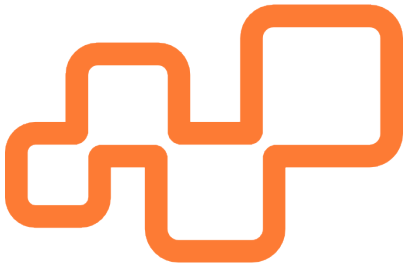


Using Static Analyses
For Continuous Quality Improvement

27. Juni 2019 – Applied Static Analysis (APSA) – TU Darmstadt

CQSE | Dr. Sven Amann

Praxis

Software-**Audits**

Kontinuierliche **Qualitäts-** und **Testkontrolle**

Teamscale

CQSE GmbH

Forschung

17+ **Promotionen** in Software Engineering

Eigene **Forschung**

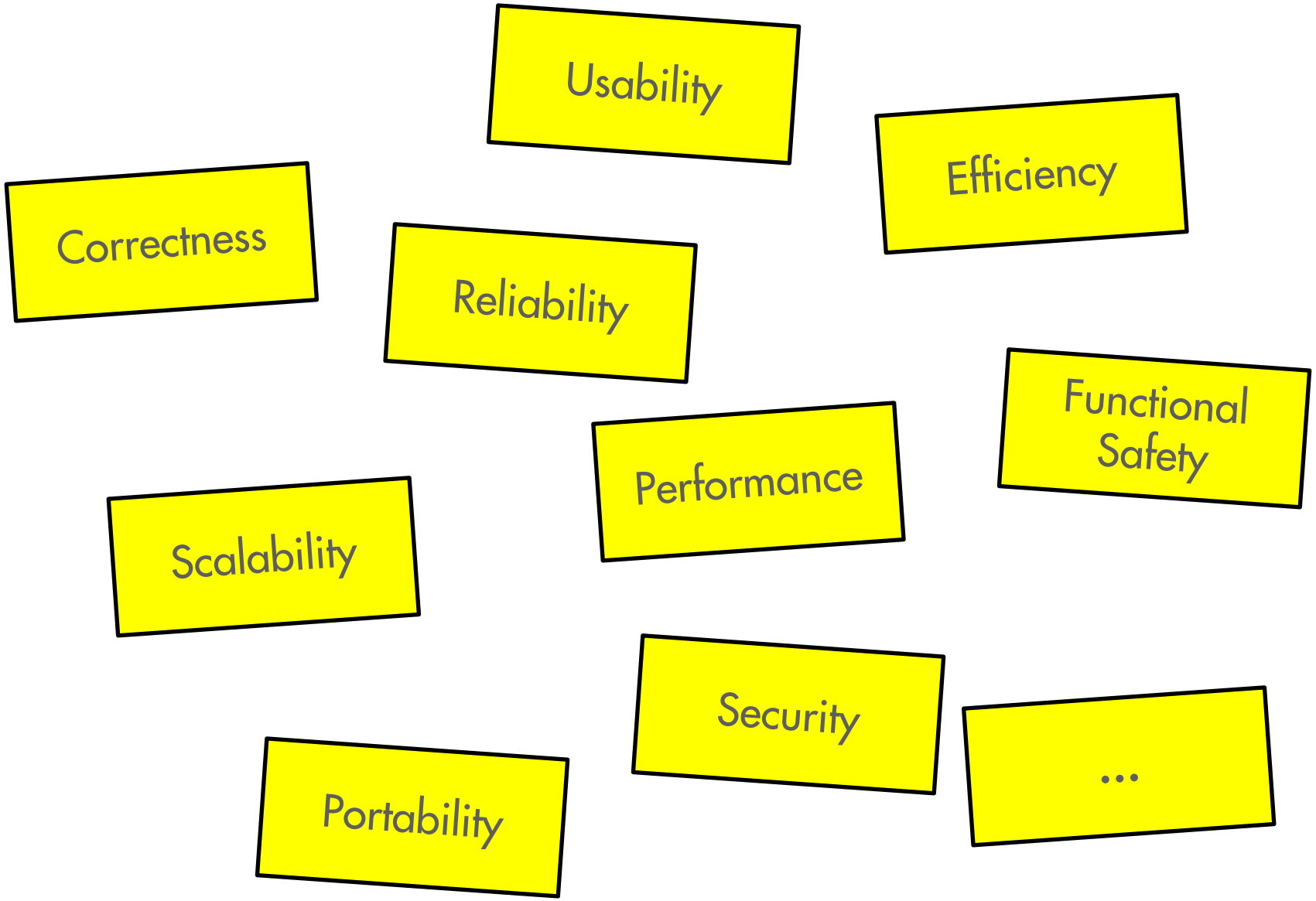
Enger Kontakt zu **Universitäten**

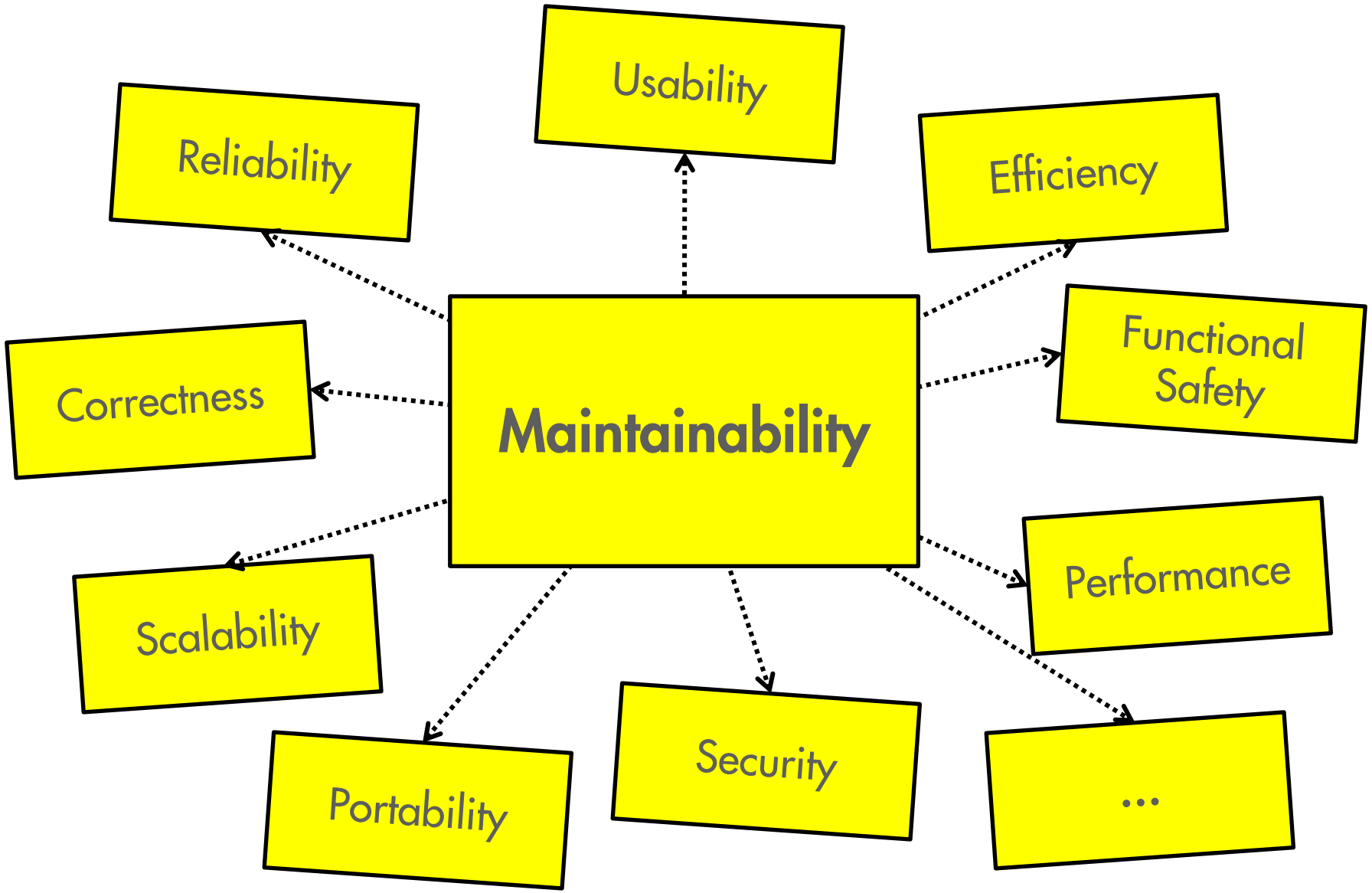
Agenda

- **11:40** **Introduction**
- 11:45 Static Analysis for Quality Improvement
- 12:05 Continuous Quality Improvement
- 12:50 Innovation & Improvement
- 13:10 Q&A
- 13:20 End of Lecture

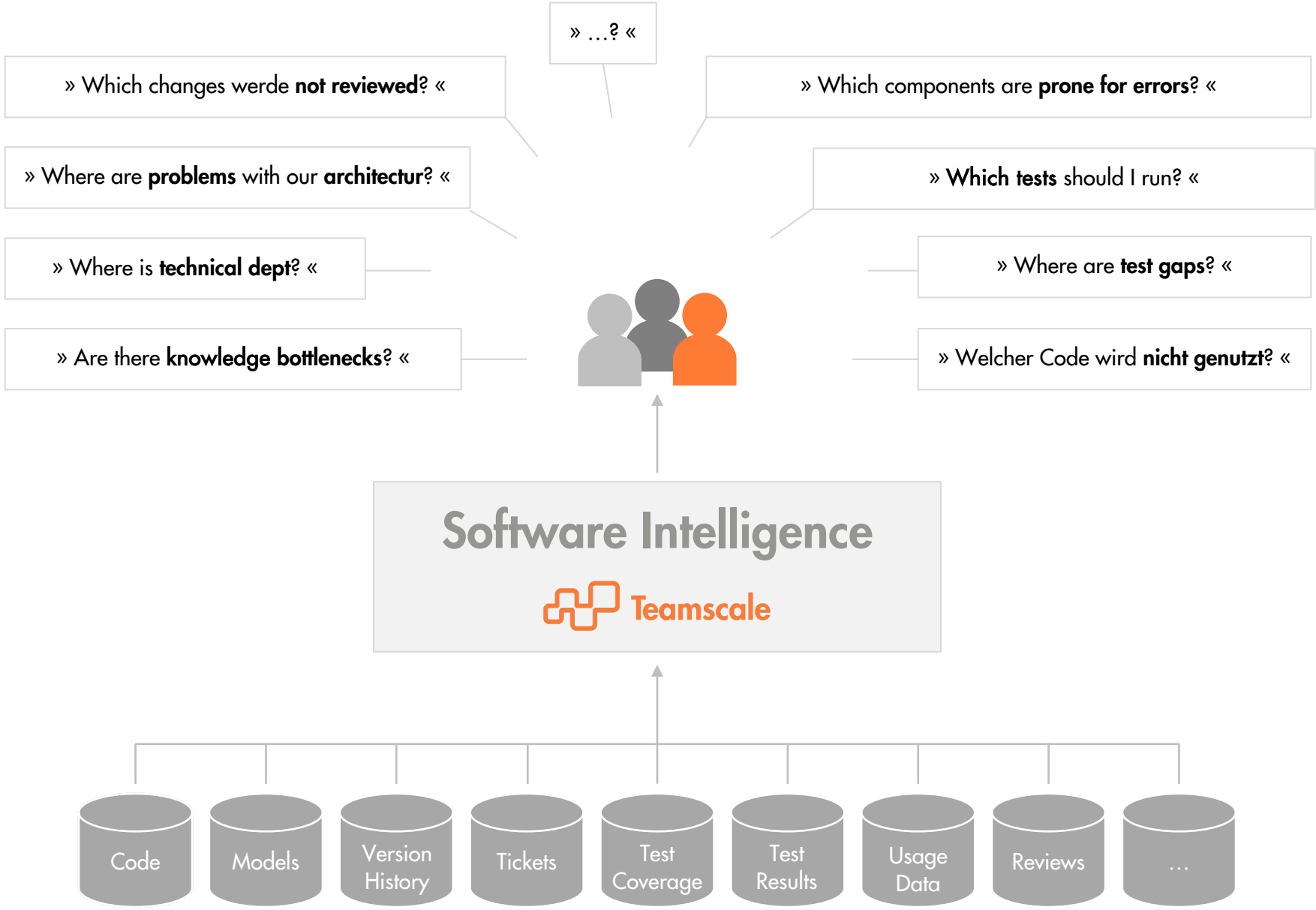
Agenda

- 11:40 Introduction
- **11:45 Static Analysis for Quality Improvement**
- 12:05 Continuous Quality Improvement
- 12:50 Innovation & Improvement
- 13:10 Q&A
- 13:20 End of Lecture

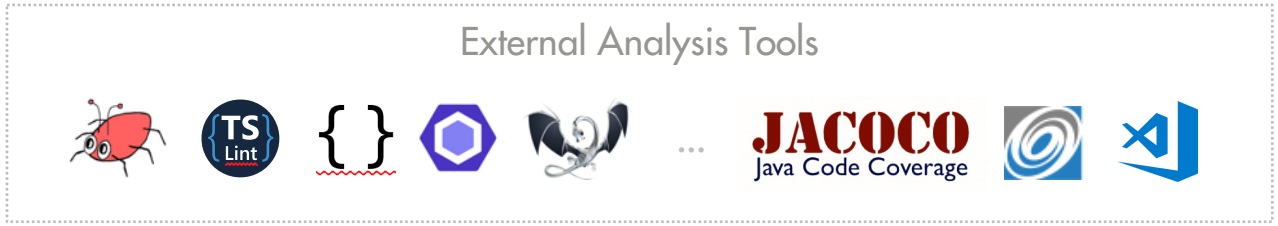

Usability
Efficiency
Correctness
Reliability
Functional Safety
Performance
Scalability
Security
...
Portability


Usability
Reliability
Efficiency
Correctness
Maintainability
Functional Safety
Scalability
Performance
Portability
Security
...

