

Continuous Quality Control

Qualität trotz immer kürzerer Releasezyklen

JFS 2025



Dr. Tobias Röhm

Dr. Tobias Röhm



Freelancer

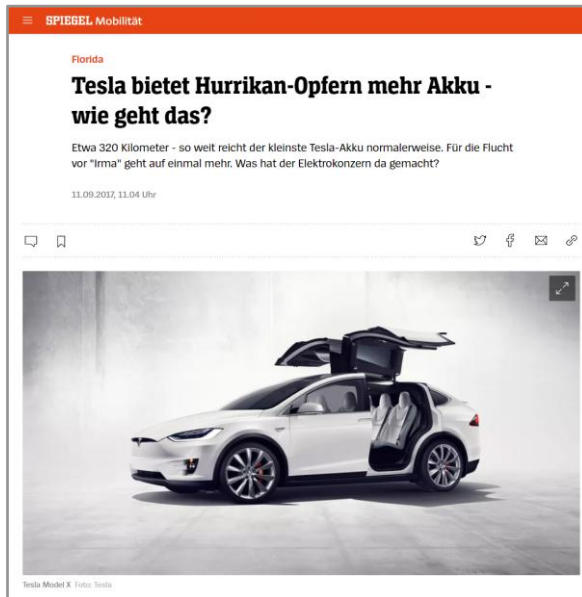


2004 - 2010

2010 - 2015

2015 - Heute

Kontext



Ereignisbasierte
Over-the-Air-Updates



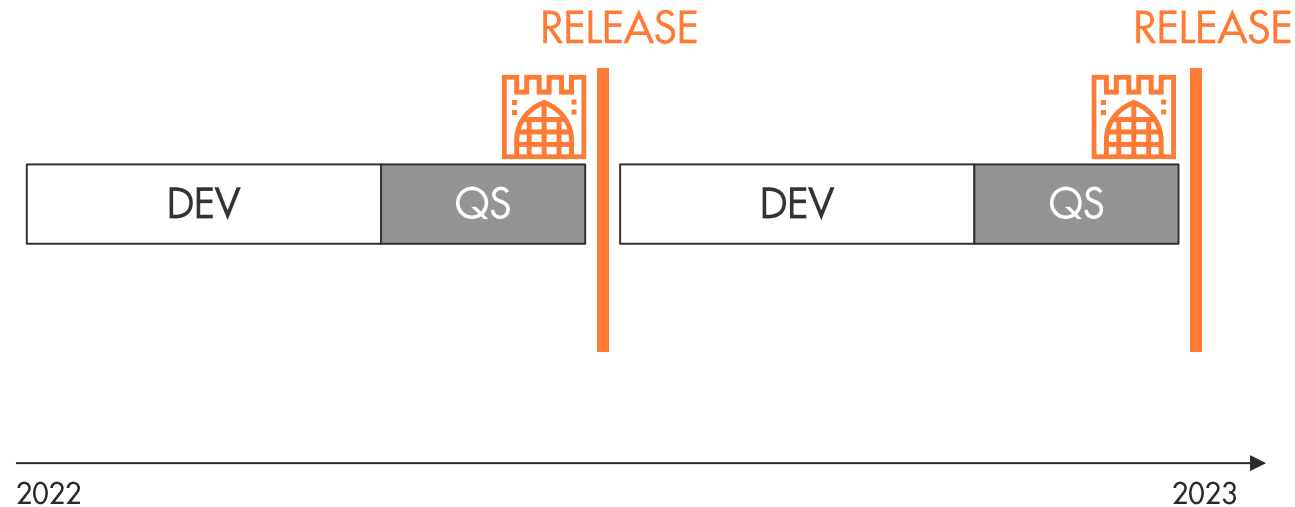
Viele, teilweise rückwirkende
Gesetzesänderungen



Zeitkritische Security-Patches

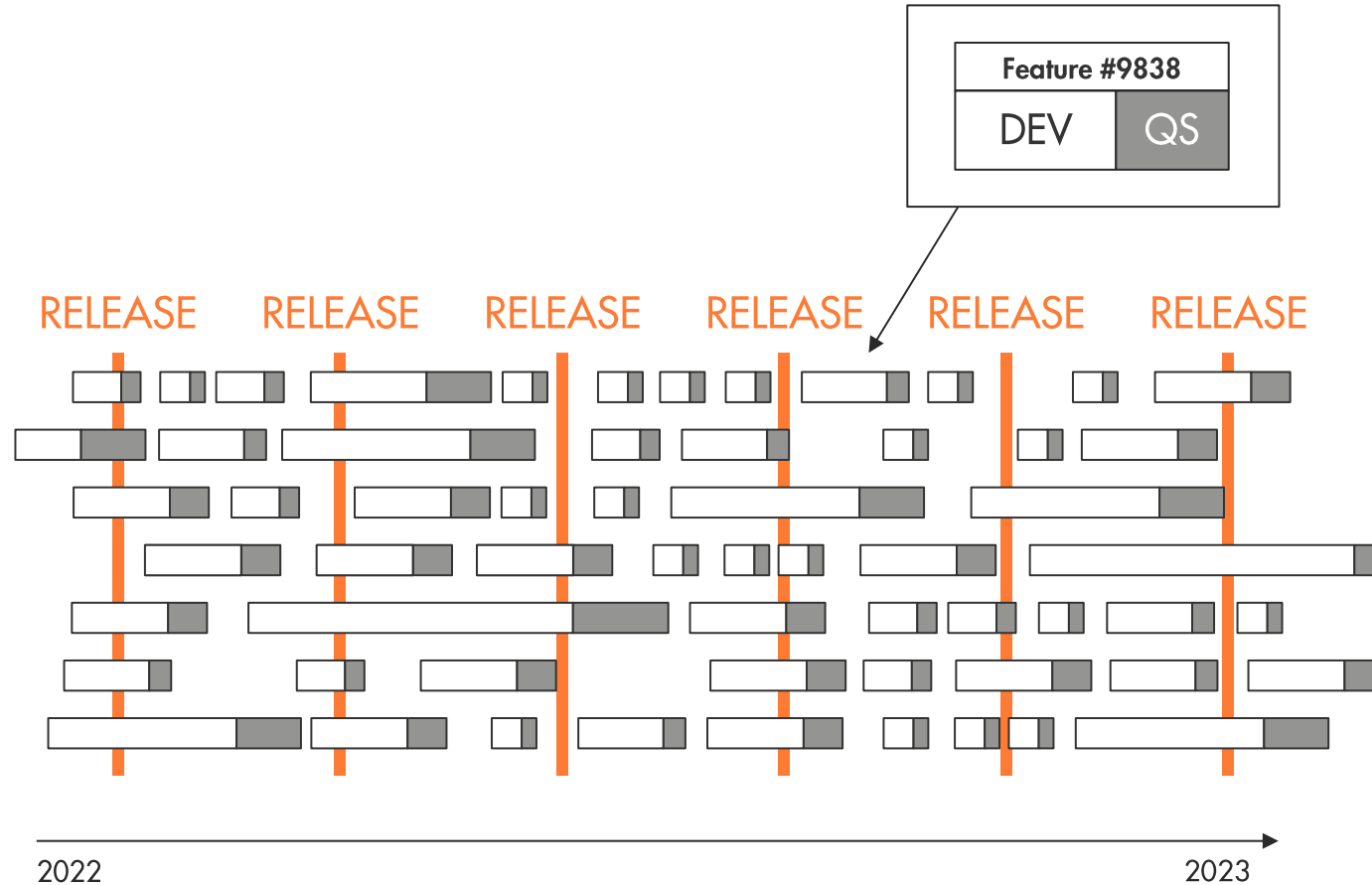
- Software-Updates müssen jederzeit möglich sein
- Software muss jederzeit ein releasefähiges Qualitätsniveau aufweisen

Problem: Traditionelle QS-Prozesse



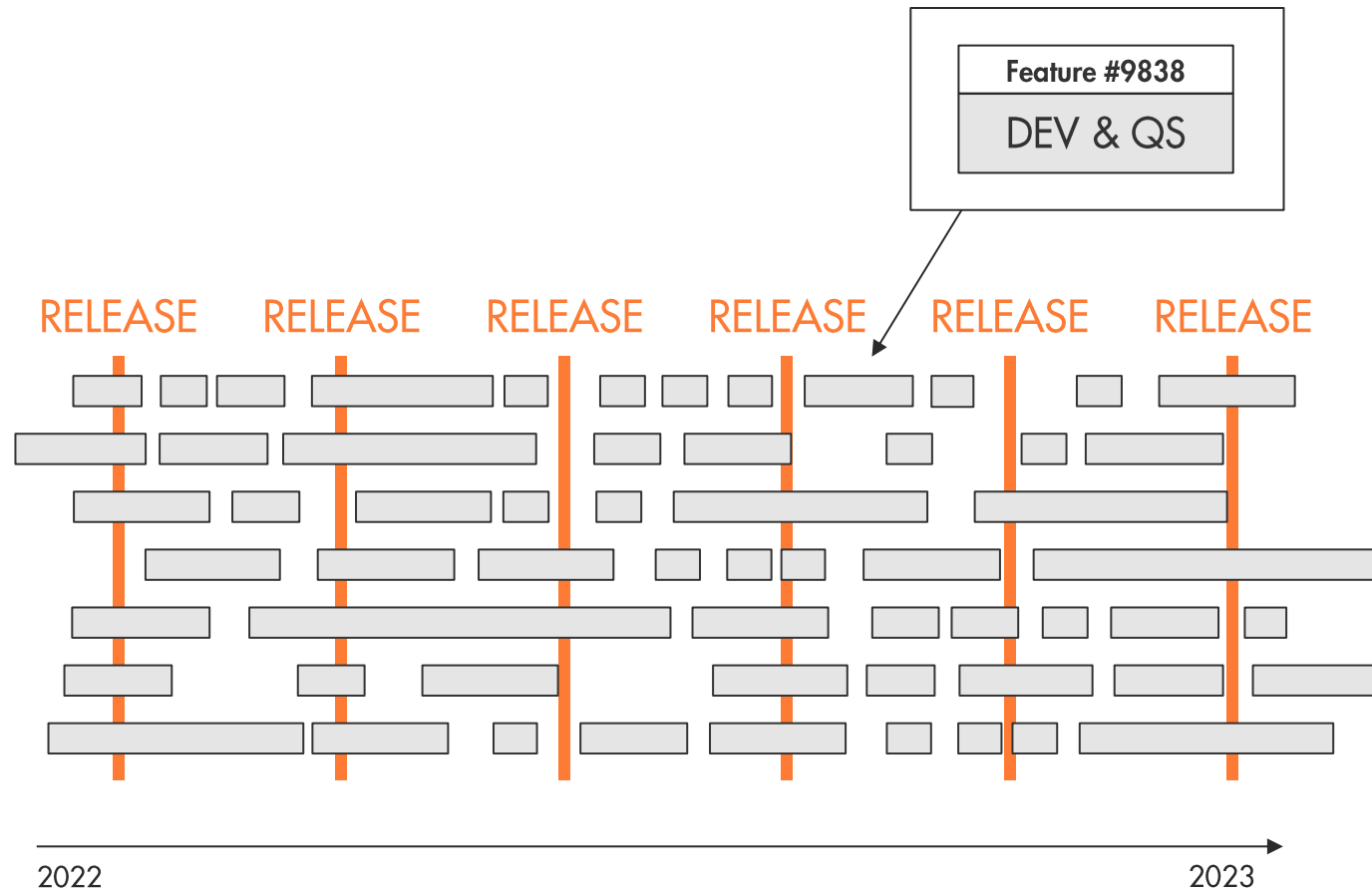
- Traditionelle QS-Prozesse sind (zu) langsam
- Software ist NICHT jederzeit auf einem releasefähigem Qualitätsniveau

