

# Bewertung von Wartbarkeit

## Jagdschein für Testleiter & Manager

**Dr. Elmar Juergens**  
**CQSE GmbH**



# Über Mich

## Forschung

- Clone Detection, Architekturanalyse
- Effektivität und Effizienz von QS-Maßnahmen



## Beratung

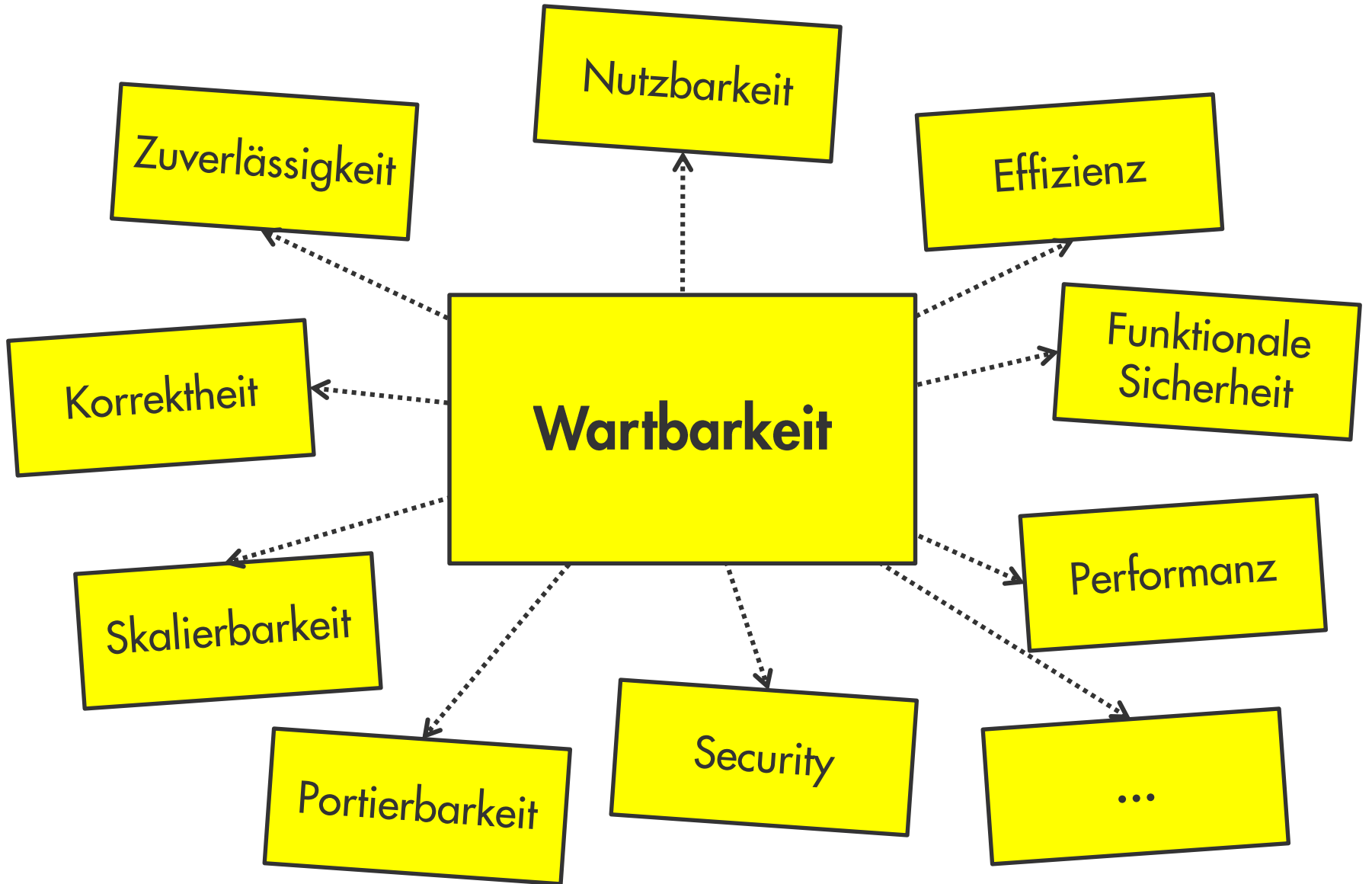
- Gründer
- Qualitäts-Bewertung & Qualitäts-Controlling



## Gesellschaft für Informatik

- Zum Junior-Fellow ernannt
- Erfahrungsaustausch Forschung <-> Praxis





```
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
        for (int i = 0; i < elements.length; i++) {
            ModelElement el = elements[i];
            res.append(showElementLink(el)).append(HTML.LINE_BREAK);
            found = true;
        }
        Index.getInstance().resetCurrentRenderer();
    }
    if (!found && nomsg != null && nomsg.length() > 0) {
        res.append(HTML.italics(nomsg));
    }
    return res.toString();
}
```

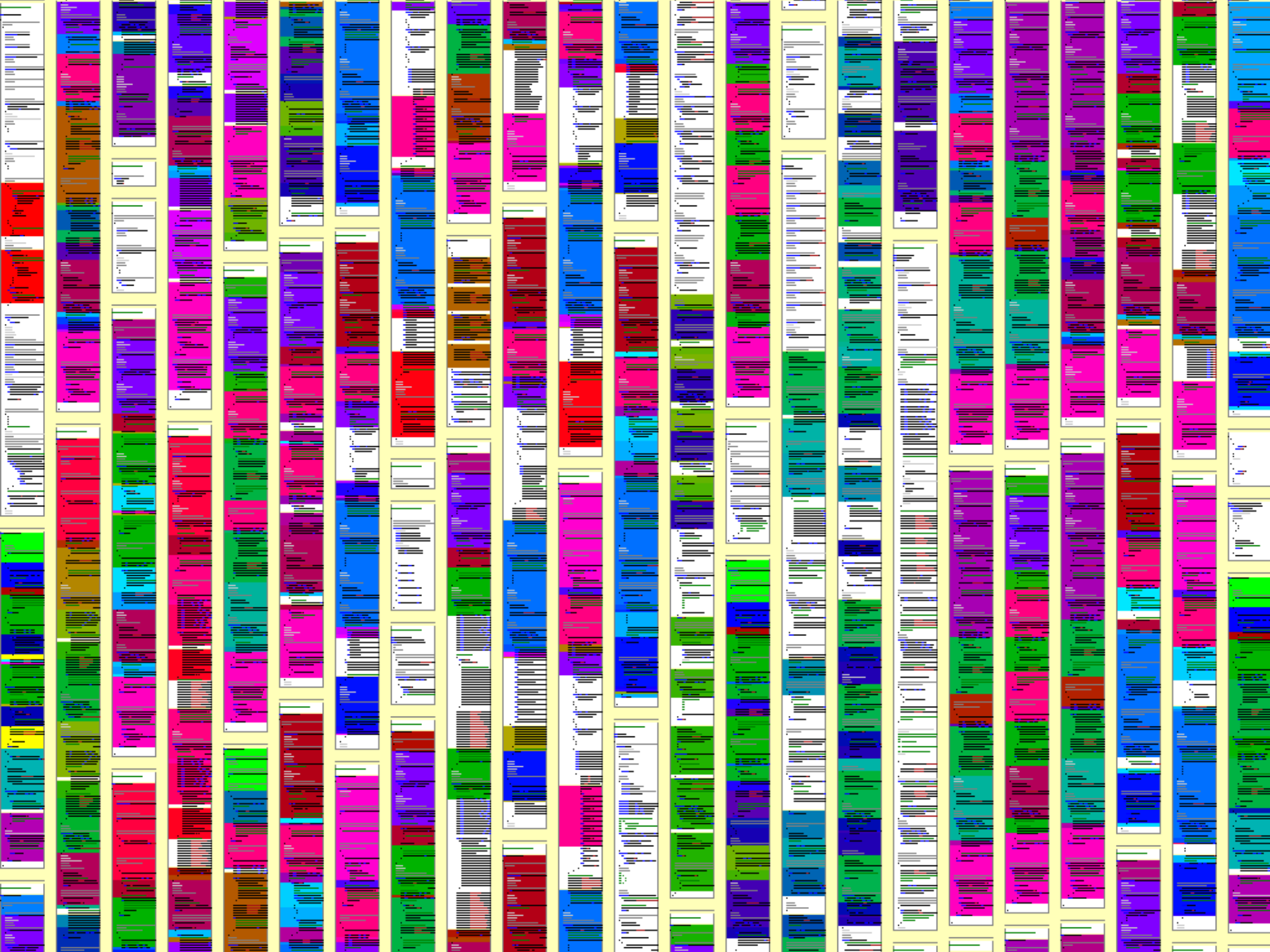
```
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
        for (int i = 0; i < elements.length; i++) {
            ModelElement el = elements[i];
            res.append(showElementLink(el)).append(HTML.LINE_BREAK);
            found = true;
        }
        Index.getInstance().resetCurrentRenderer();
    }
    if (!found && nomsg.length() > 0) {
        res.append(HTML.italics(nomsg));
    }
    return res.toString();
}
```

```
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
        for (int i = 0; i < elements.length; i++) {
            ModelElement el = elements[i];
            res.append(showElementLink(el)).append(HTML.LINE_BREAK);
            found = true;
        }
        Index.getInstance().resetCurrentRenderer();
    }
    if (!found && nomsg != null && nomsg.length() > 0) {
        res.append(HTML.italics(nomsg));
    }
    return res.toString();
}
```











## Studie

Munich Re 

- Über 100 Fehler in produktiver Software



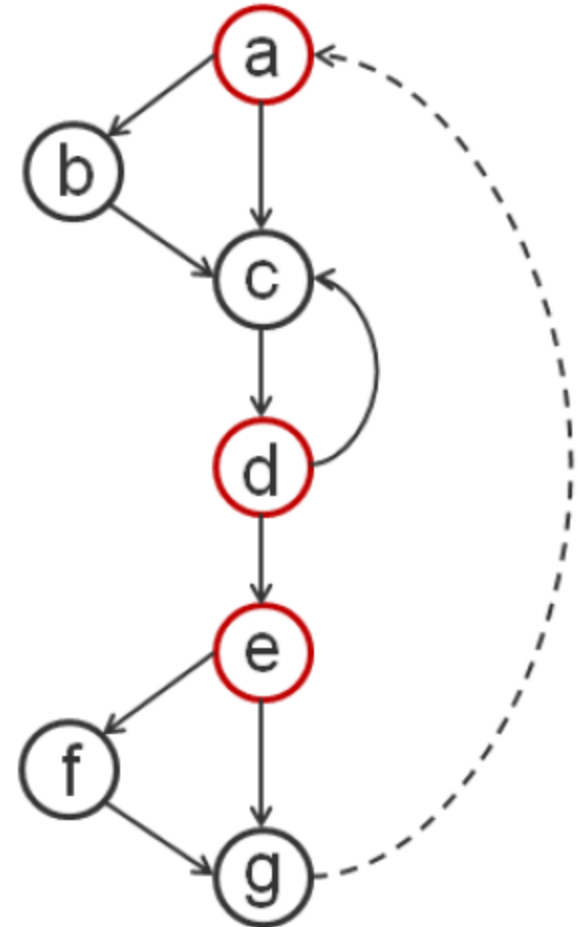
- 52% aller ungewollten Unterschiede fehlerhaft

Juergens, Deissenboeck et al: *Do Code Clones Matter?* ICSE 2009



?