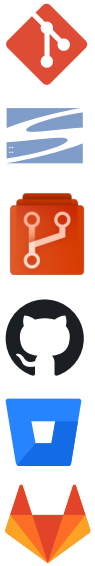
» All developers in a team
(newbies as well as veterans)
want to produce high-quality software,
i.e., write maintainable code. «


» ...? «
» Which changes werde **not reviewed**? «
» Which components are **prone for errors**? «
» Where are **problems** with our **architectur**? «
» **Which tests** should I run? «
» Where is **technical dept**? «
» Where are **test gaps**? «
» Are there **knowledge bottlenecks**? «
» Welcher Code wird **nicht genutzt**? «
Software Intelligence
Teamscale
Code
Models
Version History
Tickets
Test Coverage
Test Results
Usage Data
Reviews
...

ABAP
Ada
C/C++
C#
COBOL
Delphi
Fortran
Gosu
Groovy
IEC 61131–3 ST
Java
JavaScript
Kotlin
Matlab
Objective-C
OpenCL
OScript
PHP
PL/SQL
Python
Rust
SAP HANA
Simulink
Swift
Visual Basic
XML
Xtend



Version Control Systems



Incremental Analysis Engine

Clone Detection

Architecture Analysis

Comment Analysis

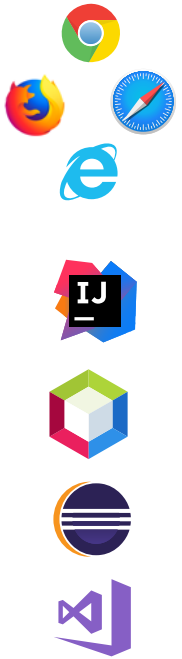
Findings Tracking

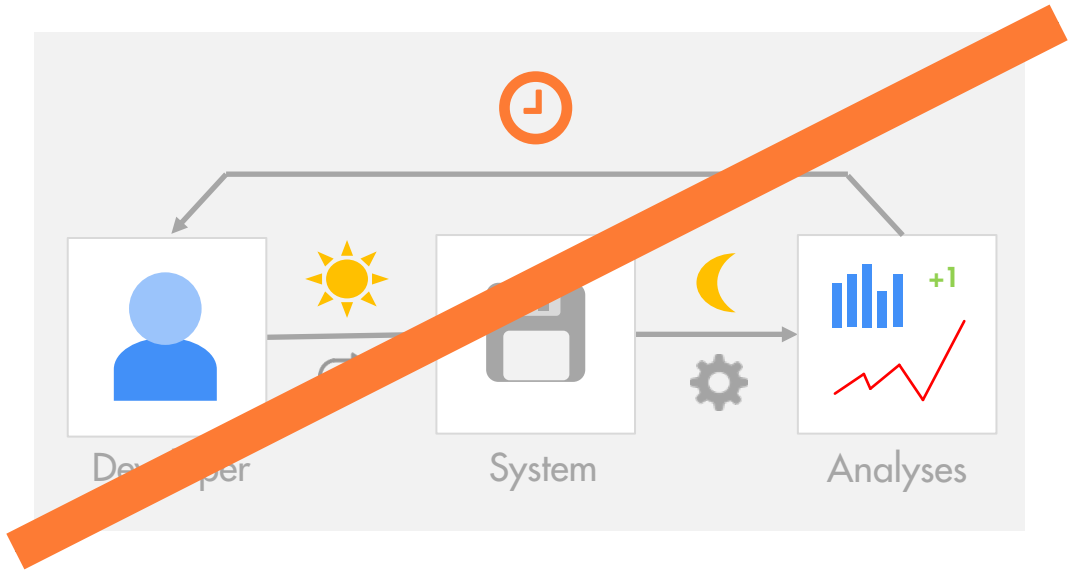
Teamscale

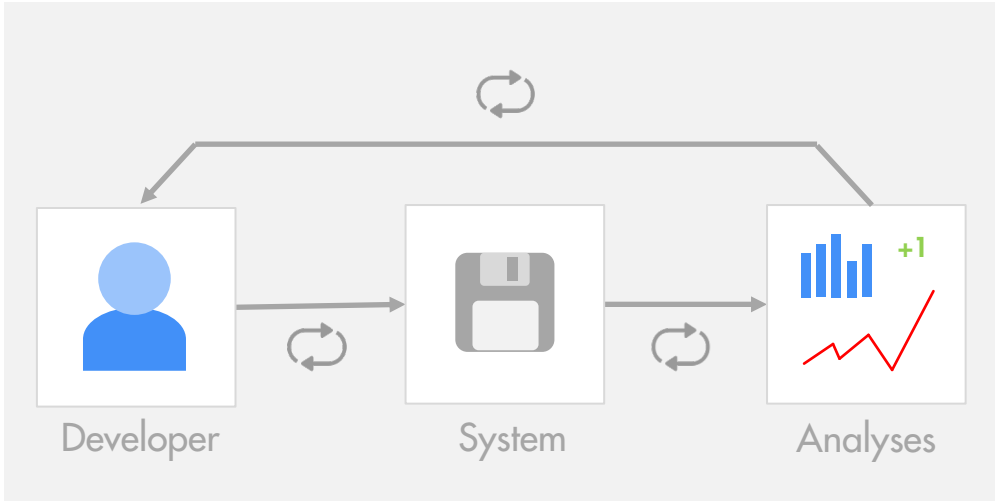
NoSQL Store

REST Service Layer

Clients




+1
System
Analyses


+1
Developer
System
Analyses

Our Goal:

Measure What Is Relevant

(as opposed to what is easy to measure)


CycC
CR
LCOM
No. Identifiers
CBO
No. Classes
No. Statements
Cohesion
Clone Coverage
NOPM
NOPA
Assertion Density
Line Coverage
Defect Density
Statement Coverage
Flow
NOC
DIT
Halstead
Abstractness
Branch Coverage
Coupling
Stability
LOC
Blow Up
Fan Out
NOAM
MCDC
RFC
WMC
Fan In
RFSS
No. Methods

$$171 - 5.2 \cdot \ln(avgHV) - 0.23 \cdot avgCC(g') -$$

$$16.2 \cdot \ln(avgLOC) + 50 \cdot \sin(\sqrt{2.4 \cdot perCM})$$

HV: Halstead Volume

LOC: lines of code

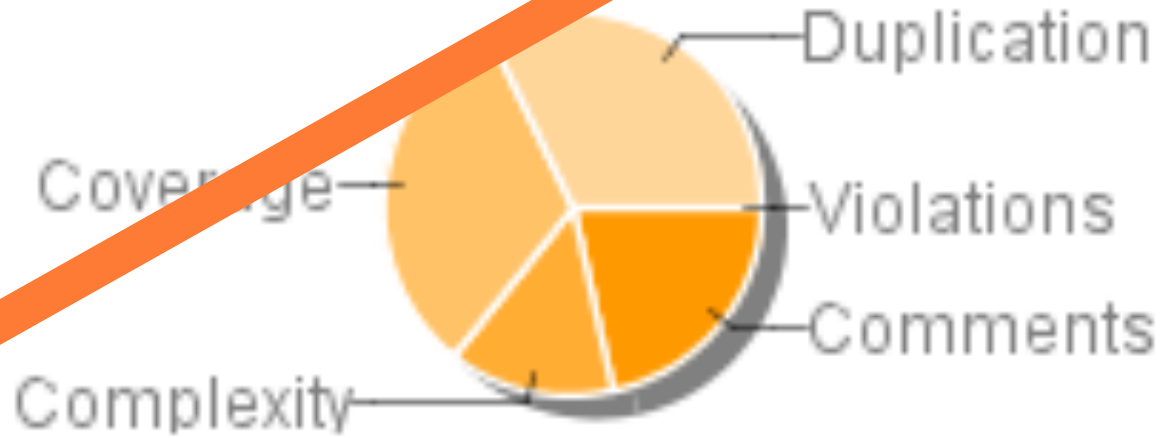
CC: Cyclomatic Complexity

perCM: % Comment Lines

Technical Debt

$ 14'605

29 man days



http://www.sonarqube.org/

Best Practice: Goal-Oriented Analyses

- Objective
- Impact on the code is clear
- Actionable
- Clear relation to maintainability

```
// 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();
}
```

