Wie hole ich die anderen ins »Qualitätsboot«?

Erfahrungen aus 10 Jahren Qualitätsretrospektiven, OOP 2025

CQSE | Dr. Tobias Röhm (@langelot)

Dr. Tobias Röhm

Freelancer — 2004 - 2010

TUM — 2010 - 2015

CQSE — 2015 - Heute

Entwicklerin

Tester

Architekt

Product Owner

Softwarequalität

Anwendungs-
verantwortliche

QS-Spezialist

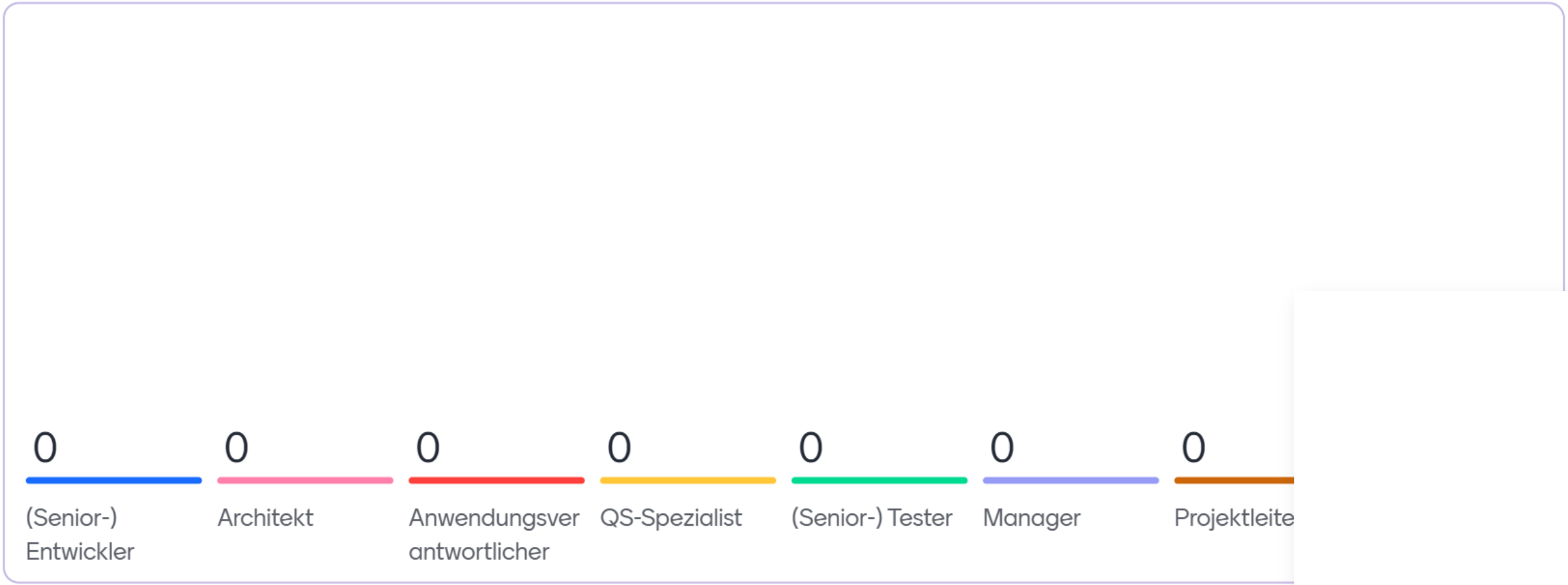
Projektleiterin

Managerin

Join at menti.com | use code

Mentimeter

Welche Bezeichnung trifft Ihre aktuelle Rolle am besten (Mehrfachnennungen möglich)?



