

# Continuous Quality Control

## Qualität trotz immer kürzerer Releasezyklen

---

Software Architecture Alliance 2024

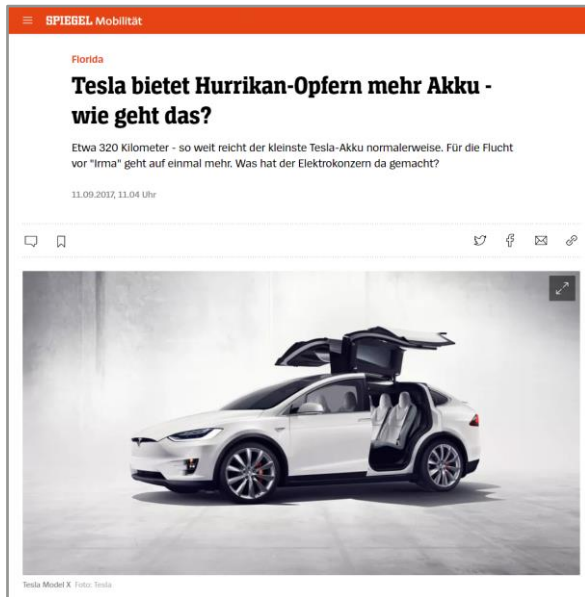
**CQSE**

Dr. Tobias Röhm (@langelot)

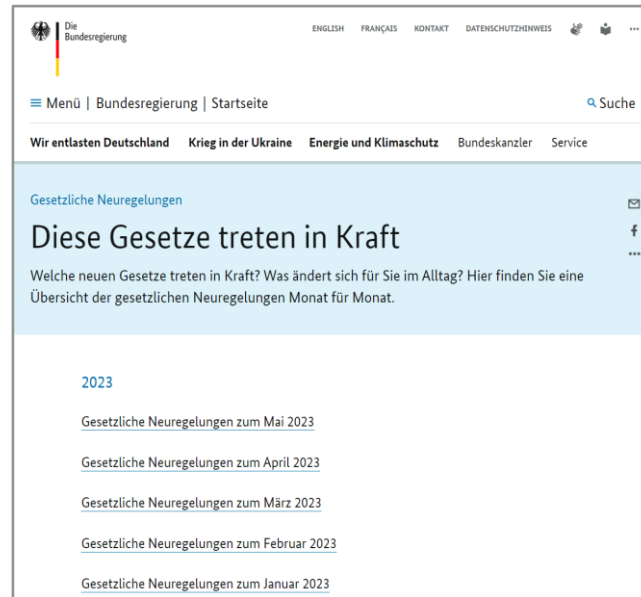
# Dr. Tobias Röhm



# Kontext



Ereignisbasierte  
Over-the-Air-Updates



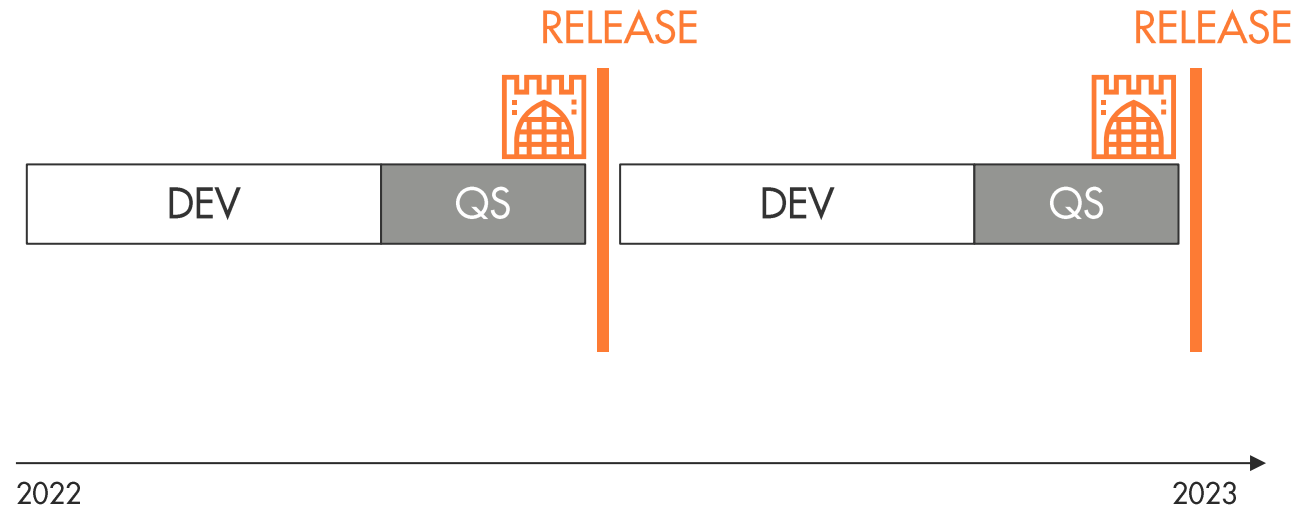
Viele, teilweise rückwirkende  
Gesetzesänderungen



Zeitkritische Security-Patches

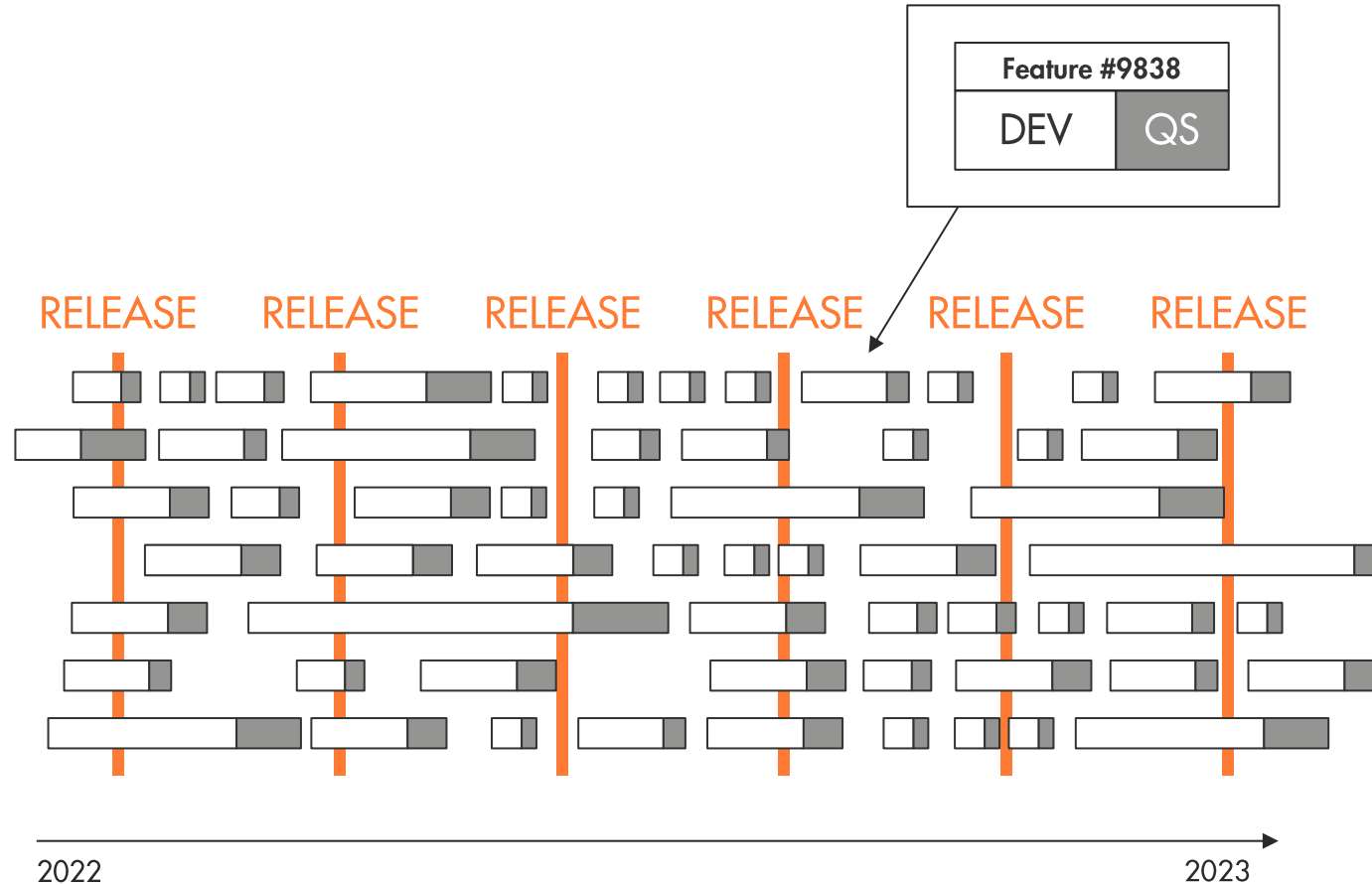
→ Software-Updates müssen jederzeit möglich sein  
→ Software muss jederzeit ein releasefähiges Qualitätsniveau aufweisen