Lösung: Continuous Quality Control (CQC)



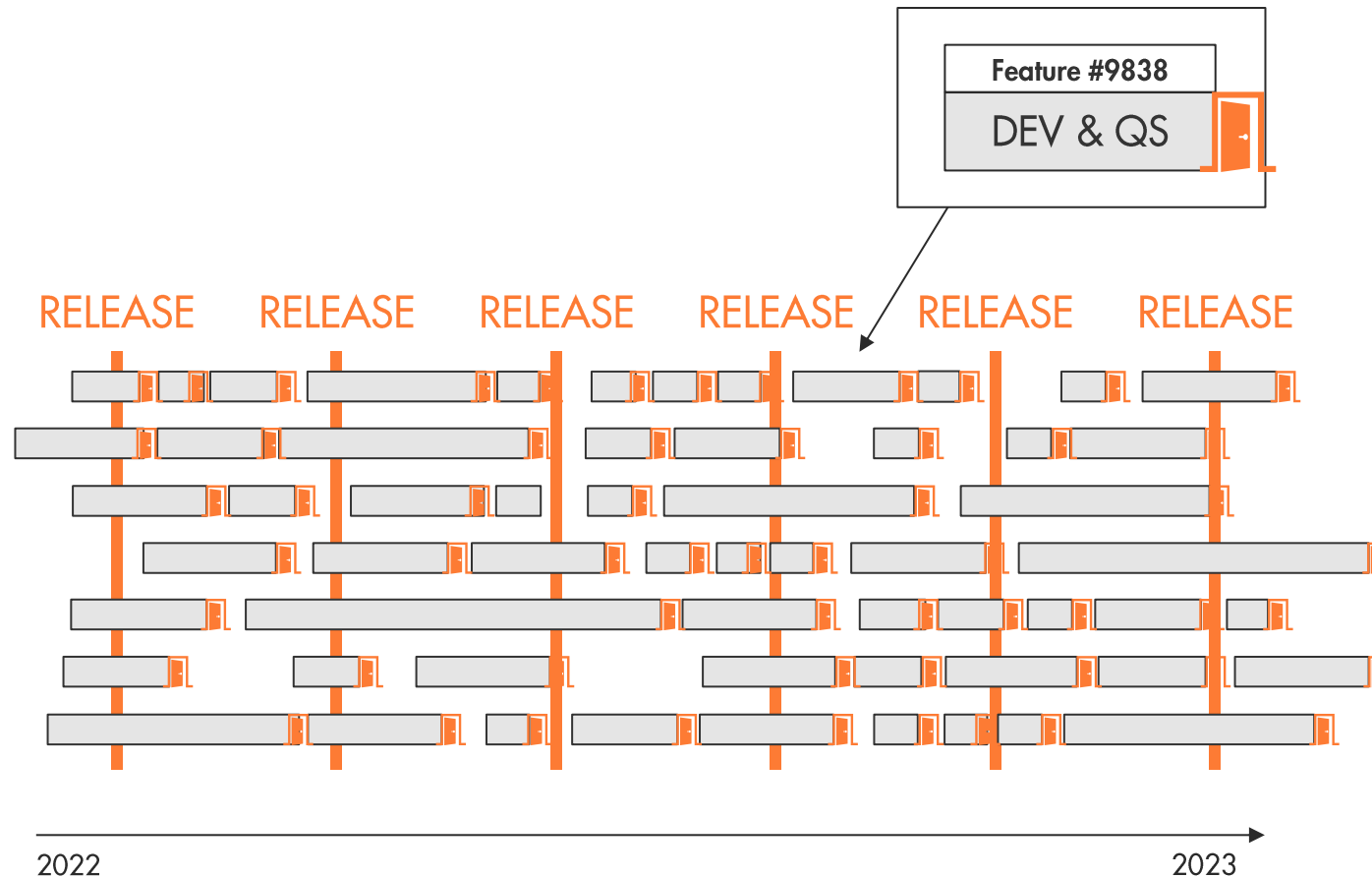
→ Qualitätssicherung auf Ticketebene um zu parallelisieren und zu beschleunigen

Lösung: Continuous Quality Control (CQC) II



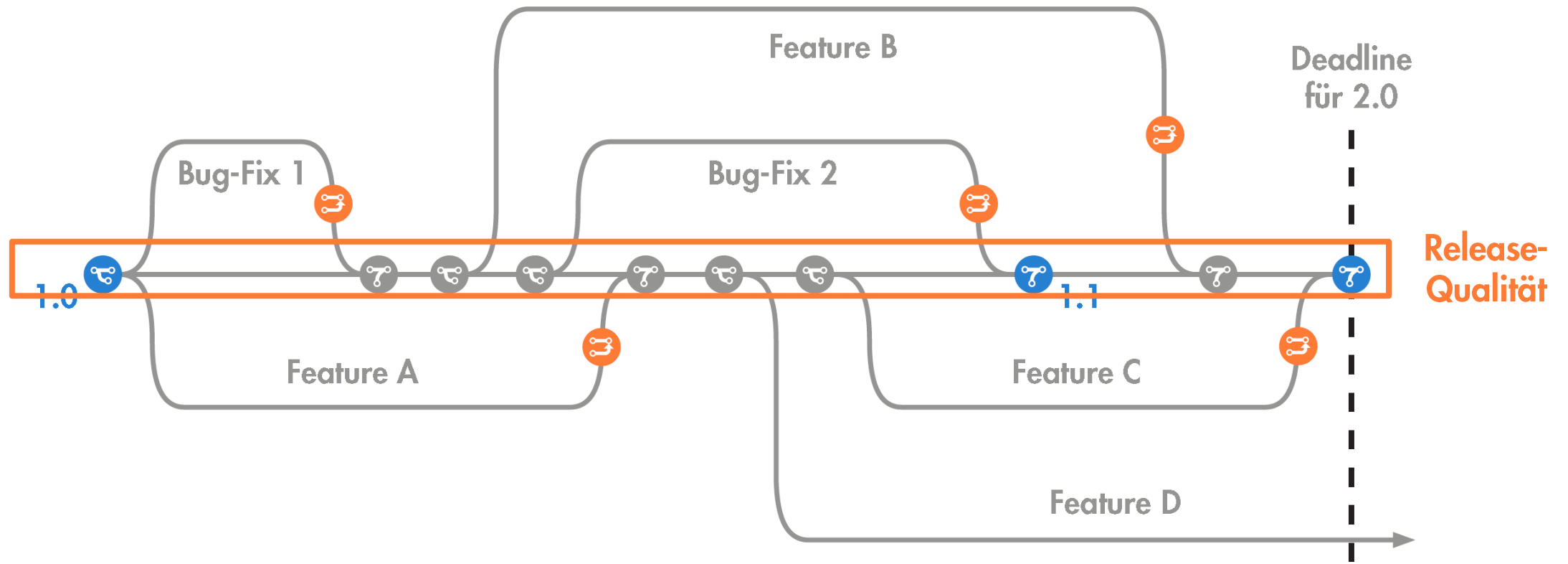
→ Verschmelzung von Entwicklung und QS um zu beschleunigen