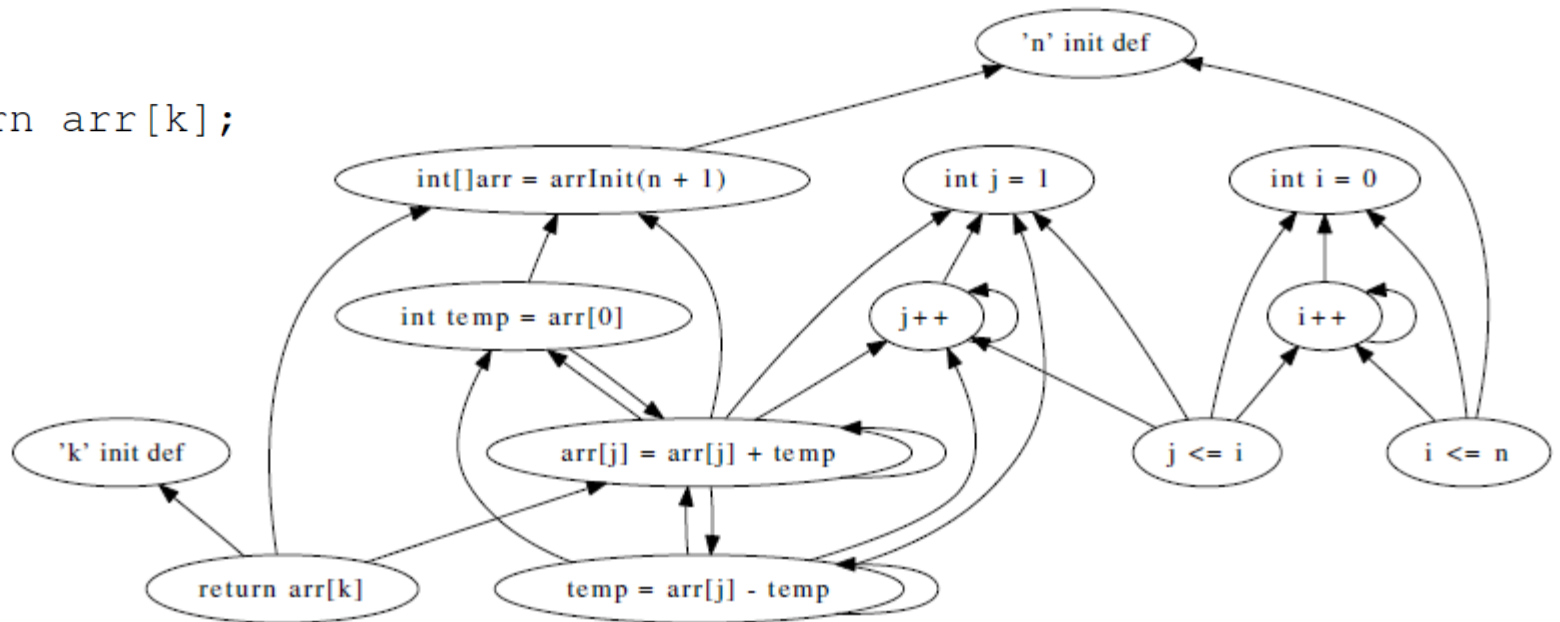
```
1 public void exampleCode(int a, int b, int c) {  
2     if(a < 0) {  
3         System.out.println("A ist negativ");  
4     }  
5  
6     do {  
7         b += 1;  
8     } while(b < 0);  
9  
10    if(c > 0) {  
11        System.out.println("C ist positiv");  
12    }  
13  
14    int z = a * b - c;  
15 }
```



```

// Require: n >= k >= 0
int bico(int n, int k) {
    int[] arr = arrInit(n+1);
    for (int i = 0; i <= n; i++) {
        int temp = arr[0];
        for (int j = 1; j < i; j++) {
            arr[j] = arr[j] + temp;
            temp = arr[j] - temp;
        }
    }
    return arr[k];
}

```



331	419	501	583	665	749	831	915	998	1078	1161	1243	1325	1407	1490	1574	1654	1735	1819	1901	1984	*	z. B. der Parameter a_display richtig gear
332	420	502	584	666	750	832	916	997	1079	1162	1244	1326	1408	1491	1575	1655	1736	1820	1902	1985	**	REFRESH lt_item_300.
333	421	503	585	667	751	833	917	998	1080	1163	1245	1327	1409	1492	1576	1656	1740	1821	1903	1986		ASSIGN ar_data_changed=&mp_mod_rowa->* TO
334	422	504	586	668	752	834	917	999	1081	1164	1246	1328	1410	1493	1577	1657	1741	1822	1904	1987		LOOP AT <<t_itema>> INTO gs_itema_300.
340	423	505	587	669	753	835	918	1000	1082	1165	1247	1329	1411	1494	1578	1658	1742	1823	1905	1988		READ TABLE gt_itema_300 INTO lt_item_300
341	424	506	588	670	754	836	919	1001	1083	1166	1248	1330	1412	1495	1579	1659	1743	1824	1906	1989		WITH KEY itm_number = gs_itema_300->itm
342	425	507	589	671	755	837	920	1002	1084	1167	1249	1331	1413	1496	1580	1660	1744	1825	1907	1990		APPEND lt_item_300 TO lt_item_300.
343	426	508	590	672	756	838	921	1003	1085	1168	1250	1332	1414	1497	1581	1661	1745	1826	1908	1991		ENDLOOP.
344	427	509	591	673	757	839	922	1004	1086	1169	1251	1333	1415	1498	1582	1662	1746	1827	1909	1992		IF sy-subrc EQ 0 AND <<t_itema>> IS ASSIGN
345	428	510	592	674	758	840	923	1005	1087	1170	1252	1334	1416	1499	1583	1663	1747	1828	1910	1993		REFRESH <<t_itema>>.
346	429	511	593	675	759	841	924	1006	1088	1171	1253	1335	1417	1500	1584	1664	1748	1829	1911	1994		<<t_itema>> = lt_item_300[].
347	430	512	594	676	760	842	925	1007	1089	1172	1254	1336	1418	1501	1584	1664	1749	1830	1912	1995		ENDIF.
348	431	513	595	677	761	843	926	1008	1090	1173	1255	1337	1419	1502	1585	1665	1750	1831	1913	1997		*
349	432	514	596	678	762	844	927	1009	1091	1174	1256	1338	1420	1503	1586	1667	1751	1832	1914	1998		Fehlerprotokoll ausgeben
350	433	515	597	679	763	845	928	1010	1092	1175	1257	1339	1421	1504	1587	1668	1752	1833	1915	1999		CALL METHOD ar_data_changed->display_protoc
351	434	516	598	680	764	846	929	1011	1093	1176	1258	1340	1422	1505	1588	1669	1753	1834	1916	2000		
352	435	517	599	681	765	847	930	1012	1094	1177	1259	1341	1423	1506	1589	1670	1754	1835	1917	2001		ENDMETHOD.
353	436	518	600	682	766	848	931	1013	1095	1178	1260	1342	1424	1507	1590	1671	1755	1836	1918	2002		"handle_data_ch
354	437	519	601	683	767	849	932	1014	1096	1179	1261	1343	1425	1508	1591	1672	1756	1837	1919			
355	438	520	602	684	768	850	933	1015	1097	1180	1262	1344	1426	1509	1592	1673	1757	1838	1920			
356	439	521	603	685	769	851	934	1016	1098	1181	1263	1345	1427	1510	1593	1674	1758	1839	1921			
357	440	522	604	686	770	852	935	1017	1099	1182	1264	1346	1428	1511	1594	1675	1759	1840	1922			
358	441	523	605	687	771	853	936	1018	1100	1183	1265	1347	1429	1512	1595	1676	1760	1841	1923			
359	442	524	606	688	772	854	937	1019	1101	1184	1266	1348	1430	1513	1596	1677	1761	1842	1924			
360	443	525	607	689	773	855	938	1020	1102	1185	1267	1349	1431	1514	1597	1678	1762	1843	1925			
361	444	526	608	690	774	856	939	1021	1103	1186	1268	1350	1432	1515	1598	1679	1763	1844	1926			
362	445	527	609	691	775	857	940	1022	1104	1187	1269	1351	1433	1516	1599	1680	1764	1845	1927			
363	446	528	610	692	776	858	941	1023	1105	1188	1270	1352	1434	1517	1600	1681	1765	1846	1928			
364	447	529	611	693	777	859	942	1024	1106	1189	1271	1353	1435	1518	1601	1682	1766	1847	1929			
365	448	530	612	694	778	860	943	1025	1107	1190	1272	1354	1436	1519	1602	1683	1767	1848	1930			
366	449	531	613	695	779	861	944	1026	1108	1191	1273	1355	1437	1520	1603	1684	1768	1849	1931			
367	450	532	614	696	780	862	945	1027	1109	1192	1274	1356	1438	1521	1604	1685	1769	1850	1932			
368	451	533	615	697	781	863	946	1028	1110	1193	1275	1357	1439	1522	1605	1686	1770	1851	1933			
369	452	534	616	698	782	864	947	1029	1111	1194	1276	1358	1440	1523	1606	1687	1771	1852	1934			
370	453	535	617	699	783	865	948	1030	1112	1195	1277	1359	1441	1524	1607	1688	1772	1853	1935			
371	454	536	618	700	784	866	949	1031	1113	1196	1278	1360	1442	1525	1608	1689	1773	1854	1936			
372	455	537	619	701	785	867	950	1032	1114	1197	1279	1361	1443	1526	1609	1690	1774	1855	1937			
373	456	538	620	702	786	868	951	1033	1115	1198	1280	1362	1444	1527	1610	1691	1775	1856	1938			
374	457	539	621	703	787	869	952	1034	1116	1199	1281	1363	1445	1528	1611	1692	1776	1857	1939			
375	458	540	622	704	788	870	953	1035	1117	1200	1282	1364	1446	1529	1612	1693	1777	1858	1940			
376	459	541	623	705	789	871	954	1036	1118	1201	1283	1365	1447	1530	1613	1694	1778	1859	1941			
377	460	542	624	706	790	872	955	1037	1119	1202	1284	1366	1448	1531	1614	1695	1779	1860	1942			
378	461	543	625	707	791	873	956	1038	1120	1203	1285	1367	1449	1532	1615	1696	1780	1861	1943			
379	462	544	626	708	792	874	957	1039	1121	1204	1286	1368	1450	1533	1616	1697	1781	1862	1944			
380	463	545	627	709	793	875	958	1040	1122	1205	1287	1369	1451	1534	1617	1698	1782	1863	1945			
381	464	546	628	710	794	876	959	1041	1123	1206	1288	1370	1452	1535	1618	1699	1783	1864	1946			
382	465	547	629	711	795	877	960	1042	1124	1207	1289	1371	1453	1536	1619	1700	1784	1865	1947			
383	466	548	630	712	796	878	961	1043	1125	1208	1290	1372	1454	1537	1620	1701	1785	1866	1948			
384	467	549	631	713	797	879	962	1044	1126	1209	1291	1373	1455	1538	1621	1702	1786	1867	1949			
385	468	550	632	714	798	880	963	1045	1127	1210	1292	1374	1456	1539	1622	1703	1787	1868	1950			
386	469	551	633	715	799	881	964	1046	1128	1211	1293	1375	1457	1540	1623	1704	1788	1869	1951			
387	470	552	634	716	800	882	965	1047	1129	1212	1294	1376	1458	1541	1624	1705	1789	1870	1952			
388	471	553	635	717	801	883	966	1048	1130	1213	1295	1377	1460	1542	1625	1706	1790	1871	1953			
389	472	554	636	718	802	884	967	1049	1131	1214	1296	1378	1461	1543	1626	1707	1791	1872	1954			
390	473	555	637	719	803	885	968	1050	1132	1215	1297	1379	1462	1544	1627	1708	1792	1873	1955			
391	474	556	638	720	804	886	969	1051	1133	1216	1298	1380	1463	1545	1628	1709	1793	1874	1956			
392	475	557	639	721	805	887	970	1052	1134	1217	1299	1381	1464	1546	1629	1710	1794	1875	1957			
393	476	558	640	722	806	888	971	1053	1135	1218	1300	1382	1465	1547	1630	1711	1795	1876	1958			
394	477	559	641	723	807	889	972	1054	1136	1219	1301	1383	1466	1548	1631	1712	1796	1877	1959			
395	478	560	642	724	808	890	973	1055	1137	1220	1302	1384	1467	1549	1632	1713	1797	1878	1960			
396	479	561	643	725	809	891	974	1056	1138	1221	1303	1385	1468	1550	1633	1714	1798	1879	1961			
397	480	562	644	726	810	892	975	1057	1139	1222	1304	1386	1469	1551	1634	1715	1799	1880	1962			
398	481	563	645	727	811	893	976	1058	1140	1223	1305	1387	1470	1552	1635	1716	1800	1963	1964			
399	482	564	646	728	812	894	977	1059	1141	1224	1306	1388	1471	1553	1636	1717	1801	196				