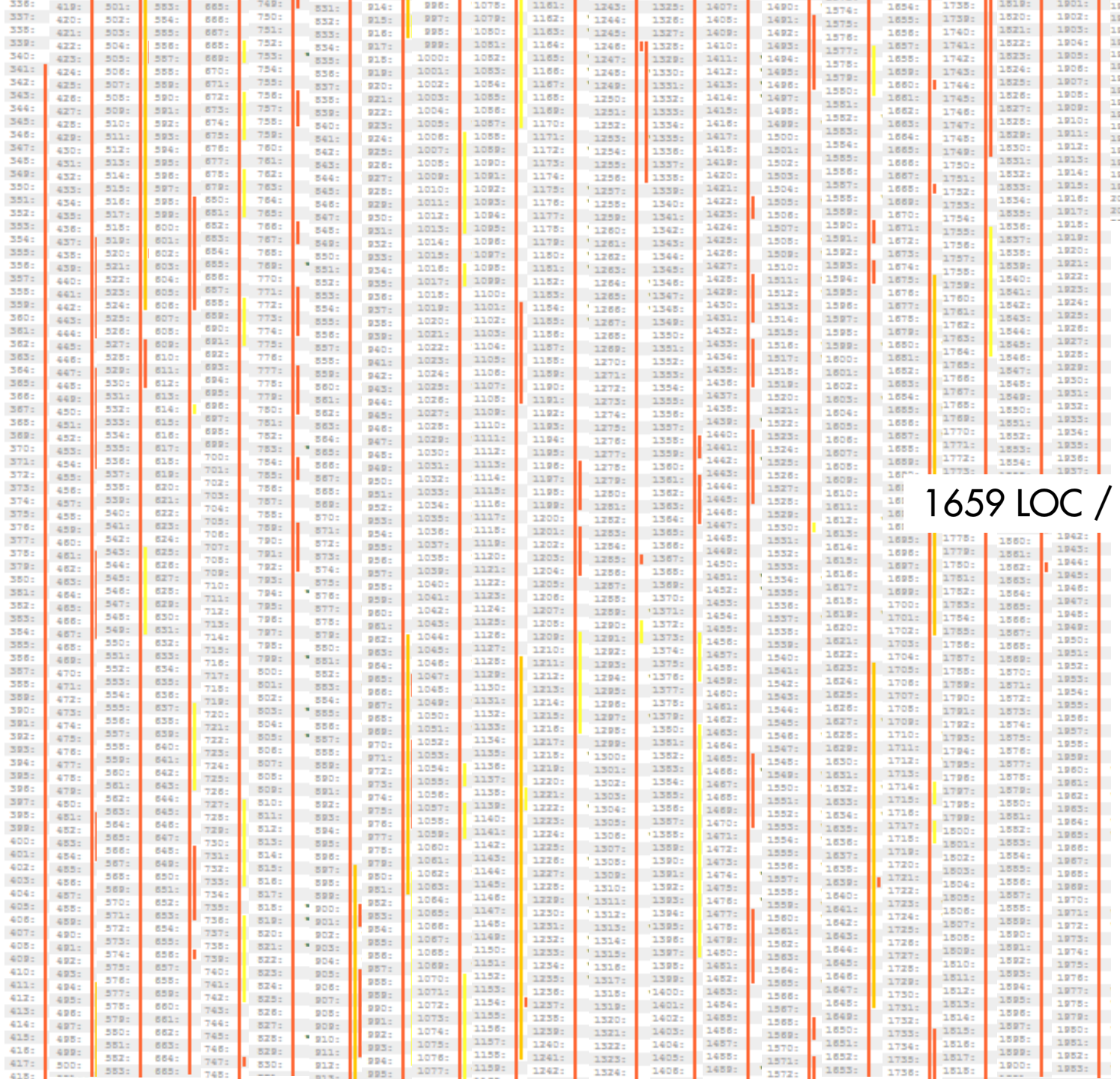
Study

Munich Re LV 1871

- Over 100 bugs in produktive Software

- 52% of all unintentional differences buggy

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



1659 LOC / 1301 SLOC

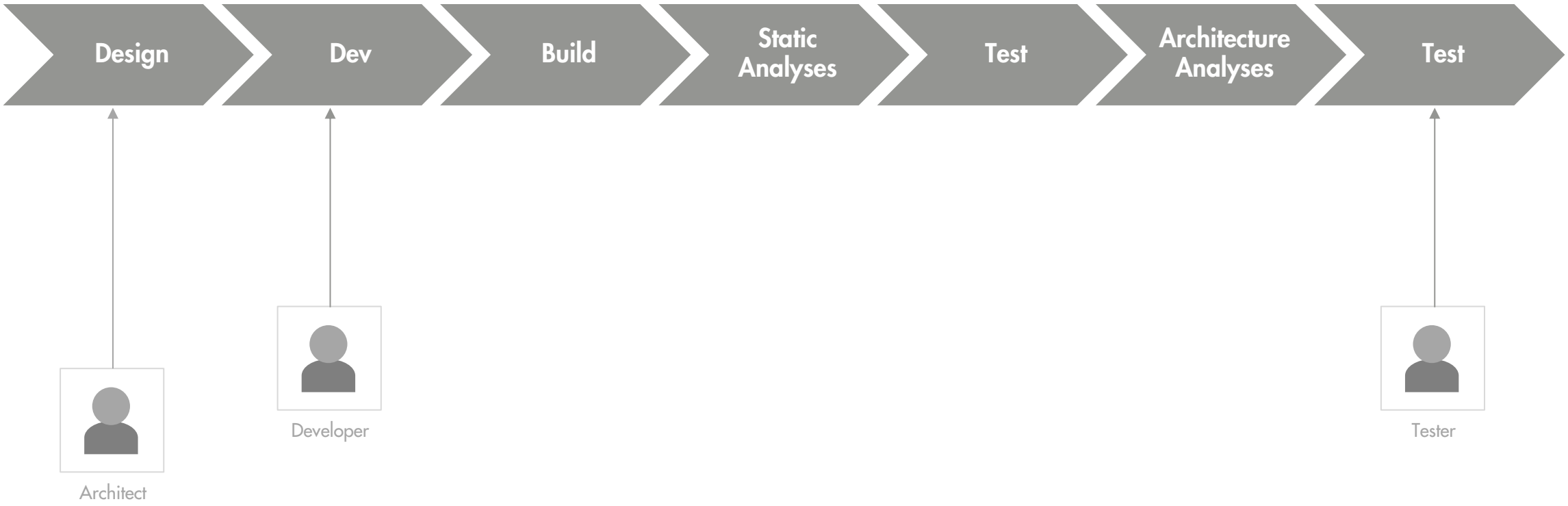
```
1418:   IF NOT lt_zeile_num IS INITIAL.
1419:   " Anzahl = 1
1420:     IF l_anz = 1.
1421:       LOOP AT lt_zeile_num INTO ls_zeile_num.
1422:         READ TABLE gt_gui_liste_angebot_0500
1423:           INTO ls_gui_liste_angebot INDEX ls_zeile_num
1424:
1425:         IF ls_gui_liste_angebot-auftyp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

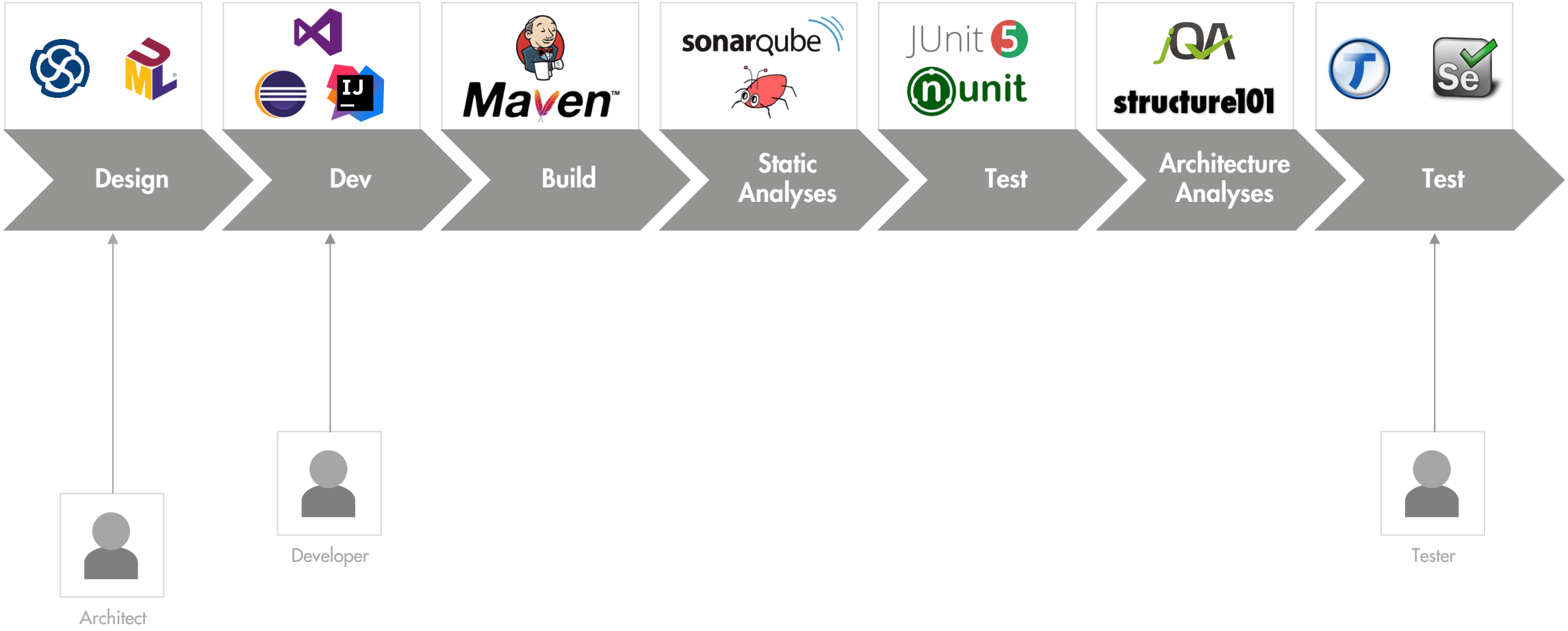
```
                                                priname2 = "REZ793LX.LST";
                                                                                    else
                                                                                            pri
                                                                            }
                                                                    }
                                                                    else
                                                27                  {
                                                                            if(NOT memcmp(print
                                                                    28      {
                                                                                    priname = '
                                                                            29      if(sub == 1
                                                                                    {
                                                                                            if(
                                                        priname2 = "REZ793.LST";
                                                                                            els

                                                                                    }
                                                                            }
                                                                            else
                                                                            {
                                                                                    priname = '
                                                                                    if(sub == 1
                                                                                            * r
                                                                {
                                                                        if(SAR(w_subrez) == 0)
                                                                        {
                                                                                if(NOT memcmp(print