# Problem: Traditionelle QS-Prozesse



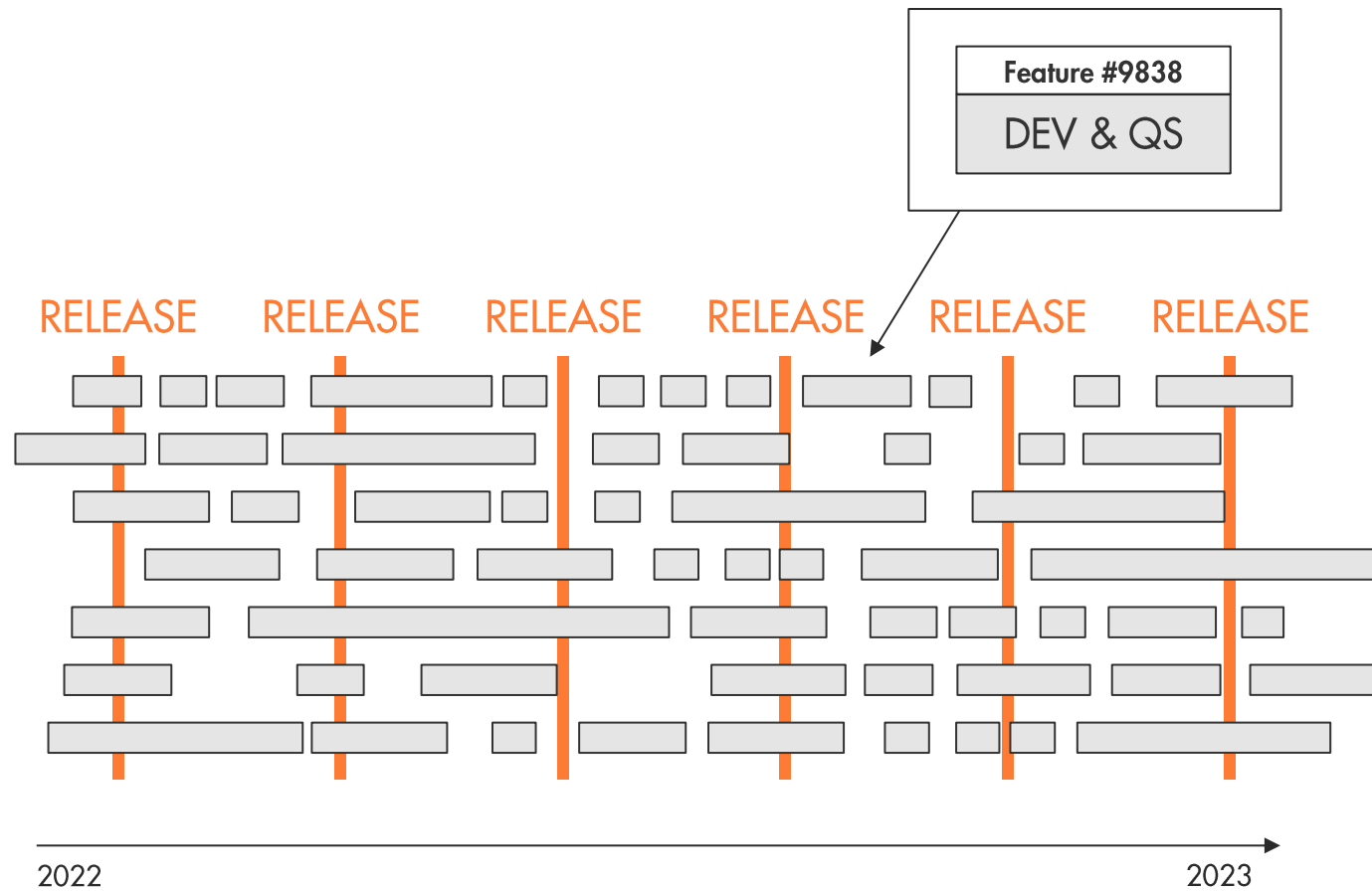
- Traditionelle QS-Prozesse sind (zu) langsam
- Software ist NICHT jederzeit auf einem releasefähigem Qualitätsniveau

# Lösung: Continuous Quality Control (CQC)



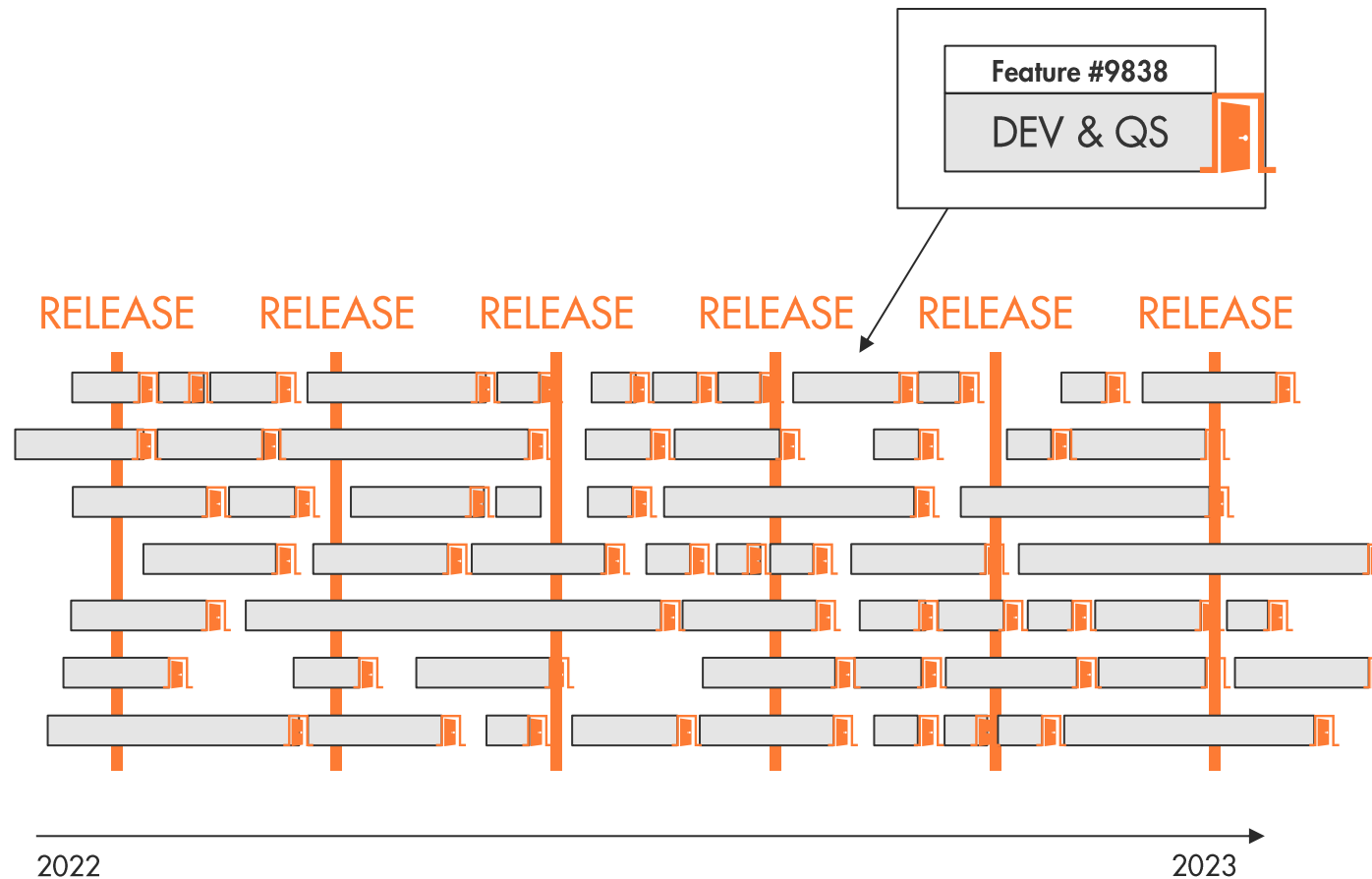
→ Qualitätssicherung auf Ticketebene um zu parallelisieren und zu beschleunigen