1418:	IF NOT lt_seile_num IS INITIAL.	1465:	
1419:	+ Anzahl = 1	1466:	IF sy-subrc IS INITIAL AND l_returncode NE con_a.
1420:	IF l_anz = 1.	1467:	READ TABLE lt_fields INTO ls_fields INDEX 1.
1421:	LOOP AT lt_seile_num INTO ls_seile_num.	1468:	IF sy-subrc IS INITIAL.
1422:	READ TABLE gt_gui_liste_angebot_0500	1469:	LOOP AT lt_guigra ASSIGNING <ls_guigra>
1423:	INTO ls_gui_liste_angebot INDEX ls_seile_	1470:	WHERE NOT verdat IS INITIAL.
1424:		1471:	IF ls_fields-value EQ space.
1425:	IF ls_gui_liste_angebot-auftyp EQ con_angebot	1472:	LEAR <ls_guigra>-verdat.
1426:	OR p_vbsto EQ con_ein.	1473:	ELSE.
1427:	+ geändertes Verschieben bis Datum in Datenbanktabell	1474:	<ls_guigra>-verdat = ls_fields-value.
1428:	CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT	1475:	ENDIF.
1429:	EXPORTING	1476:	ENDLOOP.
1430:	input = ls_gui_liste_angebot-vbeln	1477:	IF sy-subrc IS INITIAL.
1431:	IMPORTING	1478:	UPDATE / /guigra FROM TABLE lt_guigra.
1432:	output = l_vbeln.	1479:	ENDIF.
1433:		1480:	IF sy-subrc IS INITIAL.
1434:	SELECT * FROM /guigra INTO TABLE	1481:	COMMIT WORK.
1435:	WHERE vbeln EQ l_vbeln	1482:	ENDIF.
1436:	AND aktiv EQ con_ein.	1483:	
1437:		1484:	ENDIF.
1438:	IF sy-subrc IS INITIAL.	1485:	ELSE.
1439:	LOOP AT lt_guigra ASSIGNING <ls_guigra>	1486:	+ keine Änderungen vorgenommen
1440:	WHERE NOT verdat IS INITIAL.	1487:	MESSAGE i255(/ /40_reports) DISPLAY LIKE 'W'.
1441:	+ Es ist ein "Verschieben bis Datum" vorhanden	1488:	ENDIF.
1442:	EXIT.	1489:	ELSE.
1443:	ENDLOOP.	1490:	MESSAGE e258(/ /40_reports) DISPLAY LIKE 'E'.
1444:		1491:	ENDIF.
1445:	IF sy-subrc IS INITIAL.	1492:	ENDIF.
1446:	+ Es ist ein "Verschieben bis Datum" vorhanden	1493:	ELSE.
1447:	ls_fields-tabname = '/GUIGRA'	1494:	MESSAGE e256(/ /40_reports) DISPLAY LIKE 'E'.
1448:	ls_fields-fieldname = 'VERDAT'.	1495:	ENDIF.
1449:	ls_fields-value = sy-datum.	1496:	ENDLOOP.
1450:		1497:	ELSE.
1451:	APPEND ls_fields TO lt_fields.	1498:	MESSAGE e253(/ /40_reports) DISPLAY LIKE 'E'.
1452:	+ Bei Änderung Verschieben bis Datum Eingabepopup auf	1499:	ENDIF.
1453:	CALL FUNCTION 'POPUP_GET_VALUES_USER_C	1500:	ELSE.
1454:	EXPORTING	1501:	MESSAGE e028(/ /40_reports) DISPLAY LIKE 'E'.
1455:	formname = 'CHECK_DATE_VERD	1502:	ENDIF.
1456:	popup_title = text-ver	1503:	
1457:	programname = '/ GUI_LISTE_ANGEBOT'		
1458:	IMPORTING		
1459:	returncode = l_returncode		
1460:	TABLES		
1461:	fields = lt_fields		
1462:	EXCEPTIONS		
1463:	error_in_fields = 1		
1464:	OTHERS = 2.		
1465:			



```

String getMonthName (int month) {
    switch (month) {
        case 0: return "January";
        case 1: return "February";
        case 2: return "March";
        case 3: return "April";
        case 4: return "May";
        case 5: return "June";
        case 6: return "July";
        case 7: return "August";
        case 8: return "September";
        case 9: return "October";
        case 10: return "November";
        case 11: return "December";
        default: throw new IllegalArgumentException();
    }
}

```

```

int sumOfNonPrimes(int limit) {
    int sum = 0;
    OUTER: for (int i = 0; i < limit; ++i) {
        if (i <= 2) {
            continue;
        }
        for (int j = 2; j < i; ++j) {
            if (i % j == 0) {
                continue OUTER;
            }
        }
        sum += i;
    }
    return sum;
}

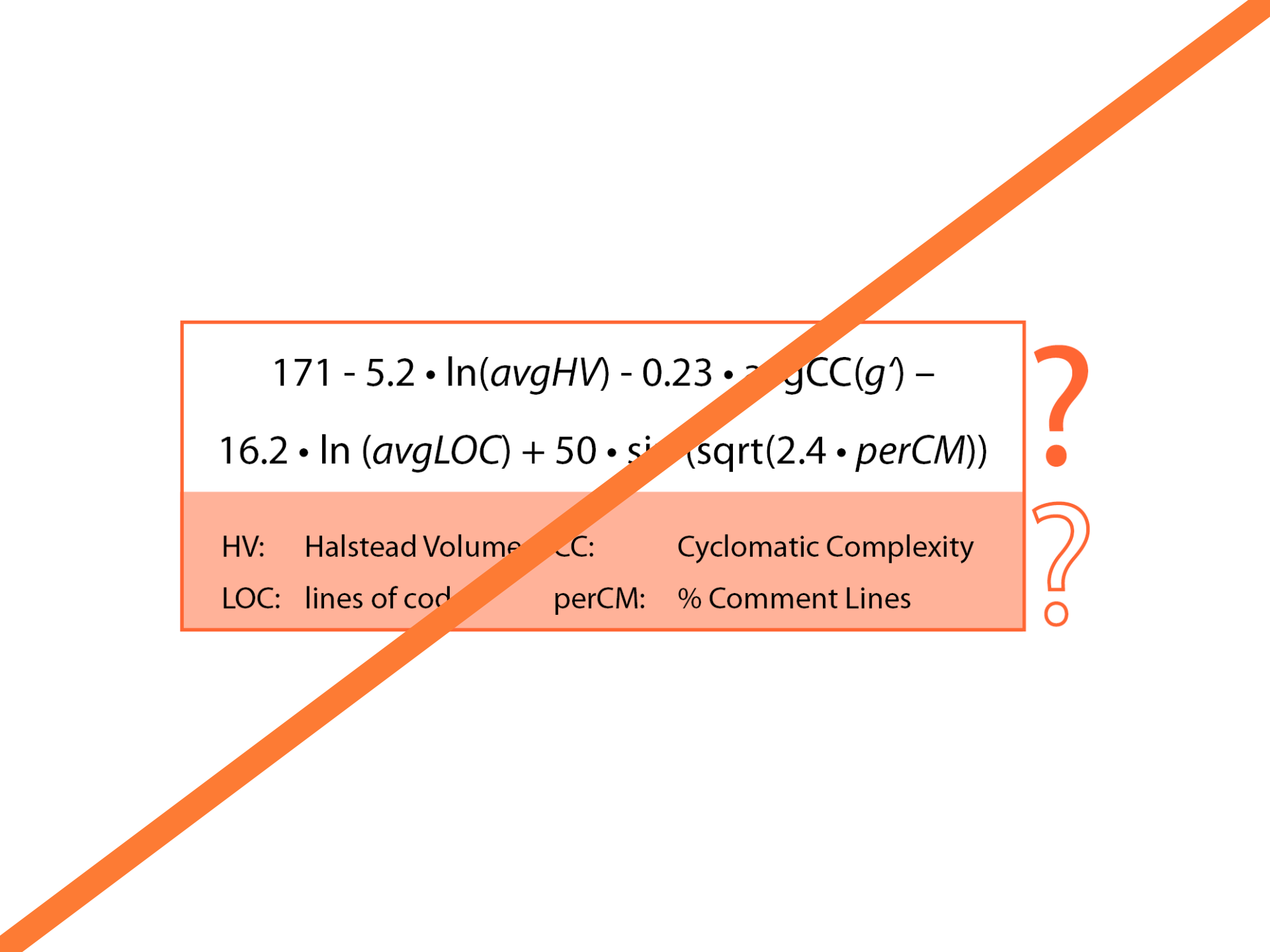
```







?


$$171 - 5.2 \cdot \ln(\text{avgHV}) - 0.23 \cdot \ln(\text{avgCC}(g)) - \\ 16.2 \cdot \ln(\text{avgLOC}) + 50 \cdot \sin(\sqrt{2.4 \cdot \text{perCM}})$$

HV: Halstead Volume      CC: Cyclomatic Complexity  
LOC: lines of code      perCM: % Comment Lines



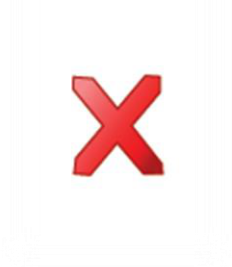
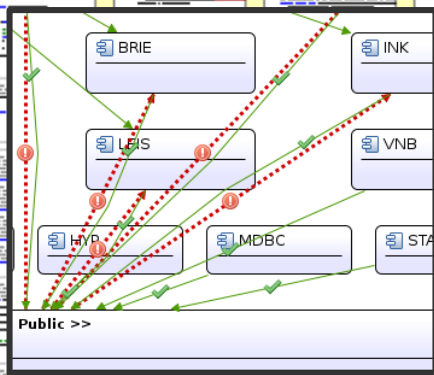
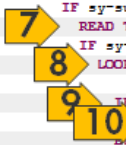