                                                                                    priname2 =
                                                                                else
                                                                                priname2 = "REZ793.
                                                                }
                                                                        else
                                                                        priname  = "PRISUB.LST";
                                                                                    }
                                                        }
                                                                        }
                                                                }
                                                        }
                                                }
                                        }
                                }
                        }
}
                }
        }
        }
```

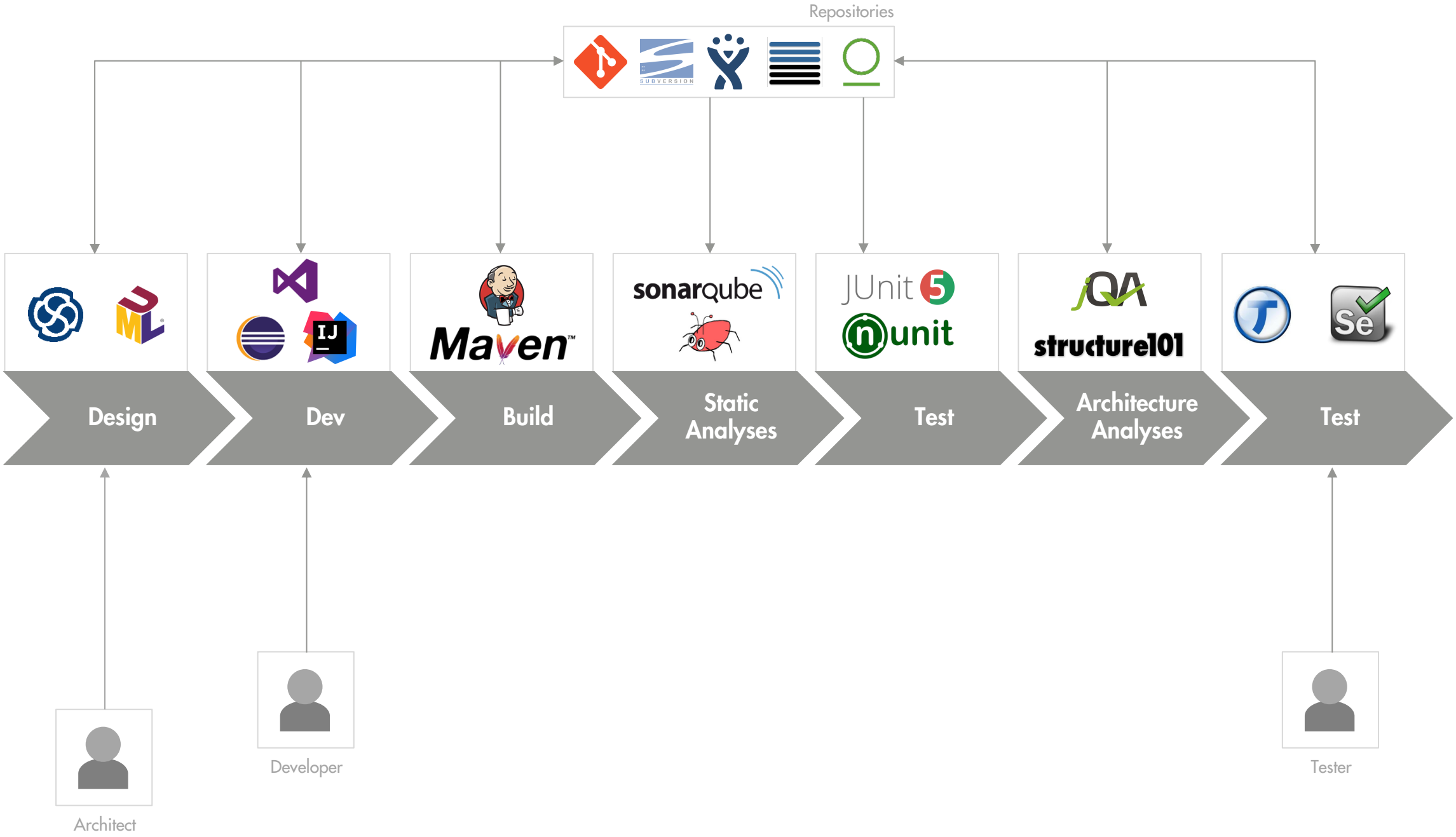
Agenda

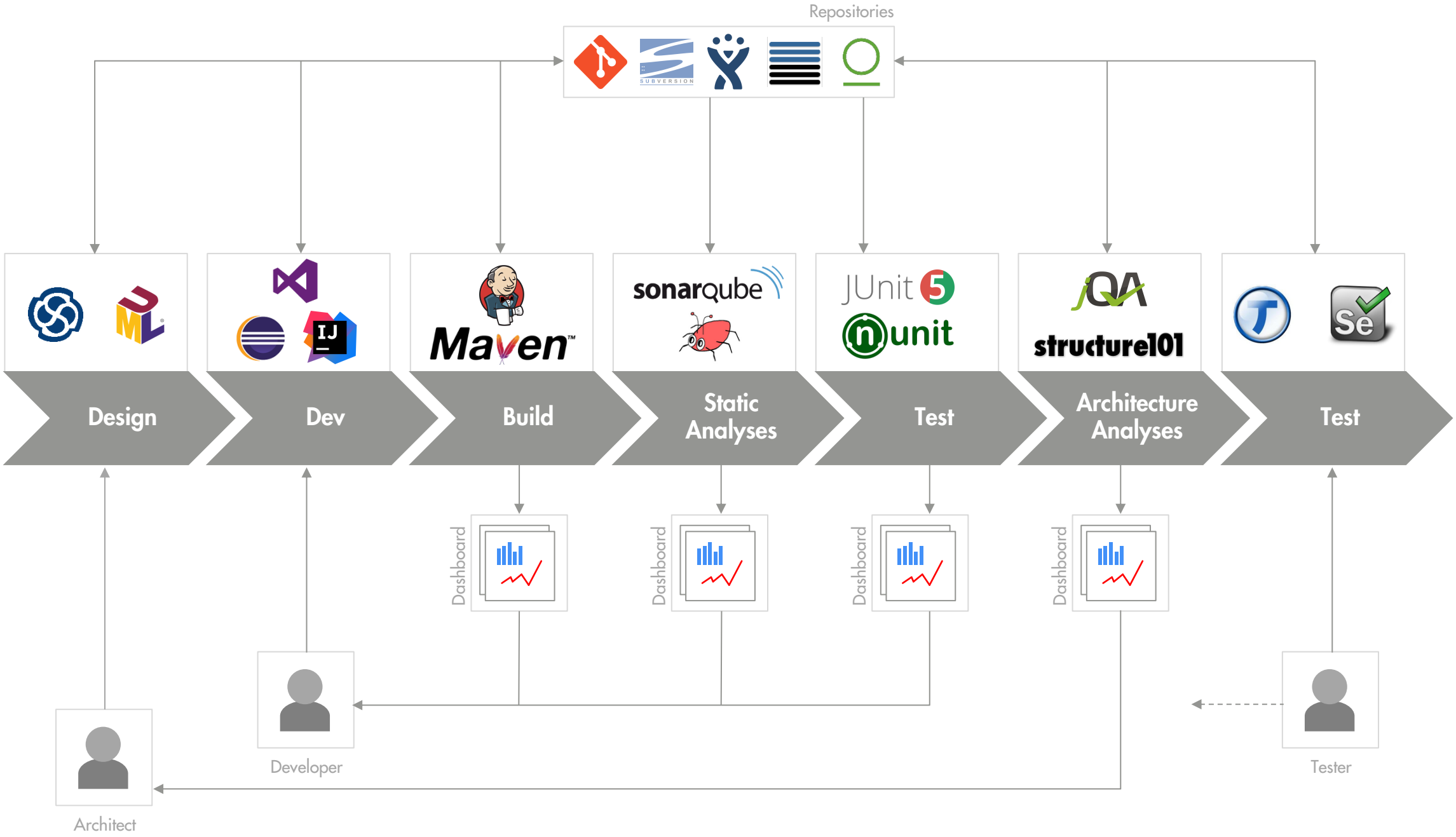
- 11:40 Introduction
- 11:45 Static Analysis for Quality Improvement