Entwicklerin

Tester

Architekt

Product Owner

Softwarequalität

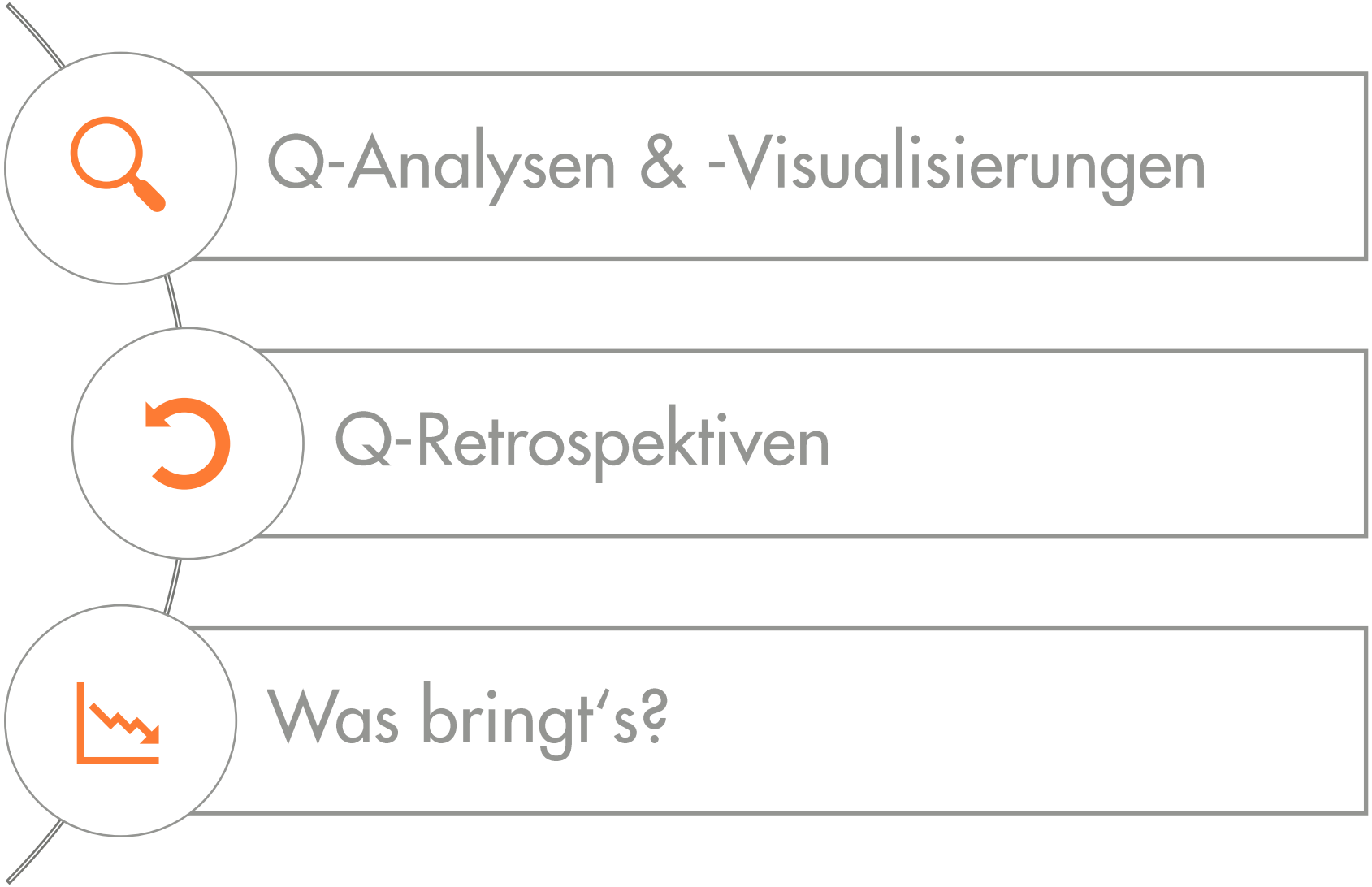
Anwendungs-
verantwortliche

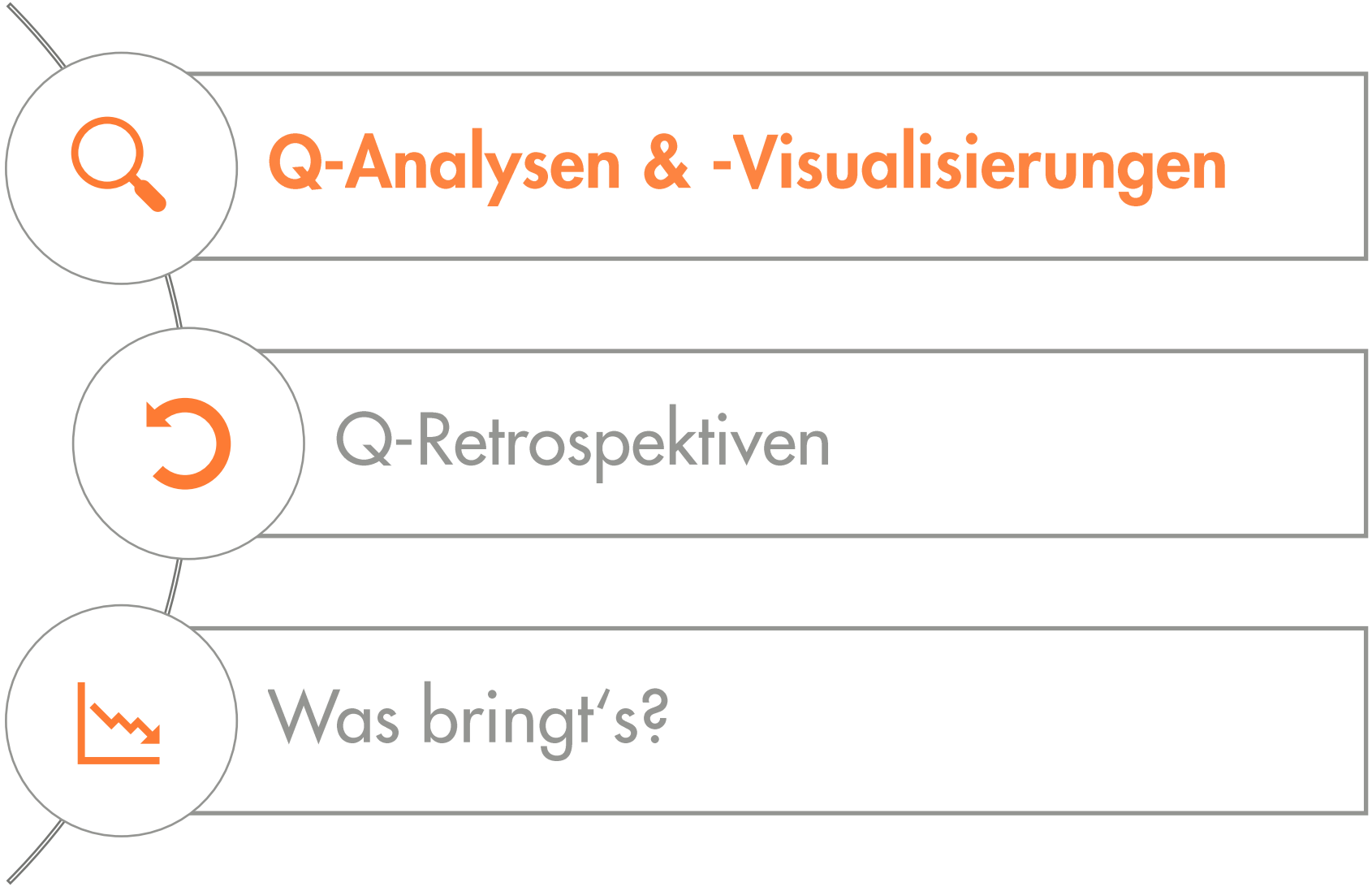
QS-Spezialist

Projektleiterin

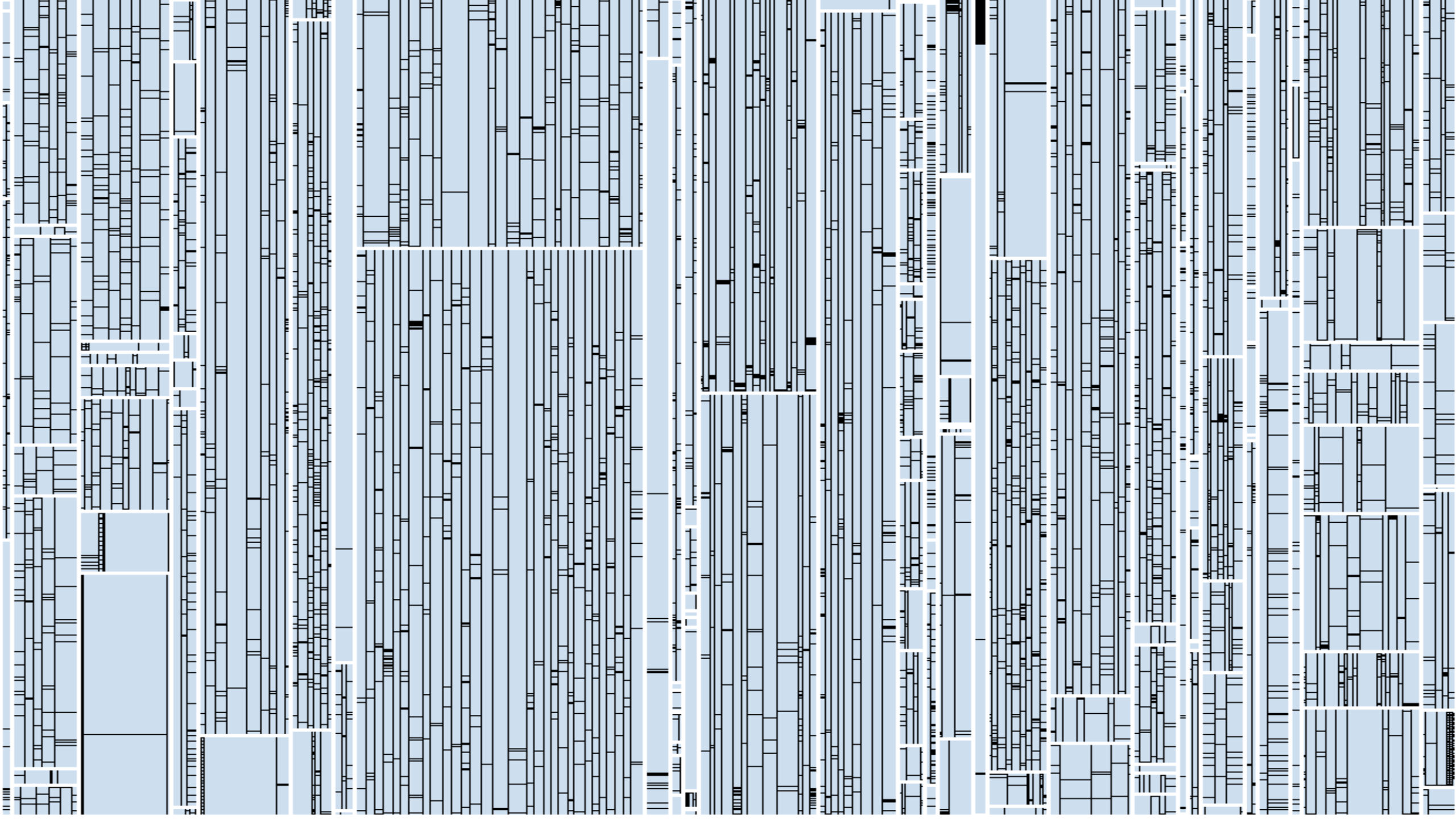
Managerin

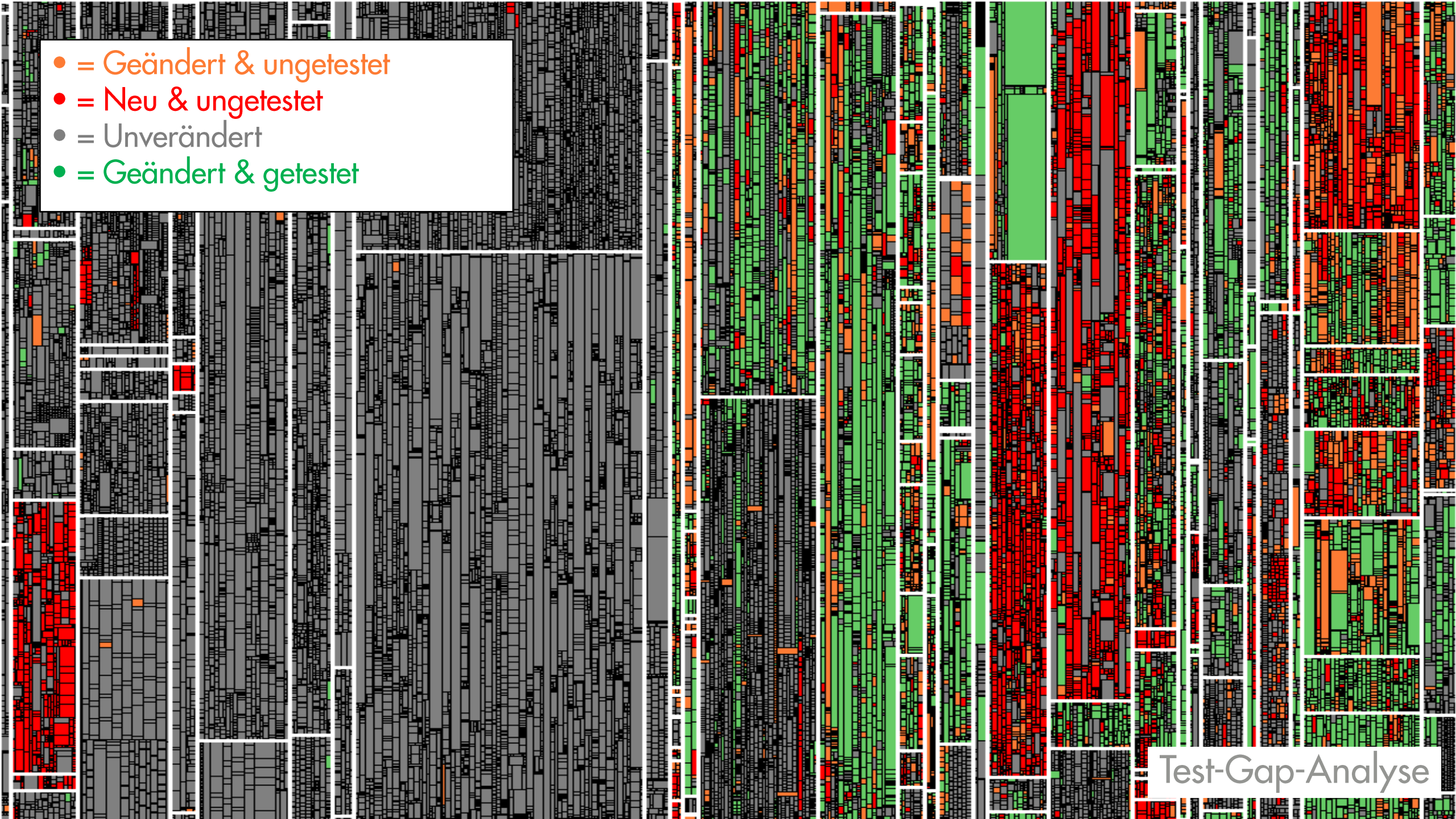
→ **Wirksame Qualitätssicherung erfordert Einbeziehung aller Beteiligten**

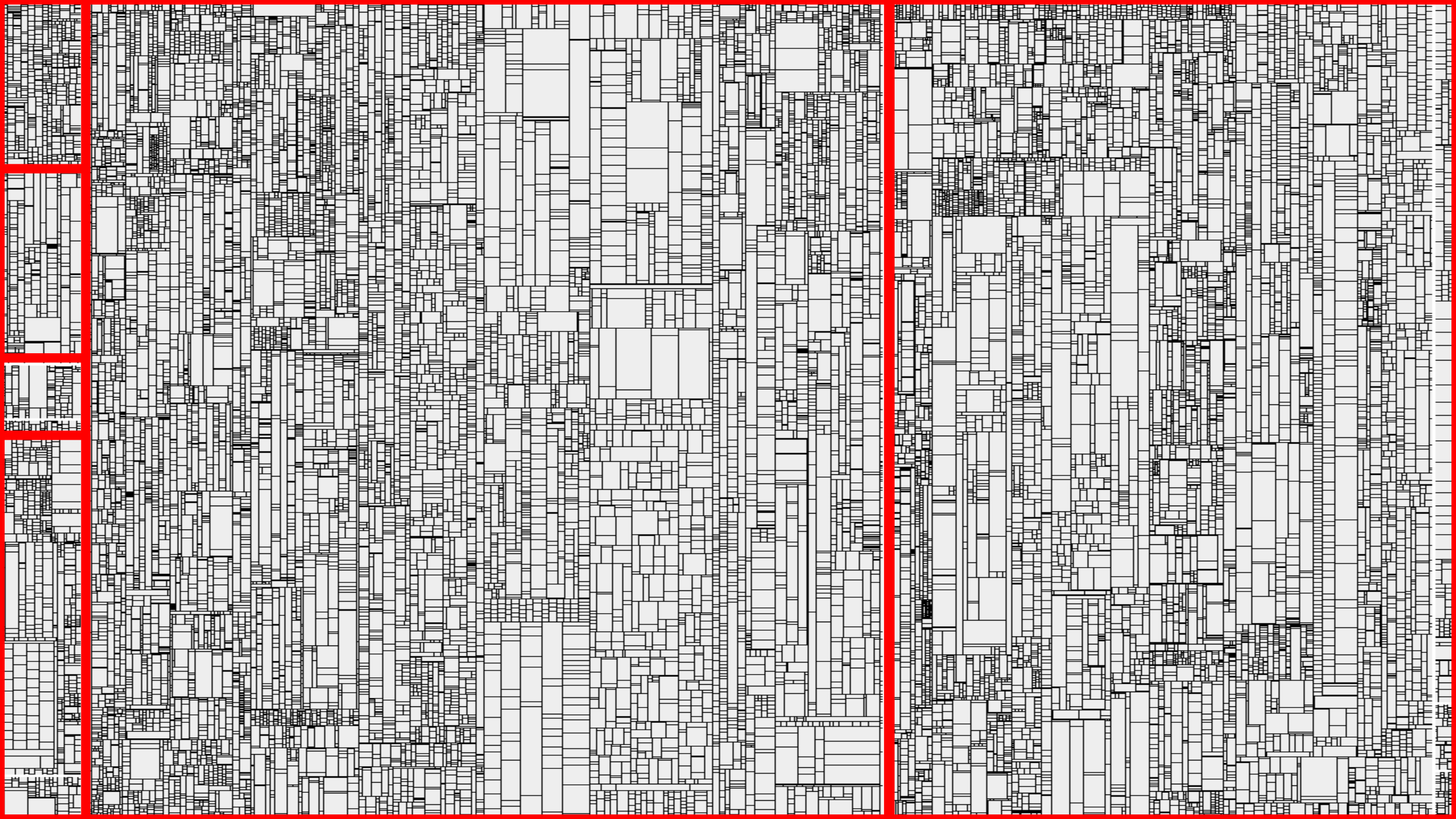

Q-Analysen & -Visualisierungen
Q-Retrospektiven
Was bringt‘s?