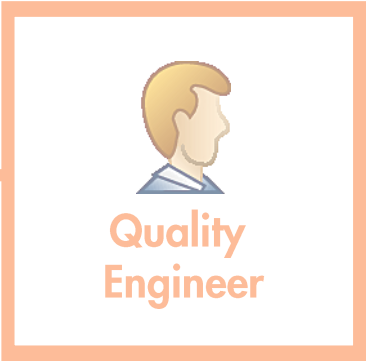
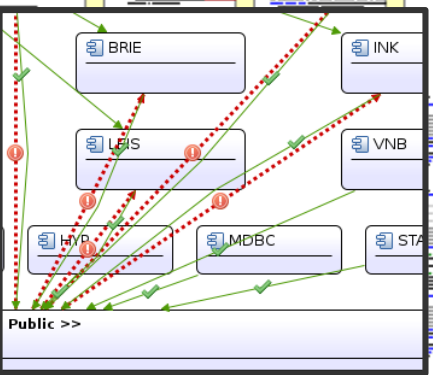
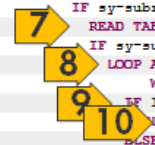


?

```
1465:
1466: IF sy-subrc IS INITIAL AND 1
1467:   READ TABLE lt_fields INTO
1468:   IF sy-subrc IS INITIAL.
1469:     LOOP AT lt_guiqra ASSIGN
1470:       WHERE NOT verdat IS
1471:         IF ls_fields-value EQ
1472:           CLEAR <ls_guiqra>-ver
1473:         ELSE.
1474:           <ls_guiqra>-verdat =
1475:             ENDIF.
1476:         ENDLIST.
1477:       IF sy-subrc IS INITIAL.
1478:         UPDATE /guigra
```



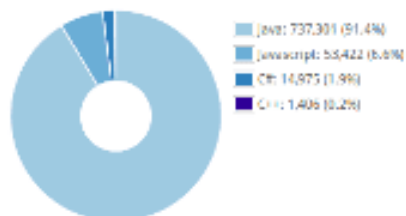
```
1465:
1466: IF sy-subrc IS INITIAL AND 1
1467:   READ TABLE lt_fields INTO
1468:   IF sy-subrc IS INITIAL.
1469:     LOOP AT lt_guiqra ASSIGN
1470:       WHERE NOT verdat IS
1471:         IF ls_fields-value EQ
1472:           CLEAR <ls_guiqra>-ver
1473:         ELSE.
1474:           <ls_guiqra>-verdat =
1475:             ENDIF.
1476:         ENDLIST.
1477:       IF sy-subrc IS INITIAL.
1478:         UPDATE / (guiqra
```



Quality Engineer



LOC Distribution for cqse-all



Clone Coverage for cqse-all



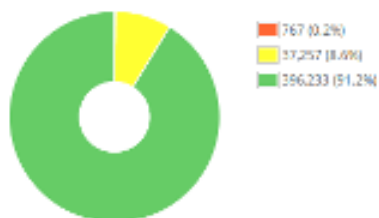
Findings Density for cqse-all



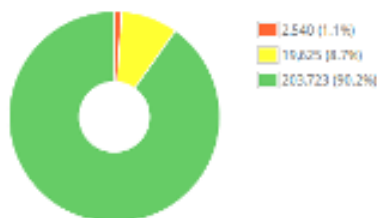
Review Rating for cqse-all



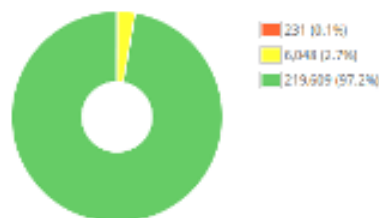
File Length for cqse-all



Method Length for cqse-all



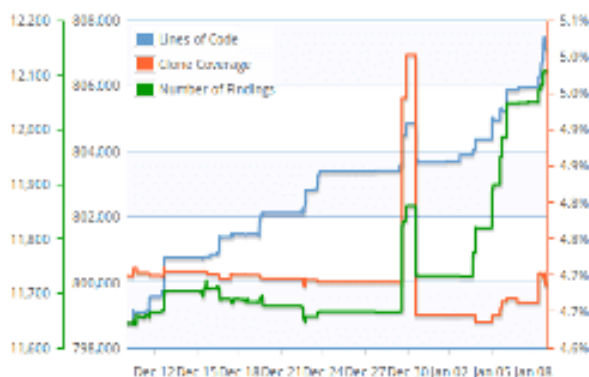
Nesting Depth for cqse-all



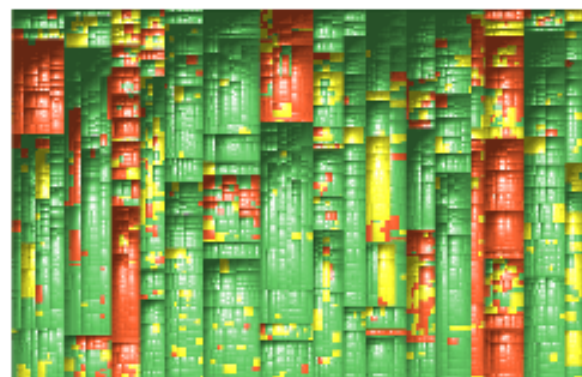
Comment Completeness for cqse-all



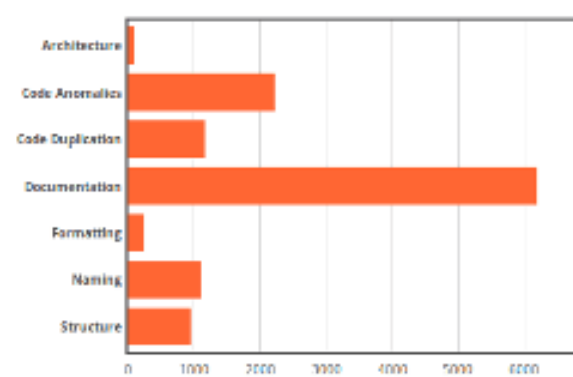
30 Day Trend for cqse-all



Rating Treemap for cqse-all



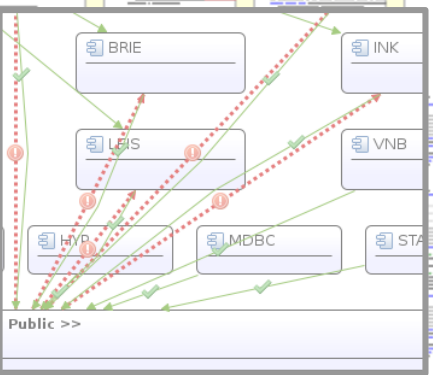
Findings Summary Bar Chart for cqse-all




# Anforderungen an zuverlässige KPIs

- Objektiv
- Auswirkungen von Code-Änderungen verständlich
- Actionable
- Nachvollziehbarer Zusammenhang zu Wartungstätigkeit

```
1465:
1466: IF sy-subrc IS INITIAL AND 1
1467:   READ TABLE lt_fields INTO
1468:   IF sy-subrc IS INITIAL.
1469:     LOOP AT lt_guiqra ASSIGNING
1470:       WHERE NOT verdat IS
1471:         IF ls_fields-value EQ s
1472:           CLEAR <ls_guiqra>-ver
1473:         ELSE.
1474:           <ls_guiqra>-verdat =
1475:             ENDIF.
1476:         ENDLIST.
1477:         IF sy-subrc IS INITIAL.
1478:           UPDATE / (guigra
```