# Lösung: Continuous Quality Control (CQC) II



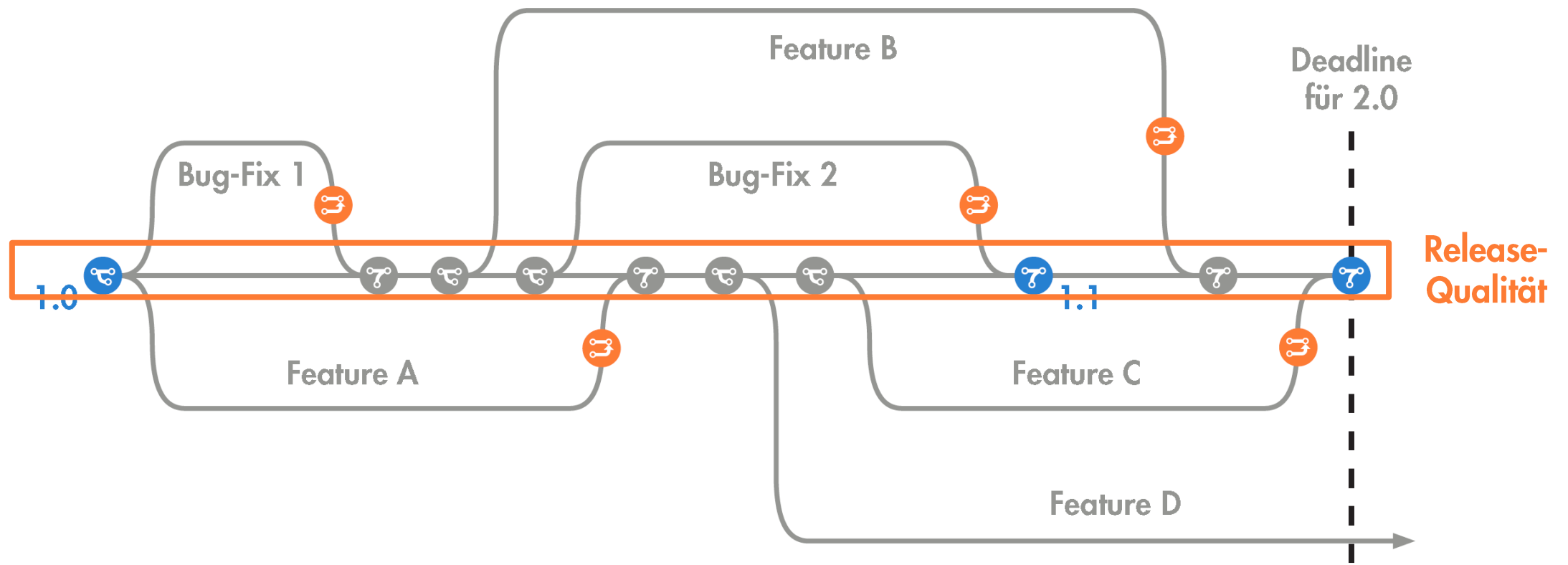
→ Verschmelzung von Entwicklung und QS um zu beschleunigen

# Lösung: Continuous Quality Control (CQC) III



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

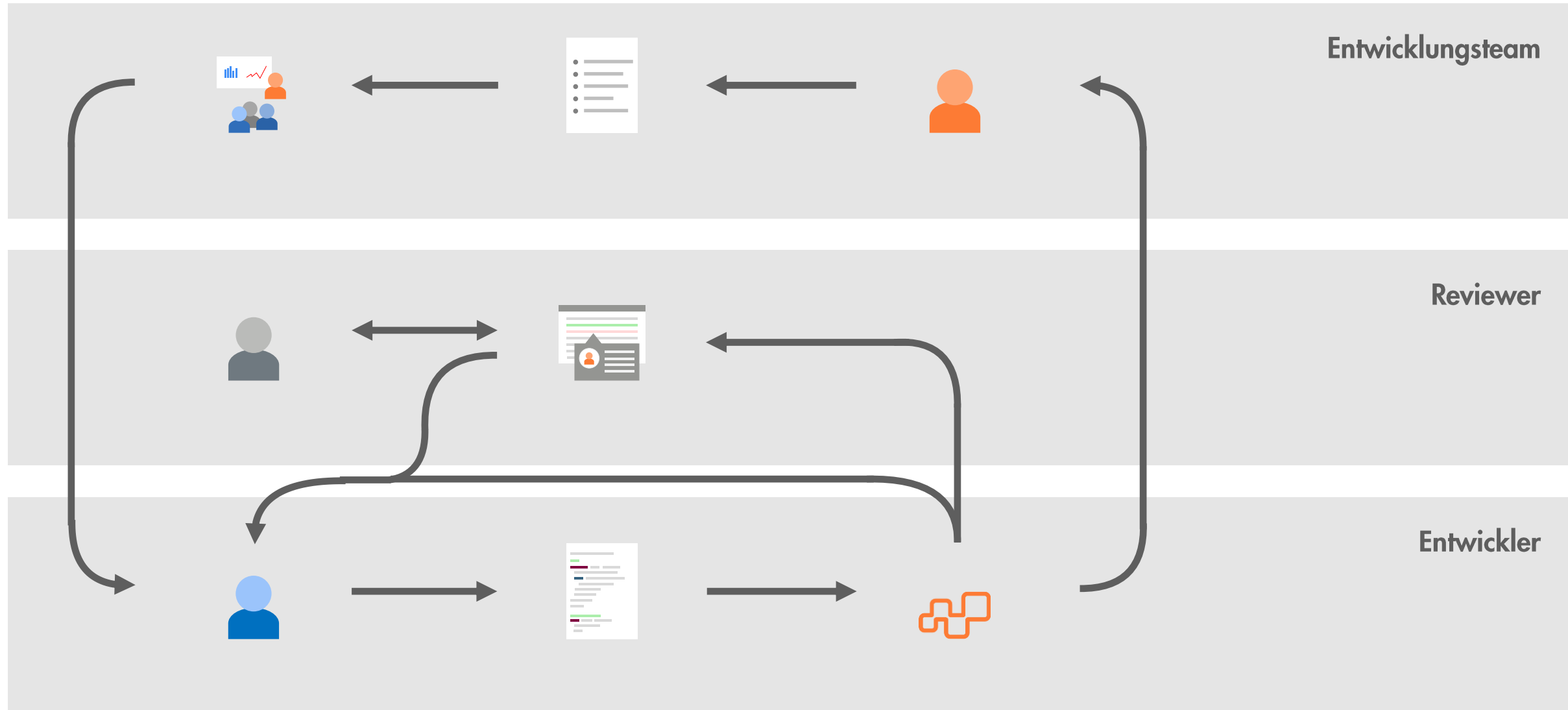
# Implementierung durch Feature Branches in Git



→ Merge Request als »Qualitätstürchen«



# Agenda: Continuous Quality Control-Feedbackschleifen



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

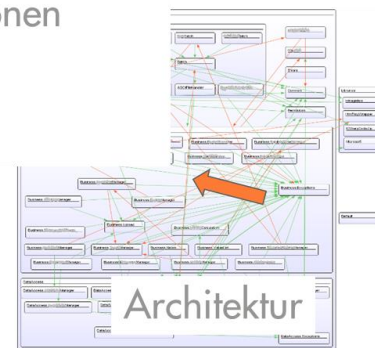
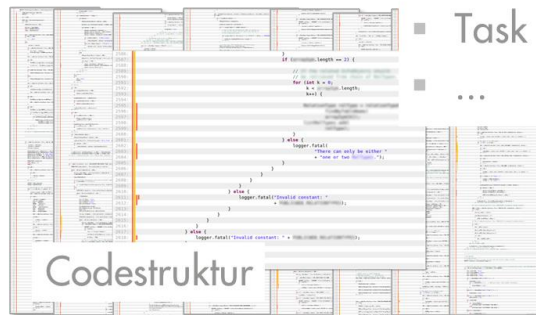
# Feedbackschleife 1: Statische Codeanalyse hilft Entwicklern, Fehler & Q-Defizite zu vermeiden



