» Where are *problems* in our *architecture*? «

» Which changes were *not reviewed*? «

» Are there *head monopolies*? «

» Where are *technical debts*? «

» Which components are *susceptible to bugs*? «

» Are *critical bugs* fixed in a timely manner? «

» Where are *gaps in our test*? «

» Which code is *unused*? «

» ...? «

**Software Intelligence**

![Teamscale Logo](Teamscale Logo)

- Code
- Models
- Version History
- Tickets
- Test Coverage
- Test Results
- Usage Data
- Reviews
- ...
Where are problems in our architecture? «

» Which changes were not reviewed? «

» Are there head monopolies? «

» Where are technical debts? «

» Which components are susceptible to bugs? «

» Are critical bugs fixed in a timely manner? «

» Where are gaps in our test? «

» Which code is unused? «

Software Intelligence

Teamscale
The variable ‘r’ may contain a null value at this point and is dereferenced

Code Anomalies → Null pointer dereference → Dataflow/Null pointer dereference

The variable that is dereferenced in this statement may contain a null value. This may lead to an exception being thrown at runtime. There are two reasons for this finding: Either the variable has explicitly been assigned a null value at some point before this statement and is now being dereferenced, or the variable is checked for a null value at some point in the

read more

in core/src/main/java/hudson/util/Service.java:110

Note: The code below refers to the last version before the finding's removal. Latest version of the file on this branch.

```java
85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116
public static <T> void load(Class<T> spi, ClassLoader cl, Collection<Class<? extends T>> result) {
    try {
        Enumeration<URL> e = cl.getResources("META-INF/services/" + spi.getName());
        while (e.hasMoreElements()) {
            BufferedReader r = null;
            URL url = e.nextElement();
            try {
                r = new BufferedReader(new InputStreamReader(url.openStream(), "UTF-8"));
                String line;
                while (r.readLine() != null) {
                    if (line.startsWith("#"))
                        continue; // comment line
                    line = line.trim();
                    if (line.length() == 0)
                        continue; // empty line. ignore.
                    try {
                        result.add(cl.loadClass(line).asSubclass(spi));
                    } catch (ClassNotFoundException x) {
                        LOGGER.log(Level.WARNING, "Failed to load "+line, x);
                    }
                }
            } catch (IOException x) {
                LOGGER.log(Level.WARNING, "Failed to load "+url, x);
            } finally {
                r.close();
            }
        }
    } catch (IOException x) {
        LOGGER.log(Level.WARNING, "Failed to look up service providers for "+spi, x);
    }
}
```
```java
if (lhs == null) return;
if (rhs == null) return;

Constructor<?> lc = findDataBoundConstructor(lhs.getClass());
Constructor<?> rc = findDataBoundConstructor(rhs.getClass());
assertEquals("Data bound constructor mismatch. Different type?", lc, rc);

List<String> primitiveProperties = new ArrayList<String>(Arrays.asList(...));
String[] names = ClassDescriptor.loadParameterNames(lc);
Class<?>[] types = lc.getParameterTypes();
assertEquals(names.length, types.length);
for (int i = 0; i < types.length; i++) {
    Object lv = ReflectionUtils.getPublicProperty(lhs, names[i]);
    Object rv = ReflectionUtils.getPublicProperty(rhs, names[i]);
    if (Iterables.class.isAssignableFrom(types[i])) {
        Iterable lvCol = (Iterable) lv;
        Iterable rvCol = (Iterable) rv;
        for (Object ltrCol : lvCol) {
            for (Object rtrCol : rvCol) {
                if (ltrCol == rtrCol) continue;
                Object ltr = ltrCol.iterator() ;
                Object rtr = rtrCol.iterator() ;
                if (litr.hasNext() && rtrr.hasNext()) {
                    Object ll = ltr.next();
                    Object lr = ltr.next();
                    Object rl = rtr.next();
                    Object rr = rtr.next();
                    if (findDataBoundConstructor(litr.getClass()) != null) {
                        assertEquals(litr, rtr);
                    } else {
                        // assertEqualDataBoundBeans(litr, rtrr);
                    }
                } else {
                    // assertEqualDataBoundBeans(litr, rtrr);
                }
            }
        }
    } else {
        Iterator ltr = lv.iterator();
        Iterator rtr = rv.iterator();
        if (ltr.hasNext() && rtr.hasNext()) {
            Object ll = ltr.next();
            Object lr = ltr.next();
            Object rl = rtr.next();
            Object rr = rtr.next();
            if (findDataBoundConstructor(litr.getClass()) != null) {
                assertEqualDataBoundBeans(litr, rtrr);
            } else {
                // assertEqualDataBoundBeans(litr, rtrr);
            }
        }
    }
}

// assertEqualDataBoundBeans(lv, rv);
// assertEqualDataBoundBeans(rv, lv);

// compare shallow primitive properties
if (!primitiveProperties.isEmpty())
    assertEqualBeans(lhs, rhs, Util.join(primitiveProperties, ", ");

Makes sure that two collections are identical via @link #assertEqualDataBoundBeans(Object collection, Object collection2, String message).
```
17,207 Findings
Version Control Systems