  
Quality Engineer



✓

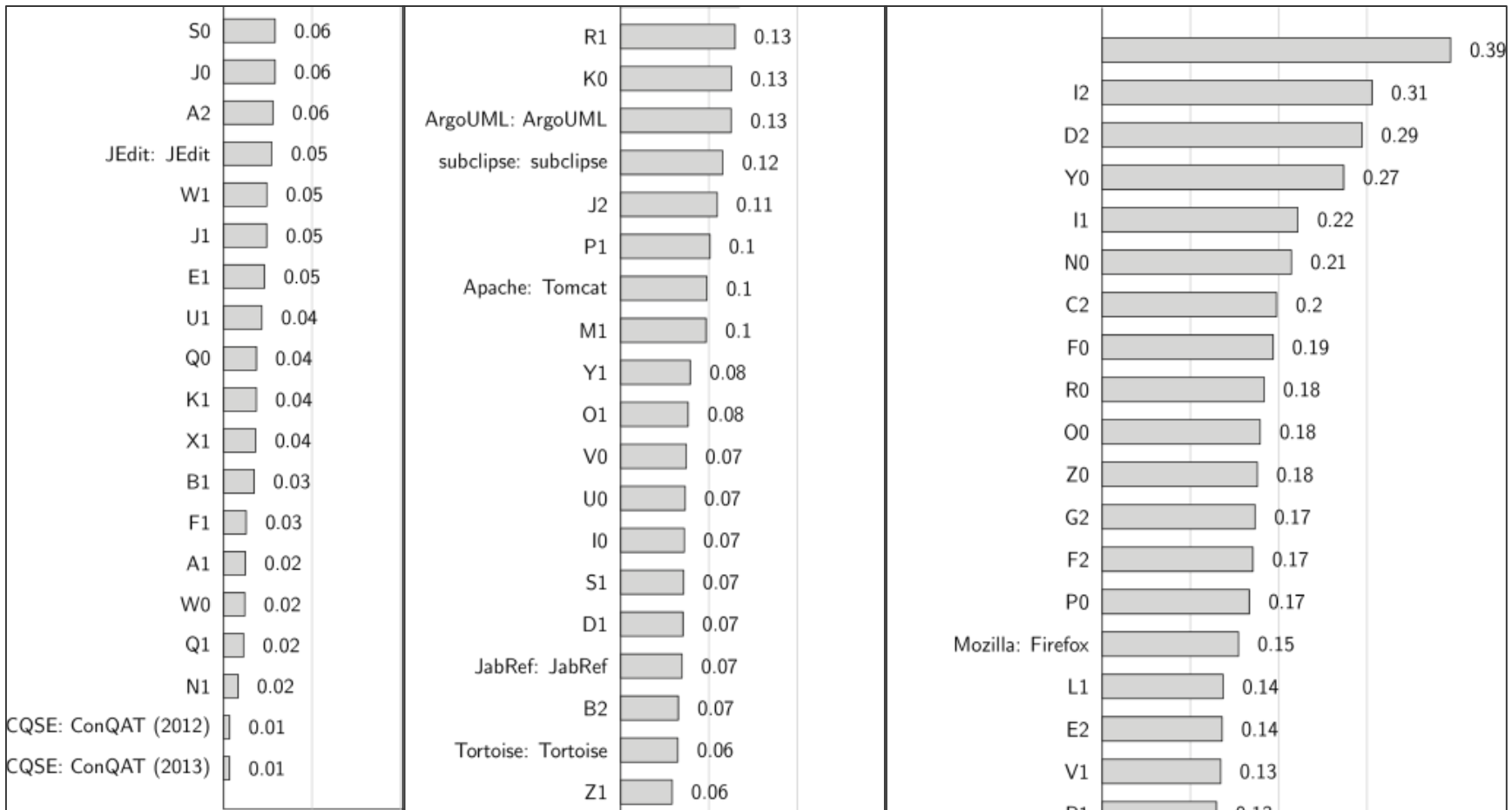
---

▬

---

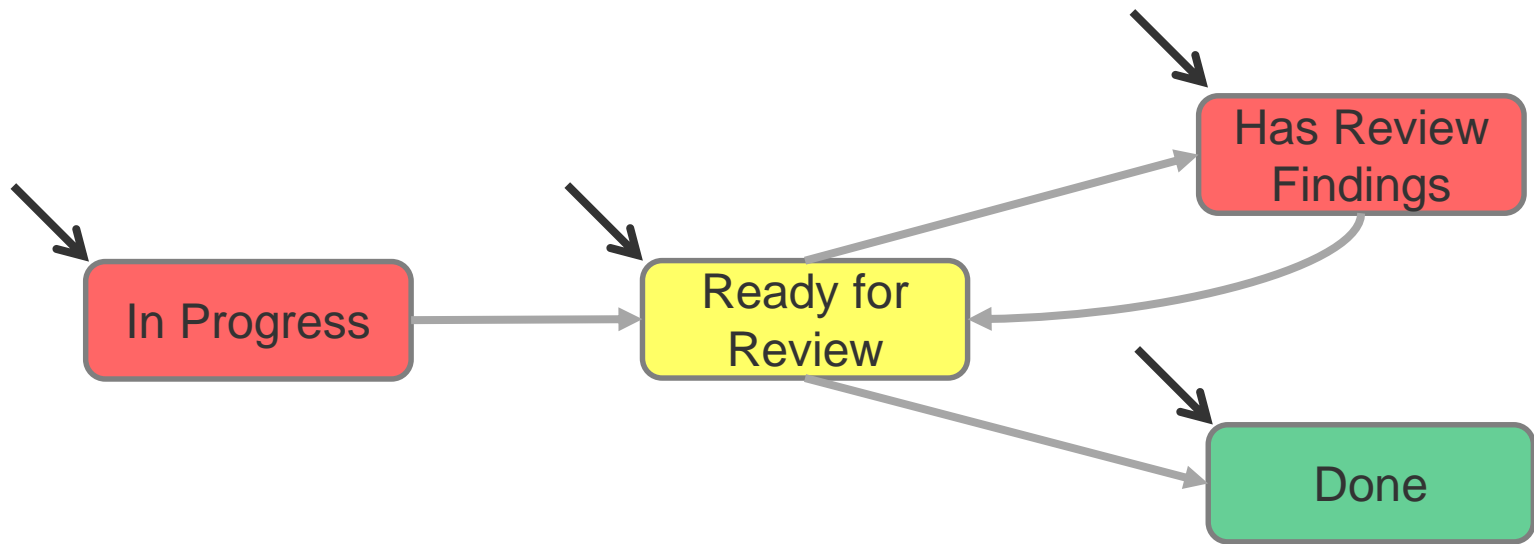
✗

# Benchmark (C#/Java/C++)

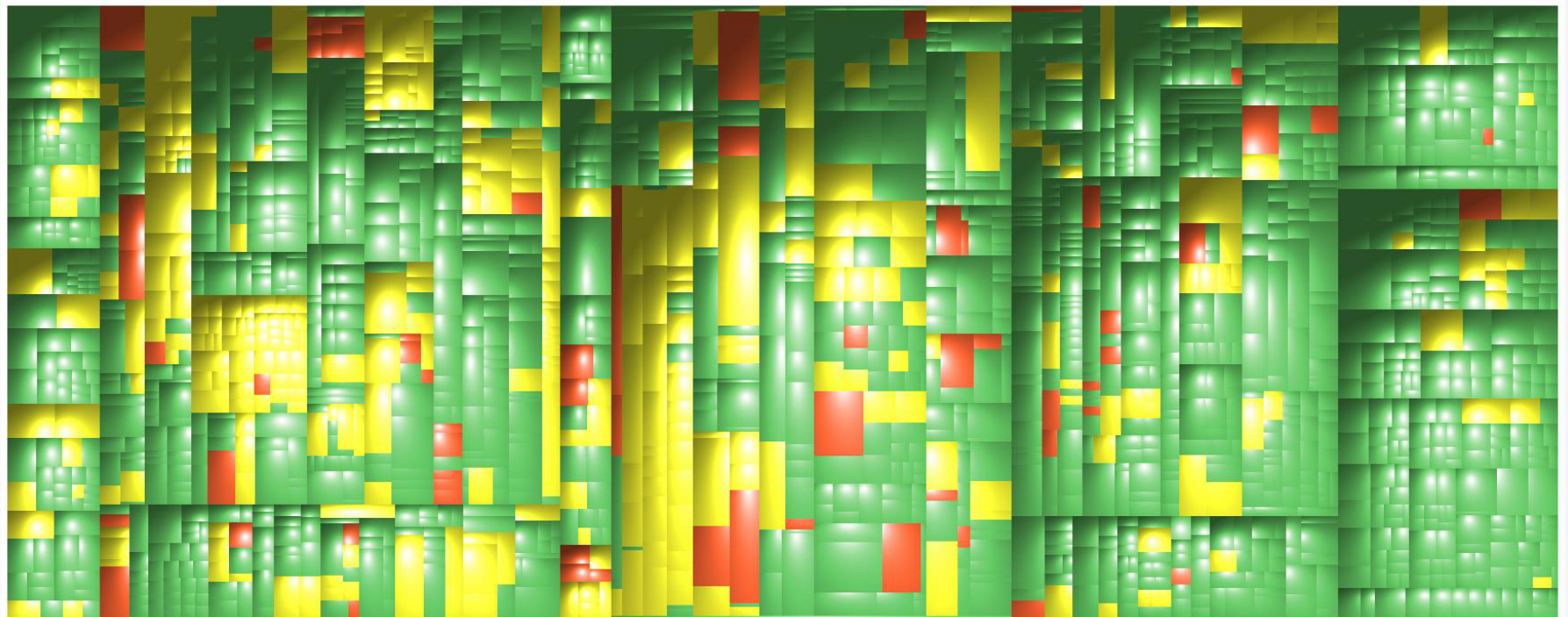






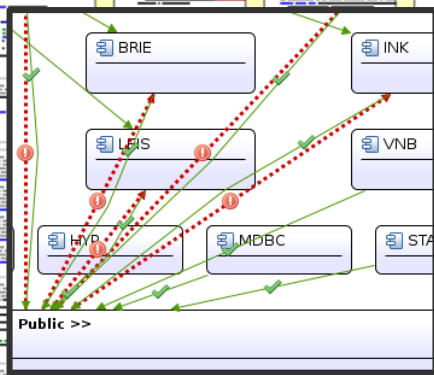


Treemap for Review Rating



```
1465:
1466:
1467:
1468:
1469:
1470:
1471:
1472:
1473:
1474:
1475:
1476:
1477:
1478:
```

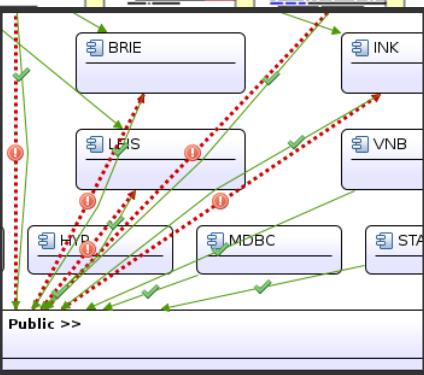
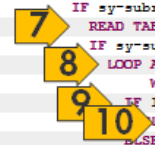
7 IF sy-subrc IS INITIAL AND 1  
8 READ TABLE lt\_fields INTO  
9 IF sy-subrc IS INITIAL.  
10 LOOP AT lt\_guiqra ASSIGN  
WHERE NOT verdat IS  
IF ls\_fields-value EQ  
CLEAR <ls\_guiqra>-ver  
ELSE.  
<ls\_guiqra>-verdat =  
ENDIF.  
ENDLOOP.  
IF sy-subrc IS INITIAL.  
UPDATE / (guiqra



Three vertically stacked rectangular boxes. The top box contains a green checkmark. The middle box contains a yellow horizontal bar. The bottom box contains a red 'X'.



```
1465:
1466: IF sy-subrc IS INITIAL AND 1
1467:   READ TABLE lt_fields INTO
1468:   IF sy-subrc IS INITIAL.
1469:     LOOP AT lt_guiqra ASSIGN
1470:       WHERE NOT verdat IS
1471:         IF ls_fields-value EQ s
1472:           CLEAR <ls_guiqra>-ver
1473:           ELSE.
1474:             <ls_guiqra>-verdat =
1475:             ENDIF.
1476:           ENDLIST.
1477:         IF sy-subrc IS INITIAL.
1478:           UPDATE / (guiqra
```



Task List

```

/** Pace statistics. */
public static Result<Pace> pace(int year) throws SQLException,
    PermissionException {
    requirePermission(currentUser().isAdmin());

    PairList<Date, Double> pace = new PairList<>();

    AggregatedSalaryData aggregatedSalaryData = new AggregatedSalaryData(
        SalaryData.getSalaryDataForEmployees(year), year);

    List<Costs> costs = Costs.getCostsDataForYear(year,
        aggregatedSalaryData.employeeHeadCount,
        aggregatedSalaryData.studentsHeadCount);
    CostDataForYear aggregatedCostData = new CostDataForYear(costs);

    RevenueData revenueData = new RevenueData(year, aggregatedSalaryData,
        aggregatedCostData);

    for (int month = 0; month < 12; ++month) {
        Date start = DateUtils.getMonthStart(year, month);
        Date end = DateUtils.getMonthEnd(year, month);

        PaceInfo paceInfo = new PaceInfo(start, end);
        int available = paceInfo.getTotalAvailableAll(false);
        if (available == 0) {
            pace.add(start, 0);
        } else {
            pace.add(start, paceInfo.getTotalBilledAll(false)
                / (double) available);
        }
    }

    Date start = DateUtils.getStartOfYear(year).getTime();
    Date end = DateUtils.getEndOfYear(year).getTime();

    PaceInfo paceInfo = new PaceInfo(start, end);
    double dailyRate = paceInfo.getAverageRate();
    if (dailyRate == 0) {
        dailyRate = 1000;
    }

    List<LabeledValue> infos = new ArrayList<>();

    VacationPredictor vacationPredictor = new VacationPredictor(year);
    int available = paceInfo.getTotalAvailableAll(false)
        - (int) (vacationPredictor.getOverallExpectedVacation() * 48 * 8);

    LicenseDevChartInfo licenseInfo = Licenses
        .obtainLicenseDevChartInfo(year);

    PairList<Date, Double> minPace = new PairList<>();
    PairList<Date, Double> bonusPace = new PairList<>();
    PairList<Date, Double> targetPace = new PairList<>();

    Date plotEnd = DateUtils.getMonthStart(year, 11);
    if (available > 0) {
        infos.add(new LabeledValue("Current pace: ", paceInfo
            .getTotalBilledAll(false) / (double) available));

        int partialAvailable = paceInfo.getTotalAvailableAll(true);
        if (partialAvailable > 0) {
            infos.add(new LabeledValue("Current partial pace: ", paceInfo
                .getTotalBilledAll(true) / (double) partialAvailable));
        }

        double cost = aggregatedSalaryData.sumTotalCosts
            + aggregatedCostData.sumExpenses;
        double survivalPace = ((cost - licenseInfo.getLicenseGain()) / dailyRate)
            * 8 * 48 / available;
        infos.add(new LabeledValue("Survival pace: ", survivalPace));
        minPace.add(start, survivalPace);
        minPace.add(plotEnd, survivalPace);

        double bonusPaceValue = ((revenueData.targetRevenueBoni - licenseInfo
            .getLicenseGain()) / dailyRate) * 8 * 48 / available;
        infos.add(new LabeledValue("Bonus pace: ", bonusPaceValue));
        bonusPace.add(start, bonusPaceValue);
        bonusPace.add(plotEnd, bonusPaceValue);

        double targetPaceValue = ((revenueData.targetRevenueFbit - licenseInfo
            .getLicenseGain()) / dailyRate) * 8 * 48 / available;
        infos.add(new LabeledValue("Target pace: ", targetPaceValue));
        targetPace.add(start, targetPaceValue);
        targetPace.add(plotEnd, targetPaceValue);
    }

    return json(new PaceStatistics(infos, pace, minPace, bonusPace,
        targetPace));
}