Lösung: Continuous Quality Control (CQC) III



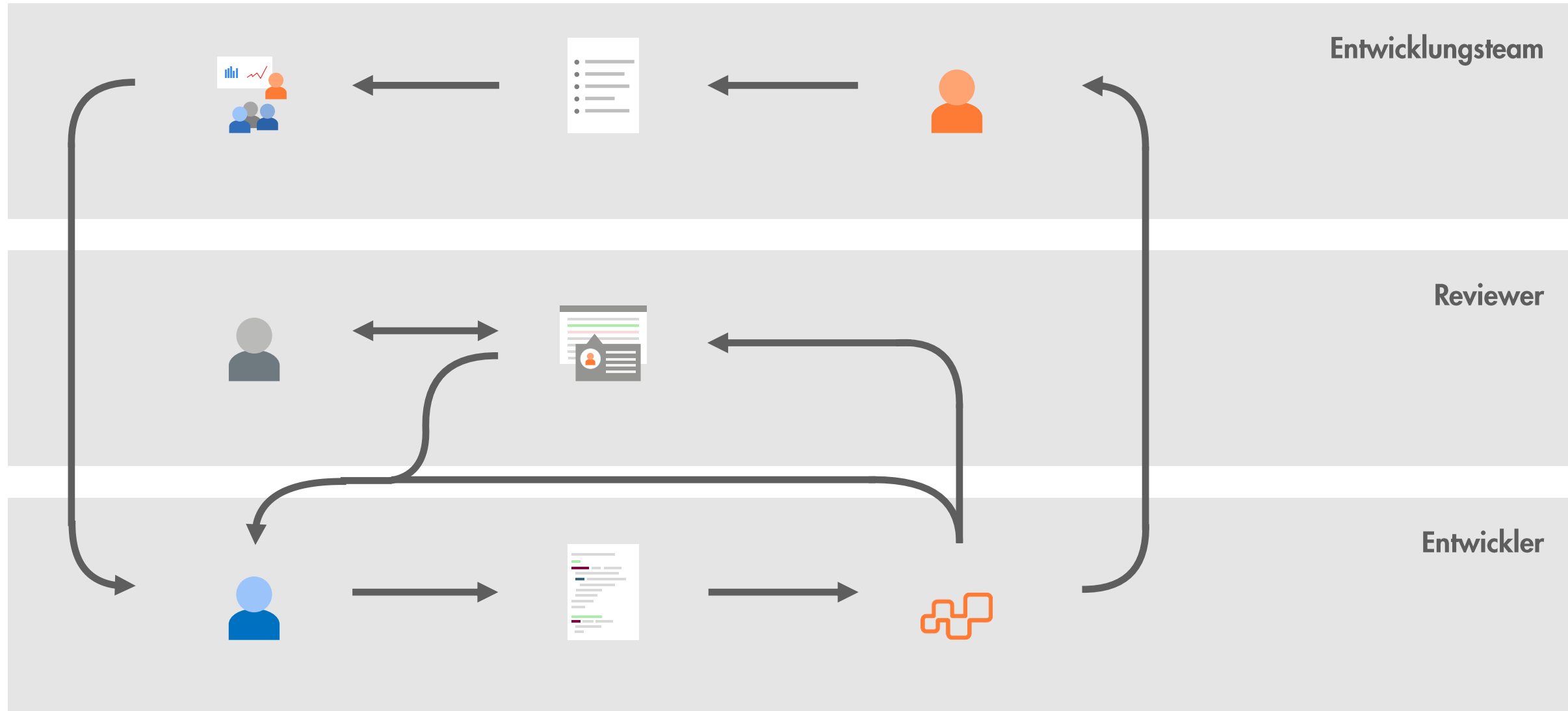
- Qualitätstürchen als frühe, leichtgewichtige Qualitätsprüfung
- Qualität ist jederzeit auf releasefähigem Niveau

Implementierung durch Feature Branches in Git



→ Merge Request als »Qualitätstürchen«

Agenda: Continuous Quality Control-Feedbackschleifen

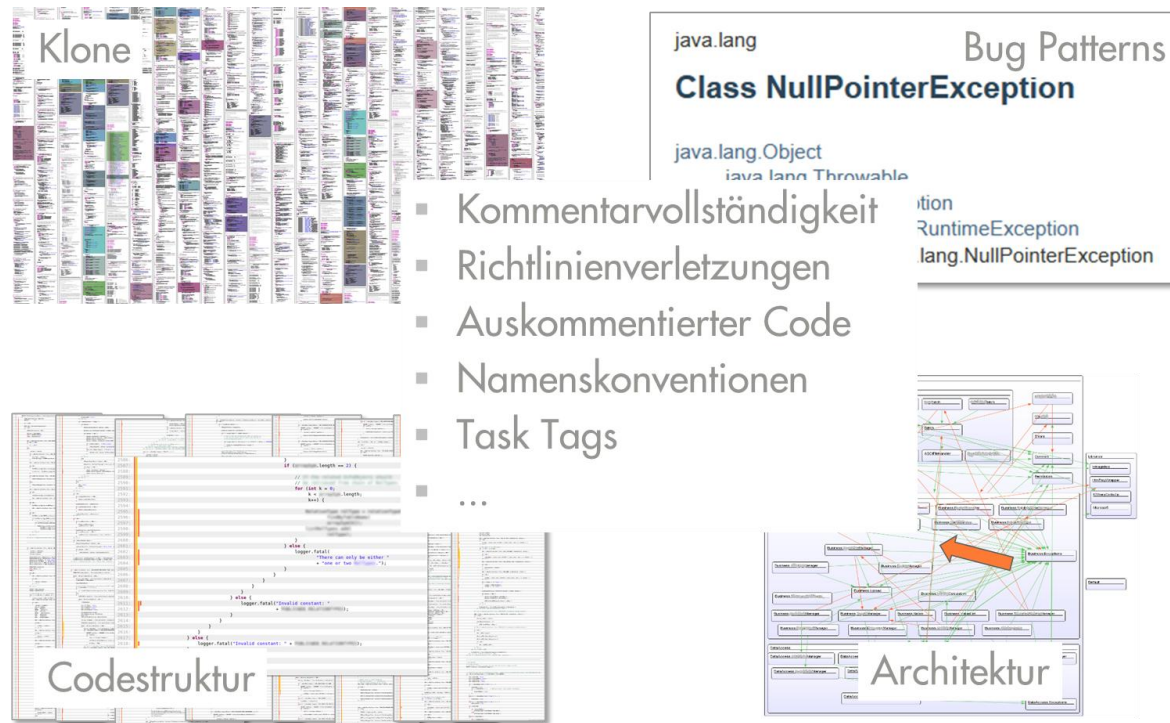


Feedbackschleife 1:
Statische Codeanalyse hilft Entwicklern,
Fehler & Q-Defizite zu vermeiden

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

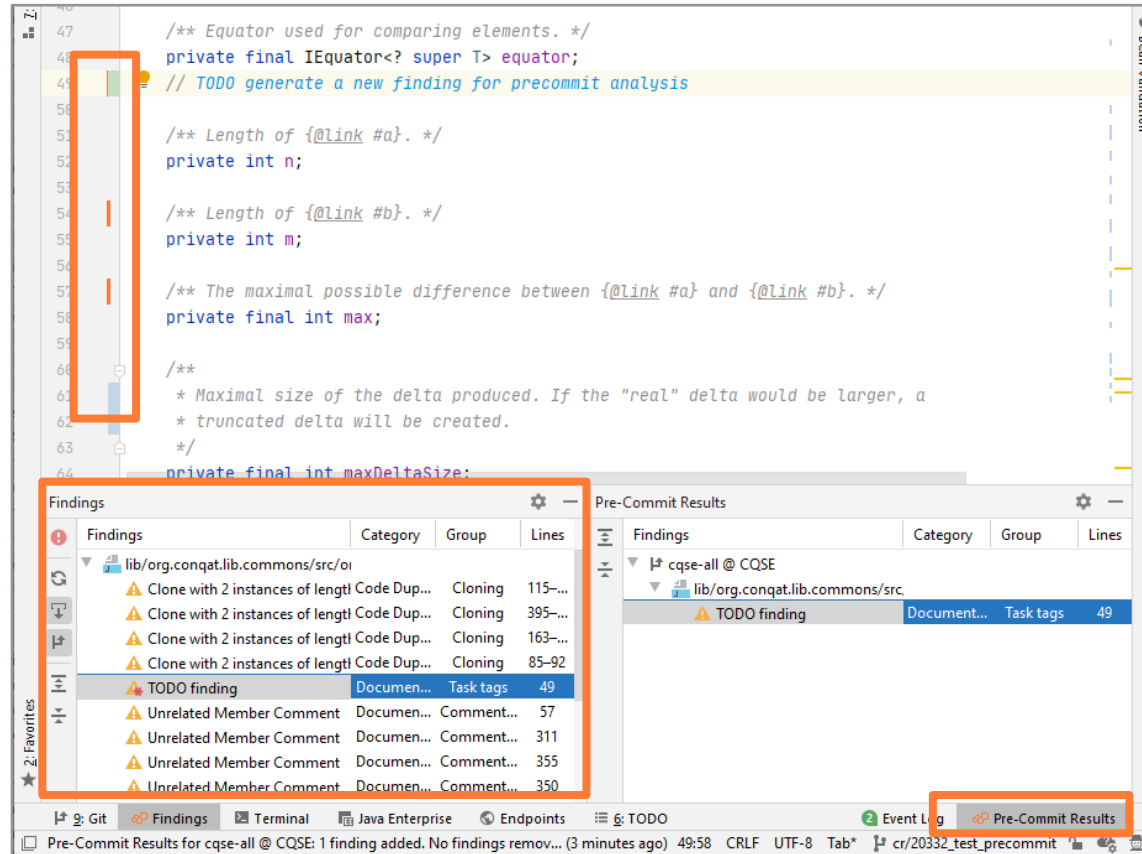


Übersicht Statische Codeanalyse



→ Statische Codeanalyse schafft Transparenz bezüglich Code- & Architekturqualität

IDE-Integration & Precommit-Analyse



Precommit-Analyse ermöglicht Q-Feedback bereits vor VCS-Commit

Inkrementelle Analyse erlaubt schnelle Ergebnisse auch für komplexe Analysen

→ Feedback aus statischer Codeanalyse ist schnell, fokussiert und integriert nutzbar

Findings-Churn & Qualitätsziele für gewachsene Systeme



Share more code between dispatchSessionStorageEvents() and dispatchLocalStora...
by Chris Dumez as revision ffa7b160 in main (github)
Files: 1 changed
Findings 

Yesterday 17:34



CachedResourceLoader::allCachedSVGImages() reparses resource URLs unnecessari...
by Ch...
Files:
Findi...