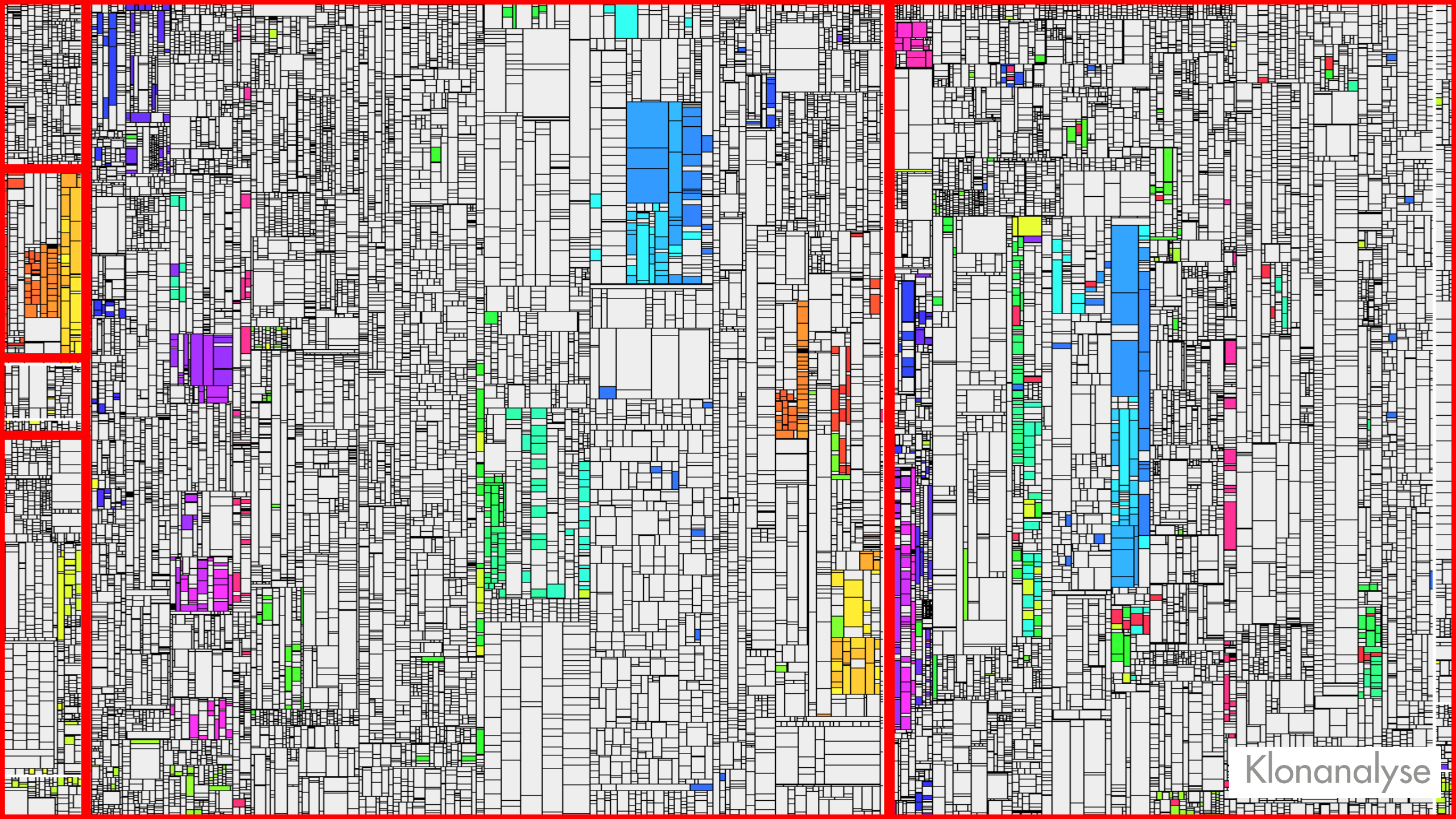

Q-Analysen & -Visualisierungen
Q-Retrospektiven
Was bringt‘s?

Teamscale




• = Geändert & ungetestet
• = Neu & ungetestet
• = Unverändert
• = Geändert & getestet
Test-Gap-Analyse




Klonanalyse

jenkins/test/src/main/java/org/jvnet/hudson/test/HudsonTestCase.java (revision 12d96a56...)

```
if (lhs==null && rhs==null)     return;
if (lhs==null)      fail("lhs is null while rhs="+rhs);
if (rhs==null)      fail("rhs is null while lhs="+lhs);

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>();

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 (Iterable.class.isAssignableFrom(types[i])) {
        Iterable lcol = (Iterable) lv;
        Iterable rcol = (Iterable) rv;
        Iterator ltr,rtr;
        for (ltr=lcol.iterator(), rtr=rcol.iterator(); ltr.hasNext() && rtr.hasNext();)
            Object litem = ltr.next();
            Object ritem = rtr.next();

            if (findDataBoundConstructor(litem.getClass())!=null) {
                assertEqualDataBoundBeans(litem,ritem);
            } else {
                assertEquals(litem,ritem);
            }
        }
        assertFalse("collection size mismatch between "+lhs+" and "+rhs, ltr.hasNext() ^
    } else
    if (findDataBoundConstructor(types[i])!=null || (lv!=null && findDataBoundConstructo
        // recurse into nested databound objects
        assertEqualDataBoundBeans(lv,rv);
    } else {
        primitiveProperties.add(names[i]);
    }
}

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

jenkins/test/src/main/java/org/jvnet/hudson/test/JenkinsRule.java (revision 3909f5ac...)

```
if (lhs==null && rhs==null)     return;
if (lhs==null)      fail("lhs is null while rhs="+rhs);
if (rhs==null)      fail("rhs is null while lhs="+lhs);

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

List<String> primitiveProperties = new ArrayList<String>();

String[] names = ClassDescriptor.loadParameterNames(lc);
Class<?>[] types = lc.getParameterTypes();
assertThat(types.length, is(names.length));
for (int i=0; i<types.length; i++) {
    Object lv = ReflectionUtils.getPublicProperty(lhs, names[i]);
    Object rv = ReflectionUtils.getPublicProperty(rhs, names[i]);

    if (lv != null && rv != null && Iterable.class.isAssignableFrom(types[i])) {
        Iterable lcol = (Iterable) lv;
        Iterable rcol = (Iterable) rv;
        Iterator ltr,rtr;
        for (ltr=lcol.iterator(), rtr=rcol.iterator(); ltr.hasNext() && rtr.hasNext();)
            Object litem = ltr.next();
            Object ritem = rtr.next();

            if (findDataBoundConstructor(litem.getClass())!=null) {
                assertEqualDataBoundBeans(litem,ritem);
            } else {
                assertThat(ritem, is(litem));
            }
        }
        assertThat("collection size mismatch between " + lhs + " and " + rhs, ltr.hasNex
                is(false));
    } else
    if (findDataBoundConstructor(types[i])!=null || (lv!=null && findDataBoundConstructo
        // recurse into nested databound objects
        assertEqualDataBoundBeans(lv,rv);
    } else {
        primitiveProperties.add(names[i]);
    }
}

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

Klonanalyse

The variable token may contain a null value at this point and is dereferenced

Flag as False Positive | Flag as Tolerated | AI Actions

Correctness › Possible Bugs › Null pointer dereference

Dereferencing a potentially null pointer or null reference can cause runtime exceptions and unexpected behavior. Consider performing null checks before dereferencing pointers or using safer dereferencing methods.

What Does This Check Look For?

This check looks for null dereferencing problems and identifies potential instances where a program may attempt to access or manipulate an object or variable that has not been properly initialized or has been set to `null` (in Java and C#) or `NULL`, `0`, or `nullptr` (in C and C++).

Some notes related to C/C++ code: first, note that it is sometimes common to dereference a pointer after calling `malloc`. However, since `malloc` can fail, Teamscale will create a finding if a pointer is dereferenced after

Show more

[redacted] ScannerUtils.java:54

Code | Introduction | Tasks | Properties

```
44        * @throws IOException
45        *             thrown if scanner throws an IO exception
46        */
47       public static void readTokens(IScanner scanner, List<IToken> tokens, List<ScannerException> exceptions)
48               throws IOException {
49           IToken token = null;
50
51           do {
52               try {
53                   token = scanner.getNextToken();
54                   if (token.getType() != ETokenType.EOF) {
55                       tokens.add(token);
56                   }
57               } catch (ScannerException e) {
58                   exceptions.add(e);
59                   continue;
60               }
61           } while (token != null && token.getType() != ETokenType.EOF);
62       }
63
64       /**
```

NULL-Check fehlt
→ NPE-Exception

Show all lines (99 more) and all findings (4 more)

Korrektheitsanalyse

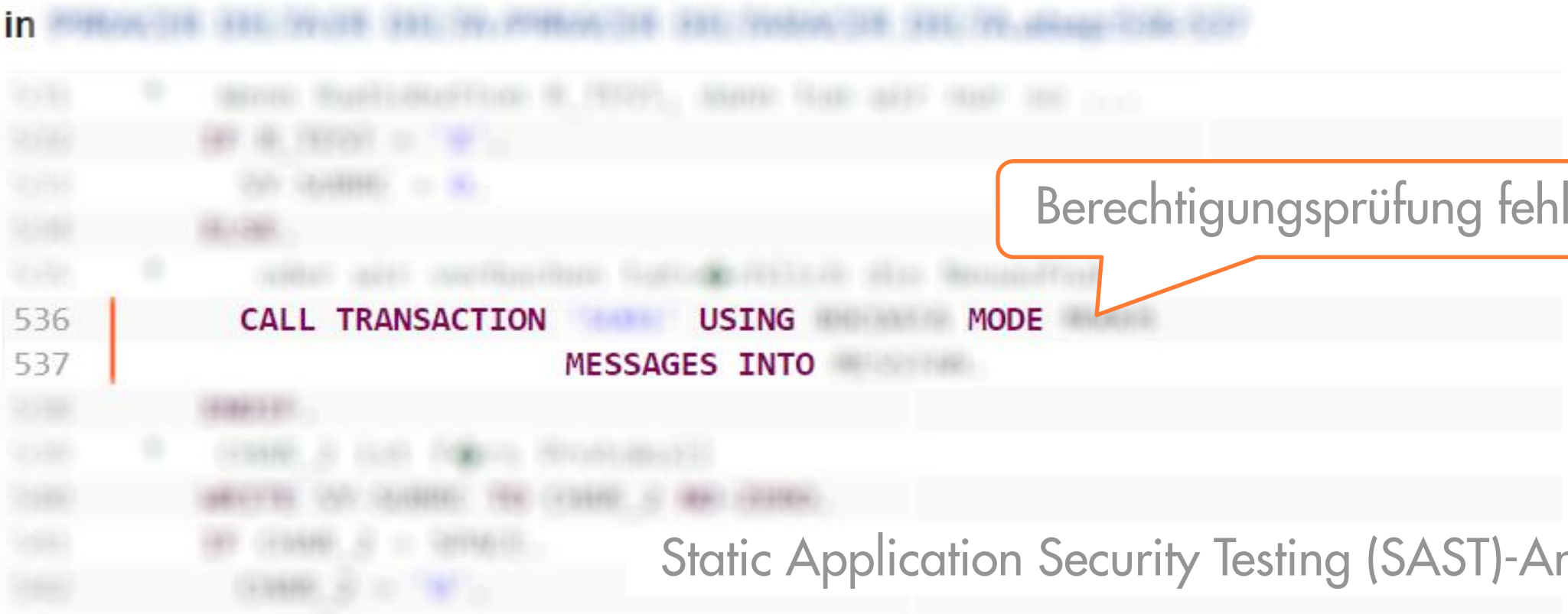
Missing authority check before CALL TRANSACTION

Code Anomalies / Security

CALL TRANSACTION executes an transaction with the given transaction code. Until SAP_BASIS release 7.40 no authority check was performed for CALL TRANSACTION, from SAP_BASIS release 7.40 on, a authority check is only performed, if the addition WITH AUTHORITY CHECK is used.