# Übersicht Statische Codeanalyse

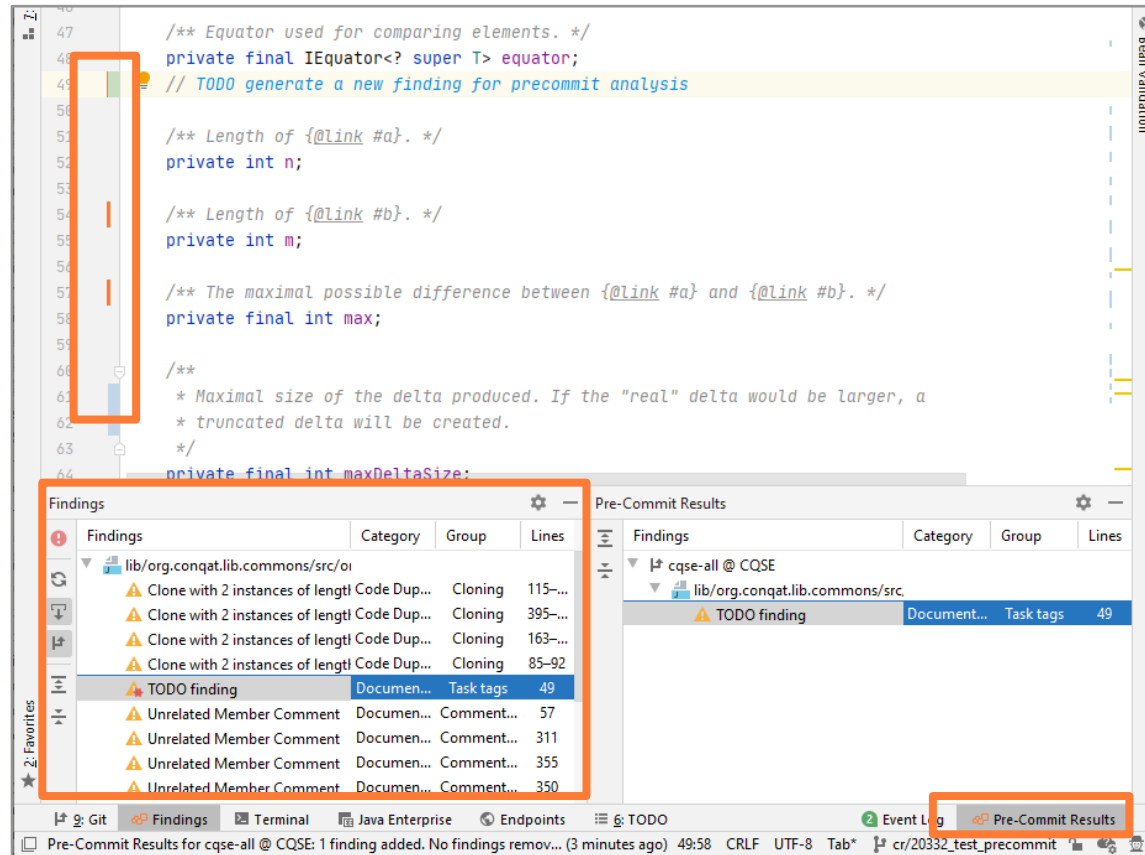


- Kommenturvollständigkeit
- Richtlinienverletzungen
- Auskommentierter Code
- Namenskonventionen
- Task Tags
- ...



→ 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...** Yesterday 17:34  
by Chris Dumez as revision ffa7b160 in main (github)  
Files: 1 changed  
Findings   

 **CachedResourceLoader::allCachedSVGImages() reparses resource URLs unnecessari...** Yesterday 17:26  
by Ch...  
Files:  
Findi...

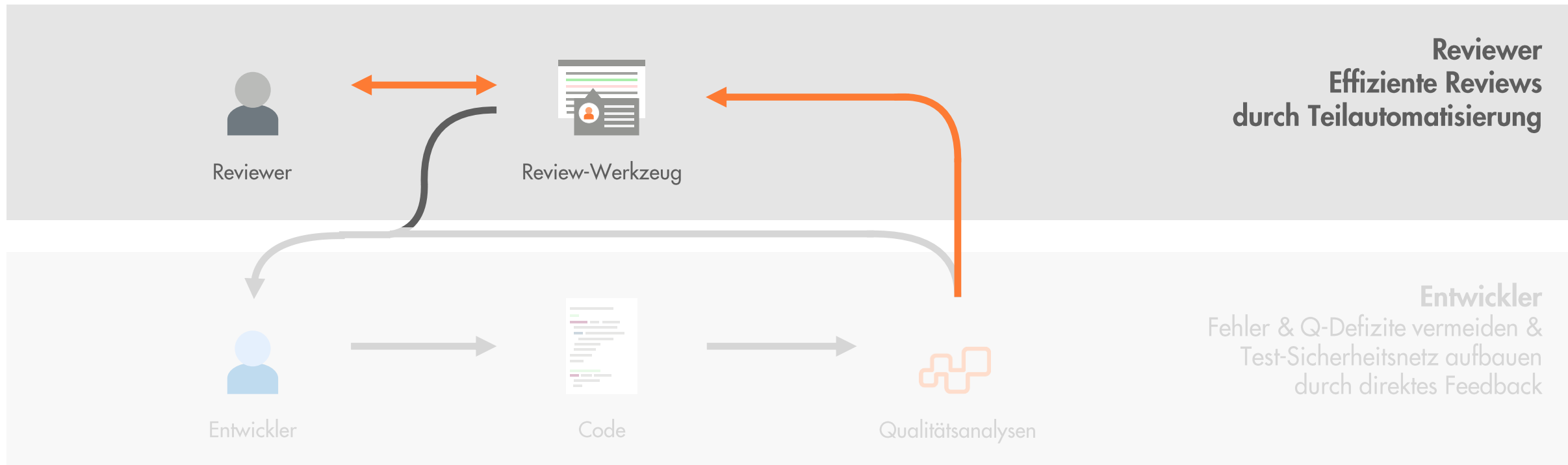
Unterscheidung zwischen »neuen Findings« und »Findings in geändertem Code« erlaubt schrittweise Q-Verbesserung von gewachsenen Codebasen mit der Pfadfinderregel.

 **[WinCairo][WK2] animations/par... is timing out https://bug...** Yesterday 13:20  
by Fujii Hironori as revision 2d29f... in main (github)  
Files: 2 changed  
Findings   

 **[git-webkit] Link to GitHub wiki https://bugs.webkit.org/show\_bug.cgi?id=2370...** Yesterday 13:11

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

The screenshot shows a pull request titled "Modify the code #7" by "asteckermeier". It includes a navigation bar with "Code", "Pull requests 2", "Projects", "Security", and "Insights". Below the title, it says "Open" and "asteckermeier wants to merge 2 commits into Sydro from merge\_request\_tga\_demo\_5". There are statistics for "Conversation 0", "Commits 2", "Checks 1", and "Files changed 1". A warning icon indicates a "Fix method not being called" issue. The "Teamscale (SWE)" section is expanded, showing "teamscale-findings" with a "Resolve" link. Under "Added Findings", it states "This pull request would introduce 1 new finding". The "DETAILS" section shows a Teamscale logo and a finding summary: "WhiteBoard/WBProcessManager.m#L115: TODO (AS) We should switch this to always log." A red box highlights the Teamscale logo and the finding summary. The "ANNOTATIONS" section shows a warning icon and the text: "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.", and a link to the finding details. A "View more details on Teamscale (SWE)" link is at the bottom.

The screenshot shows a pull request titled "PR #4" by "nkhater". It includes a navigation bar with "Conversation 0", "Commits 22", "Checks 1", and "Files changed 3". The "Changes from all commits" section shows a diff for "jabref/JabRefGUI.java" and "jabref/logic/bibtex/BibEntry/Writer.java". The diff shows changes to the Logger class and the catch clause of an exception handler. Two Teamscale findings are highlighted with red boxes: 1. A warning on line 38 in jabref/JabRefGUI.java: "Interface comment missing". 2. A failure on line 168 in jabref/logic/bibtex/BibEntry/Writer.java: "Catch clause catches generic exception 'Exception'".



Reviewer






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

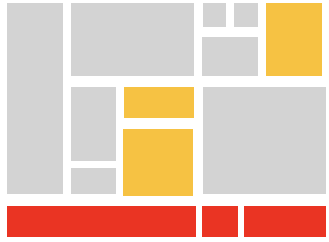


Funktioniert Ticket-Implementierung?  
Ging aus Versehen etwas kaputt?

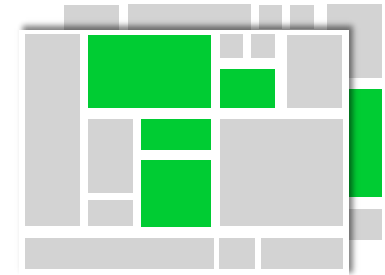
Wurden alle Codeänderungen getestet?