- **12:05 Continuous Quality Improvement**
- 12:50 Innovation & Improvement
- 13:10 Q&A
- 13:20 End of Lecture

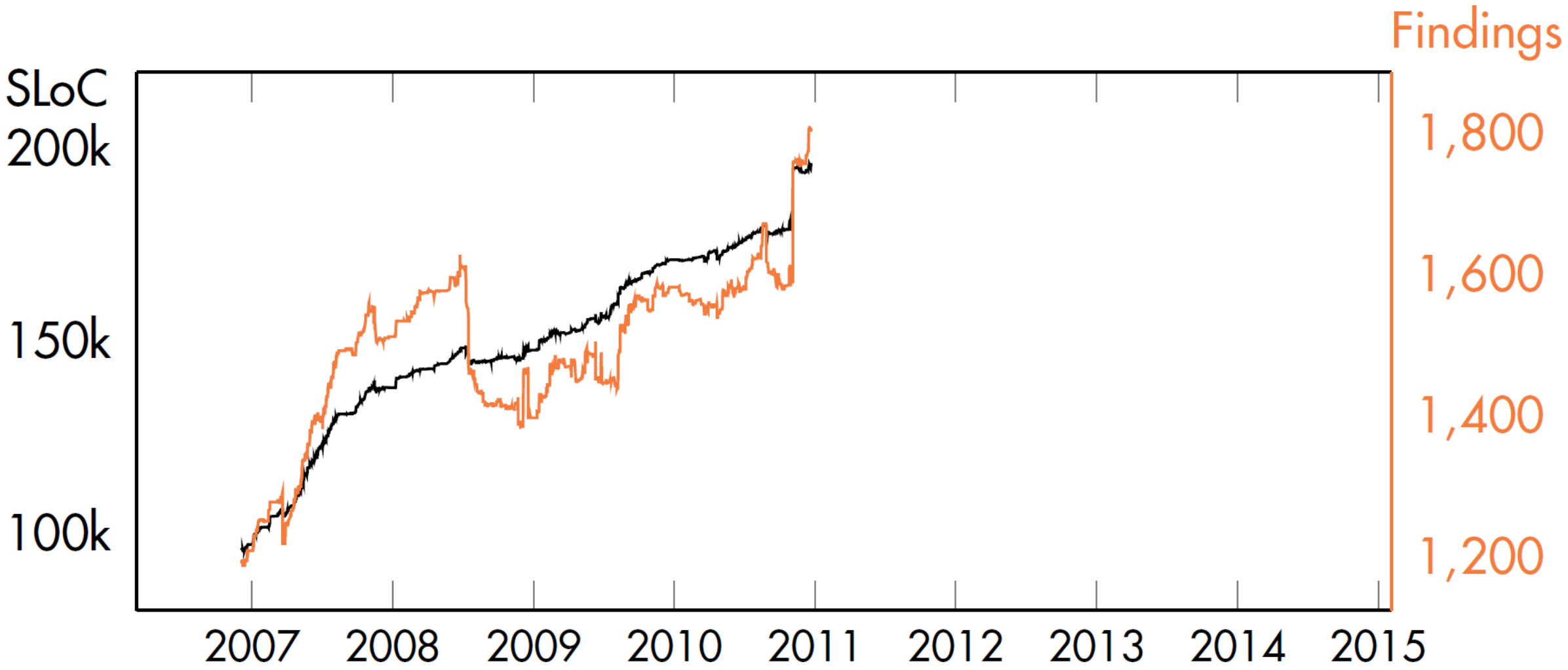

Design
Dev
Build
Static Analyses
Test
Architecture Analyses
Test

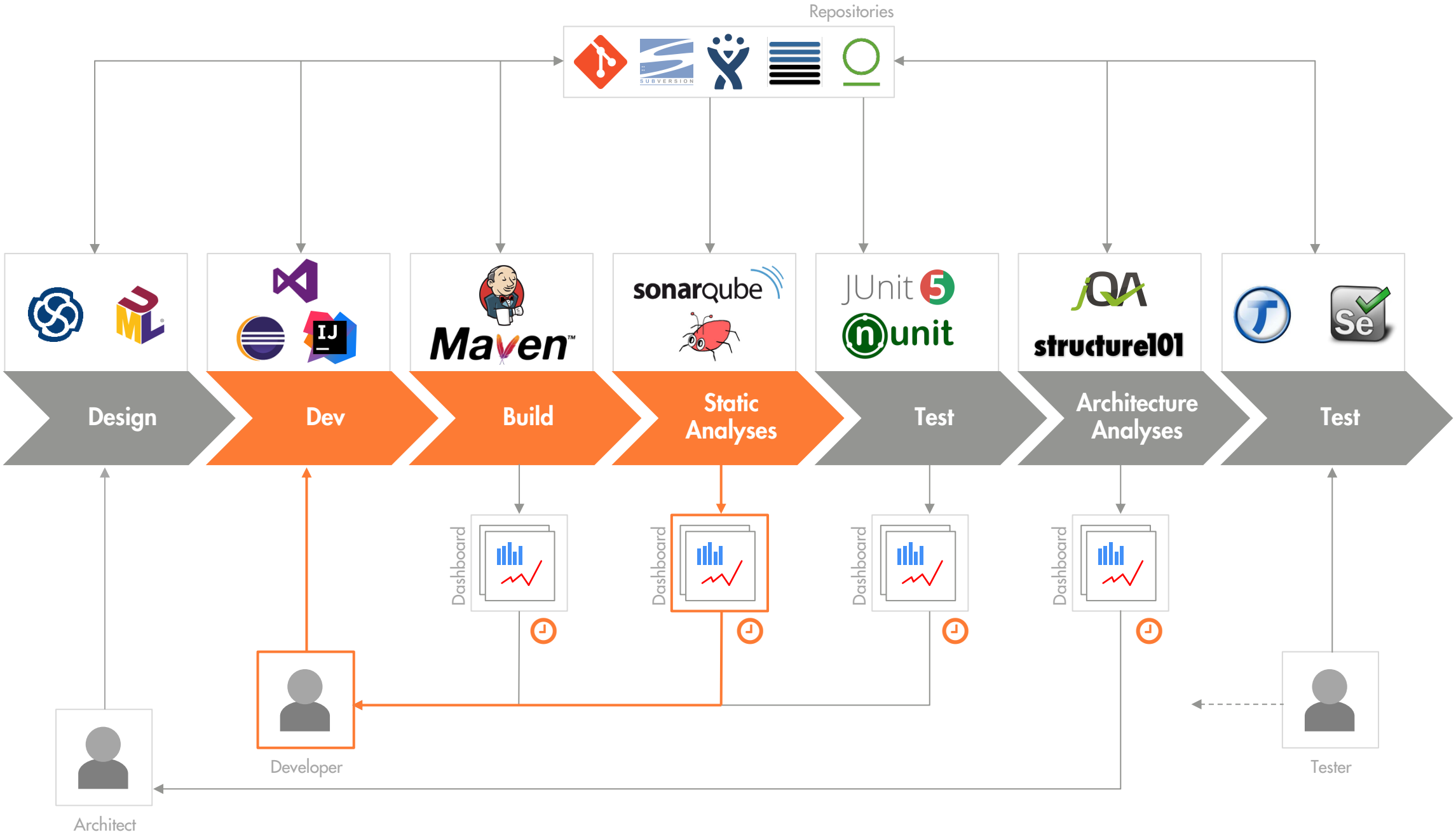

Design
Dev
Build
Static Analyses
Test
Architecture Analyses
Test
Architect
Developer
Tester

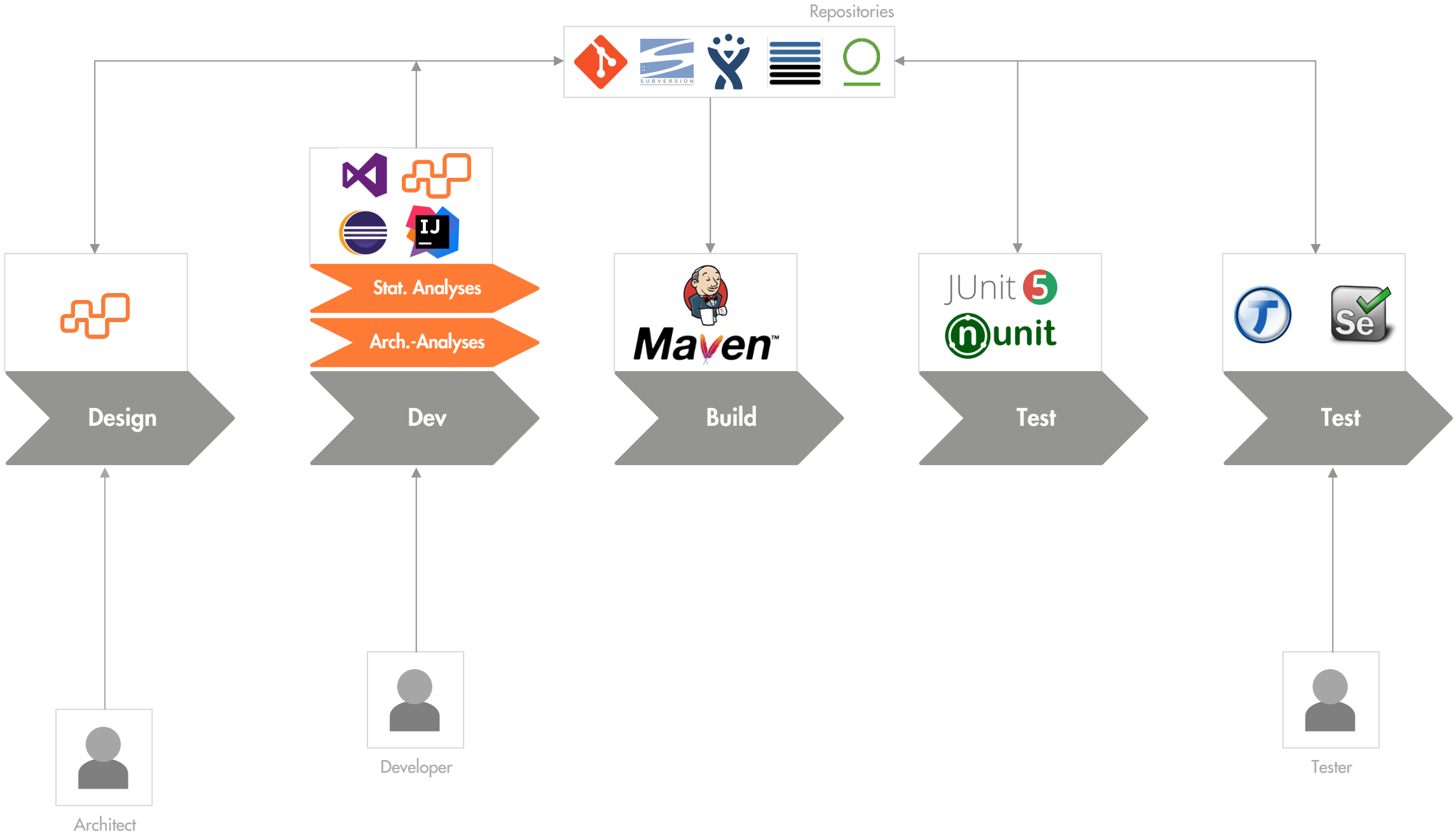

Design
Dev
Build
Static Analyses
Test
Architecture Analyses
Test
Architect
Developer
Tester

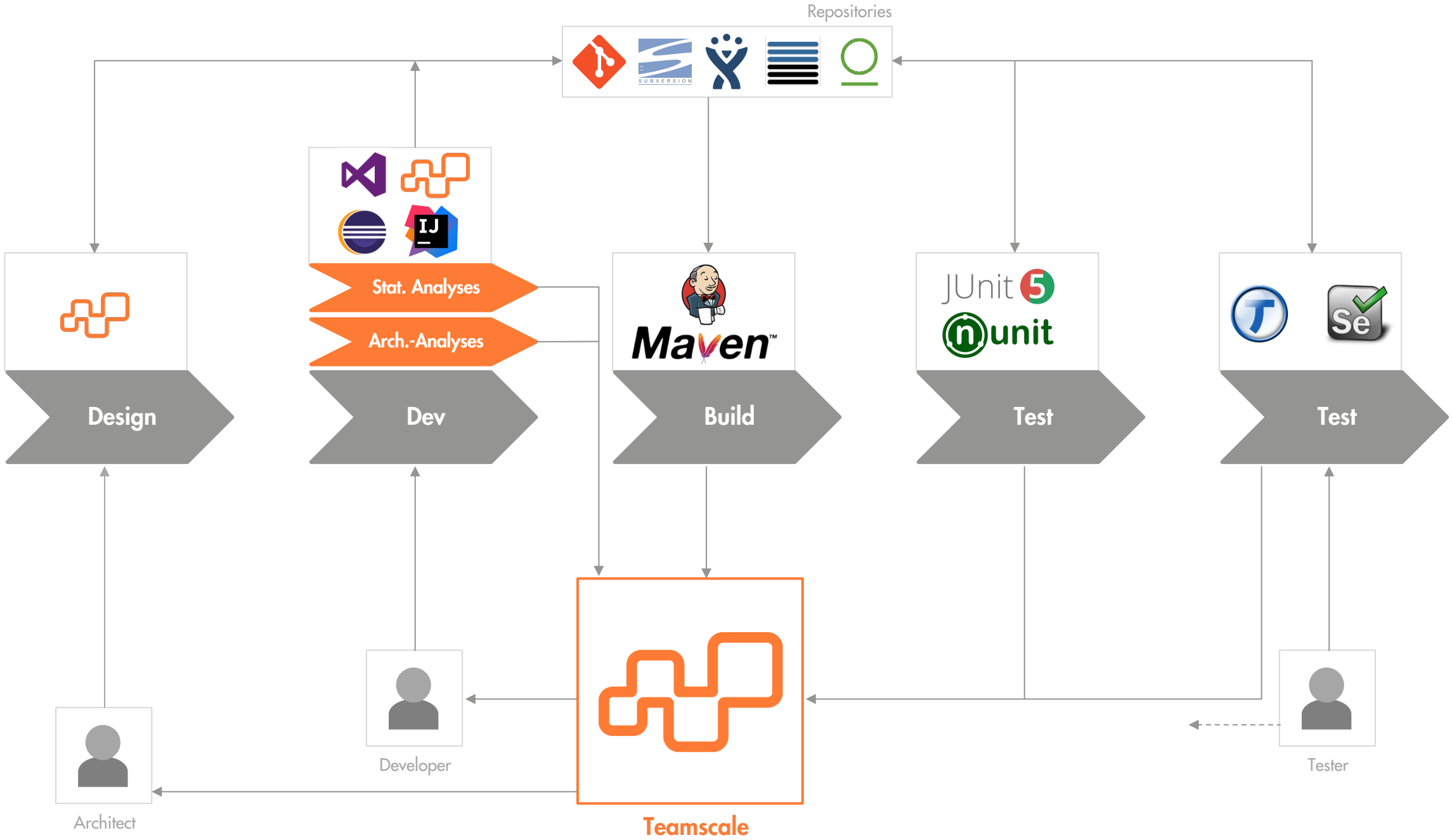

Repositories
Design
Dev
Build
Static Analyses
Test
Architecture Analyses
Test
Architect
Developer
Tester

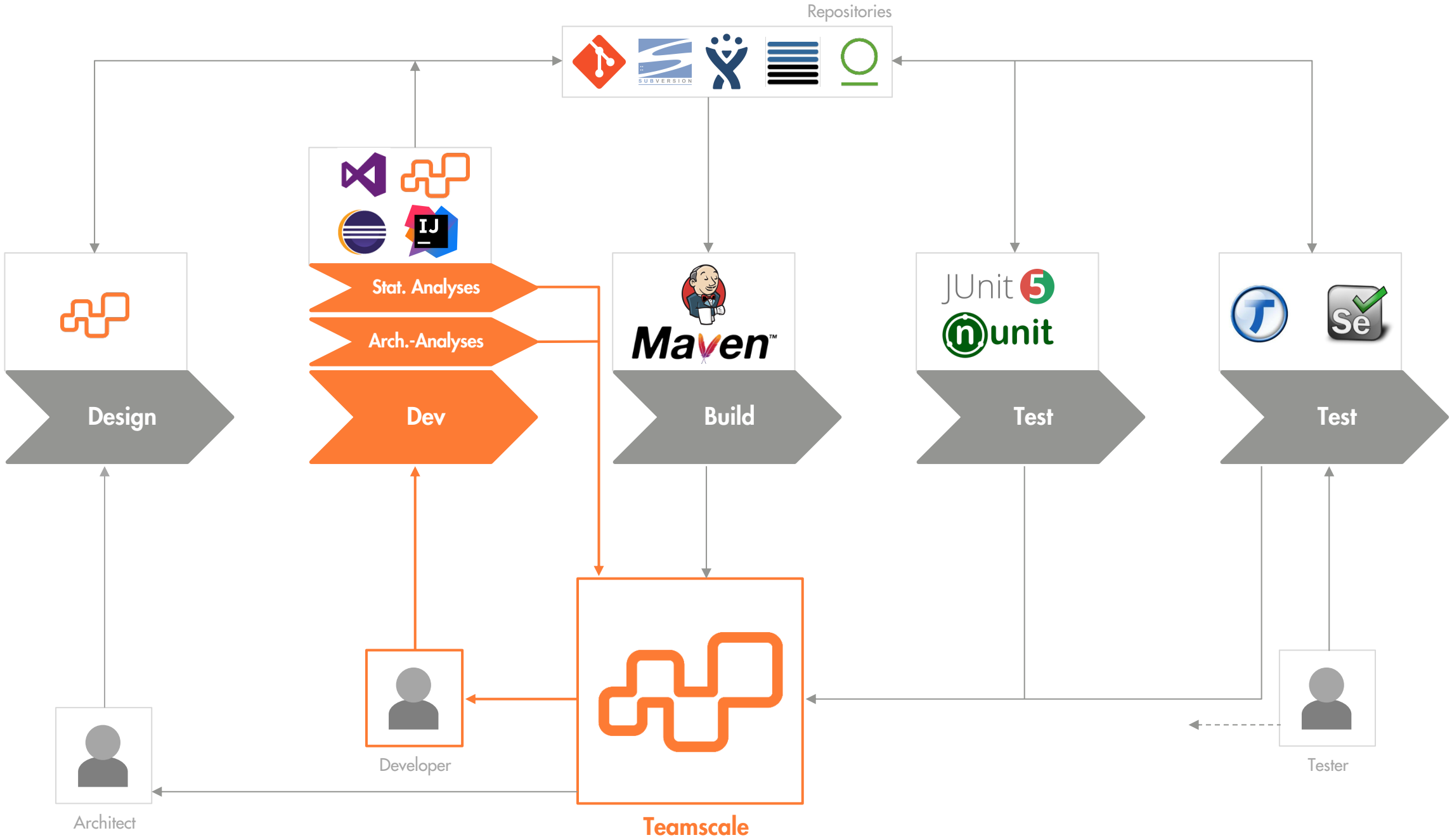

Repositories
Design
Dev
Build
Static Analyses
Test
Architecture Analyses
Test
Dashboard
Dashboard
Dashboard
Dashboard
Architect
Developer
Tester


Findings
SLoC
200k
150k
100k
1,800
1,600
1,400
1,200
2007
2008
2009
2010
2011
2012
2013
2014
2015


Repositories
Design
Dev
Build
Static Analyses
Test
Architecture Analyses
Test
Dashboard
Dashboard
Dashboard
Dashboard
Developer
Architect
Tester


Repositories
Stat. Analyses
Arch.-Analyses
Design
Dev
Build
Test
Test
Architect
Developer
Tester

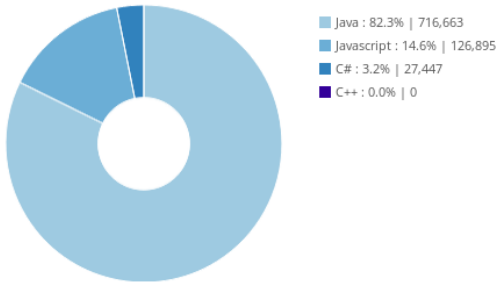

Repositories
Stat. Analyses