To solve this issue, the addition WITH AUTHORITY CHECK should be added (if SAP_BASIS version is 7.40 or higher)or the function module AUTHORITY_CHECK_TCODE should be called before.

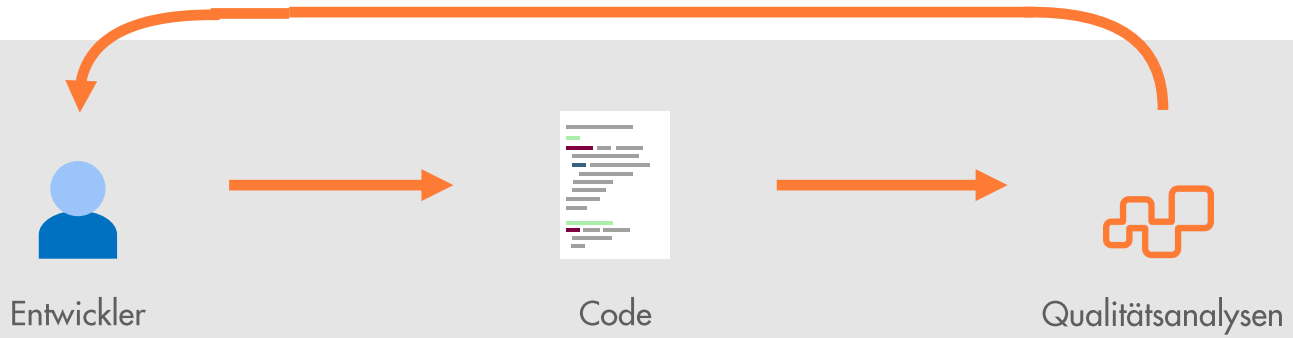
in



Berechtigungsprüfung fehlt

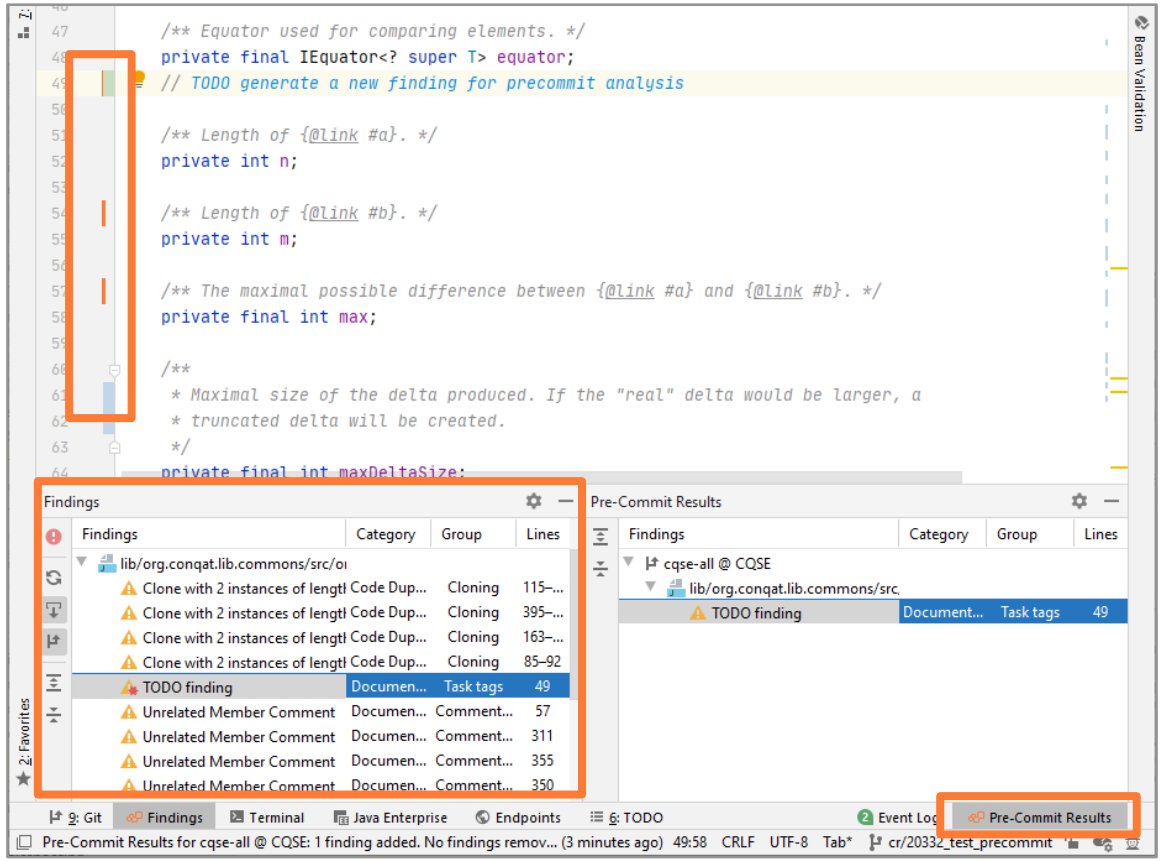
Static Application Security Testing (SAST)-Analyse

Feedbackschleife 1: Statische Codeanalyse hilft Entwicklern, Fehler & Q-Defizite zu vermeiden