Anwendung

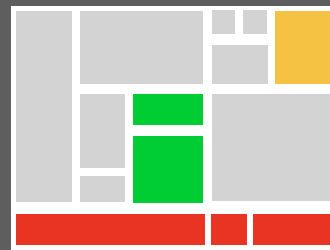
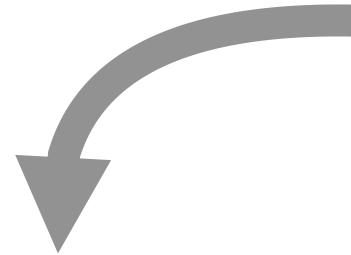
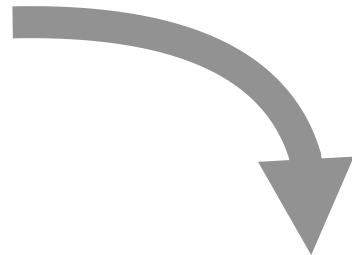
# Test-Gap-Analyse



Codeänderungen



Testcoverage



Test Gaps =  
Ungetestete Codeänderungen

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

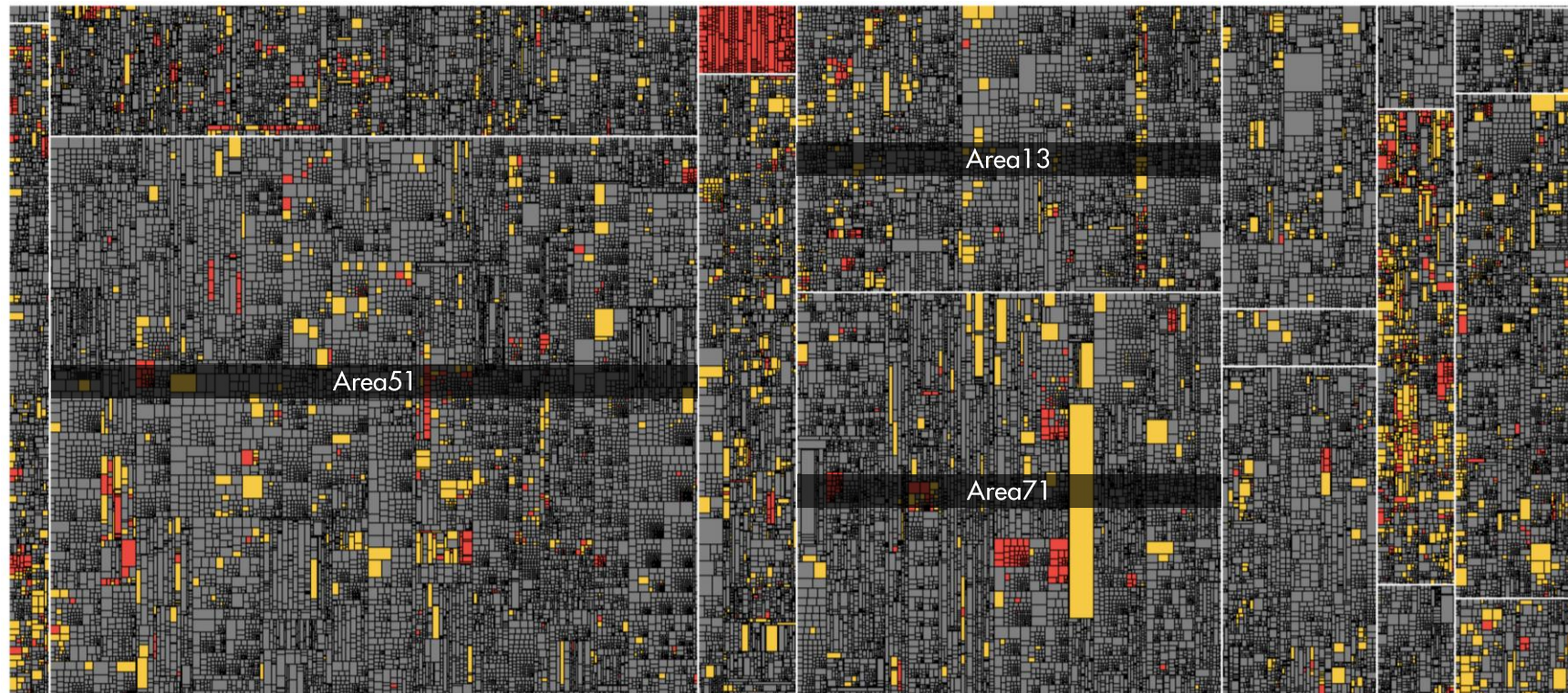
Did we test our changes? Assessing alignment between tests and development in practice. (Eder et al. 2013)



# Codeänderungen

Churn

Jul 10 2022 - Aug 10 2022



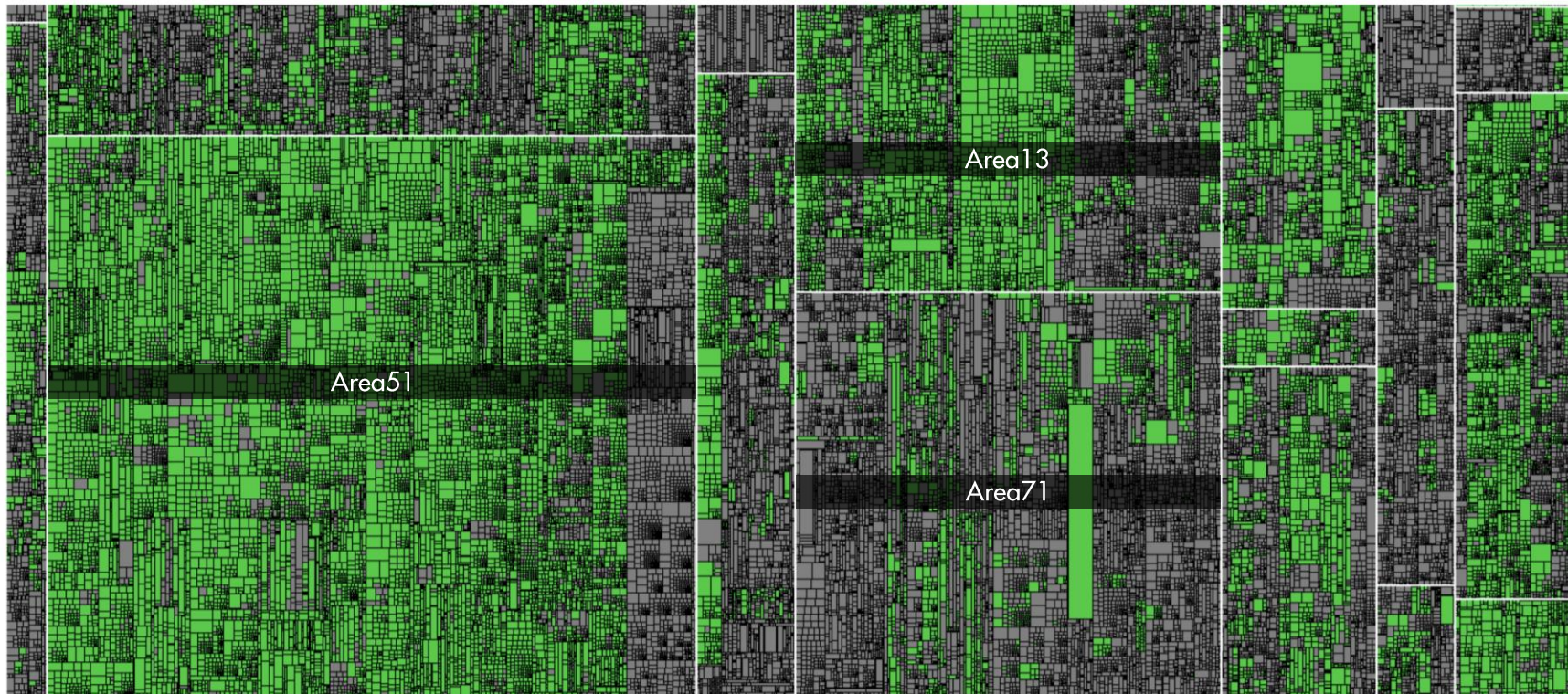
Methoden wurden...

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



# Testcoverage

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



Methoden wurden...