Arch.-Analyses
Design
Dev
Build
Test
Test
Architect
Developer
Teamscale
Tester


Repositories
Stat. Analyses
Arch.-Analyses
Dev
Design
Build
Test
Test
Developer
Tester
Architect
Teamscale

Demo

https://demo.teamscale.com

LOC Distribution for cqse-all



Clone Coverage for cqse-all

2.4%

+2.40%

Findings Density for cqse-all

21.9

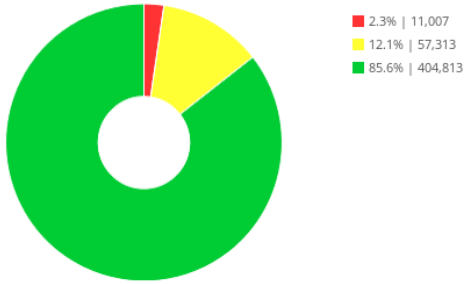
+21.9

Test Coverage for cqse-all

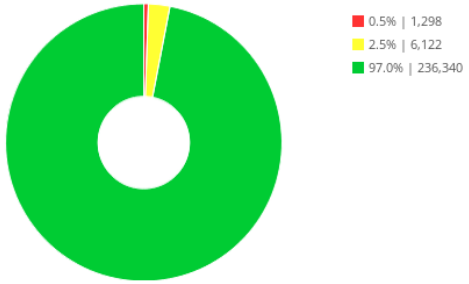
71.8%

+0.05%

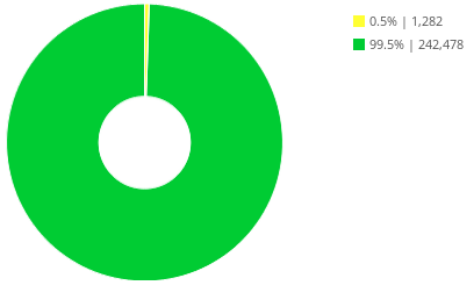
File Length for cqse-all



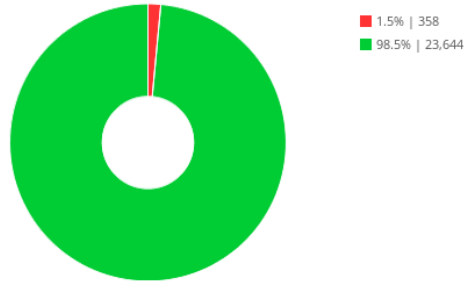
Method Length for cqse-all



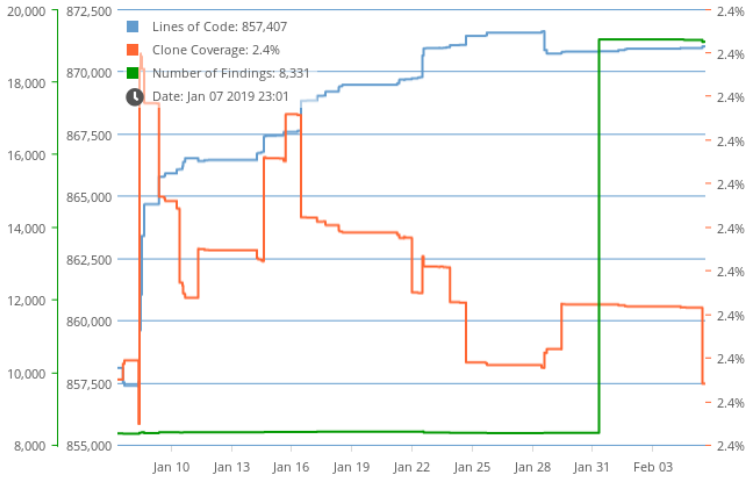
Nesting Depth for cqse-all



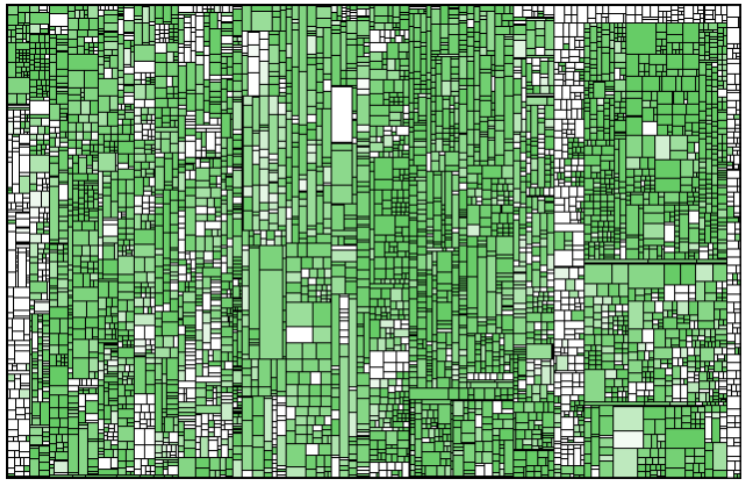
Comment Completeness for cqse-all



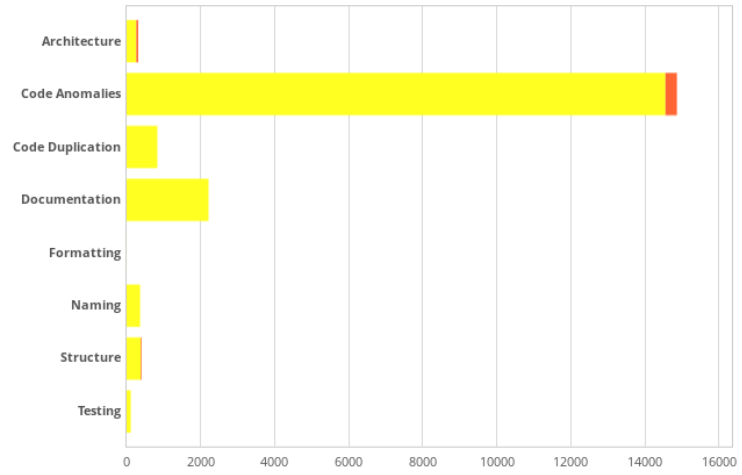
30 Day Trend for cqse-all

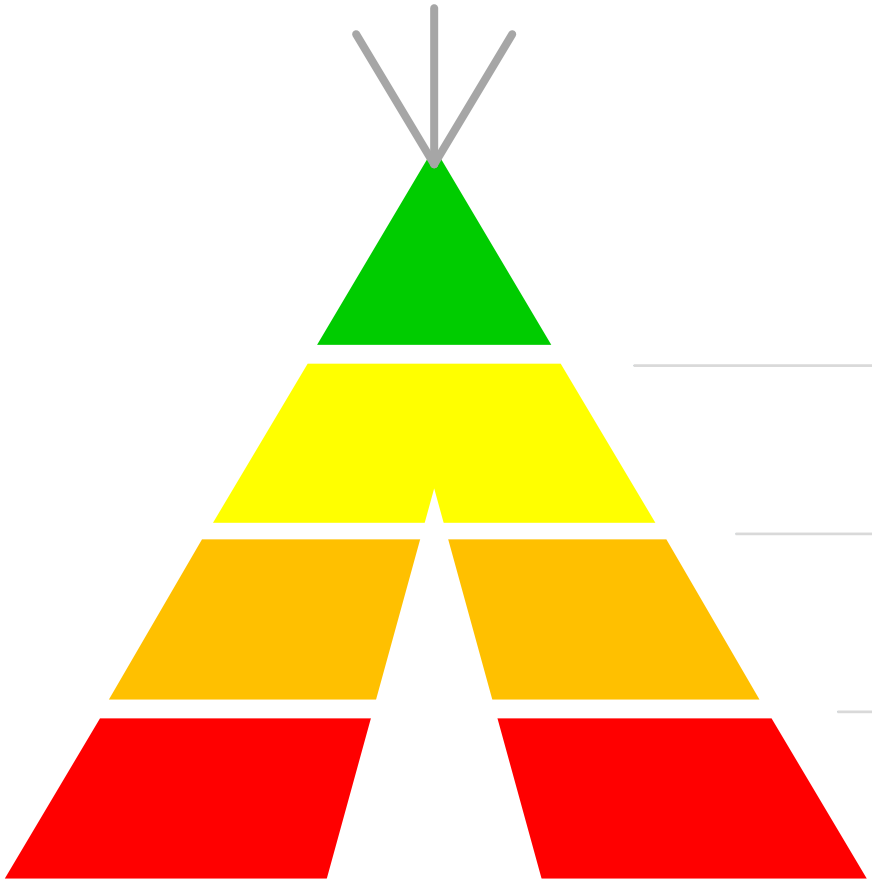


Test Coverage Treemap for cqse-all



Findings Summary Bar Chart for cqse-all



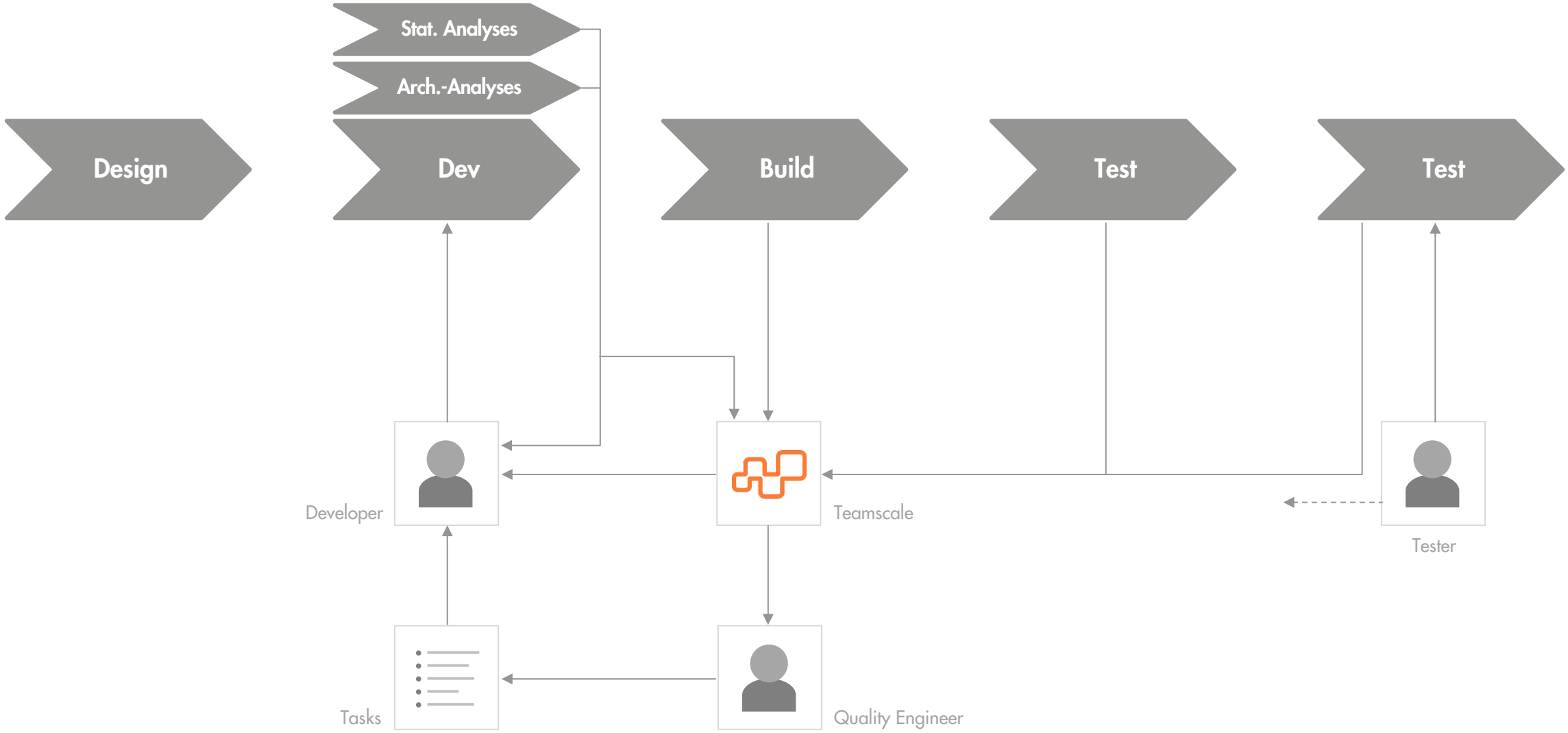


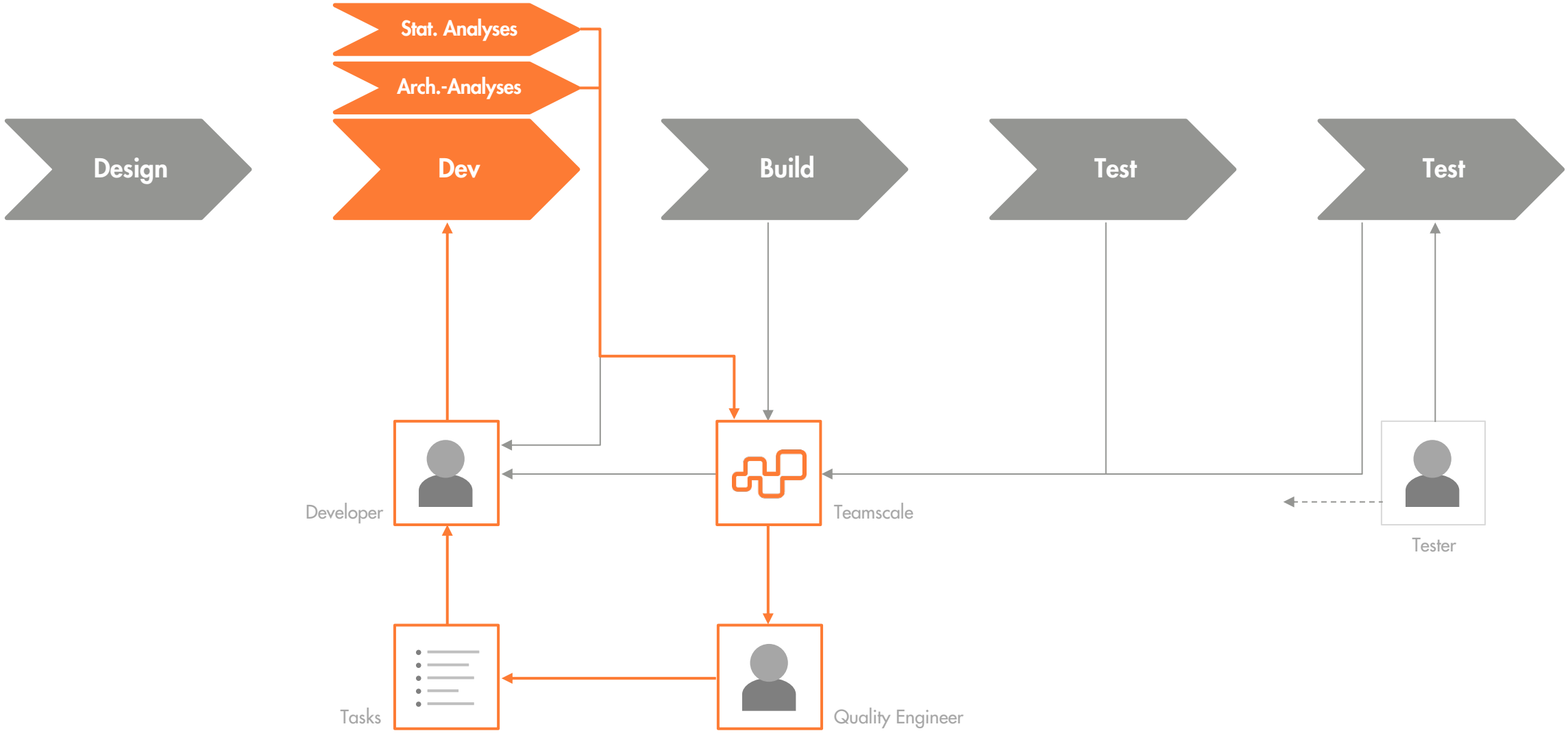
No Findings

No Findings in Changed Code

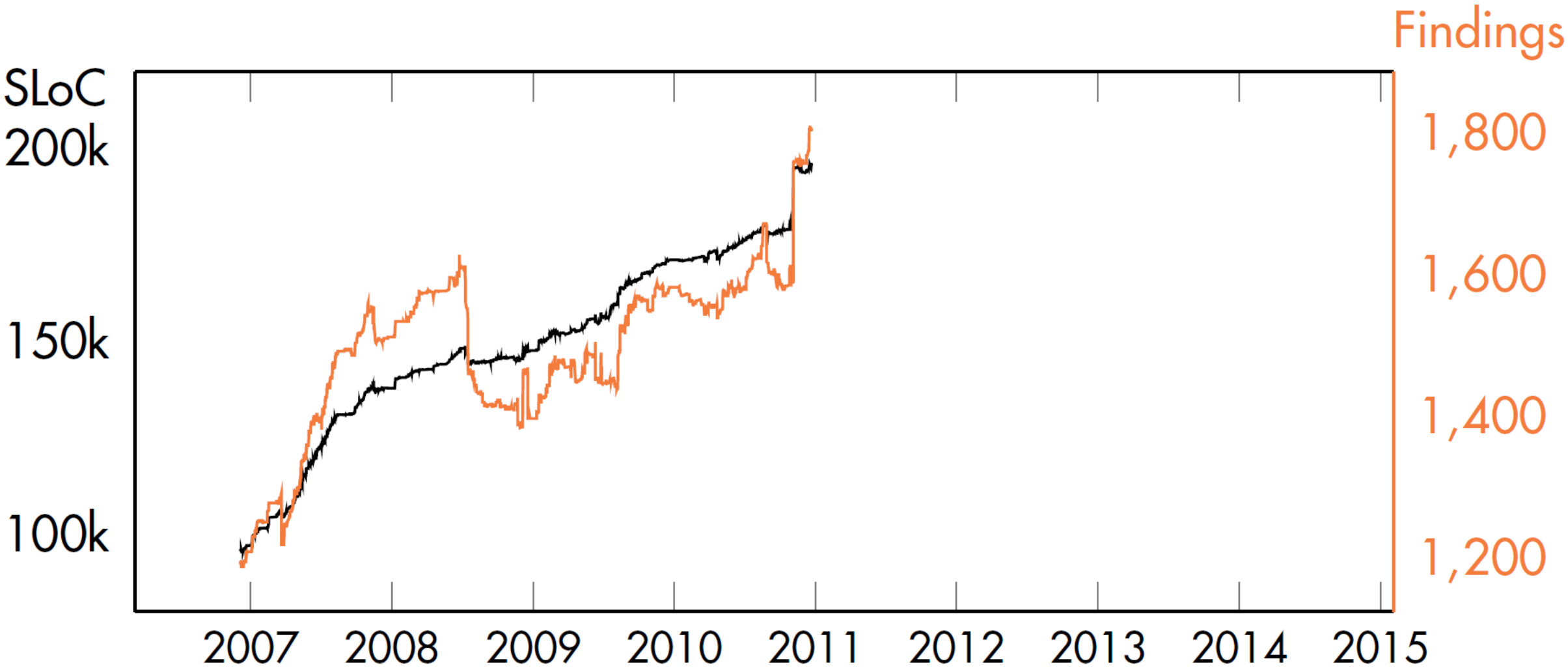
No New Findings

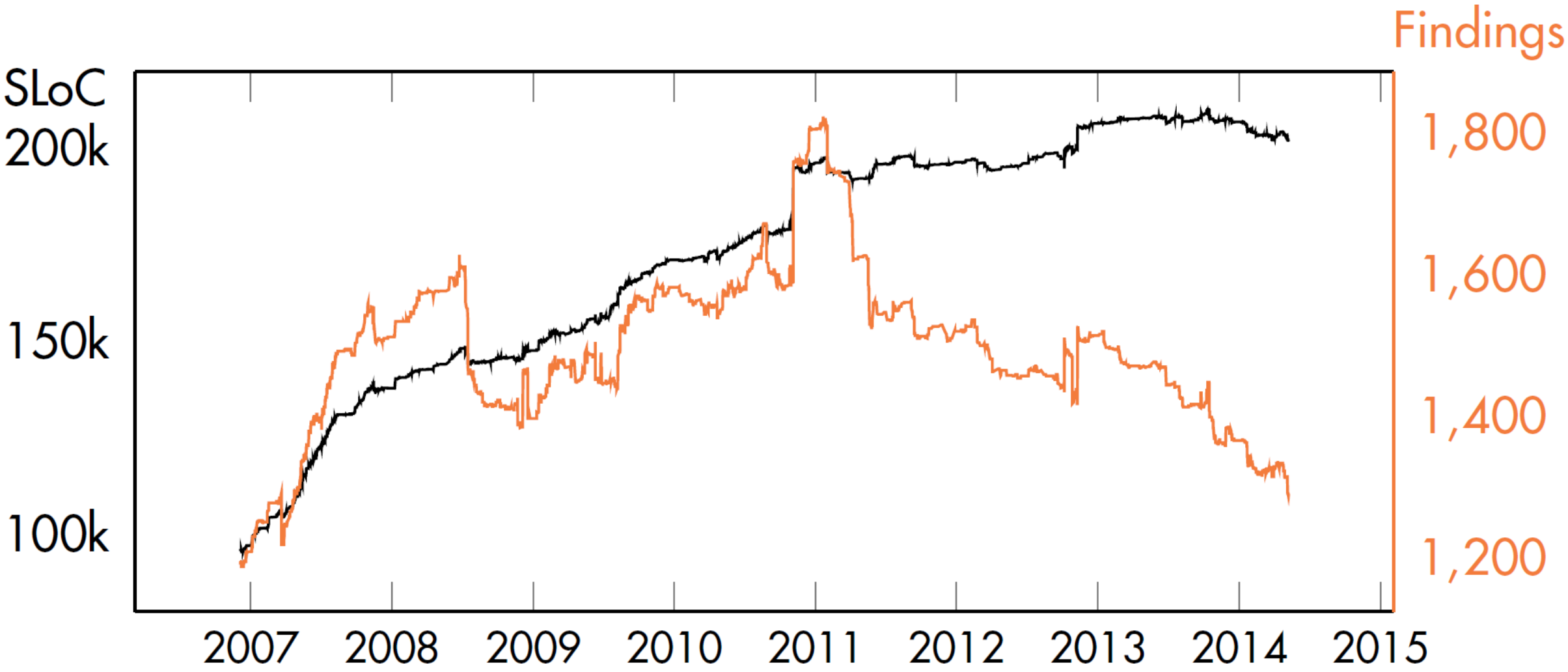
I don’t care

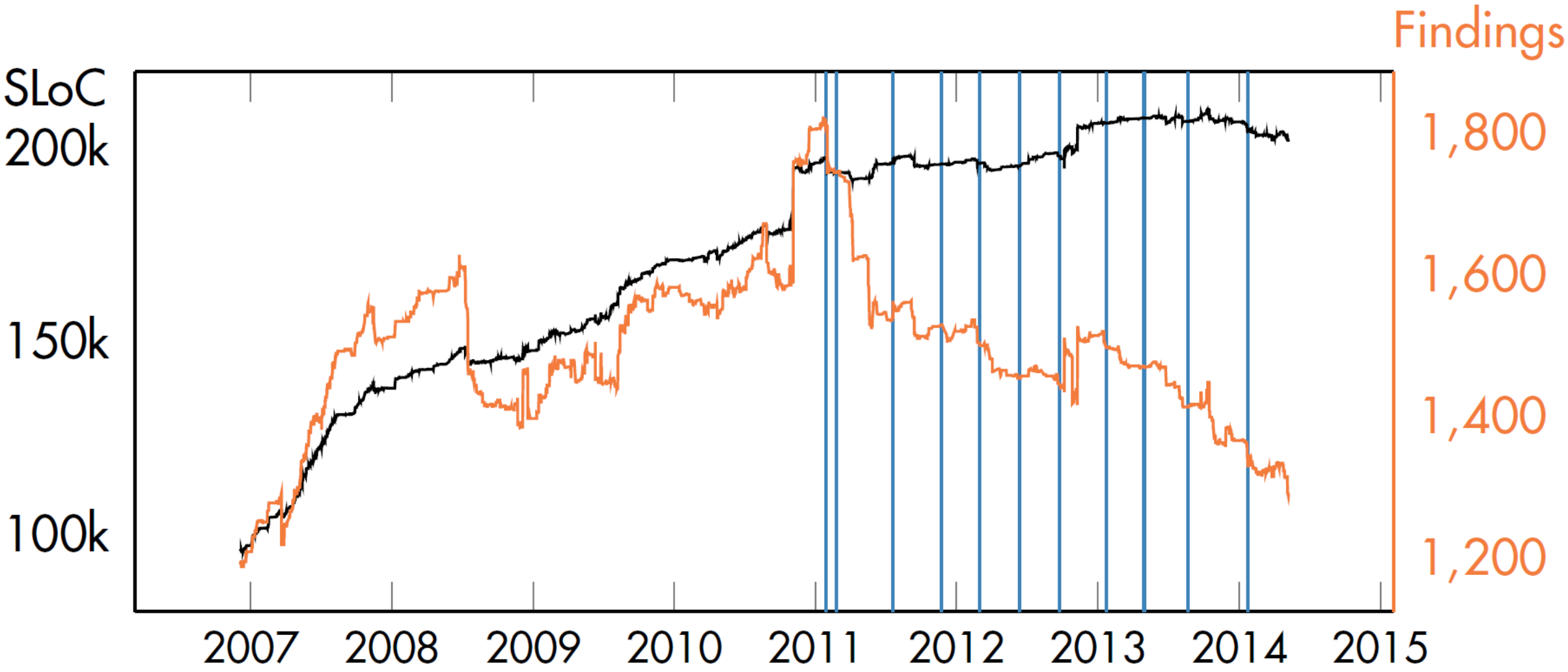

Stat. Analyses