```

```

/** Pace statistics. */
public static Result pace(int year) throws SQLException,
    PermissionException {
    requirePermission(currentUser().isMa());

    PairList<Date, Double> pace = new PairList<>();

    AggregatedSalaryData aggregatedSalaryData = new AggregatedSalaryData(
        SalaryData.getSalaryDataForEmployees(year), year);

    List<Costs> costs = Costs.getCostsDataForYear(year,
        aggregatedSalaryData.employeeHeadCount,
        aggregatedSalaryData.studentsHeadCount);
    CostDataForYear aggregatedCostData = new CostDataForYear(costs);

    RevenueData revenueData = new RevenueData(year, aggregatedSalaryData,
        aggregatedCostData);

    for (int month = 0; month < 12; ++month) {
        Date start = DateUtils.getMonthStart(year, month);
        Date end = DateUtils.getMonthEnd(year, month);

        PaceInfo paceInfo = new PaceInfo(start, end);
        int available = paceInfo.getTotalAvailableAll(false);
        if (available == 0) {
            pace.add(start, 0.);
        } else {
            pace.add(start, paceInfo.getTotalBilledAll(false)
                / (double) available);
        }
    }

    Date start = DateUtils.getStartOfYear().getTime();
    Date end = DateUtils.getEndOfYear(year).getTime();

    PaceInfo paceInfo = new PaceInfo(start, end);
    double dailyRate = paceInfo.getAverageRate();
    if (dailyRate == 0) {
        dailyRate = 1000;
    }

    List<LabeledValue> infos = new ArrayList<>();

    VacationPredictor vacationPredictor = new VacationPredictor(year);
    int available = paceInfo.getTotalAvailableAll(false)
        - (int) (vacationPredictor.getOverallExpectedVacation() * 40 * 8);

    LicenseDevCharInfo licenseInfo = licenses
        .obtainLicenseDevCharInfo(year);

    PairList<Date, Double> minPace = new PairList<>();
    PairList<Date, Double> bonusPace = new PairList<>();
    PairList<Date, Double> targetPace = new PairList<>();

    Date plotEnd = DateUtils.getMonthStart(year, 12);
    if (available > 0) {
        infos.add(new LabeledValue("Current pace: ", paceInfo
            .getTotalBilledAll(false) / (double) available));

        int partialAvailable = paceInfo.getTotalAvailableAll(true);
        if (partialAvailable > 0) {
            infos.add(new LabeledValue("Current partial pace: ", paceInfo
                .getTotalBilledAll(true) / (double) partialAvailable));
        }

        double cost = aggregatedSalaryData.sumTotalCosts
            + aggregatedCostData.sumExpenses;
        double survivalPace = ((cost - licenseInfo.getLicenseGain()) / dailyRate)
            * 8 * 40 / available;
        infos.add(new LabeledValue("Survival pace: ", survivalPace));
        minPace.add(start, survivalPace);
        minPace.add(plotEnd, survivalPace);

        double bonusPaceValue = ((revenueData.targetRevenueBonI - licenseInfo
            .getLicenseGain()) / dailyRate) * 8 * 40 / available;
        infos.add(new LabeledValue("Bonus pace: ", bonusPaceValue));
        bonusPace.add(start, bonusPaceValue);
        bonusPace.add(plotEnd, bonusPaceValue);

        double targetPaceValue = ((revenueData.targetRevenueBtI - licenseInfo
            .getLicenseGain()) / dailyRate) * 8 * 40 / available;
        infos.add(new LabeledValue("Target pace: ", targetPaceValue));
        targetPace.add(start, targetPaceValue);
        targetPace.add(plotEnd, targetPaceValue);
    }

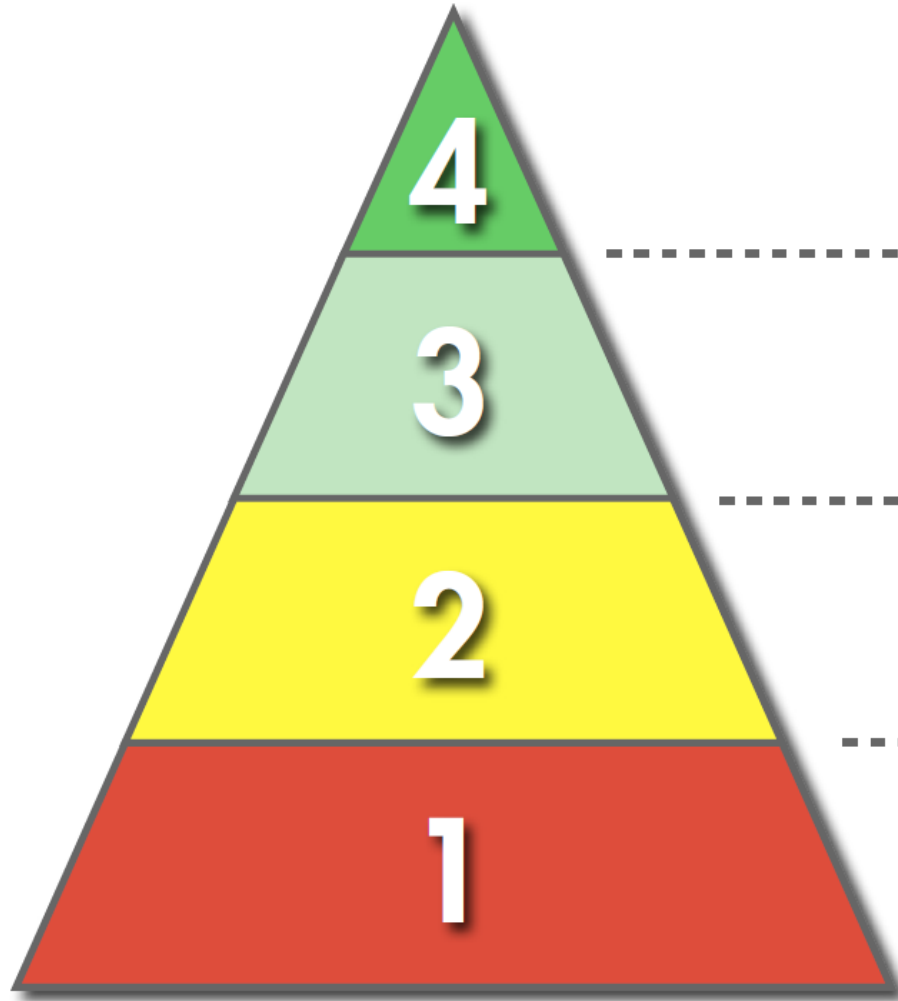
    return json(new PaceStatistics(infos, pace, minPace, bonusPace,
        targetPace));
}

```

```

185     /** Pace statistics. */
186     public static Result pace(int year) throws SQLException,
187         PermissionException {
188         requirePermission(currentUser().isMa());
189
190         PairList<Date, Double> pace = new PairList<>();
191
192         AggregatedSalaryData aggregatedSalaryData = new AggregatedSalaryData(
193             SalaryData.getSalaryDataForEmployees(year), year);
194
195         List<Costs> costs = Costs.getCostsDataForYear(year,
196             aggregatedSalaryData.employeeHeadCount,
197             aggregatedSalaryData.studentsHeadCount);
198         CostDataForYear aggregatedCostData = new CostDataForYear(costs);
199
200         RevenueData revenueData = new RevenueData(year, aggregatedSalaryData,
201             aggregatedCostData);
202
203         for (int month = 0; month < 12; ++month) {
204             Date start = DateUtils.getMonthStart(year, month);
205             Date end = DateUtils.getMonthEnd(year, month);
206
207             PaceInfo paceInfo = new PaceInfo(start, end);
208             int available = paceInfo.getTotalAvailableAll(false);
209             if (available == 0) {
210                 pace.add(start, 0.);
211             } else {
212                 pace.add(start, paceInfo.getTotalBilledAll(false)
213                     / (double) available);
214             }
215         }
216