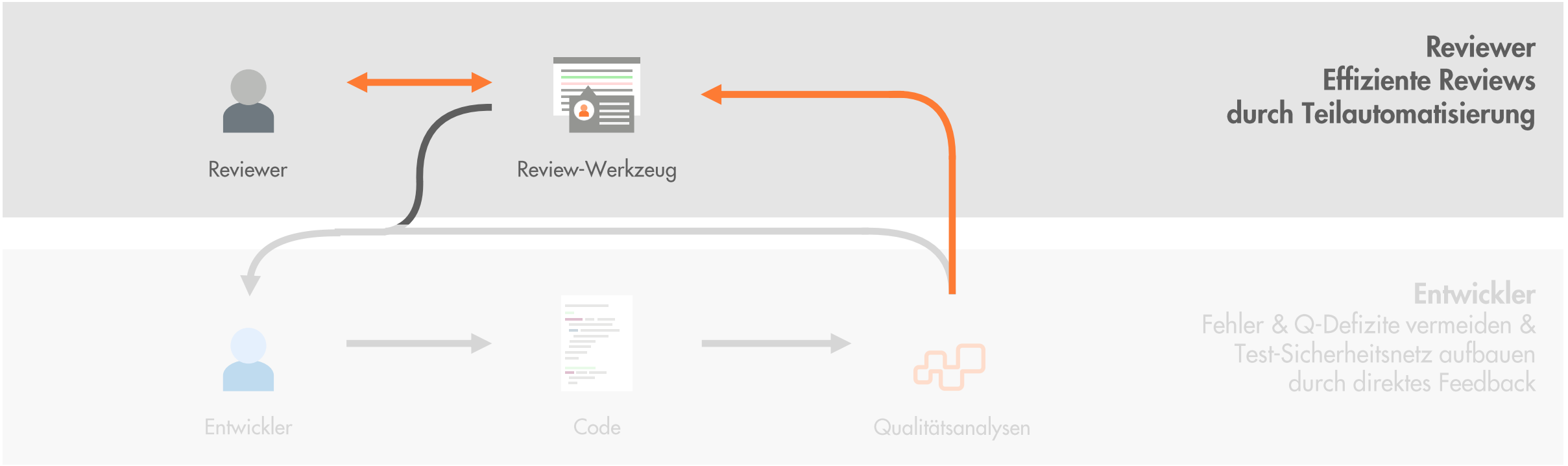


Entwickler-Feedback
Fehler & Q-Defizite vermeiden
durch schnelles Feedback

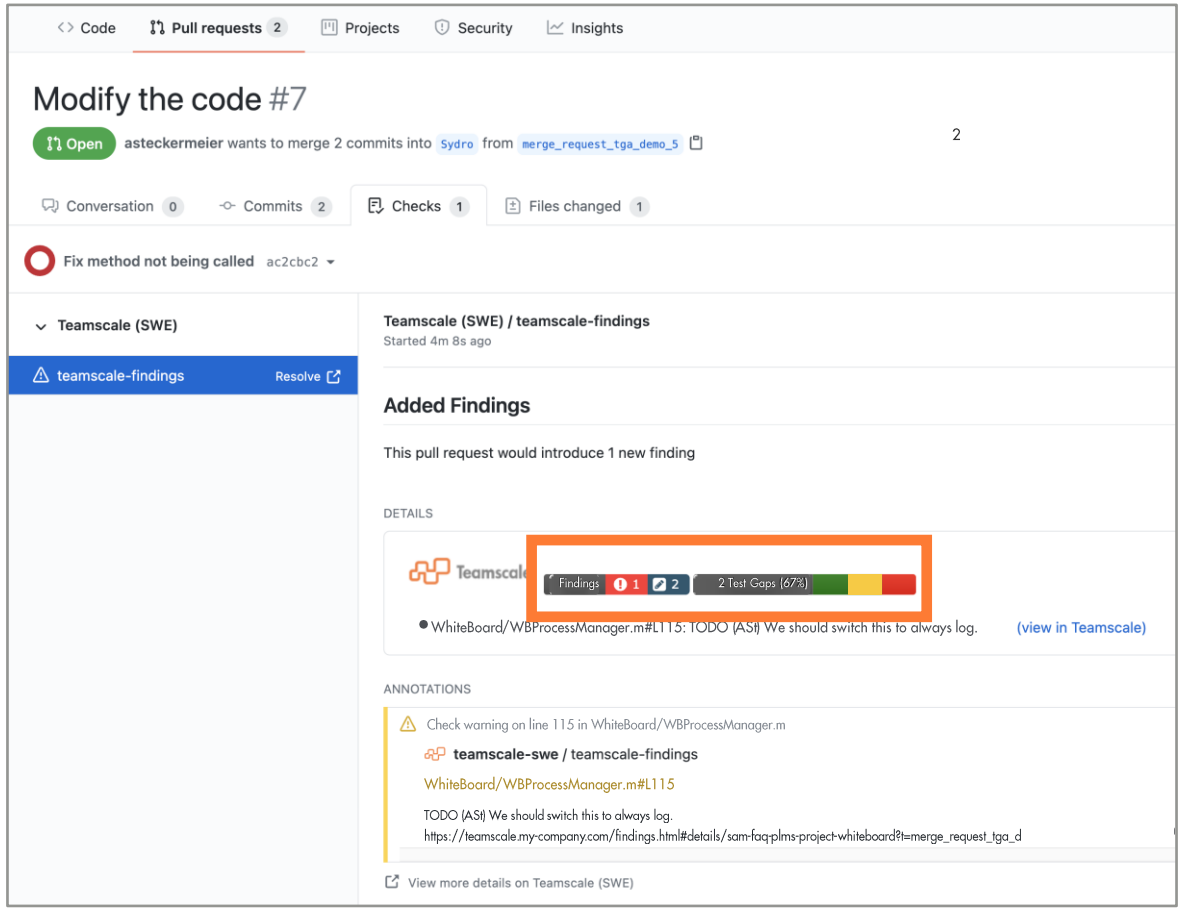
IDE-Integration



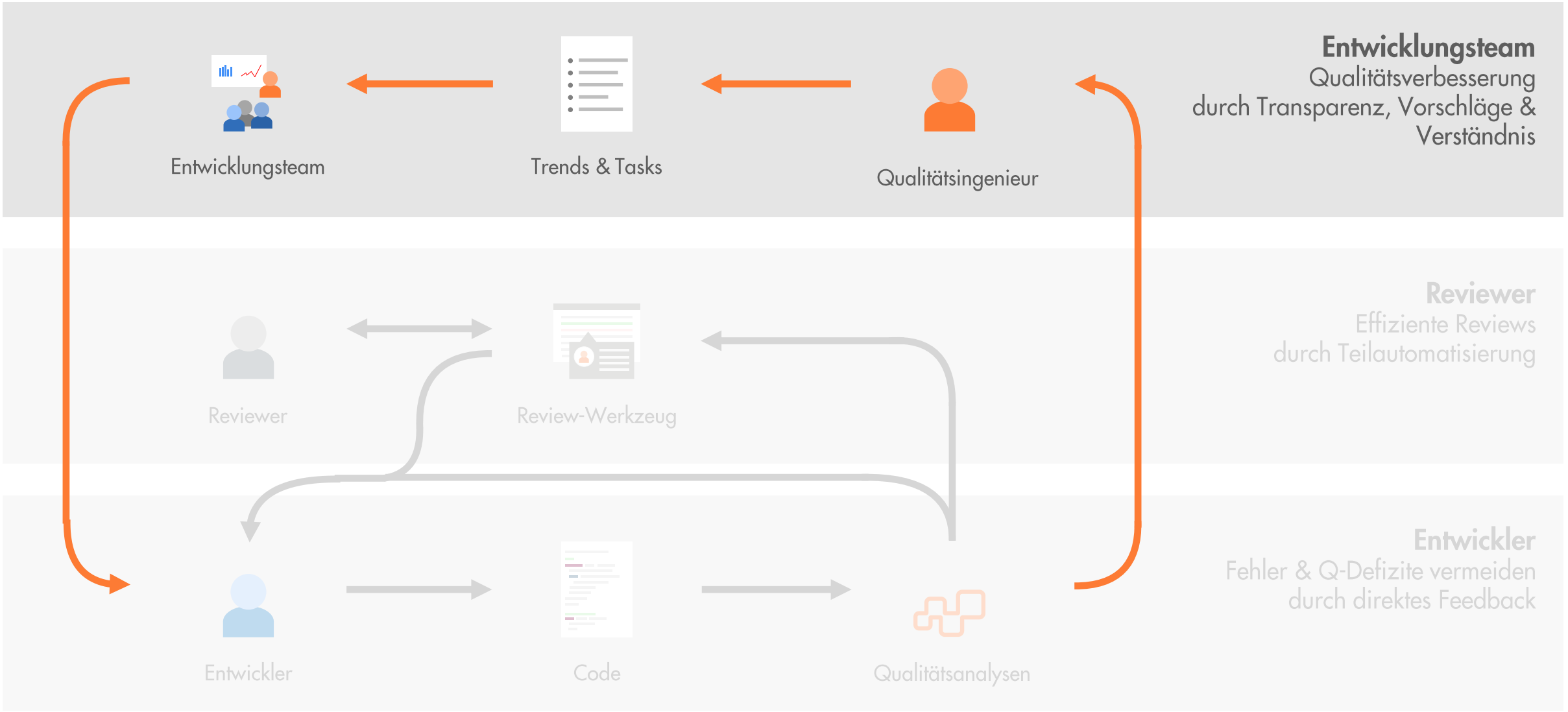
Feedbackschleife 2: Stat. Codeanalyse und Test-Gap-Analyse ermöglichen Reviewern effiziente Code Reviews



CCP-Integration durch Pull Request-Annotation



Feedbackschleife 3: Q-Retrospektiven unterstützen Entwicklungsteams bei QS

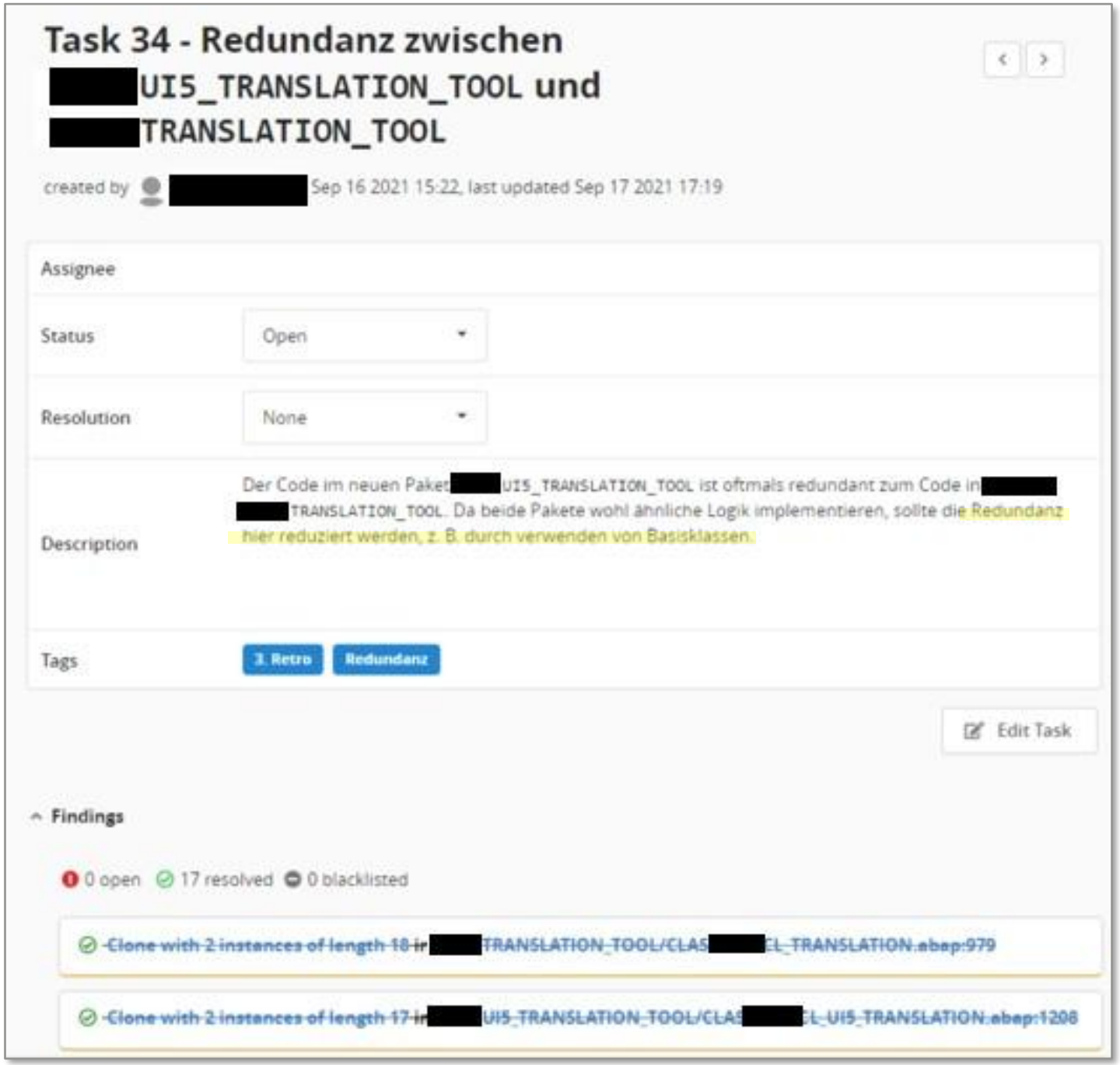


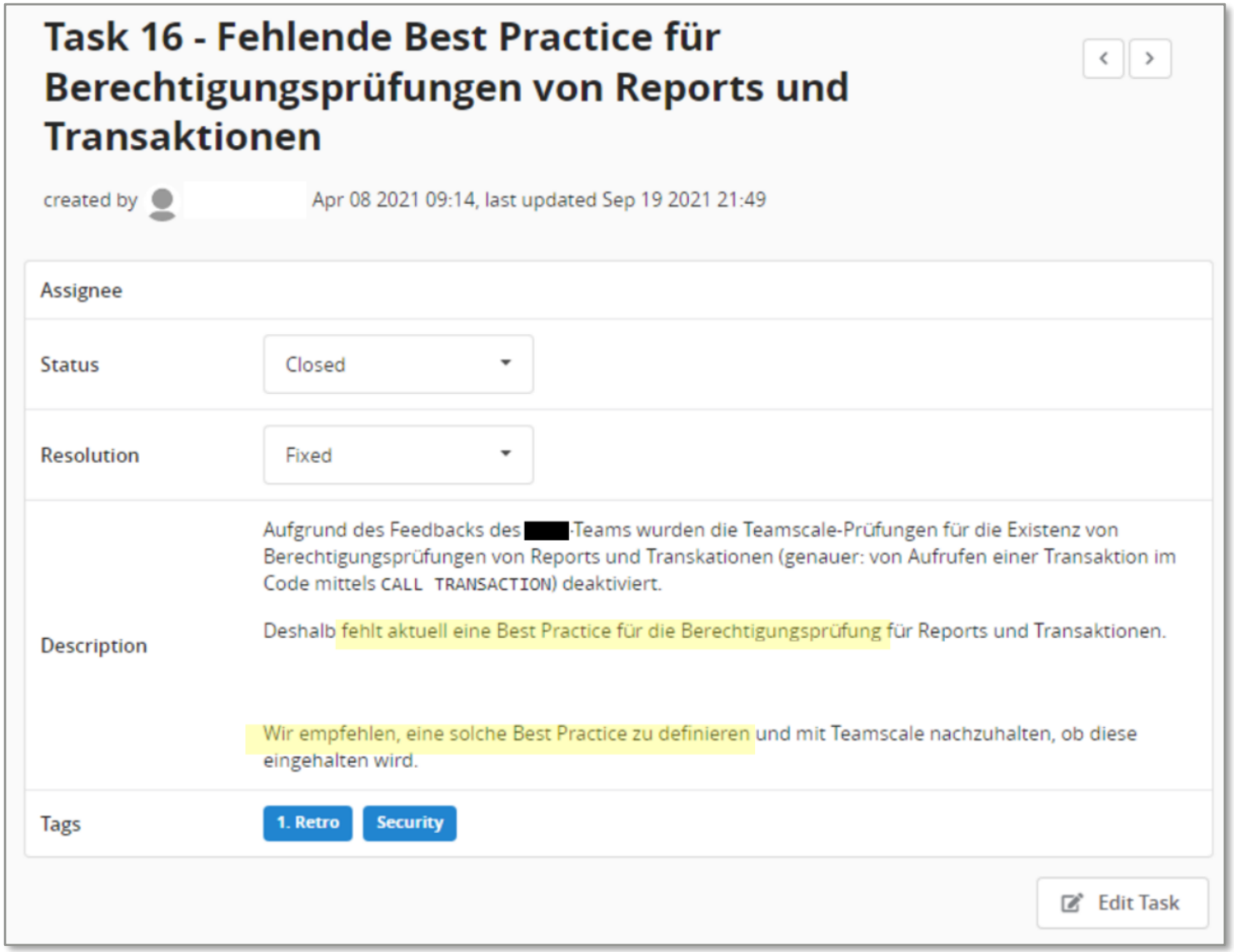

Q-Analysen & -Visualisierungen
Q-Retrospektiven
Was bringt‘s?

