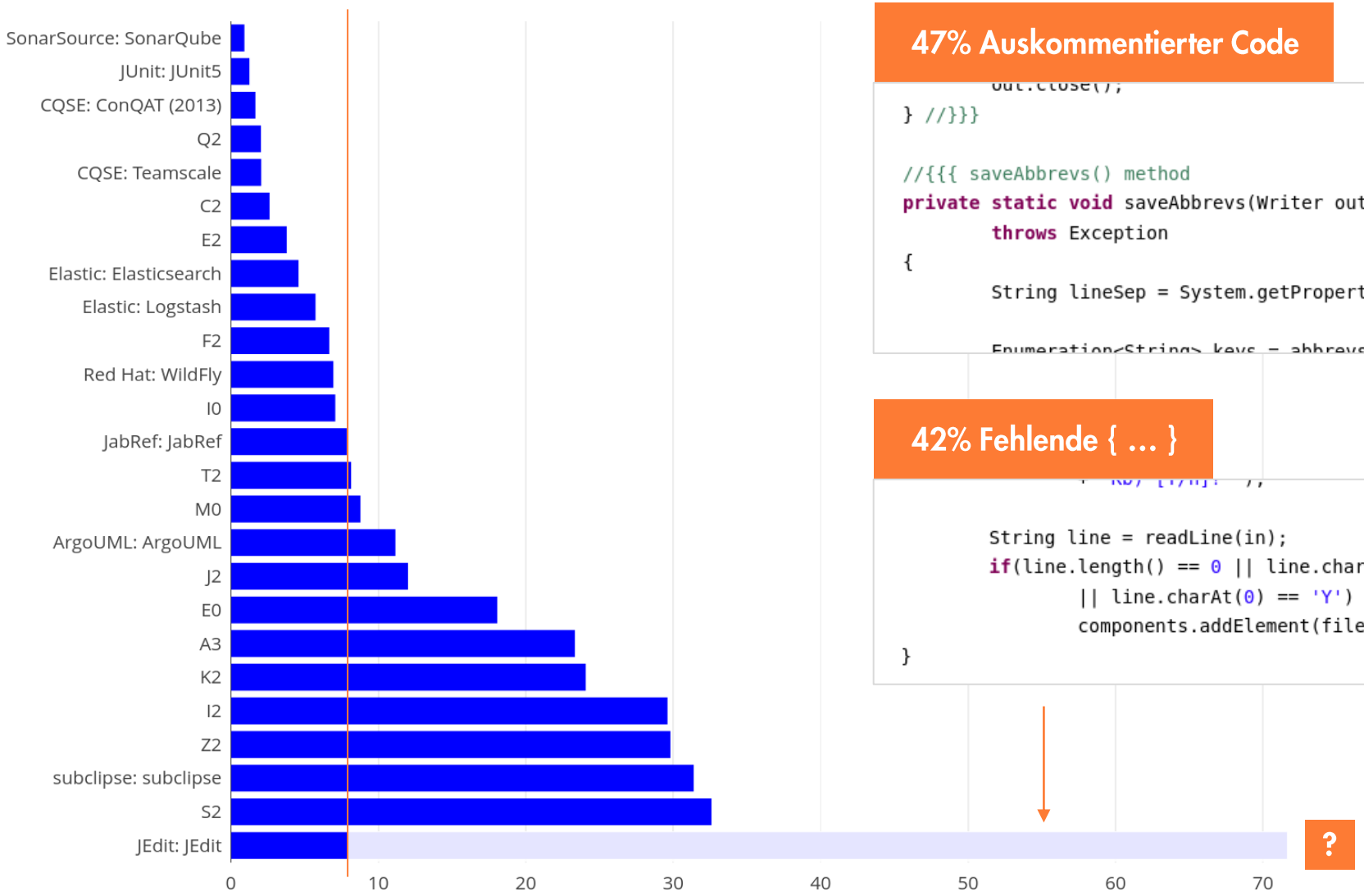


```
showCompareButton.setSelection(showCompare);
showCompareButton.addSelectionListener(
        new SelectionListener(){
            public void widgetSelected(SelectionEvent e) {
                showComparePane(!showCompare);
                if (showCompare) {
                    IStructuredSelection selection = (IStructuredSelection)resourceSelectionTree.getTreeViewer().getSelection();
                    if (!selection.isEmpty()) {
                        Object sel0 = selection.getFirstElement();
                        if (sel0 instanceof IFile) {
                            final ISVNLocalResource localResource= SVNWorkspaceRoot.getSVNResourceFor((IFile)sel0);
                            try {
                                // if any alternate compare actions are defined from the extension point
                                // then call those actions instead of showing the default compare dialog
                                if (alternateCompareActions.length > 0) {
                                    StructuredSelection localResourceSelection = new StructuredSelection(localResource);
                                    for (int i = 0; i < alternateCompareActions.length; i++) {
                                        // make sure the selection is up to date
                                        alternateCompareActions[i].selectionChanged(localResourceSelection);
                                        alternateCompareActions[i].run();
                                    }
                                } else {
                                    setCompareInput(new SVNLocalCompareInput(localResource, SVNRevision.BASE, true));
                                }
                            } catch (Exception e1) {
                            }
                        }
                    }
                }
```