Yesterday 17:26



**[WinCairo][WK2] animations/par...
... is timing out https://bug...**
by Fujii Hironori as revision 2d29f... in main (github)
Files: 2 changed
Findings 

Yesterday 13:20



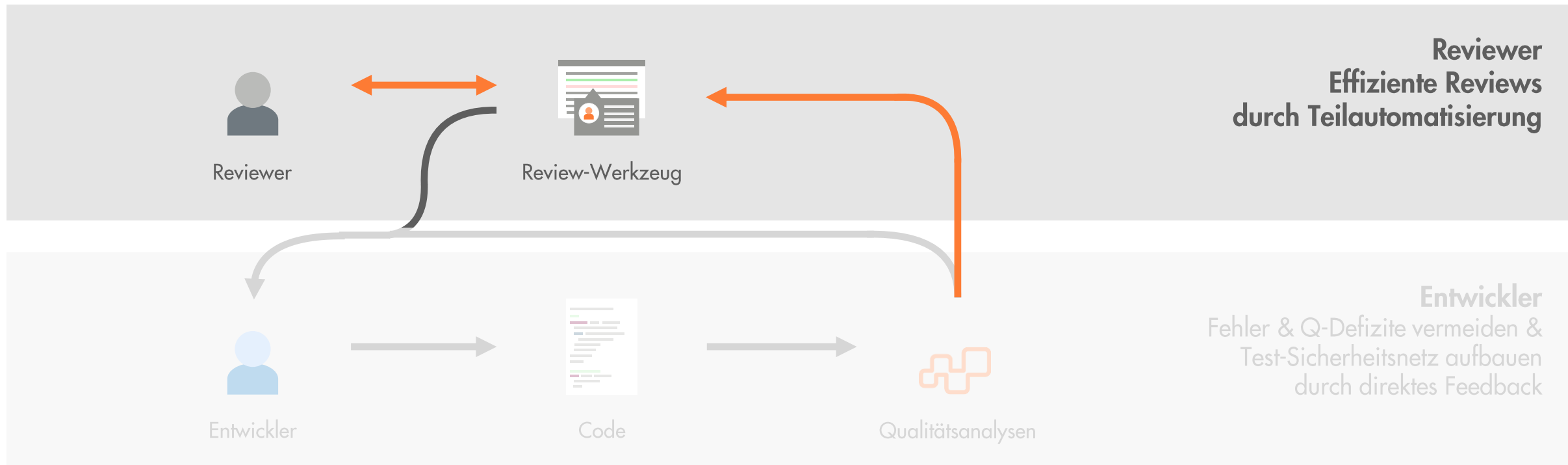
[git-webkit] Link to GitHub wiki https://bugs.webkit.org/show_bug.cgi?id=2370...

Yesterday 13:11

Unterscheidung zwischen »neuen Findings« und »Findings in geändertem Code« erlaubt schrittweise Q-Verbesserung von gewachsenen Codebasen mit der Pfadfinderregel.

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

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



Annotation von Findings aus statischer Codeanalyse an Merge/Pull Requests

Code Pull requests 2 Projects Security Insights

Modify the code #7

Open asteckermeier wants to merge 2 commits into from merge_request_tga_demo_5

Conversation 0 Commits 2 Checks 1 Files changed 1

Fix method not being called ac2cbc2

Teamscale (SWE)

teamscale-findings Resolve

Added Findings

This pull request would introduce 1 new finding

DETAILS

Teamscale Findings 1 2

WhiteBoard/WBProcessManager.m#L115: TODO (AS) We should switch this to always log.

ANNOTATIONS

Check warning on line 115 in WhiteBoard/WBProcessManager.m

teamscale-swe / teamscale-findings

WhiteBoard/WBProcessManager.m#L115

TODO (AS) We should switch this to always log.

View more details on Teamscale (SWE)

PR #4

Open inkhater wants to merge 22 commits into master from firstbranch

Conversation 0 Commits 22 Checks 1 Files changed 3

Changes from all commits File filter... Jump to... ⚙

4 jabref/JabRefGUI.java

```
35 @@ -35,7 +35,7 @@
36 public class JabRefGUI {
37
38 - private static final Logger LOGGER = LoggerFactory.getLogger(JabRefGUI.class);
39 + public static final Logger LOGGER = LoggerFactory.getLogger(JabRefGUI.class);
```

Check warning on line 38 in jabref/JabRefGUI.java

Teamscale App / Teamscale Findings

jabref/JabRefGUI.java#L38

Interface comment missing

2 jabref/logic/bibtex/BibEntry/Writer.java

```
163 @@ -163,7 +163,7 @@ private void writeField(BibEntry entry, Writer out, String name, int indentation
164 try {
165 out.write(fieldFormatter.format(field.get(), name));
166 out.write('.') + OS.NEWLINE);
167 } catch (IOException ex) {
168 throw new IOException("Error in field '" + name + "': " + ex.getMessage(), ex);
169 }
```

Check failure on line 168 in jabref/logic/bibtex/BibEntry/Writer.java

Teamscale App / Teamscale Findings

jabref/logic/bibtex/BibEntry/Writer.java#L166-L168

Catch clause catches generic exception 'Exception'



Reviewer/Test(er)

Funktioniert Ticket-Implementierung?
Ging aus Versehen etwas kaputt?



Anwendung

Annotation von Testergebnissen an Merge/Pull Requests

Cr/20002 new report template

Edited 9 hours ago by build

 **Request to merge** `cr/20002_new_report_t...`  **into** `teamscale/v5.4.x`

Open in Web IDE

Check out branch



The source branch is 31 commits behind the target branch

 **Pipeline #89447 failed for 040f1ce9 on** `cr/20002_new_report_t...`



 Test summary contained 5 failed test results out of 4318 total tests

Expand

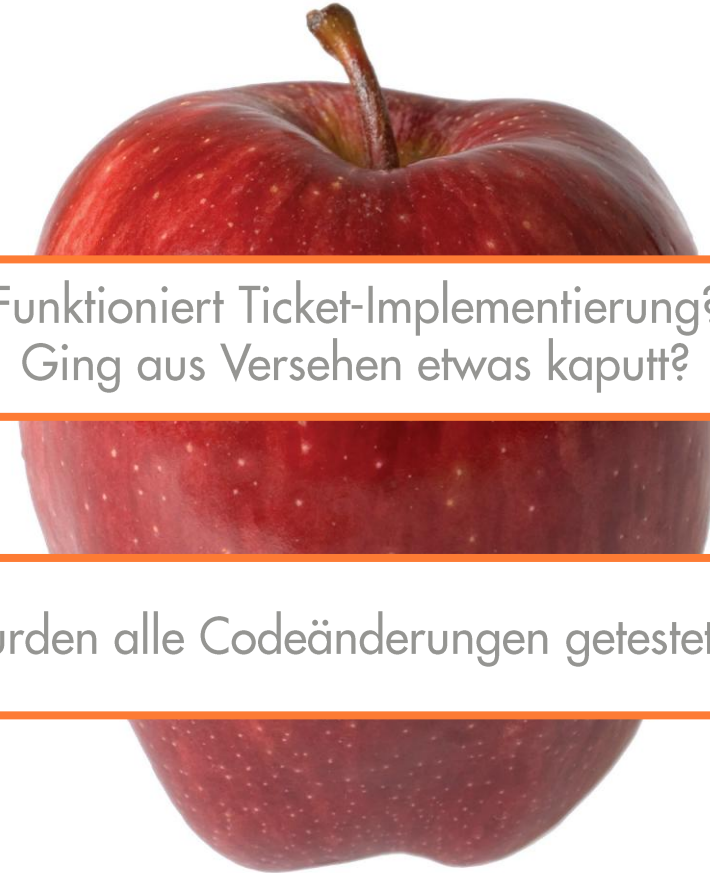
 **Merge** ☒ Delete source branch ☐ Squash commits 

> **8 commits** and **1 merge commit** will be added to `teamscale/v5.4.x`. [Modify merge commit](#)

You can merge this merge request manually using the [command line](#)



Reviewer/Test(er)

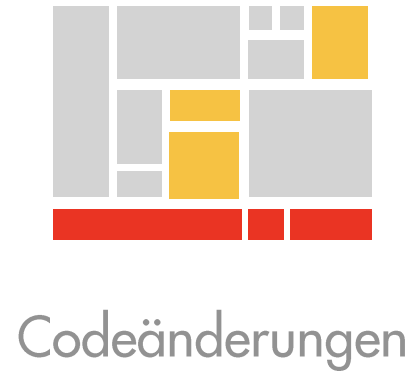


Funktioniert Ticket-Implementierung?
Ging aus Versehen etwas kaputt?

Wurden alle Codeänderungen getestet?

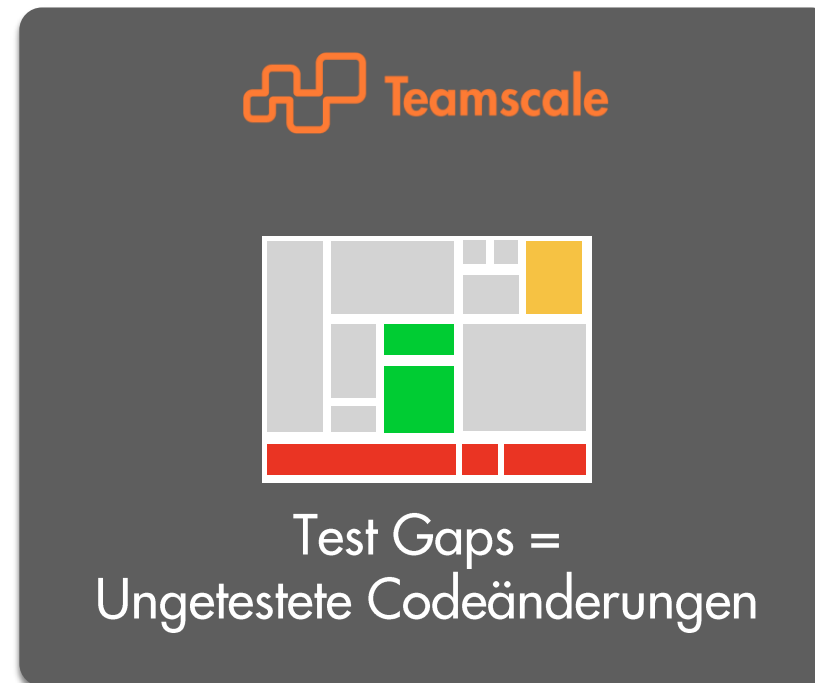
Anwendung

Test-Gap-Analyse



Test-Gap-Analyse identifiziert ungetestete Codeänderungen. Diese haben ein hohes Fehlerrisiko.