Was ist eine Q-Retrospektive?

- 📖 Regelmäßiger Workshop zur Rückschau auf Qualität
- 🎯 Team abholen, anhaltendes Q-Bewusstsein, gemeinsames Q-Verständnis
- 👥 Entwicklungs-/Testteam & Qualitätsingenieur & ggf. weitere Beteiligte
- 👤 Vorbereitet, moderiert und nachbereitet von Qualitätsingenieur
- 💬 Wartbarkeit, Security, Korrektheit, Testen, …
- 🗓️ Monatlich/ quartärlich bzw. je Sprint/ Release
- 🕑 60-120 min
- 🔎 Basiert auf Q-Analysen

Diskussion von Q-Verbesserungsvorschlägen


Task 34 - Redundanz zwischen [redacted]UI5_TRANSLATION_TOOL und [redacted]TRANSLATION_TOOL
created by [redacted] Sep 16 2021 15:22, last updated Sep 17 2021 17:19
Assignee
Status
Open
Resolution
None
Description
Der Code im neuen Paket [redacted]UI5_TRANSLATION_TOOL ist oftmals redundant zum Code in [redacted]TRANSLATION_TOOL. Da beide Pakete wohl ähnliche Logik implementieren, sollte die Redundanz hier reduziert werden, z. B. durch verwenden von Basisklassen.
Tags
3. Retro
Redundanz
Edit Task
Findings
0 open
17 resolved
0 blacklisted
Clone with 2 instances of length 18 in [redacted]TRANSLATION_TOOL/CLAS[redacted]CL_TRANSLATION.abap:979
Clone with 2 instances of length 17 in [redacted]UI5_TRANSLATION_TOOL/CLAS[redacted]CL_UI5_TRANSLATION.abap:1208



→ Verbesserungsvorschläge ermöglichen konkrete Verbesserung und Einplanung

→ Diskussionen schaffen gemeinsames Qualitätsverständnis und Best Practices

Diskussion von Q-Indikatoren

Übersicht Produktqualität (Allgemein)

Quality Indicator (QI)	Trend
Architekturkonformität	
Verletzungen	↘
Nicht abgedeckte Typen	↗
OWASP Dependency Findings	
Paketabhängigkeiten (ROT)	↘
Paketabhängigkeiten (Gelb)	↘
Redundanz	
Duplizierter Code	↗

Quality Indicator (QI)	Trend
Dokumentation	
Kommentarvollständigkeit	↘
Struktur	
Methodenlänge	↗
Schachtelungstiefe	↗
Dateigröße	↗
Testabdeckung	
Testabdeckung	

Übersicht Produktqualität (Codefindings)

Quality Indicator (QI)	Trend
Security	
Kritische Security Findings	→
Security Findings (Dichte)	→
Korrektheit	
Kritische Korrektheitsfindings	↘
Korrektheitsfindings (Dichte)	↗
Effizienz	
Kritische Effizienzfindings	→
Effizienzfindings (Dichte)	→

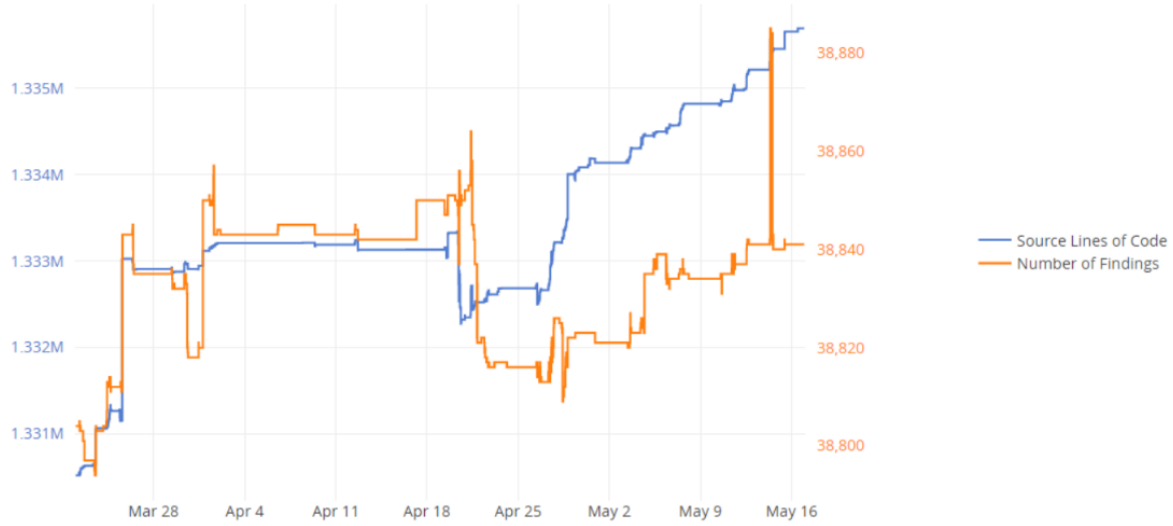
Quality Indicator (QI)	Trend
Verständlichkeit	
Kritische Verständlichkeitfindings	→
Verständlichkeitfindings (Dichte)	↘
Fehlerbehandlung	
Kritische Fehlerbehandlungen	↘
Fehlerbehandlungsfindings (Dichte)	↘

→ Q-Transparenz für Entwicklungsteam und Auftraggeber

→ Einhaltung von Q-Indikatoren kann überprüft und ggfs. gegengesteuert werden

Diskussion von Q-Trends

Trend 22.März - 17. Mai



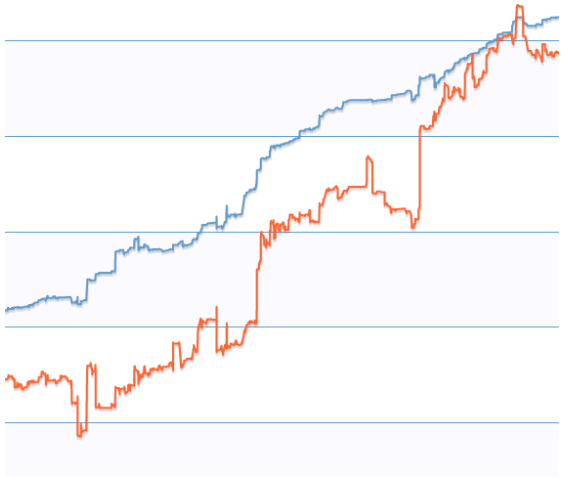
Ca. +5000 Zeilen (SLOC) Code, mehr neue Findings. Sprung am 14.05.: Code in `DG_APLOUT_TO_SCE` **kopiert, welcher wieder auskommentiert wurde.**

Findings-Delta

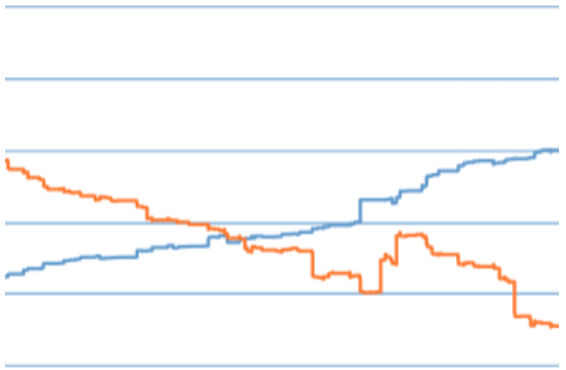
Findings | 213 | 176

Anzahl Codezeilen
Anzahl Findings

Trend-Analyse: QS-Auswirkung



»Proportionales Wachstum«
→ Keine oder wirkungslose QS
=> Analyse & Gegensteuerung notwendig



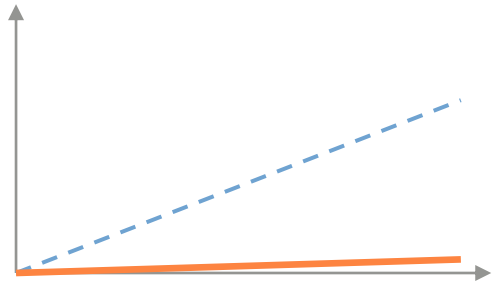
»X-Form«
→ Q-Verbesserung bei Codewachstum
=> Lob notwendig

→ Auswirkung der QS kann überprüft und ggfs. gegengesteuert werden

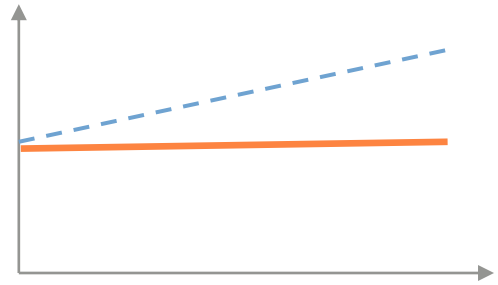
Anzahl Codezeilen
Anzahl Findings

Trend-Analyse: Erreichung von Q-Ziel

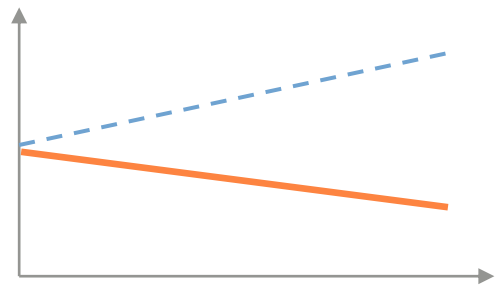
Neuentwicklung,
Q-Ziel »Perfekte Qualität«