Kommentierung

Codeanomalien / 1.000 SLOC



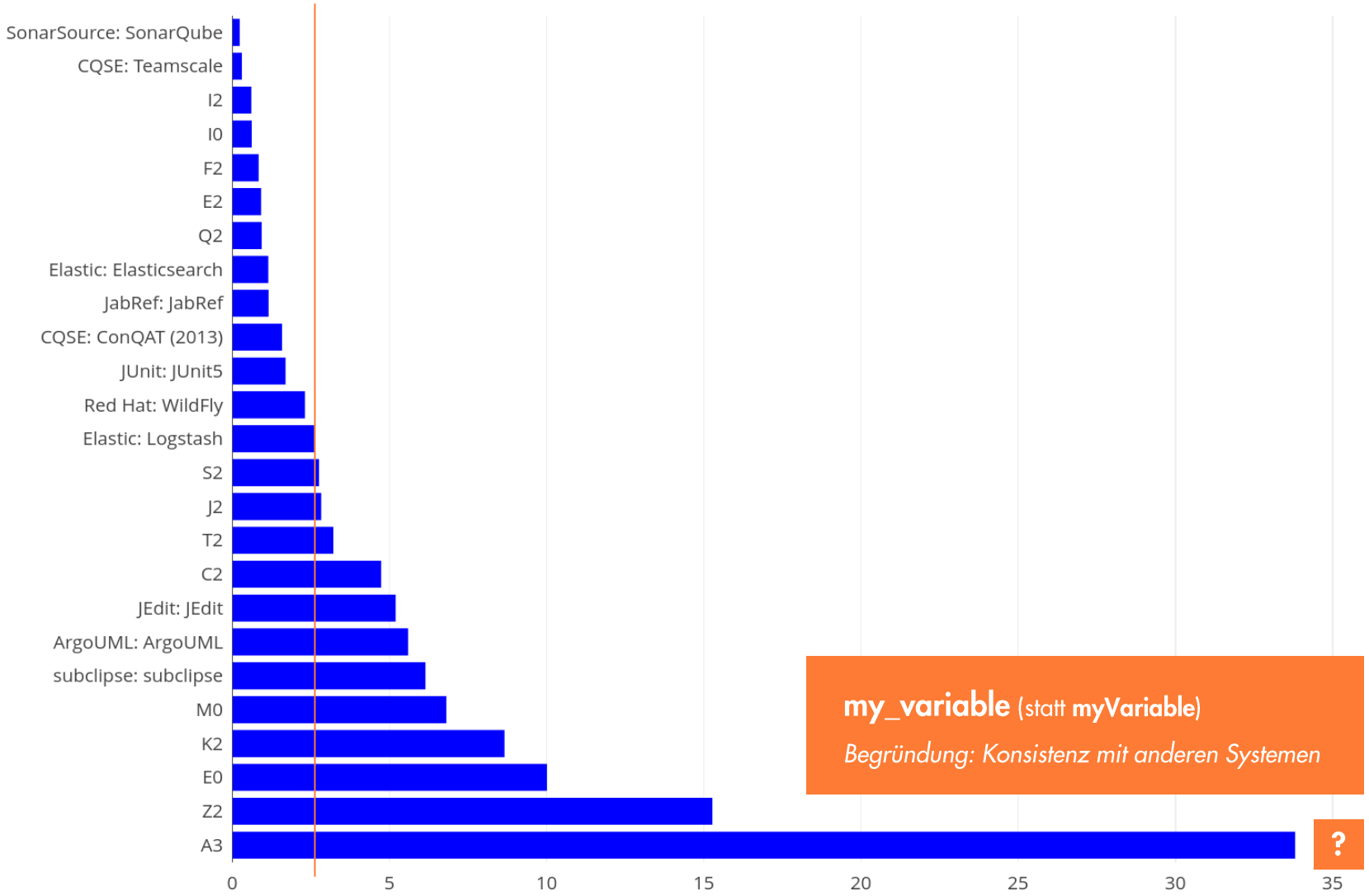
47% Auskommentierter Code

```
        out.close();
} //}}}

//{{{ saveAbbrevs() method
private static void saveAbbrevs(Writer out, Hashtable<String,Str
        throws Exception
{
        String lineSep = System.getProperty("line.separator");

        Enumeration<String> keys = abbrevs.keys();
```