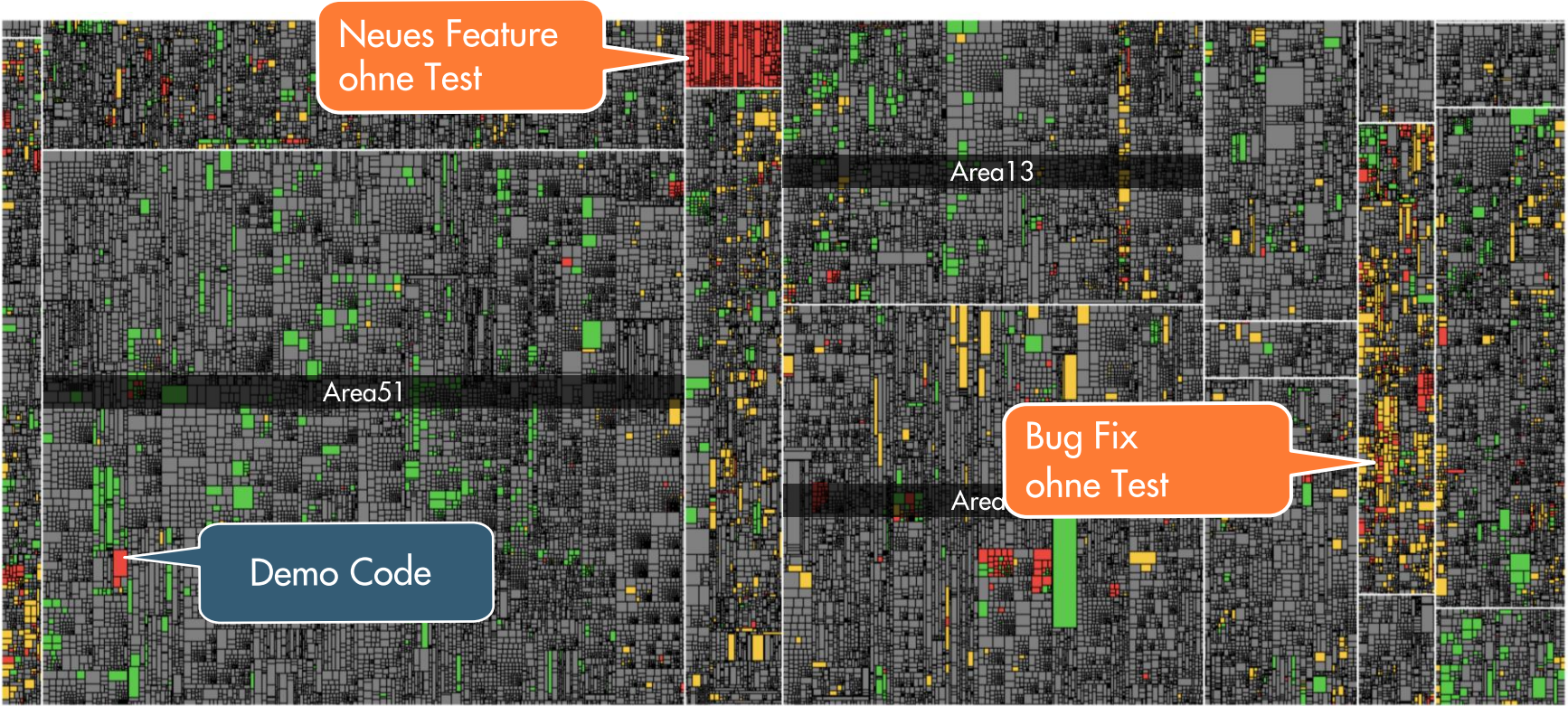
■ ... nicht getestet

■ ... 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 Sydro 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 Sydro OPEN NEGATIVE

This change contains 2 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

Findings 1 2 2

New findings

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

# Test Gaps je Ticket

**Done** Issue TS-15286 - Allow upload of backups to S3 storage

Creator: Andreas Sewe (on May 23 2018 16:00) Last update: May 29 2018 15:54  
Assignee: Andreas Sewe Parent: Issue TS-14733

project	Type	Priority	Resolution	Fix Version	Component
TS	Feature	Normal	Green	Teamscale 4.3	Backend
Affected Version	Customer	Customer issue	Epic Name	Freshdesk URL	

Merge Request [https://git.cqse.eu/cqse/teamscale/merge\\_requests/2963/](https://git.cqse.eu/cqse/teamscale/merge_requests/2963/) PDash Task QA-Contact  
kinnen

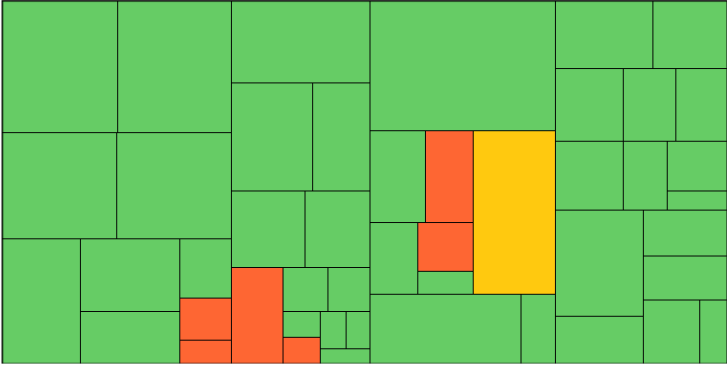
**Description**

This is part of TS-14733. Initial goal for 4.3 (as the feature freeze is soon) is to support special {{s3}} URI as *Path to backup.zip*.  
Credentials can be stored either in the URI or (if possible) under *Admin > Settings > System Accounts*.

⌵ Affected files (15)

⌵ Test Gap Treemap

May 28 2018 13:40–now | Test Gap: 14.89% Coverage sources: **All**



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

# Code Review-Kommentar in GitLab

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.

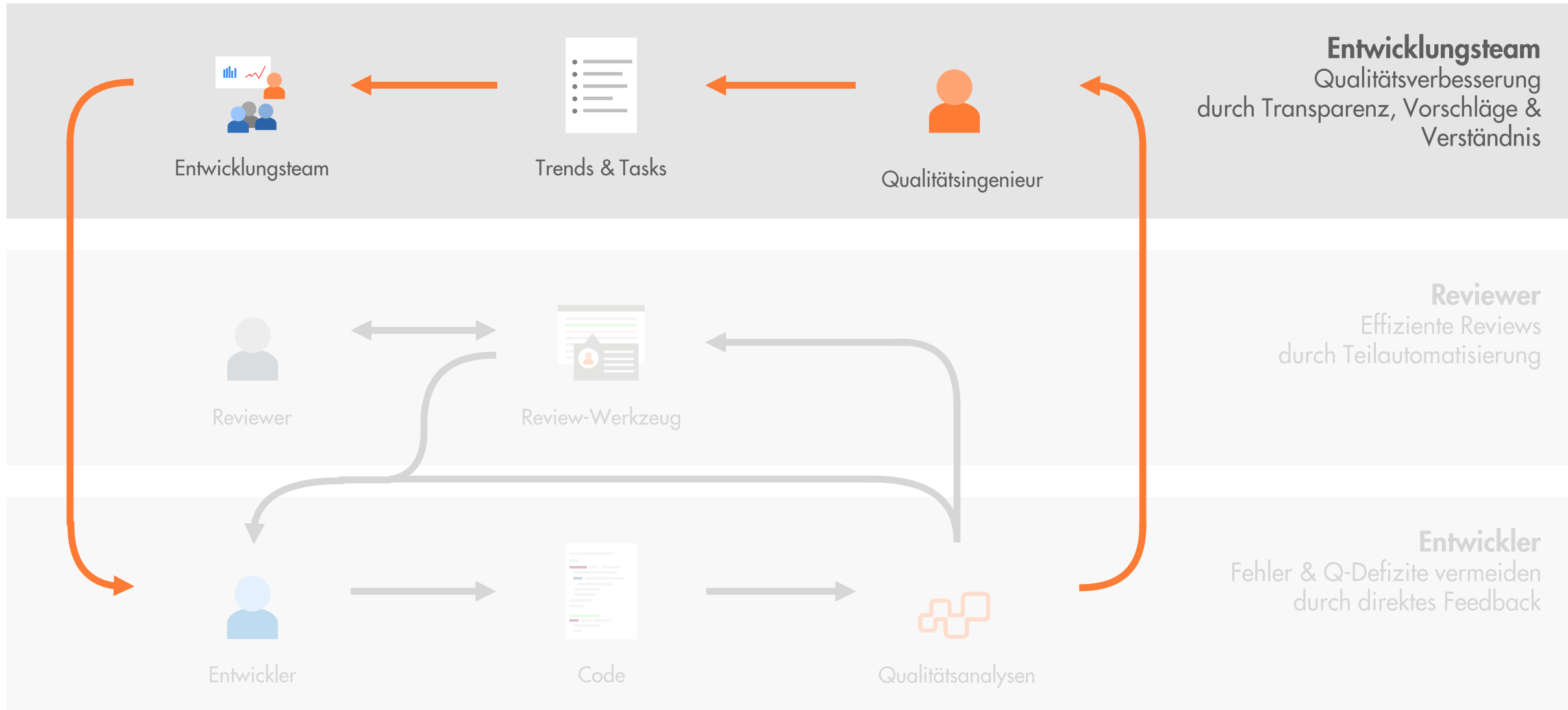
```
computeChangeEntry(requiredNode, changeSet, path, element).ifPresent(result::add);
```