Gewachsene Anwendung,
Q-Ziel »Qualität halten«



Gewachsene Anwendung,
Q-Ziel »Qualität mit Pfad-finderregel verbessern«



→ Erreichung von Q-Zielen kann überprüft und ggfs. gegengesteuert werden

Anzahl Codezeilen
Anzahl Findings

Trend-Analyse: Analyse von Q-Sprüngen

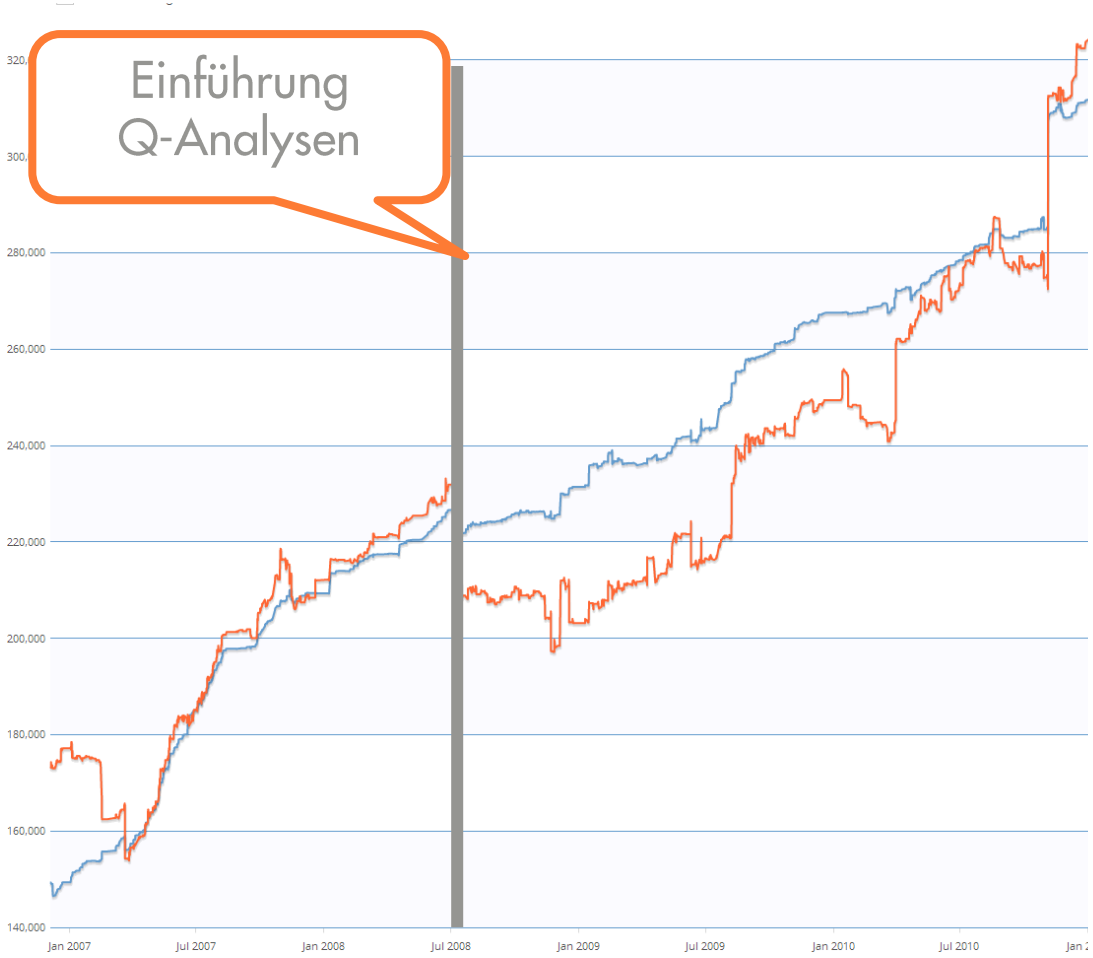
→ Ursachen für Q-Sprünge können verstanden und ggf. adressiert werden


WARNING
BRIGHT LIGHT
– Do not stare into light beam.
– Do not view at close range.
Failure to do so will cause
permanent eye damage.

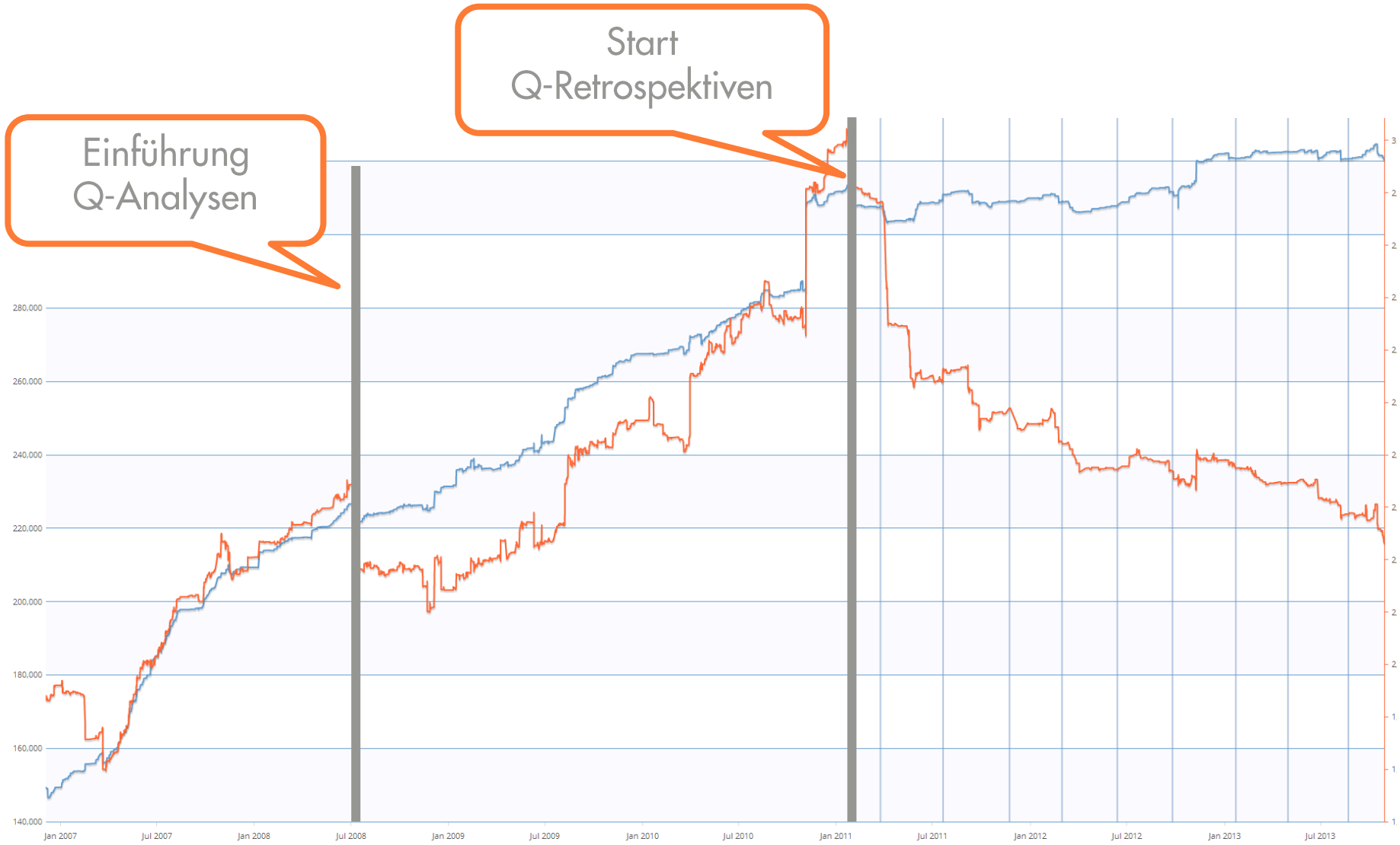
- Qualität regelmäßig auf der Agenda
  → **Anhaltendes Q-Bewusstsein**
- Qualitätsdiskussionen und Erarbeitung von Best Practices
  → **Gemeinsames Q-Verständnis**
- Transparenz bezüglich Qualitätstrends &
  ggf. Identifizierung von Handlungsbedarf
  → **Mögliche Q-Steuerung**

→ **Abholung von & Unterstützung für Entwicklungsteam**
→ **Abholung von Management bezüglich Q-Steuerung**

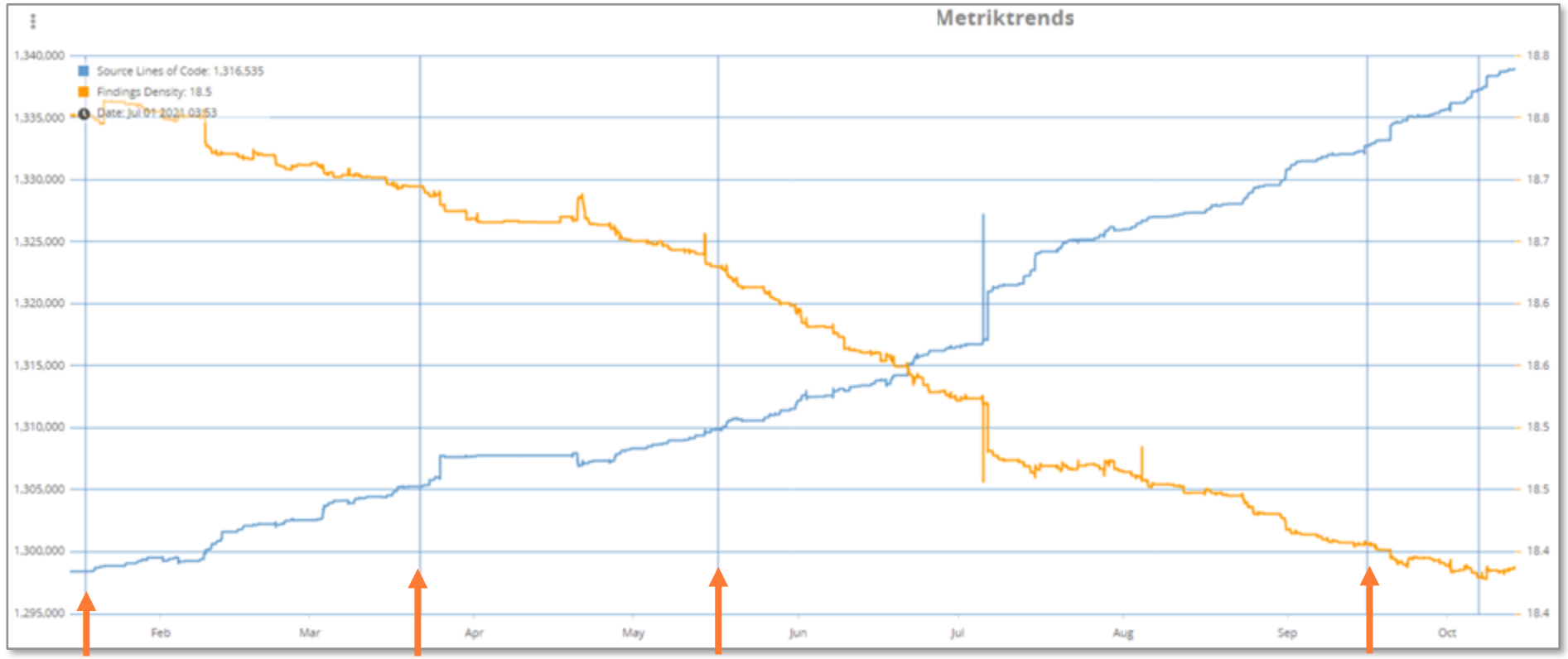

Q-Analysen & -Visualisierungen
Q-Retrospektiven
Was bringt's?