```

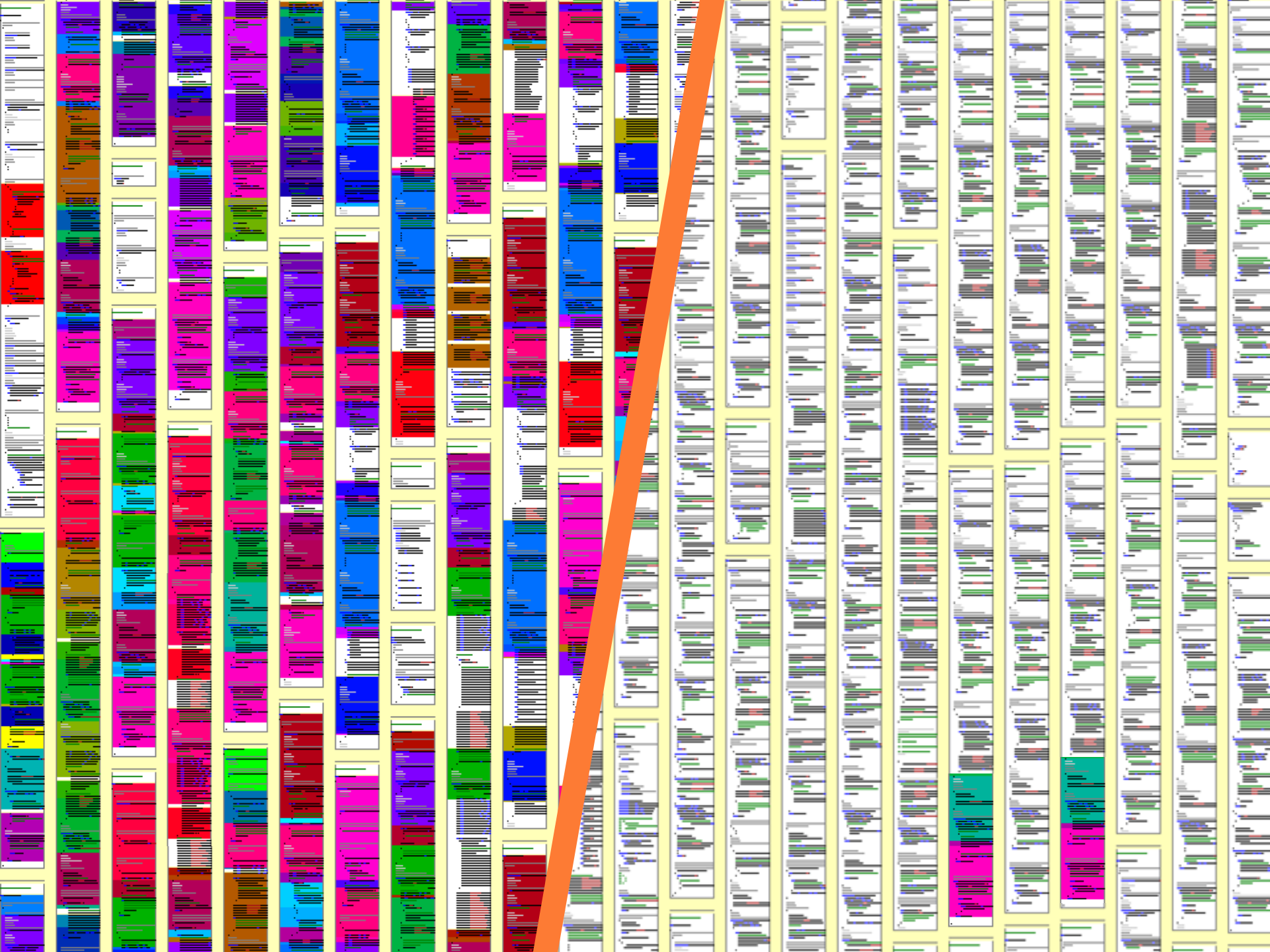


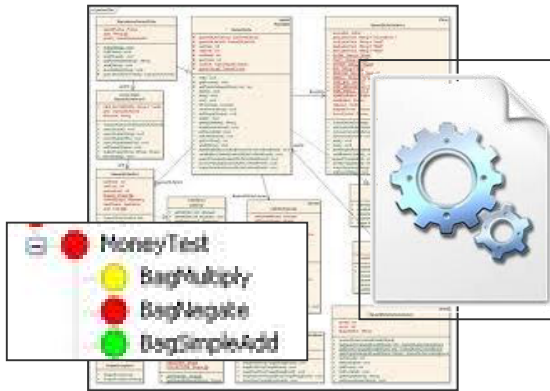
**Keine Defizite**

**Keine Defizite in geändertem Code**

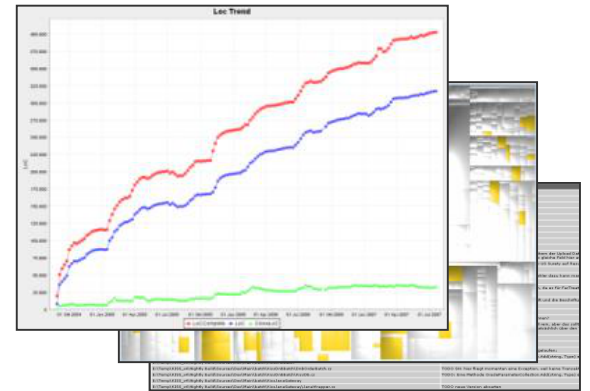
**Keine neuen Defizite**

**Egal**

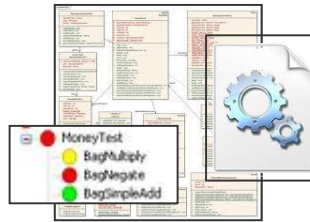




conQAT



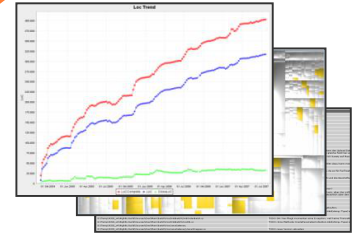
Baseline



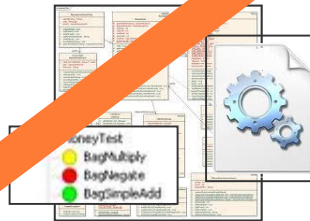
QAT



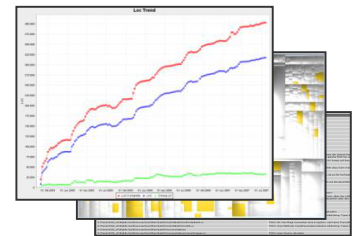
Baseline Dashboard

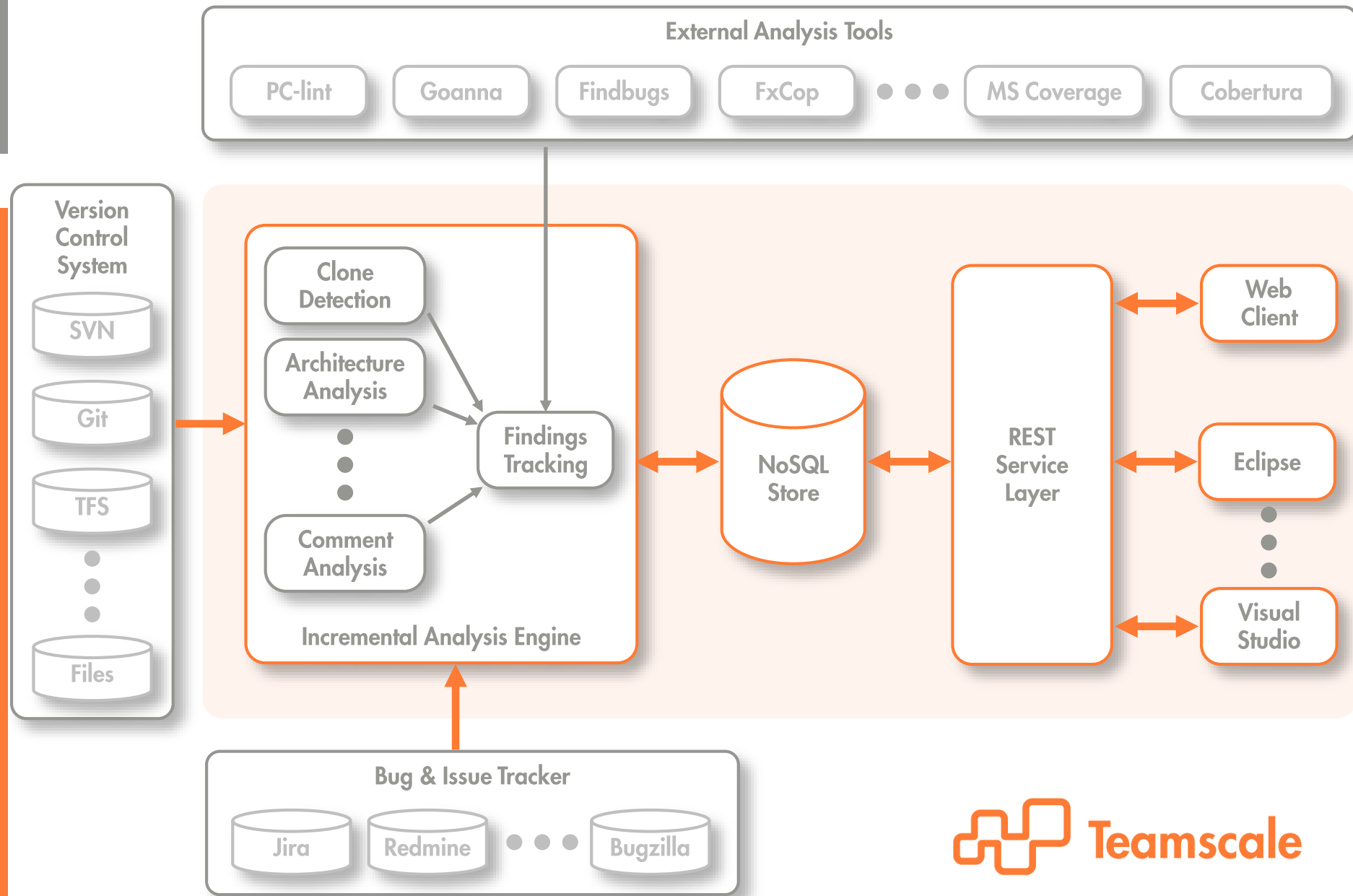


Latest Version



Dashboard  
inl Delta









### ACT-1270 Fixing Inconsistent handling of serializable process variables

by [Victoria King](#) in revision [e1aa41b4b133d269980fff3f81d008da8f21a109](#) (git)

Jun 29 2012

16:05

changed 2 files

**-4** findings



### ACT-1258 Merging Pablo's work into trunk

by [Jacob Nelson](#) in revision [9e664a1f0676cedcbe03415a253e8c3e4a58944c](#) (git)

Jun 29 2012

14:41

added 3 files, changed 2 files

**-1** findings



### Fix for ACT-1059: Task#setDelegationState(DelegationState) was not saved in database

by [Michael Harris](#) in revision [1f48dcad04bc4a621e60af047fb121ae161bca30](#) (git)

Jun 28 2012

21:45

changed 3 files

**+2** findings



### ACT-991 Removed user id from exception message in order not to leak sensitive information

by [Michael Harris](#) in revision [e9a09424e6309c854c44ac5d08740a8ffb082fc9](#) (git)