Arch.-Analyses
Design
Dev
Build
Test
Test
Developer
Teamscale
Tester
Tasks
Quality Engineer


Stat. Analyses
Arch.-Analyses
Design
Dev
Build
Test
Test
Developer
Teamscale
Tester
Tasks
Quality Engineer

Demo


Findings
SLoC
200k
150k
100k
1,800
1,600
1,400
1,200
2007
2008
2009
2010
2011
2012
2013
2014
2015


Findings
SLoC
200k
150k
100k
1,800
1,600
1,400
1,200
2007
2008
2009
2010
2011
2012
2013
2014
2015


Findings
SLoC
200k
150k
100k
1,800
1,600
1,400
1,200
2007
2008
2009
2010
2011
2012
2013
2014
2015

Agenda

- 11:40 Introduction
- 11:45 Static Analysis for Quality Improvement
- 12:05 Continuous Quality Improvement
- **12:50 Innovation & Improvement**
- 13:10 Q&A
- 13:20 End of Lecture


DEPARTMENT OF INFORMATICS
DEPARTMENT OF INFORMATICS
Department of Informatics
Technische Universität München
FAKULTÄT FÜR INFORMATIK
TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK
Technische Universität München
Technische Universität München
Fakultät für Informatik
FAKULTÄT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK
TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK
Technische Universität München
Technical University
FAKULTÄT FÜR INFORMATIK
DEPARTMENT OF INFORMATICS
TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK