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

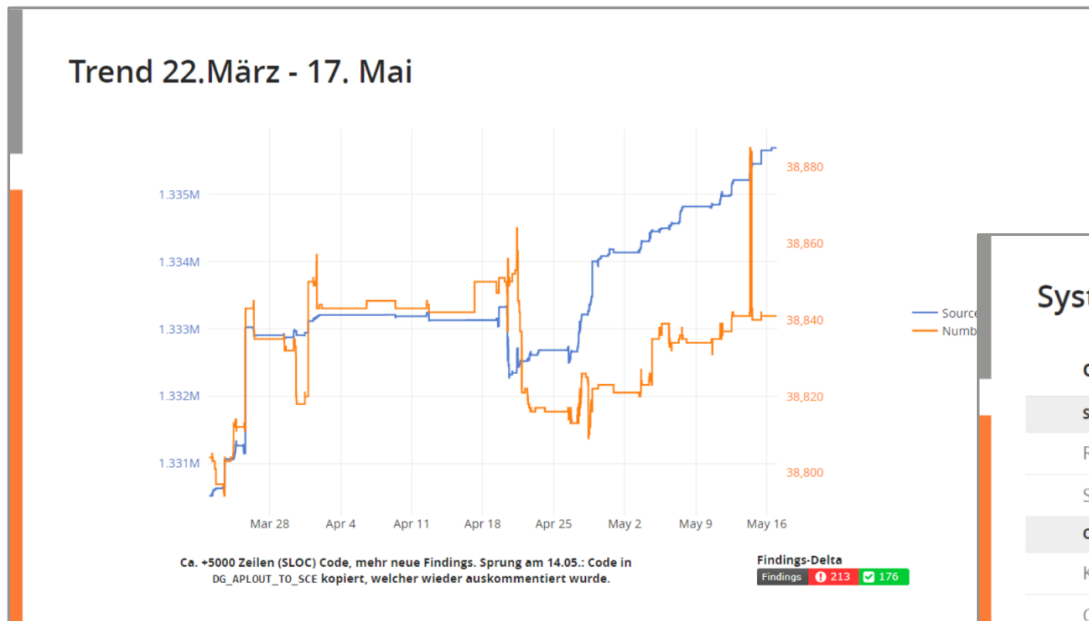
→ Hohe Eingangsqualität erlaubt effiziente, menschliche Code Reviews