→ Q-Analysen haben i.d.R. nur einen kurzfristigen Effekt



→ Kombination aus Q-Analysen und Q-Retrospektiven hat i.d.R. einen anhaltenden Effekt

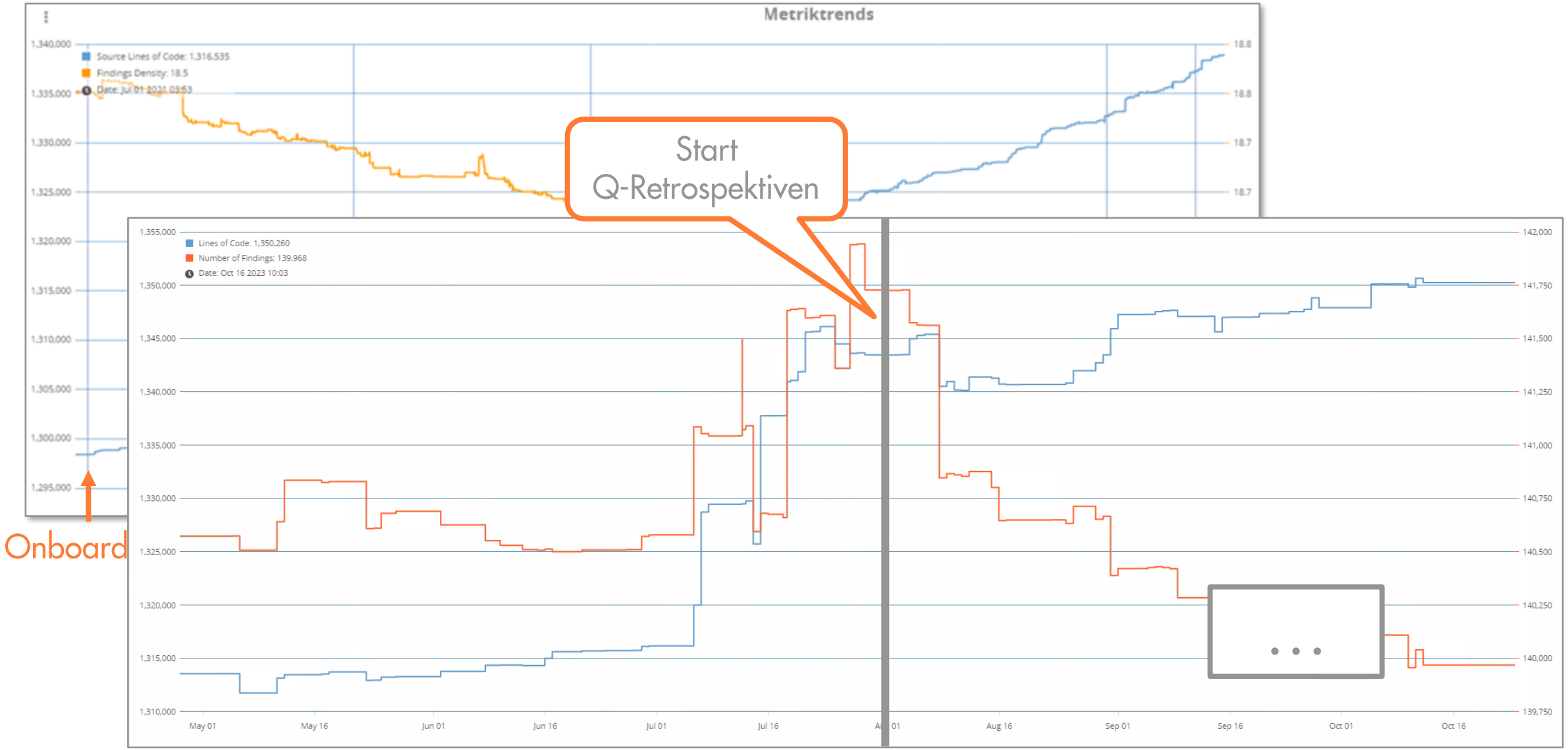


Onboarding

1. Q-Retro

2. Q-Retro

3. Q-Retro


Metriktrends
Source Lines of Code: 1,316,535
Findings Density: 18.5
Date: Jul 01 2021 03:53
1,340,000
1,335,000
1,330,000
1,325,000
1,320,000
1,315,000
1,310,000
1,305,000
1,300,000
1,295,000
18.8
18.8
18.7
18.7
Start
Q-Retrospektiven
Onboard
Lines of Code: 1,350,260
Number of Findings: 139,968
Date: Oct 16 2023 10:03
1,355,000
1,350,000
1,345,000
1,340,000
1,335,000
1,330,000
1,325,000
1,320,000
1,315,000
1,310,000
142,000
141,750
141,500
141,250
141,000
140,750
140,500
140,250
140,000
139,750
May 01
May 16
Jun 01
Jun 16
Jul 01
Jul 16
Aug 16
Sep 01
Sep 16
Oct 01
Oct 16

Zusammenfassung

1. Q-Retrospektive

Quality Indicator (QI)	Trend
Security	
Rote Security-Findings	→
Security-Findings je 1.000 Codezeilen	→
Codeanomalien	
Korrektheit	→
Codeanomalien je 1000 SLOC	↘

Quality Indicator (QI)	Trend
Redundanz	
Clone Coverage	↘
Codestruktur	
Struktur: Prozedurlänge	↘
Struktur: Schachtelungstiefe	↘

System Quality Overview

3. Q-Retrospektive

Quality Indicator (QI)	Trend
Security	
Rote Security-Findings	→
Security-Findings je 1.000 Codezeilen	↗
Codeanomalien	
Korrektheit	↗
Codeanomalien je 1000 SLOC	→

Quality Indicator (QI)	Trend
Redundanz	
Clone Coverage	↘
Codestruktur	
Struktur: Prozedurlänge	→
Struktur: Schachtelungstiefe	→

2. Q-Retrospektive

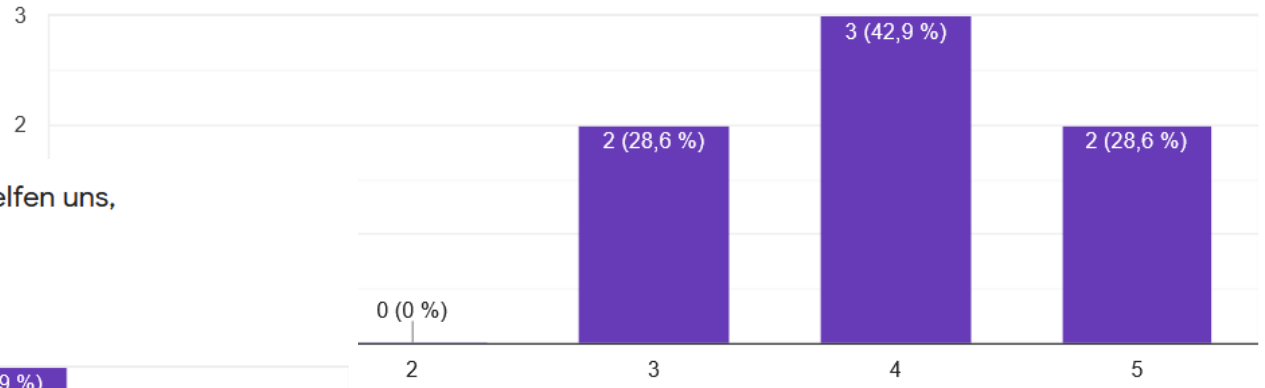
Quality Indicator (QI)	Trend
Security	
Rote Security-Findings	→
Security-Findings je 1.000 Codezeilen	→
Codeanomalien	
Korrektheit	↗
Codeanomalien je 1000 SLOC	↗

Quality Indicator (QI)	Trend
Redundanz	
Clone Coverage	→
Codestruktur	
Struktur: Prozedurlänge	→
Struktur: Schachtelungstiefe	→

Entwicklerumfrage* zu Q-Retrospektiven

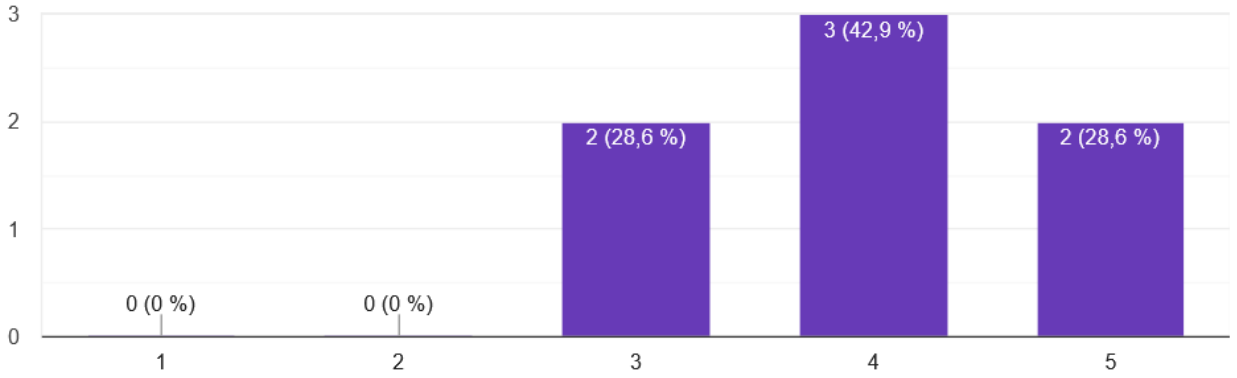
Wie ist Deine Einschätzung zu der Aussage "Durch die Qualitätsretrospektiven können wir unseren Code verbessern"?

7 Antworten



Wie ist Deine Einschätzung zu der Aussage "Die Qualitätsretrospektiven helfen uns, gemeinsame Best Practices und Standards zu etablieren"?

7 Antworten



* 7 Antworten im Mai 2021, 1: „Stimme überhaupt nicht zu“ – 5 „Stimme voll und ganz zu“

»Der Weg zu einer **nachhaltigen Qualitätsverbesserung** geht **nicht über ein Werkzeug**, sondern **über den Prozess**«

Christian Finkbeiner,
Softwarearchitekt,
SEW-EURODRIVE

Interview bzw. Vortrag mit Christian Finkbeiner

Vom Wiegen allein wird die Sau nicht fett

Wie ProSiebenSat.1, Munich Re, Allianz und Co. Softwarequalität verbessern

Dienstag, 11. März

11:00 bis 12:00 Uhr

Online und kostenlos

Jetzt anmelden: www.tmscl.me/sit-253-oop

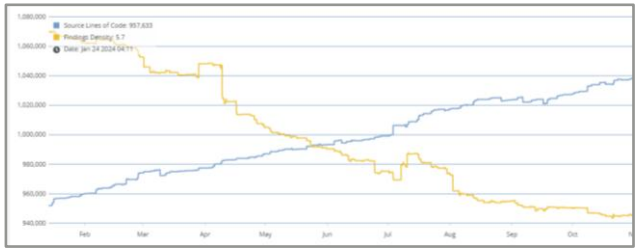
Dr. Sven Amann
(CQSE)

Zusammenfassung

→ Abholung aller Beteiligten ist essenziell für wirksame QS

→ Q-Analysen & -Visualisierungen machen Qualität sichtbar und diskutierbar

→ Q-Retrospektiven schaffen gemeinsames Q-Verständnis im Entwicklungsteam und Q-Steuerbarkeit für Management



→ Kombination von Q-Analysen und Q-Retrospektiven führt zu wirksamer, nachweisbarer Qualitätsverbesserung

Folien & Kontakt - Ich freue mich auf Fragen & Austausch ☺

Vortragsfolien:
tmscl.me/3OO8JnF

Dr. Tobias Röhm
roehm@cqse.eu
tmscl.me/coffee-tobias-roehm