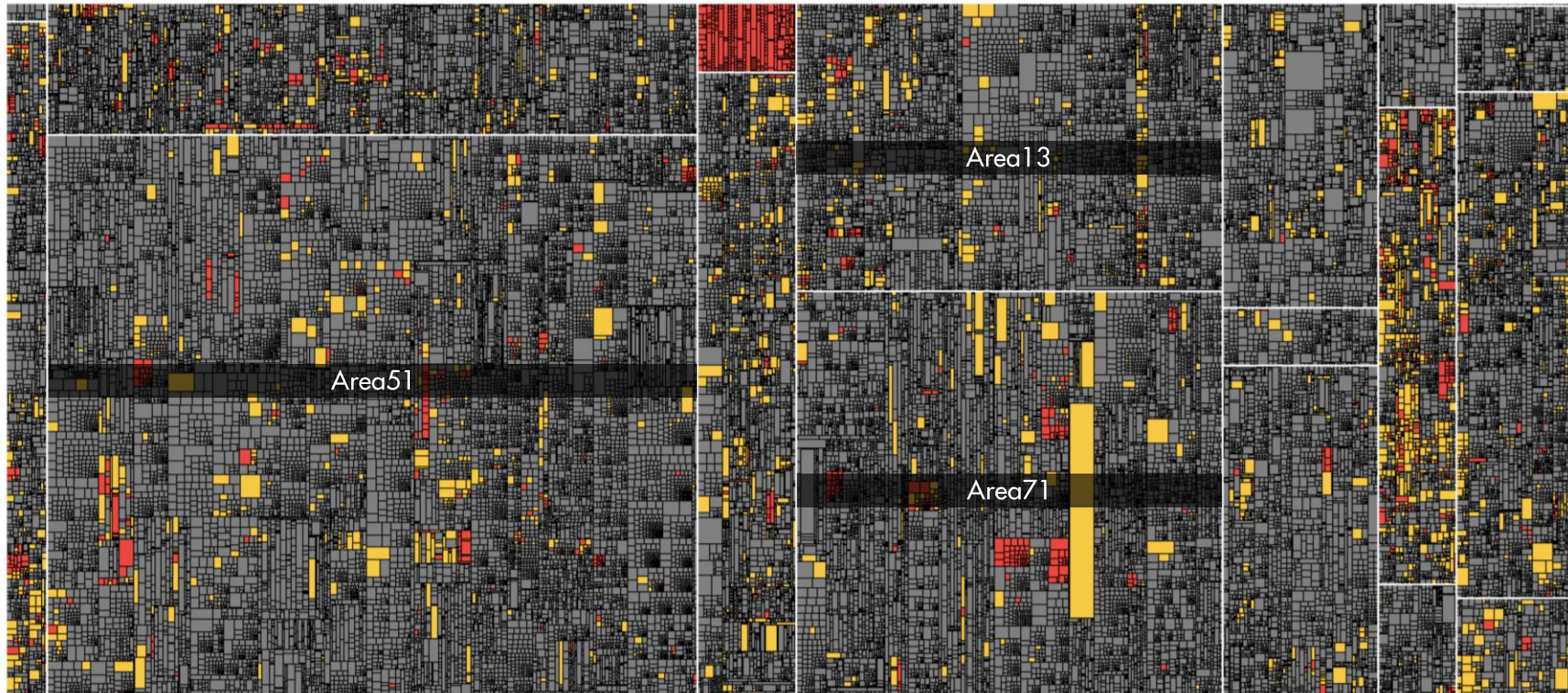
Did we test our changes? Assessing alignment between tests and development in practice. (Eder et al. 2013)



Codeänderungen

Churn

Jul 10 2022 - Aug 10 2022



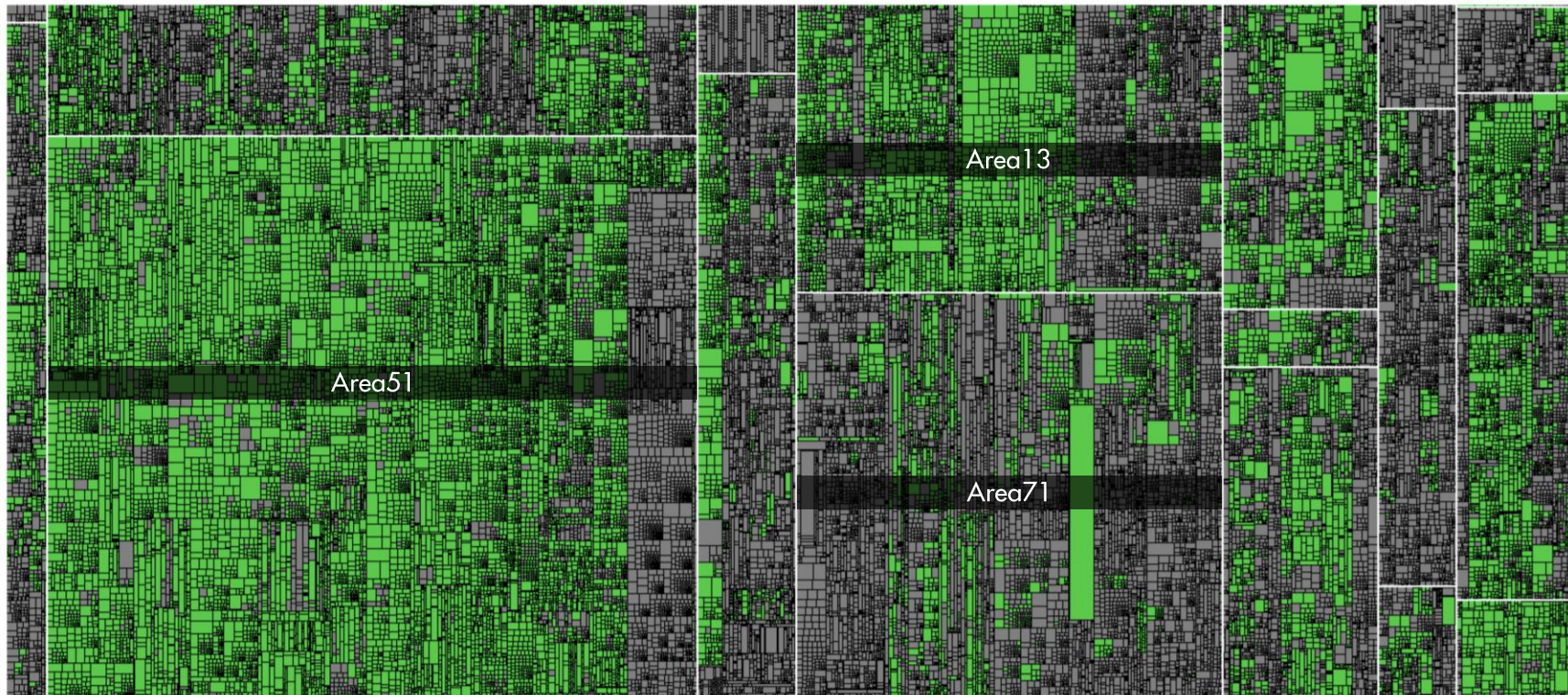
Methoden wurden...

- ... nicht geändert
- ... hinzugefügt
- ... geändert

Testcoverage

Function Execution

Jul 10 2022 - Aug 10 2022 | Execution Ratio: 43.79%



Methoden wurden...

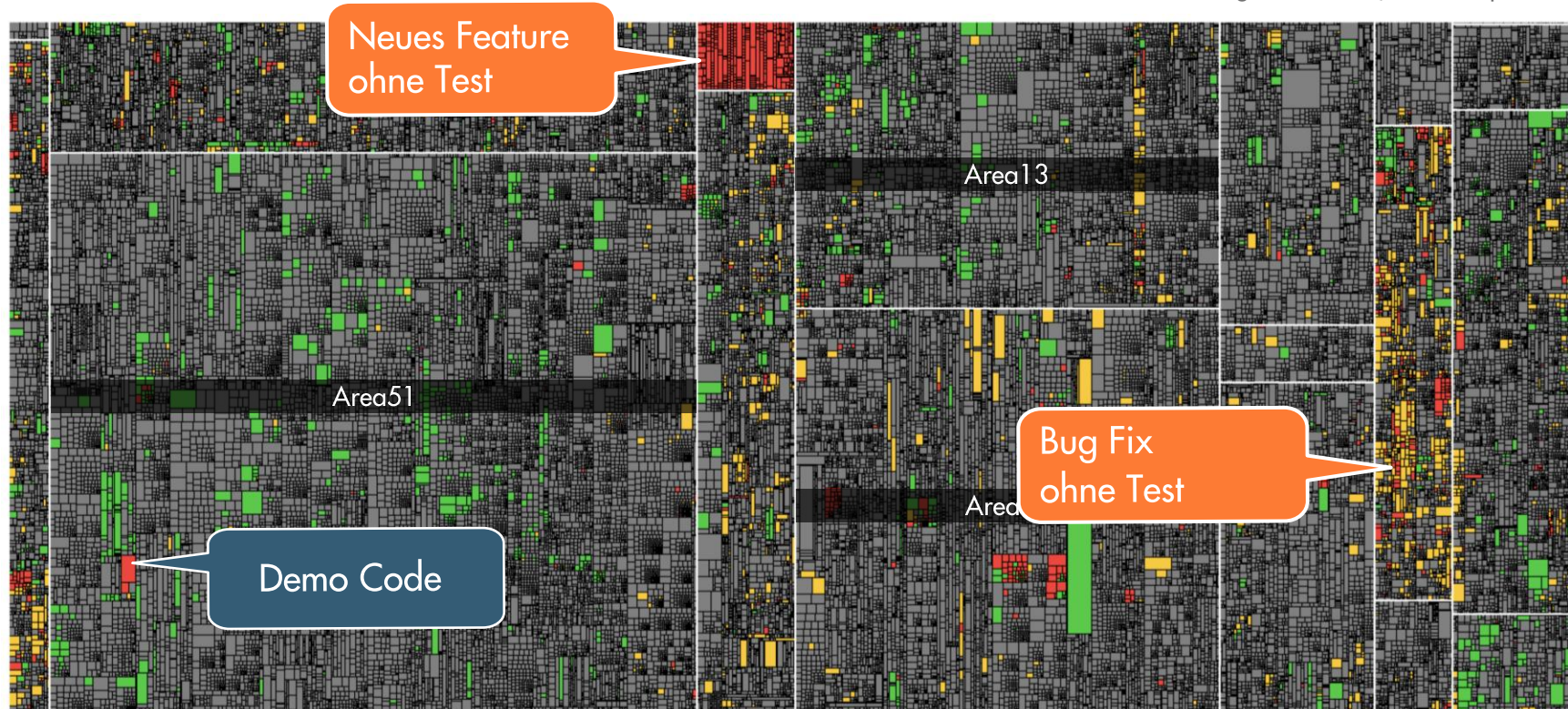
■ ... nicht getestet

■ ... im Test ausgeführt

Test Gaps (= geänderter, ungetesteter Code)

Test Gaps

Jul 10 2022 - Aug 10 2022 | Test Gap: 63%



Methoden wurden...