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

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



# Analyse & Diskussion von Qualitätstrends



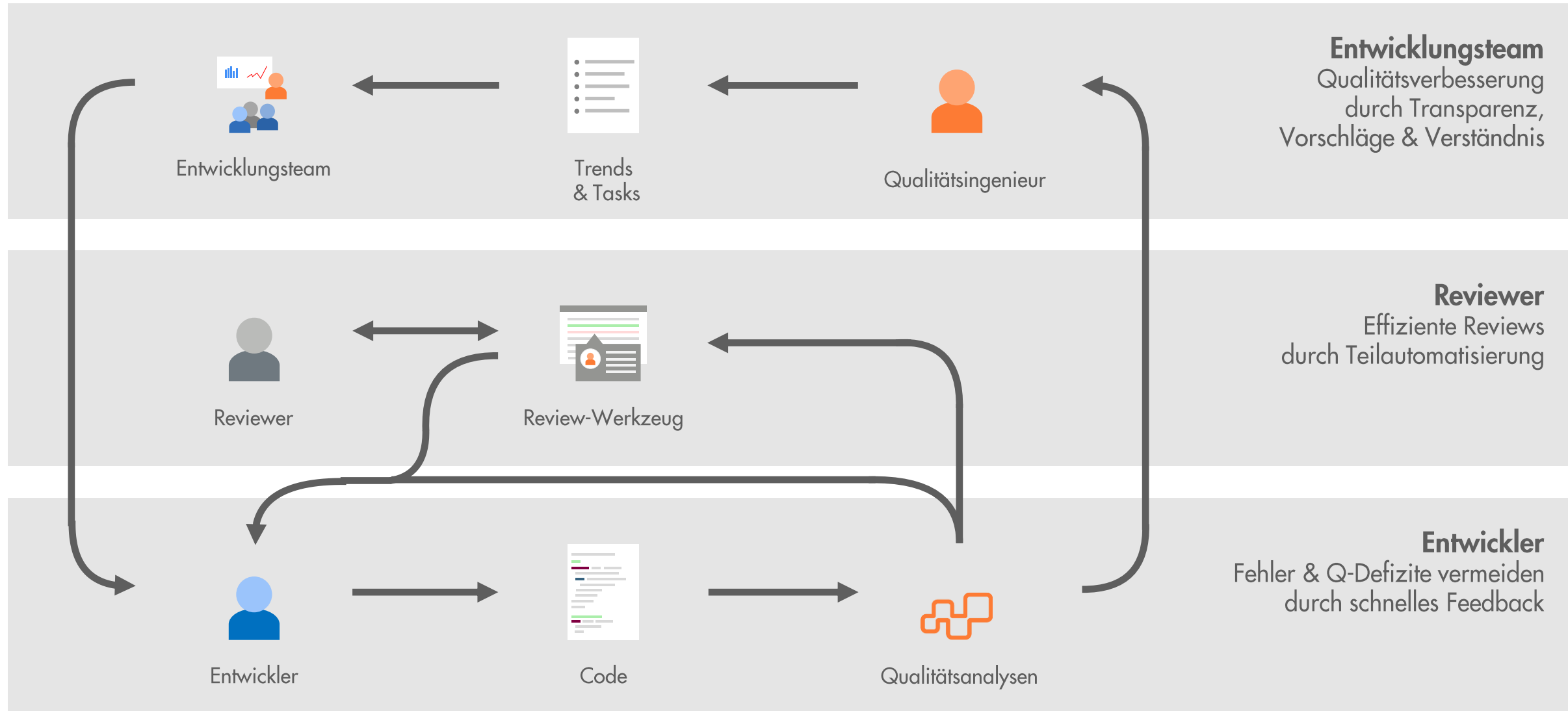
### System Quality Overview

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

→ Trendanalyse schafft Transparenz für Team und Management

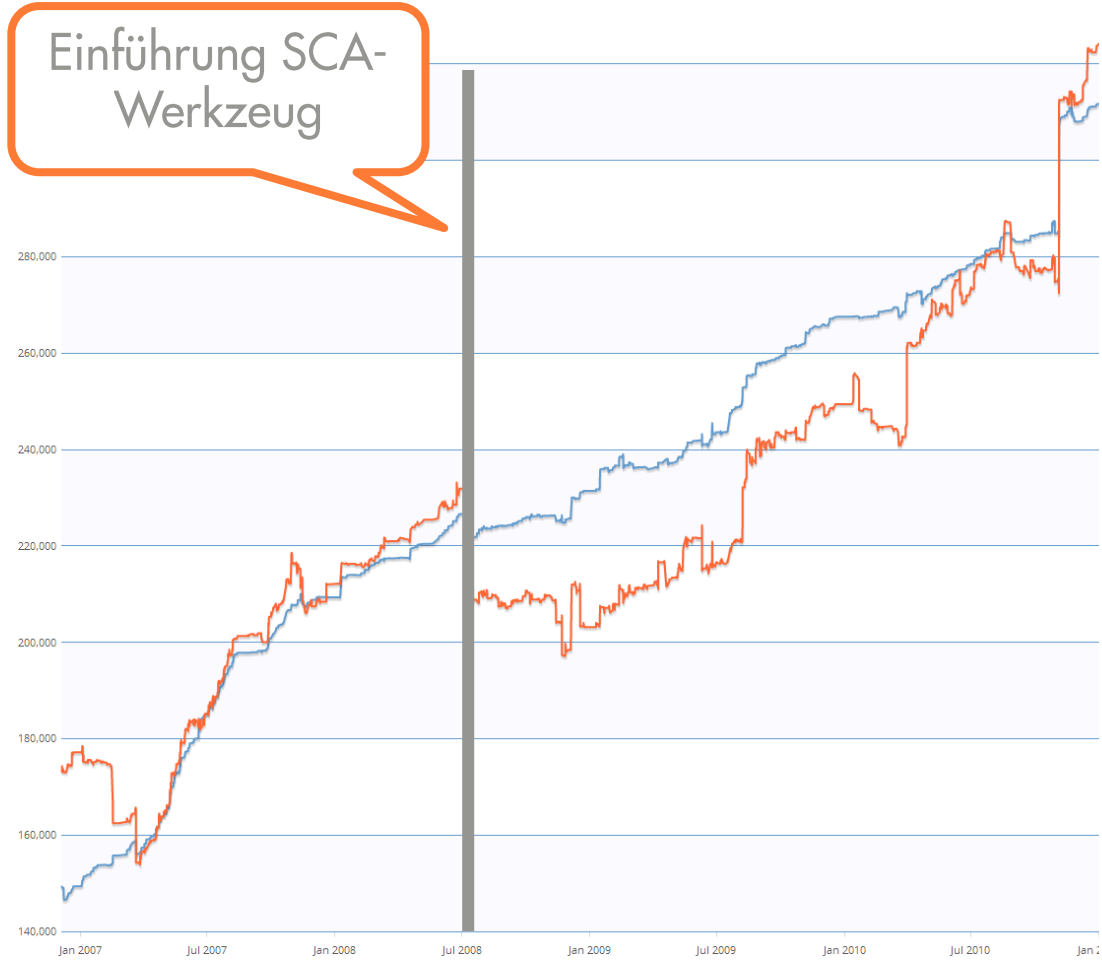


# Continuous Quality Control-Feedbackschleifen

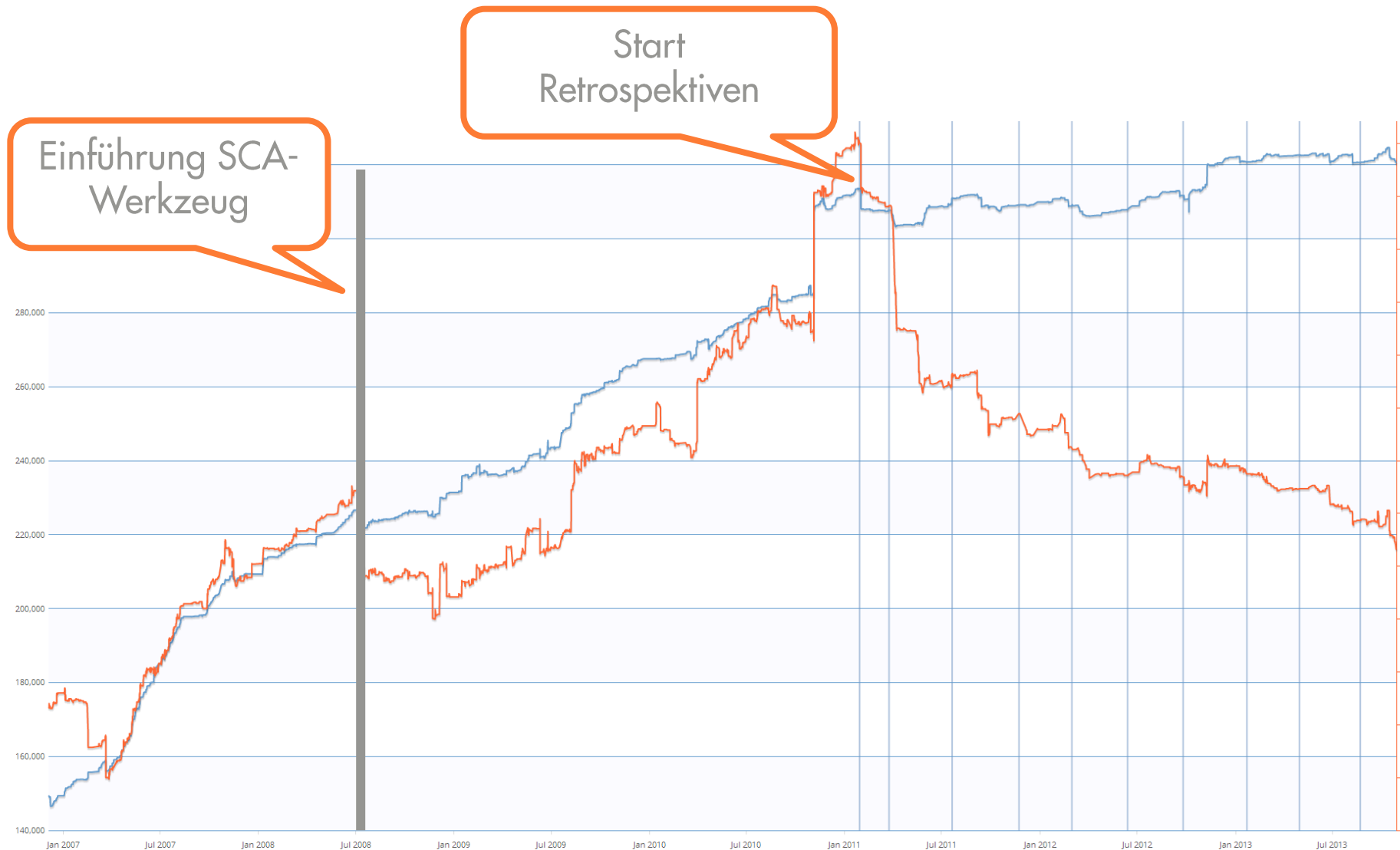




Was brought Continuous Quality Control?



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



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

# Nutzen der Klonanalyse für Munich Re

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

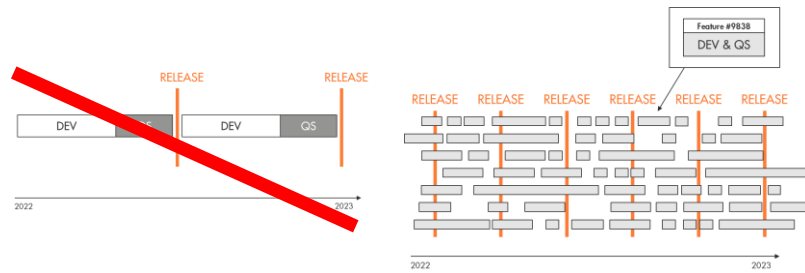


→ Technische, organisatorische und soziale Faktoren berücksichtigen

# Onboarding-Prozess



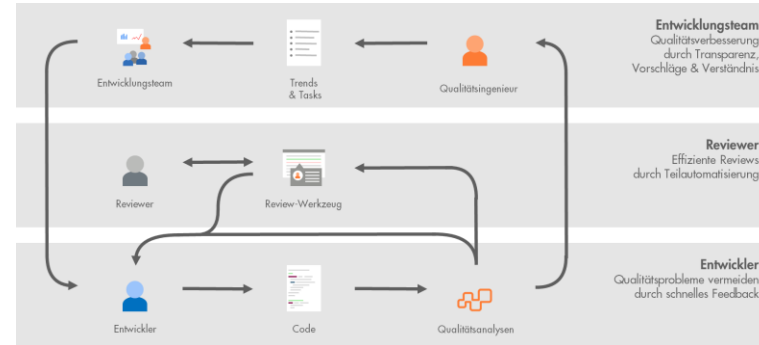
# Zusammenfassung



→ 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



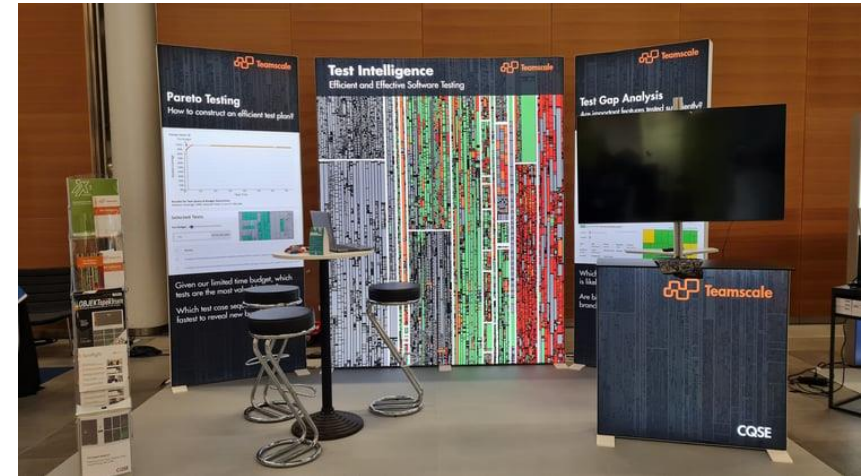
→ Einführung von CQC erfordert planvolles Vorgehen



# Möchten Sie mehr wissen?



Dr. Tobias Röhm  
[roehm@cqse.eu](mailto:roehm@cqse.eu)  
<https://tmscl.me/coffee-tobias-roehm>



CQSE-Stand auf SAA

# CQSE Workshop

## Continuous Quality Control

Quality Despite Ever-Shorter Release Cycles



November, 5  
10:30-12:00 CET  
[tmscl.me/cqc-2411-saa](https://tmscl.me/cqc-2411-saa)



# Folien & Kontakt - Ich freue mich auf Fragen 😊



Vortragsfolien:  
[tmscl.me/saa\\_cqc](https://tmscl.me/saa_cqc)



Dr. Tobias Röhm  
[roehm@cqse.eu](mailto:roehm@cqse.eu)  
<https://tmscl.me/coffee-tobias-roehm>