Incremental Analysis Engine

NoSQL Store

REST Service Layer

Clients

*Teamscale*
Merge pull request #3801 from daniel-beck/update-triggerRemotely-documentation
by Oleq Nenashev in revision 6dd877ab in branch master (repository1)
Merged from daniel-beck/update-triggerRemotely-documentation
Findings:

[JENKINS-55197] Correction of the InstallUncaught NPE (#3806)
by Wadecck Follonier in revision 0x6a3142 in branch master (repository1)
Files: 1 changed
Issue: JENKINS-55197
Findings:

Merge pull request #3804 from jgllick/timeout-diagnosis
by Jesse Glick in revision b826d6b2 in branch master (repository1)
Merged from jgllick/timeout-diagnosis
Files: 5 changed
Findings:

Merge pull request #3800 from jeffret-b/remoting/3.28
by Jeff Thompson in revision 2853de4c in branch master (repository1)
Merged from jeffret-b/remoting/3.28
Findings:

Fixing a SIOOBE encountered during Windows-jdk11 tests (but not reproducible ... by Jesse Glick in revision 08d273f3 in branch jgllick/timeout-diagnosis (repository1)
Files: 1 changed
Findings:

Blocking Lifecycle restart from within JenkinsRule.
by Jesse Glick in revision 208b42a5 in branch jgllick/timeout-diagnosis (repository1)
Files: 3 changed
Findings:
Fix potential NPE on exception path
Findbug reported NP_NULL_ON_SOME_PATH_EXCEPTION, Priority: Normal
by Matthias Vach in revision 3d309db in branch mvach/potential-npe (repository)

Affected files (1)

<table>
<thead>
<tr>
<th>Path</th>
<th>Change type</th>
</tr>
</thead>
<tbody>
<tr>
<td>core/src/main/java/hudson/util/Service.java</td>
<td>changed</td>
</tr>
</tbody>
</table>

New findings

Old findings in changed methods

Removed findings

<table>
<thead>
<tr>
<th>Message</th>
<th>Location</th>
<th>Finding Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>The variable 't' may contain a null value</td>
<td>core/src/main/java/hudson/util/Service.java:1-10</td>
<td>Code Anomalies › Null pointer dereference...</td>
</tr>
<tr>
<td>Redundant literal 'UTF-8' occurs 60 times</td>
<td>core/src/main/java/hudson/util/Service.java:92</td>
<td>Code Duplication › Redundant Literals › Co...</td>
</tr>
</tbody>
</table>

Metric changes
Free Academic License

- Academic users, including students and research groups, can get a **free license** for Teamscale.

- Please contact me directly or send a mail to sales@teamscale.com
Contact

Roman Haas · haas@cqse.eu · +49 1590 458 3831

CQSE GmbH
Centa-Hafenbraedl-Strasse 59
81249 Munich
Germany