42% Fehlende { ... }

```
        String line = readLine(in);
        if(line.length() == 0 || line.charAt(0) == 'y'
                || line.charAt(0) == 'Y')
                components.addElement(fileset);
}
```

Verletzungen der Namenskonventionen / 1.000 SLOC



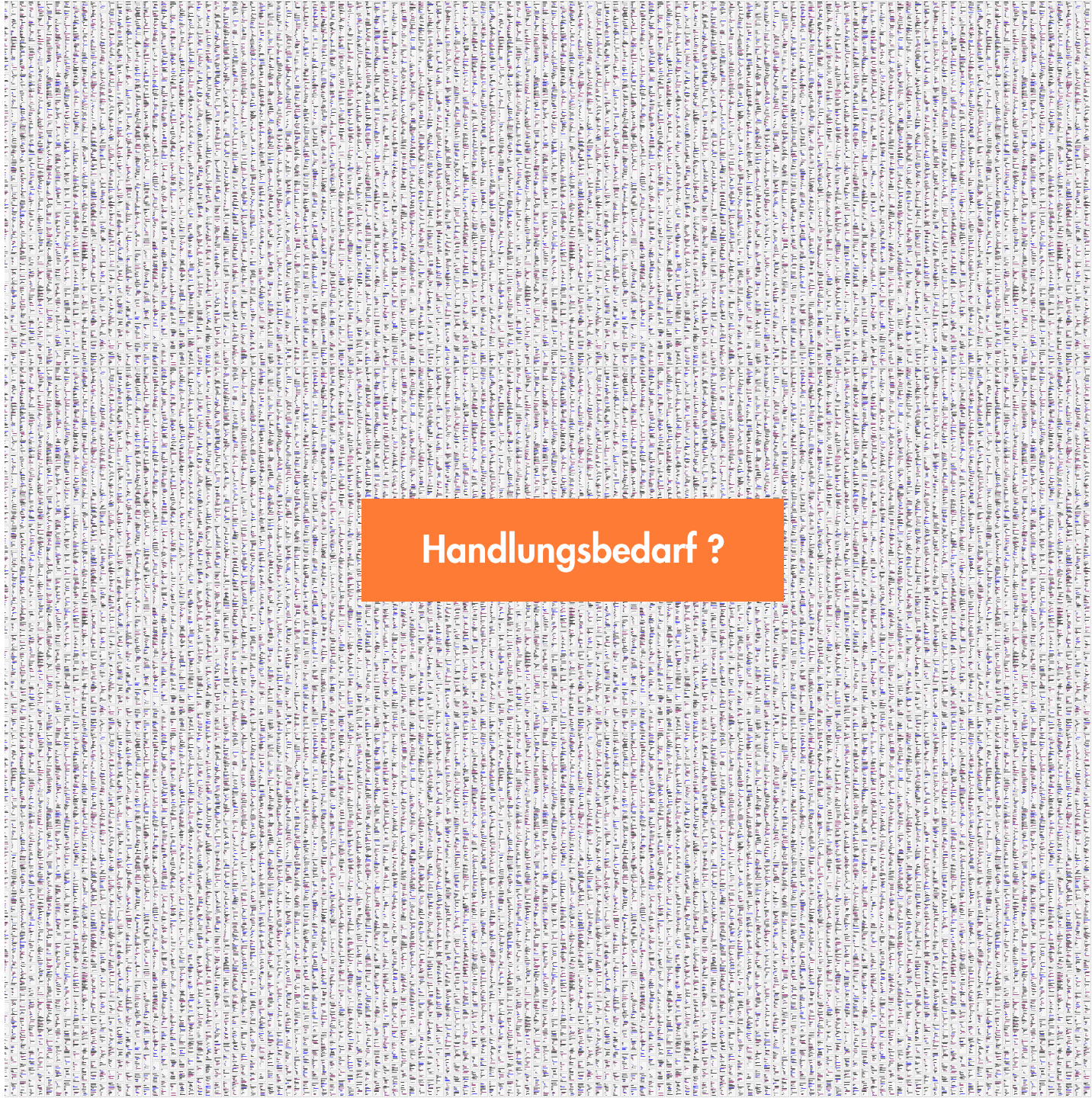
Ein Benchmark …

… ist ein wichtiger Baustein um von der **Messung zur Bewertung** zu kommen

… erlaubt von konkreten **Thresholds** und Check-Auswahl zu **abstrahieren**

… ist **keine automatische Bewertung** und muss interpretiert werden

… muss **gewissenhaft kuratiert** werden



Bring Your Own Code Workshop: Teamscale am eigenen Quelltext kennenlernen

23.09.2020 by Jakob Rott

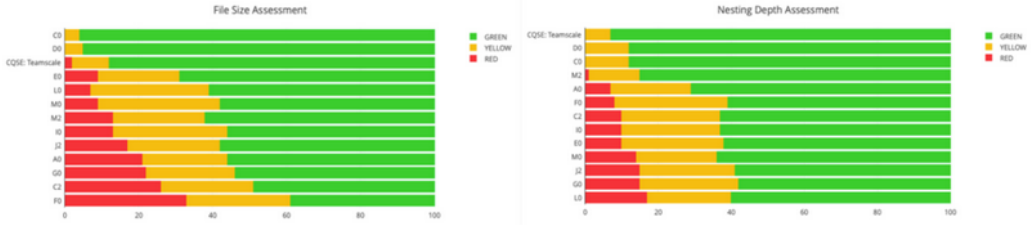
Häufig erreichen uns Anfragen von Interessenten, die unsere Analysen zu Software Qualität spannend finden, aber bisher noch im Unklaren darüber sind, wie sie diese in das eigene Projekt einfach und sinnvoll integrieren können.

In **Bring Your Own Code** (BYOC) Workshops möchten wir innerhalb von kurzer Zeit

- deutlich machen, dass die Einrichtung von Analysen und Integrationen in bestehende Tools und Workflows mit beschränktem Zeitaufwand möglich ist. Dafür setzen wir eine Teamscale-Instanz zur Analyse des Projekt-Codes auf.
- die Analysen mit einem Entwickler aus dem Team validieren.
- in einem Anschlusstreffen die Analyseergebnisse gemeinsam inspizieren und interpretieren.

Dashboardansicht — Ausschnitt der Analyseergebnisse

- Kennzahlen hinsichtlich der Softwarequalität des Projekts in unseren Benchmark einordnen: Durch frühere Analysen vergleichbarer (z. B. Größe und Sprache) Systeme lassen sich Metriken gegenüberstellen.



Einordnung der Kennzahlen aus dem eigenen Projekt in den Benchmark

Ablauf BYOC-Workshop

Dr. Nils Göde · goede@cqse.eu · +49 176 10452662

CQSE GmbH
Centa-Hafenbrädl-Straße 59
81249 München

cqse.eu/codedays22021

CQSE
Continuous Quality in Software Engineering