- ... nicht geändert
- ... hinzugefügt & nicht getestet
- ... geändert & nicht getestet
- ... geändert & getestet

Annotation von Test Gaps an Merge/Pull Requests

Code Pull requests 2 Projects Security Insights

Modify the code #7

Open asteckermeier wants to merge 2 commits into from merge_request_tga_demo_5

Conversation 0 Commits 2 Checks 1 Files changed 1

Fix method not being called ac2cbc2

Teamscale (SWE)

teamscale-findings Resolve

Teamscale (SWE) / teamscale-findings
Started 4m 8s ago

Added Findings

This pull request would introduce 1 new finding

DETAILS

Teamscale Findings 1 2 2 Test Gaps (67%)

WhiteBoard/WBProcessManager.m#L115: TODO (AS!) We should switch this to always log. (view in Teamscale)

ANNOTATIONS

Check warning on line 115 in WhiteBoard/WBProcessManager.m

teamscale-swe / teamscale-findings

WhiteBoard/WBProcessManager.m#L115

TODO (AS!) We should switch this to always log.
https://teamscale.my-company.com/findings.html#details/sam-faq-plms-project-whiteboard?merge_request_tga_d

View more details on Teamscale (SWE)

7 Modify the code

Request to merge merge_request_tga_demo_5 into OPEN NEGATIVE

This change contains 3 commits and introduces 1 new finding

Test Gaps

May 21 2022 12:22-Now | Test Gap: 67% Coverage sources Sydney-IOS-TOT

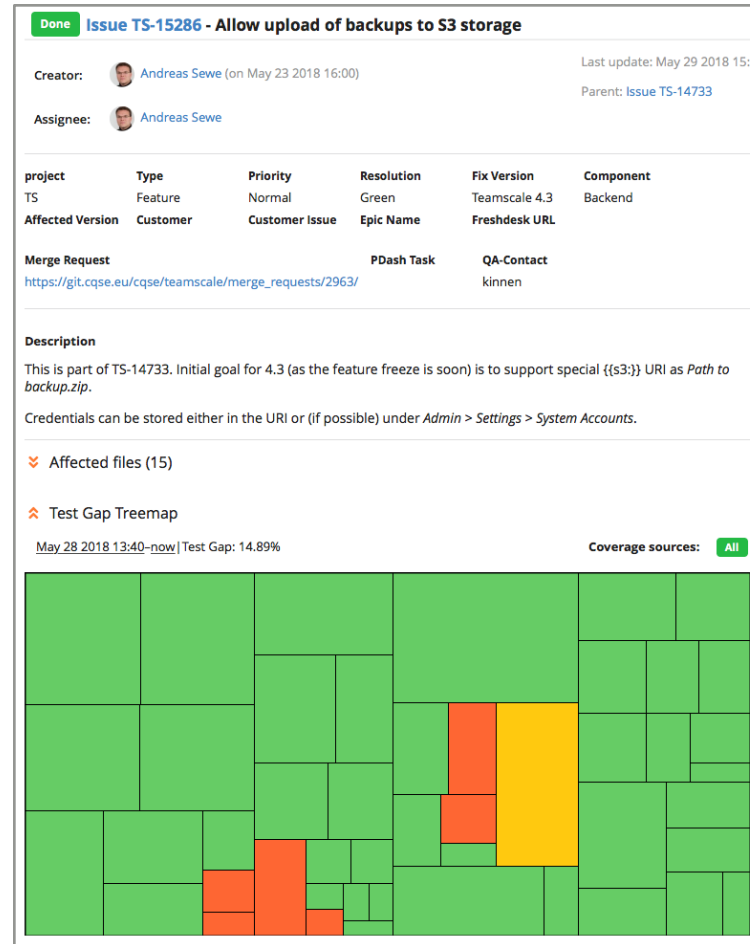
WhiteBoard/WBProcessManager.m

Method log_demo_details

State Method was added after the baseline and was not tested after its latest modification

Message	Location	Finding Group
TODO More implementation needed	FrontBoard/FBProcessManager.m:136	Documentation > Task Tags > Comments/Task ...

Test Gaps je Ticket



- Test-Gap-Analyse schafft Transparenz über Test Gaps
- Test-Gap-Analyse unterstützt beim effizienten Aufbau von Test-Sicherheitsnetz

Code Review-Kommentar in GitLab

The screenshot shows a GitLab interface for a code review. At the top, there are tabs for Discussion (1), Commits (2), Pipelines (2), and Changes (1). A dropdown menu shows 'Show all activity' and a status '1/1 thread resolved'. The discussion is started by Andreas Sewe (@sewe) 4 months ago, resolved by Lars Heinemann. The code being reviewed is in the file `engine/org.conqat.engine.index/src/org/conqat/engine/index/repository/RepositoryContentUpdaterBase.java`. The code diff shows changes to the `addChangeEntry` method. A comment by Andreas Sewe suggests re-ordering the message signature to make it clearer, specifically mentioning the `result` parameter. Lars Heinemann responds by changing the line in version 2 of the diff to use `computeChangeEntry` and `ifPresent`.

Discussion 1 Commits 2 Pipelines 2 Changes 1 Show all activity 1/1 thread resolved

Andreas Sewe @sewe started a thread on an old version of the diff 4 months ago
Resolved by Lars Heinemann 4 months ago [Toggle thread](#)

`engine/org.conqat.engine.index/src/org/conqat/engine/index/repository/RepositoryContentUpdaterBase.java`

```
293 293 List<BasicTokenElementInfo> tokenElements = tokenElementIndex.getValues(paths);
294 294 for (int i = 0; i < paths.size(); i++) {
295 295     String path = paths.get(i);
296 - RepositoryChangeInfo changeInfo = changeSet.getChangeInfo(path);
297 - ERepositoryChangeType changeType = patchChangeType(changeInfo,
298 - tokenElements.get(i));
299 - if (changeType != null) {
300 -     result.add(new RepositoryChangeEntry(path,
301 - requiredNode.getRevision().getRevision(), changeType,
302 - commit, changeInfo.getOriginPath(),
303 - changeInfo.getOriginCommit()));
304 - }
305 + BasicTokenElementInfo element = tokenElements.get(i);
306 + addChangeEntry(requiredNode, result, changeSet, path, element);
```

Andreas Sewe @sewe · 4 months ago Maintainer
I would re-order the message signature. ATM, it's hard to tell which parameter the change entry is added to. If you have `result` first, as in `result.add(...)` things become clearer.

Alternatively, let the extracted method return `Optional<RepositoryChangeEntry>` and do the `add` here.

```
computeChangeEntry(requiredNode, changeSet, path, element).ifPresent(result::add);
```

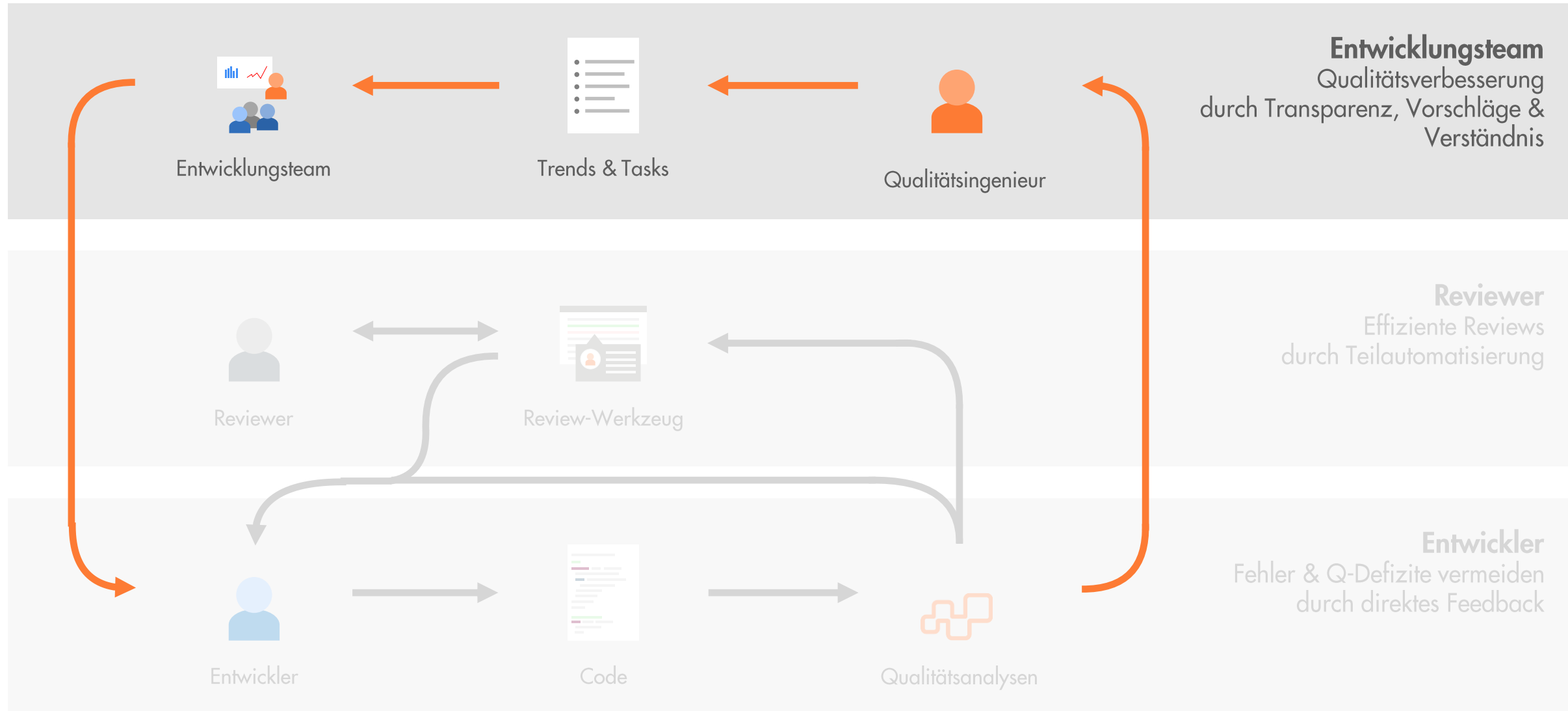
Lars Heinemann @heinemann changed this line in version 2 of the diff 4 months ago