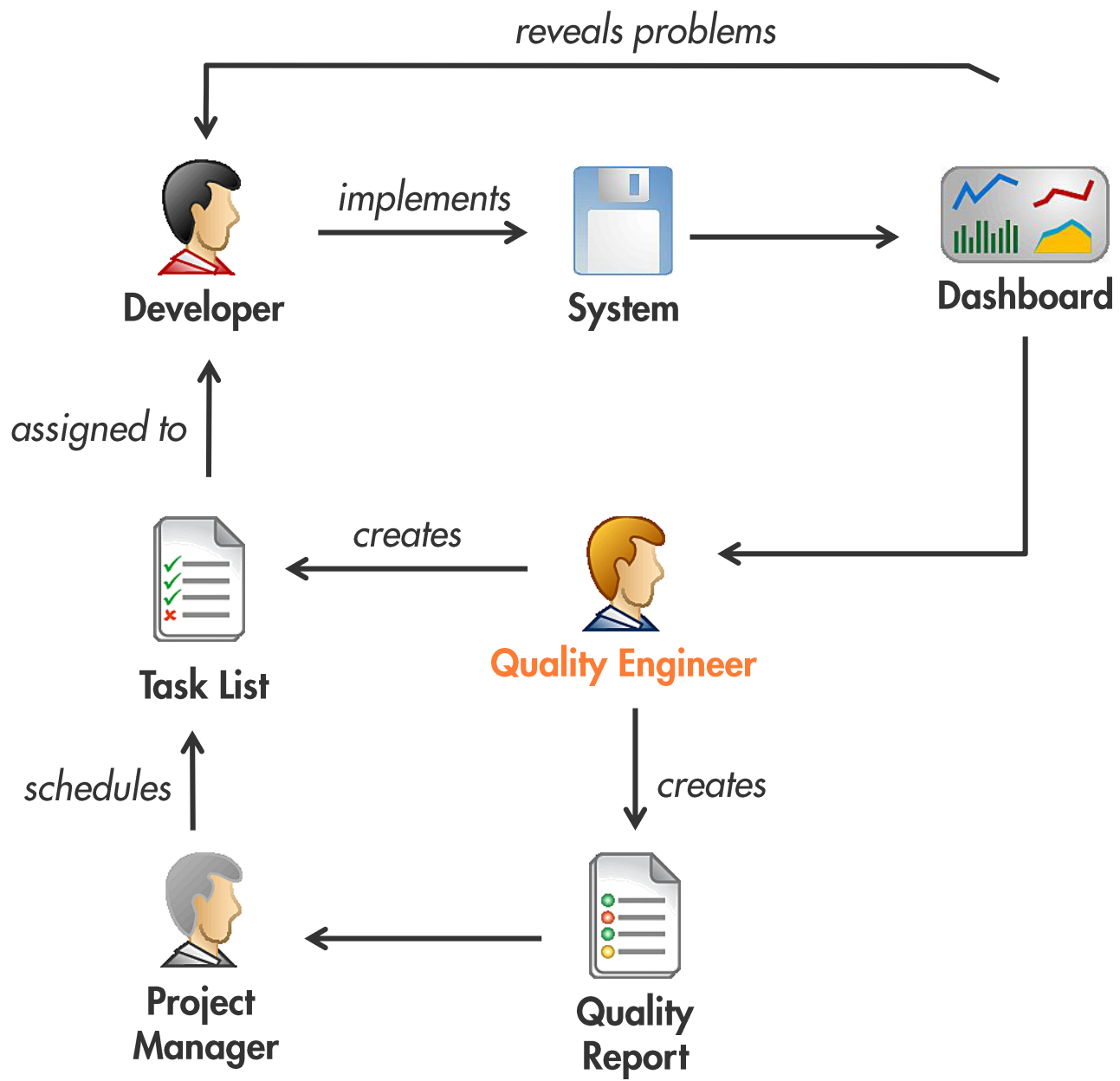
Jun 28 2012

15:26

changed 2 files

**-2** findings





# Fazit

Zuverlässige KPIs sind die Voraussetzung für aussagekräftiges, wirksames Software Quality Control.



# Fazit

Zuverlässige KPIs sind die Voraussetzung für aussagekräftiges, wirksames Software Quality Control.

Nachhaltige Verbesserung erfordert die Unterstützung von Entwicklern und Management und die Integration in den Entwicklungsprozess.

Mindestens einer muss sich hierfür verantwortlich fühlen.

# Wie sag ichs meinem Entwicklungsleiter?

Mit Vorträgen über Softwarequalität sind wir regelmäßig auf Industriekonferenzen oder Kunden-internen Workshops vertreten.



## Impulsvorträge

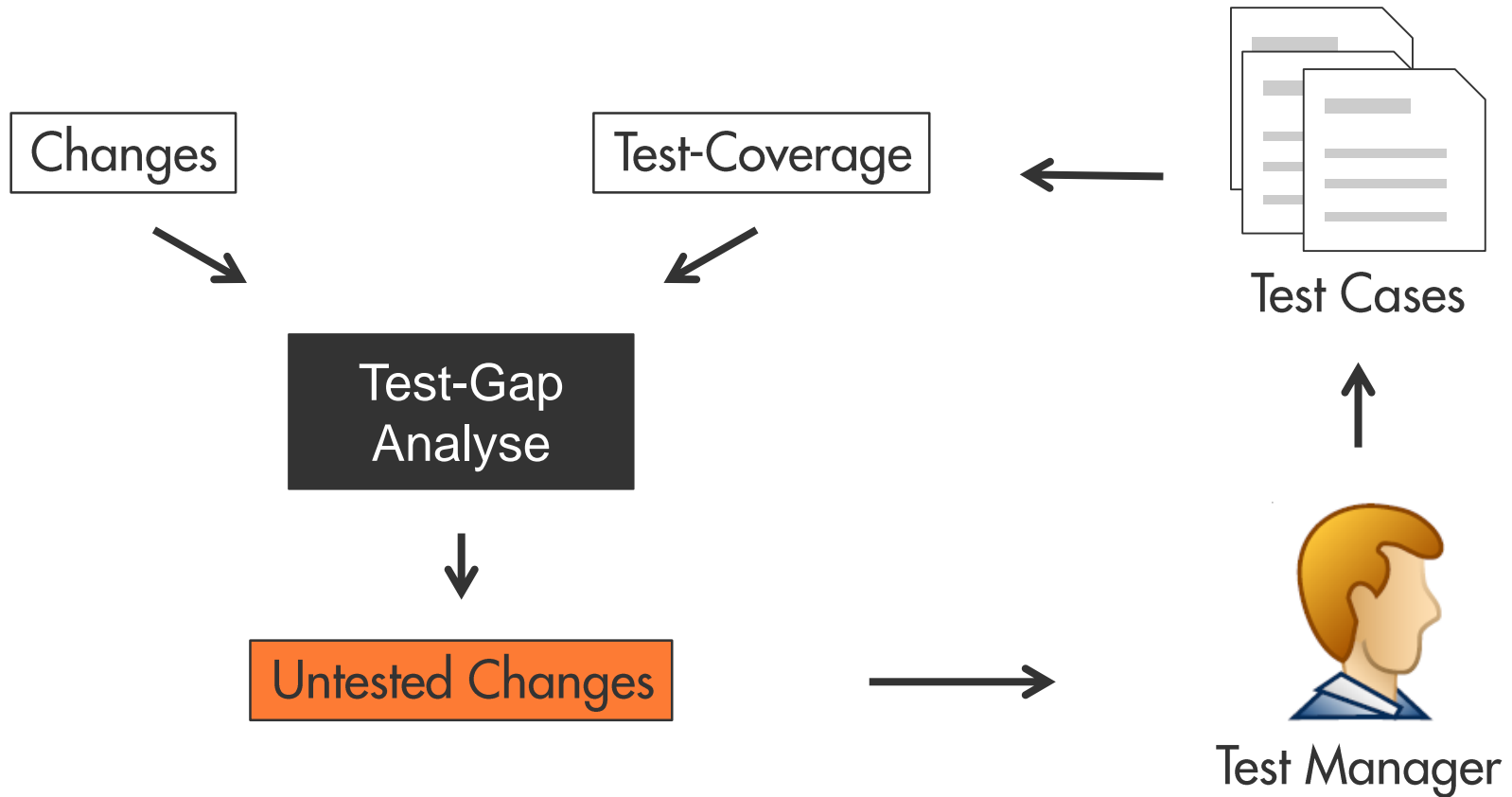
Gerne kommen wir auch zu Ihnen ins Haus, beispielsweise für interne Konferenzen oder Workshops. Unsere Themen reichen von Qualitätsanalysen über Qualitätscontrolling bis hin zu Testcontrolling oder der Einführung von Reviews. Oder aber Sie schlagen uns ein Thema Ihrer Wahl vor.

### DAS ANGEBOT

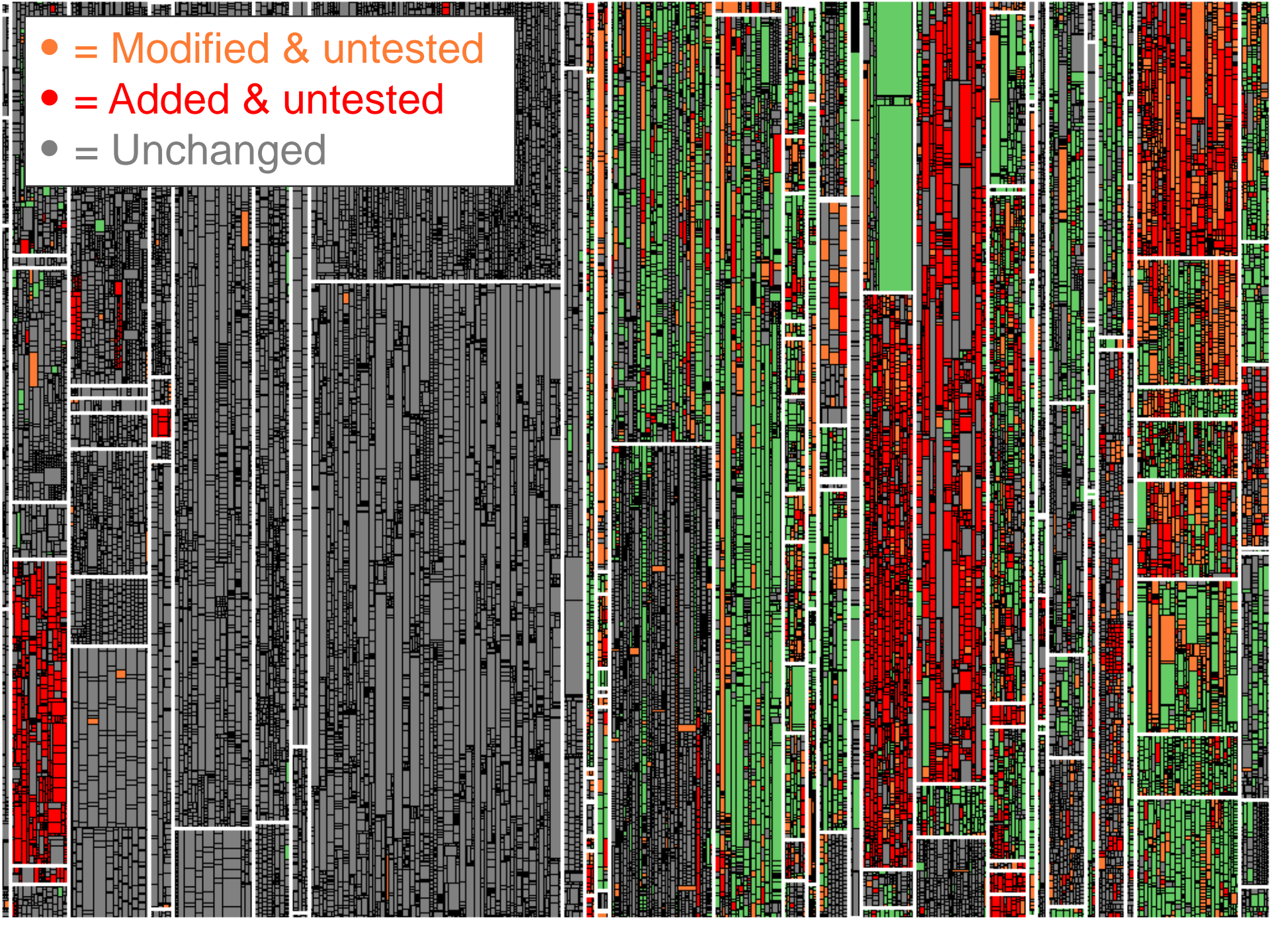
- ⌘ 60-90 MIN VORTRAG
- 💬 SOFTWAREQUALITÄT ALS THEMA
- 📍 BEI IHNEN IM HAUS
- € NUR UNSERE ANREISEKOSTEN
- 📅 TERMIN NACH VEREINBARUNG

[IMPULSVORTRAG ANFRAGEN](#)

# Test-Gap-Analyse



- = Modified & untested
- = Added & untested
- = Unchanged



# Kontakt

Dr. Elmar Juergens · juergens@cqse.eu · +49 179 675 3863

Ich freue mich auf Diskussionen.

CQSE GmbH  
Lichtenbergstraße 8  
85748 Garching bei München