→ Hohe Eingangsqualität erlaubt effiziente, menschliche Code Reviews



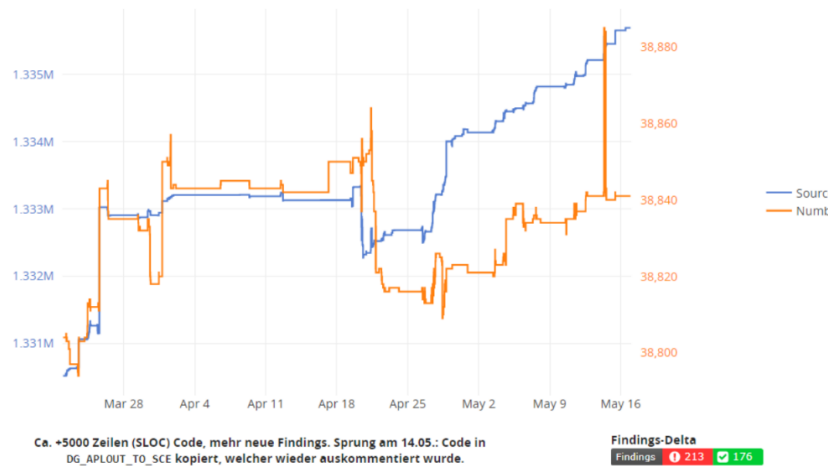
Feedbackschleife 3: Q-Retrospektiven unterstützen Entwicklungsteams bei Qualitätssicherung & -verbesserung

Feedbackschleife 3: Q-Retrospektiven unterstützen Entwicklungsteams bei Qualitätssicherung & -verbesserung



Analyse & Diskussion von Qualitätstrends

Trend 22.März - 17. Mai



System Quality Overview

Quality Indicator (QI)	Trend	Quality Indicator (QI)	Trend
Security		Redundanz	
Rote Security-Findings	→	Clone Coverage	↘
Security-Findings je 1.000 Codezeilen	↗	Codestruktur	
Codeanomalien		Struktur: Prozedurlänge	→
Korrektheit	↗	Struktur: Schachtelungstiefe	→
Codeanomalien je 1000 SLOC	→		

→ Trendanalyse schafft Transparenz für Team und Management

Konkrete Verbesserungsvorschläge

Task 34 - Redundanz zwischen
■■■■■UI5_TRANSLATION_TOOL und
■■■■■TRANSLATION_TOOL

created by ■■■■■ Sep 16 2021 15:22, last updated Sep 17 2021 17:19

Assignee ■■■■■

Status Open

Resolution None

Description

Der Code im neuen Paket ■■■■■UI5_TRANSLATION_TOOL ist oftmals redundant zum Code in ■■■■■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 ■■■■■TRANSLATION_TOOL/CLAS■■■■■EL_TRANSLATION.abap:979

Clone with 2 instances of length 47 in ■■■■■UI5_TRANSLATION_TOOL/CLAS■■■■■EL_UI5_TRANSLATION.abap:1208

Task 16 - Fehlende Best Practice für
Berechtigungsprüfungen von Reports und
Transaktionen

created by ■■■■■ Apr 08 2021 09:14, last updated Sep 19 2021 21:49

Assignee

Status Closed

Resolution Fixed

Description

Aufgrund des Feedbacks des ■■■■■Teams wurden die Teamscale-Prüfungen für die Existenz von Berechtigungsprüfungen von Reports und Transaktionen (genauer: von Aufrufen einer Transaktion im Code mittels CALL TRANSACTION) deaktiviert.

Deshalb fehlt aktuell eine Best Practice für die Berechtigungsprüfung für Reports und Transaktionen.

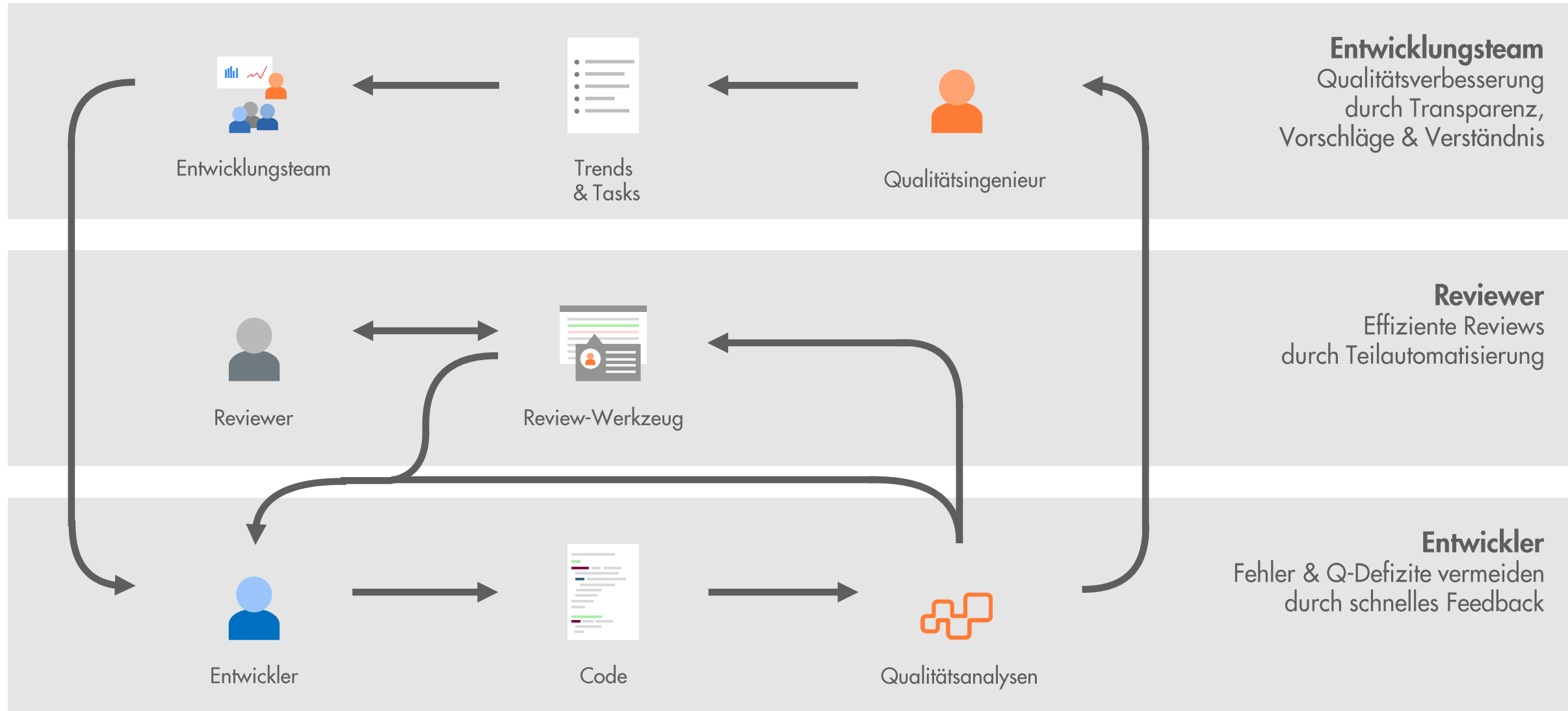
Wir empfehlen, eine solche Best Practice zu definieren und mit Teamscale nachzuhalten, ob diese eingehalten wird.

Tags 1. Retro Security

Edit Task

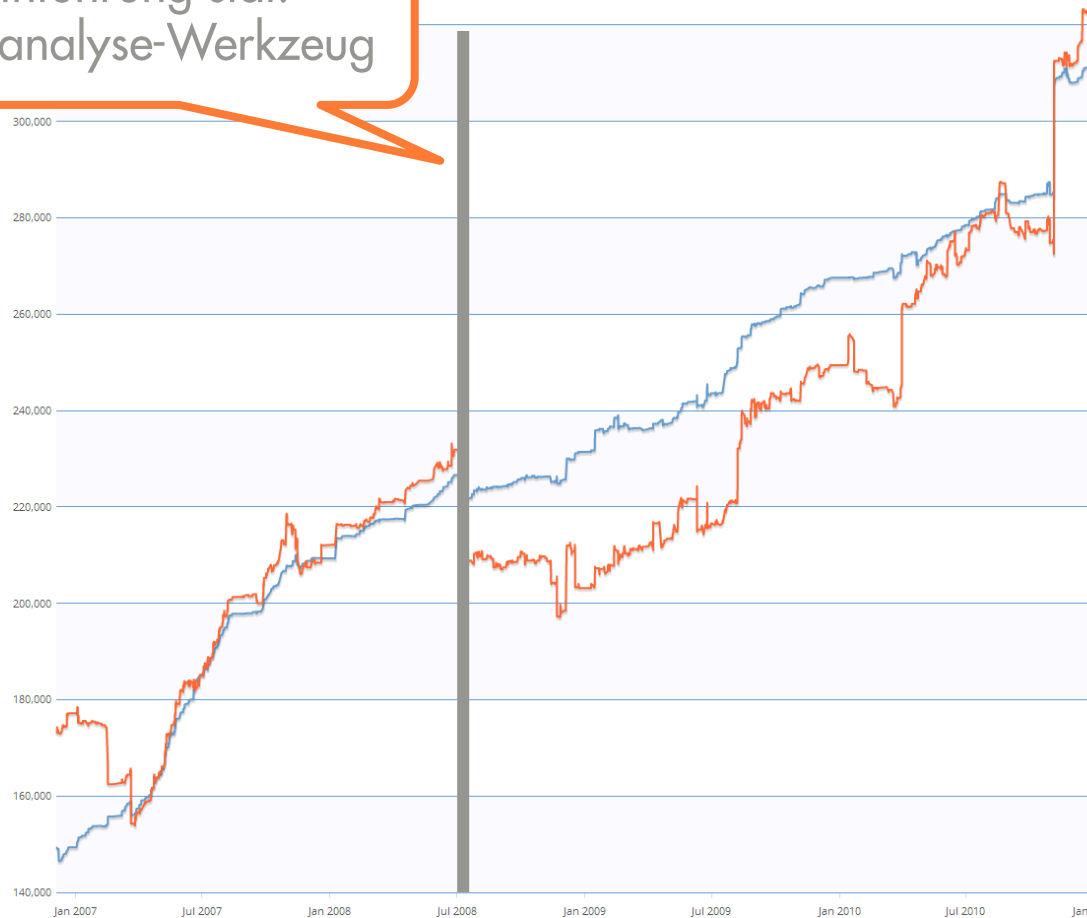
- Verbesserungsvorschläge ermöglichen Einplanung und konkrete Verbesserung
- Diskussionen schaffen gemeinsames Qualitätsverständnis und Best Practices

Continuous Quality Control-Feedbackschleifen



Was brought Continuous Quality Control?

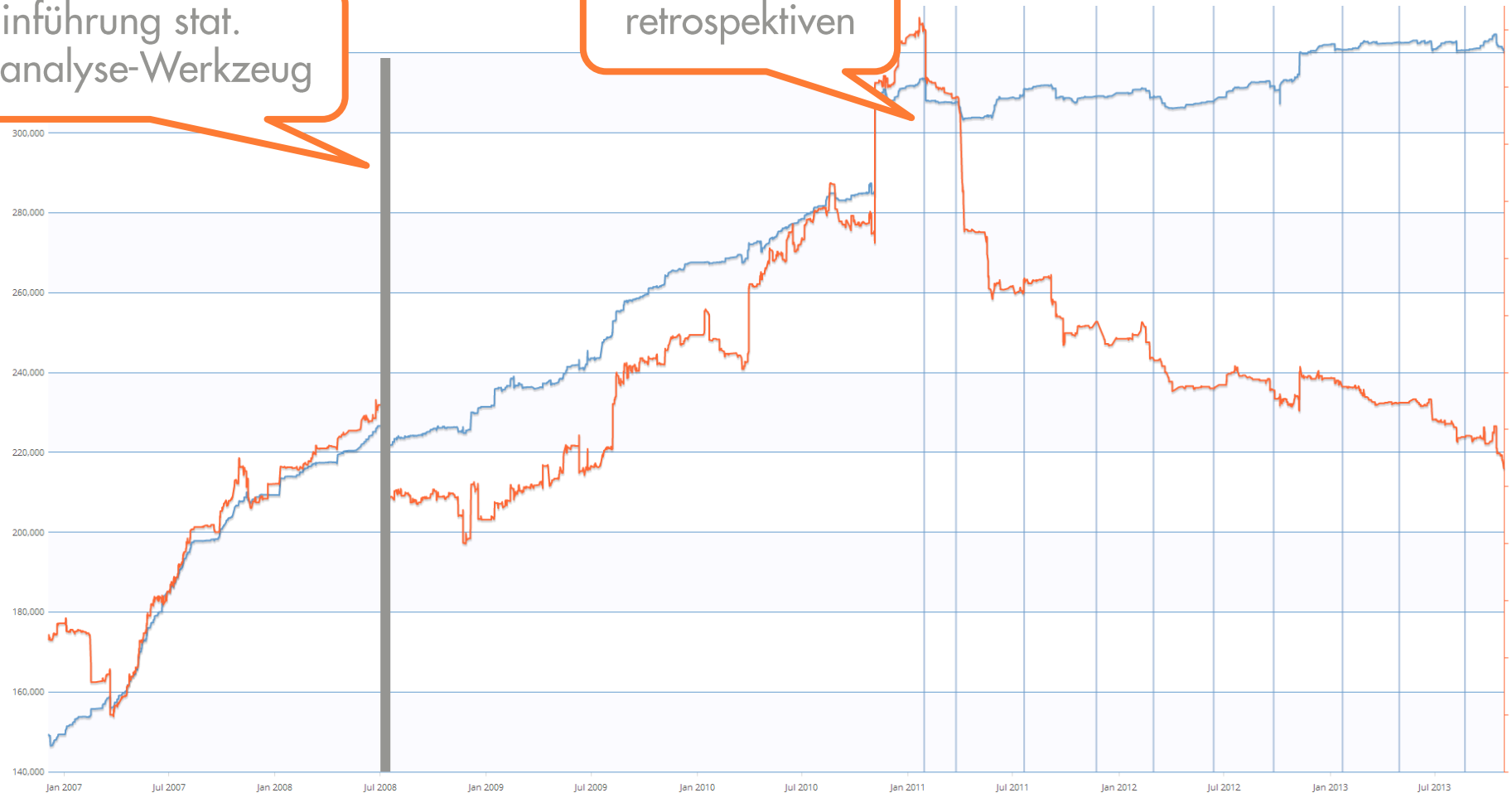
Einführung stat.
Codeanalyse-Werkzeug



→ Werkzeug hat i.d.R. nur einen kurzfristigen Effekt

Einführung stat.
Codeanalyse-Werkzeug

Start Qualitäts-
retrospektiven



→ Kombination aus Werkzeug und Retrospektiven
hat i.d.R. einen nachhaltigen Effekt

Nutzen der Klonanalyse für Munich Re

☒ All	7178
☒ Architecture	290
☒ ▶ Architecture Conformance	290
☒ Code Duplication	1608
☒ ▶ Code Clones	1608
☒ Comprehensibility	2180
☒ ▶ Bad Practice	1313
☒ ▶ Design Flaws	74
☒ ▶ Explicit Findings Management	21
☒ ▶ Formatting	225
☒ ▶ Modernization	3
☒ ▶ Naming	327
☒ ▶ Test Smells	76
☒ ▶ Unused Code	141
☒ Correctness	550
☒ ▶ API Misuse	42
☒ ▶ Concurrency	8
☒ ▶ Discouraged APIs	307
☒ ▶ Error-prone Practices	84
☒ ▶ Possible Bugs	108
☒ ▶ Resource Leaks	1
☒ Disabled Tests	19
☒ ▶ Disabled Tests	19
☒ Documentation	818
☒ ▶ Comment Completeness	762
☒ ▶ Malformed Comments	25
☒ ▶ Task Tags	31

$$500 \frac{PT}{Jahr}^*$$

Munich Re spart durch Einsatz von Clone Detection
jährlich ca. 500 PT Aufwand für Fehlerbehebung

$$\text{Ersparnis Aufwand} = 6\%^*$$

Munich Re spart durch Einsatz von Clone Detection
jährlich 6% Aufwand durch vermiedene Redundanz ein.

→ Continuous Quality Control hat großen Nutzen

* Übertragbarkeit muss geprüft werden; für ausführliche, konservative Herleitung siehe [CQSE-Workshop „Copy, Paste und dann ...?“](#)

Wie führt man Continuous Quality Control ein?

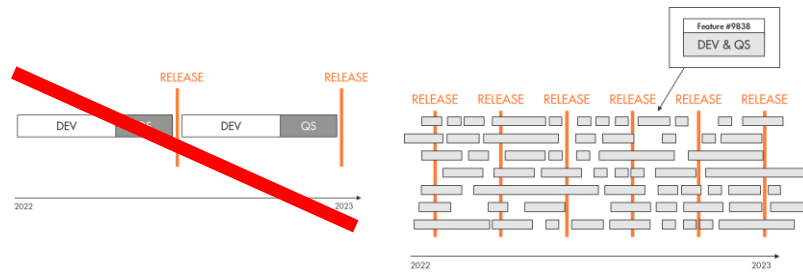


→ Technische, organisatorische und soziale Faktoren berücksichtigen

Onboarding-Prozess



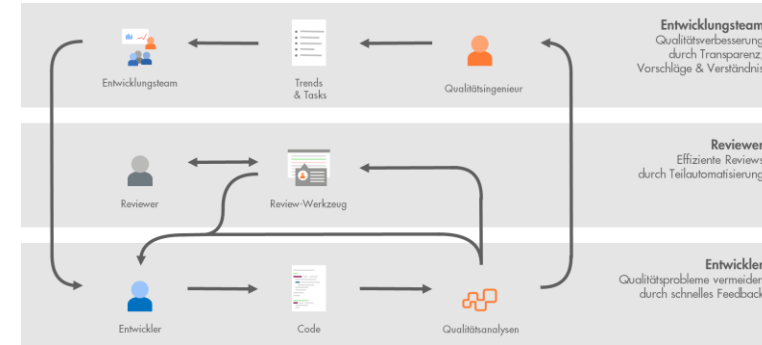
Zusammenfassung



→ Releasefähigkeit erfordert jederzeitige Releasequalität und kontinuierliche QS



→ CQC hat großen Nutzen (auch bei gewachsenen Anwendungen)



→ CQC ermöglicht dies durch Feedbackschleifen für Entwickler, Reviewer und Entwicklungsteam



→ Einführung von CQC erfordert planvolles Vorgehen

Continuous Quality Control

Qualität trotz immer kürzerer Releasezyklen

Dienstag, 11. November 2025

10:30 bis 12:00 Uhr

online und kostenlos



Jetzt anmelden: tmscl.me/cqc2511-tjfs



Dr. Tobias Röhm



Tobias Wiese

Folien & Kontakt - Ich freue mich auf den Austausch 😊



Vortragsfolien:
tmscl.me/jfs_2025_releasezyklen



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



CQSE-Stand