FAKULTÄT FÜR INFORMATIK
DEPARTMENT OF INFORMATICS
TECHNISCHE UNIVERSITÄT MÜNCHEN
DEPARTMENT OF INFORMATICS
TECHNICAL UNIVERSITY OF MUNICH
Master's Thesis in Informatics
Cross-IDE Quick-Fixes to Support Automatic Refactorings
André Kupka
Continuous Quality Control of Long-Lived Software Systems
Florian Deißenböck
Technische Universität München
Why and How to Control Cloning in Software Artifacts
Elmar Juergens
Technische Universität München
Integrated Behavior Modeling of Space-Intensive Mechatronic Systems
Benjamin Hummel
Technische Universität München
Cost-Effective Quality Assurance For Long-Lived Software Using Automated Static Analysis
Daniela Steidl
Technische Universität München
Technische Universität München
Erkennung von Prozessverletzungen in Scrum-Teams

www.thesisguide.org

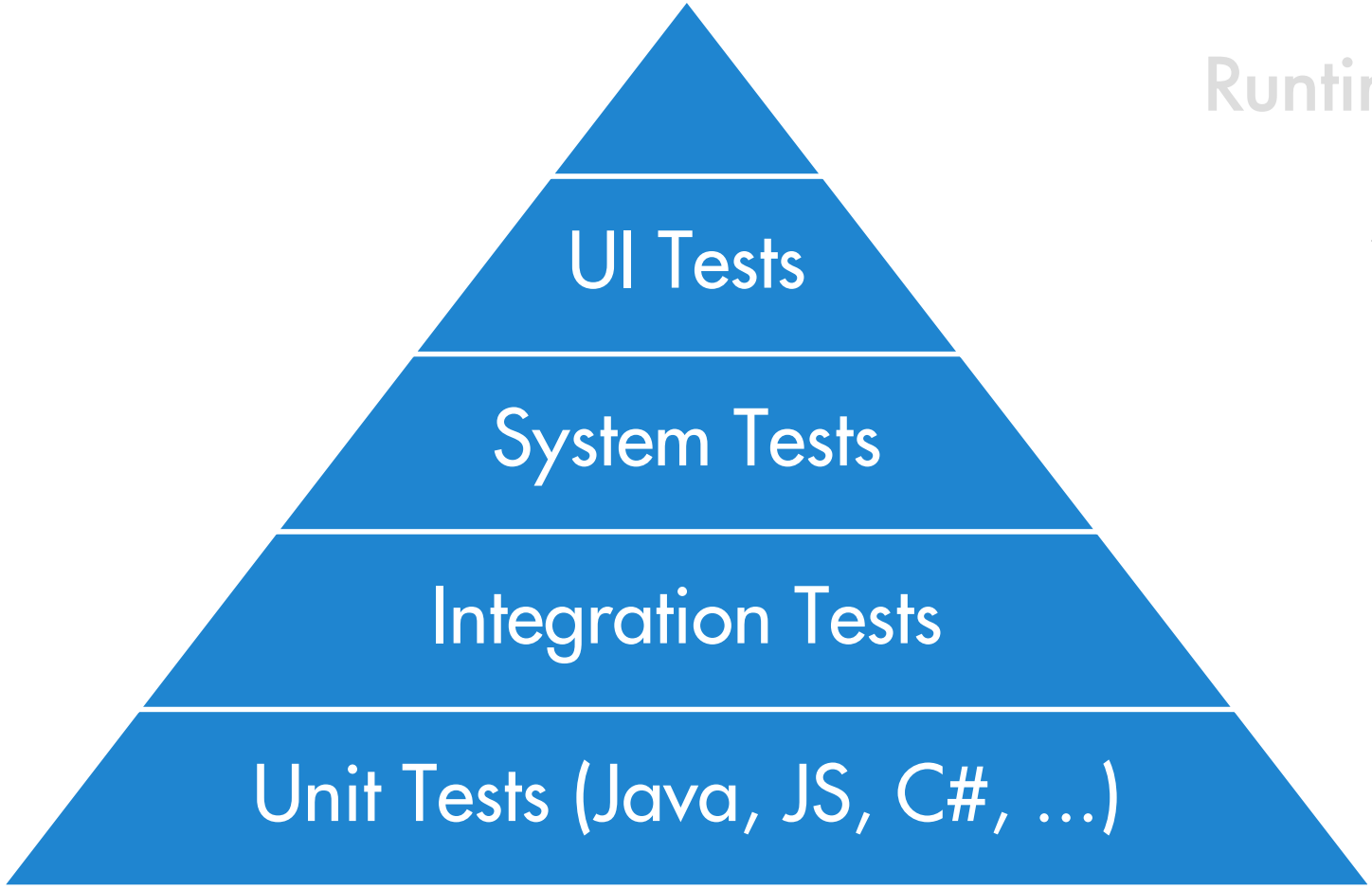
Test-Impact Analysis

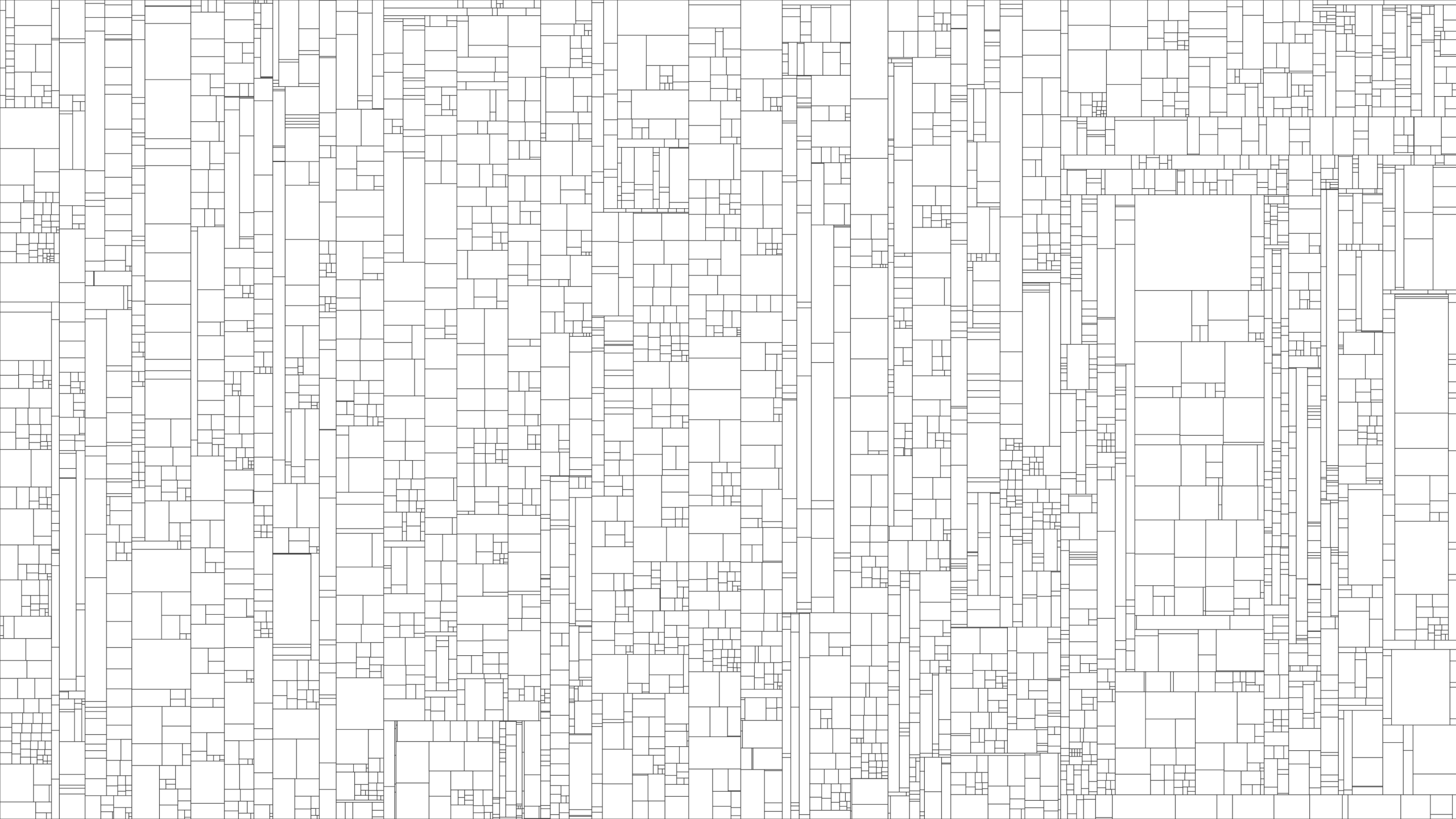
Master Thesis by Jakob Rott

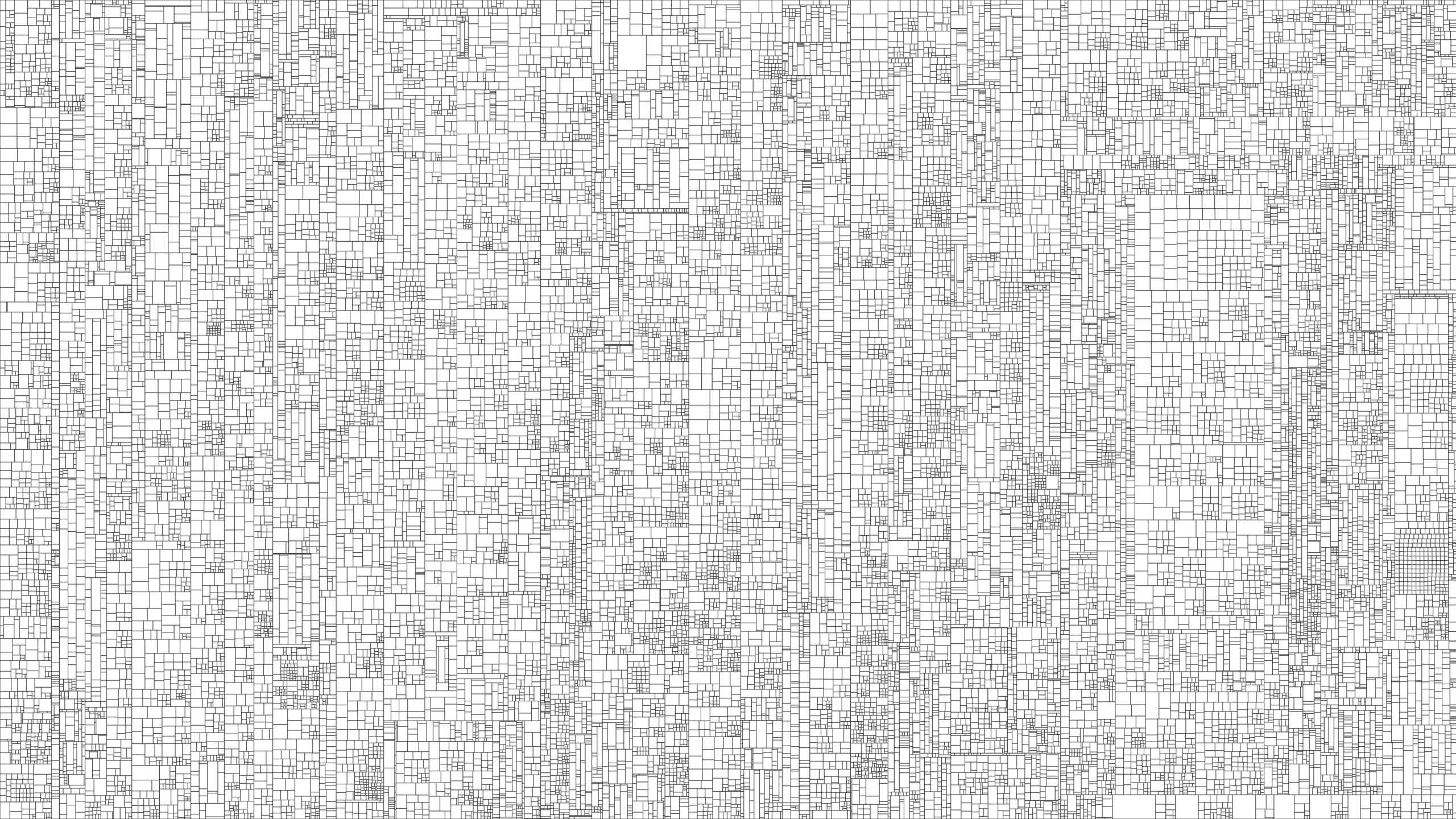
CQSE
Continuous Quality in Software Engineering

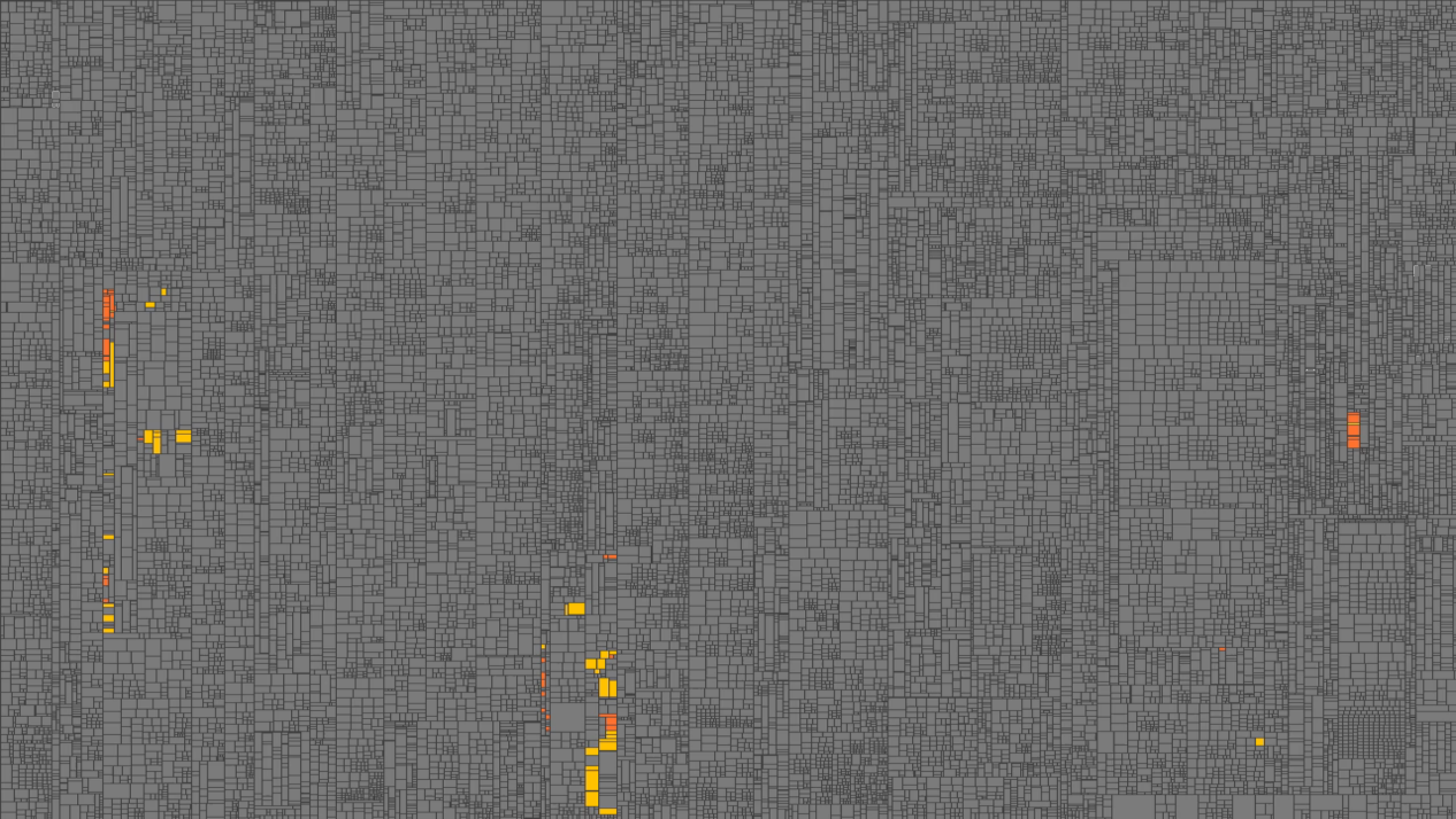
How We Test Teamscale

Number		Runtime [s]
267	UI Tests	1.800
198	System Tests	400
1.273	Integration Tests	370
4.812	Unit Tests (Java, JS, C#, ...)	180

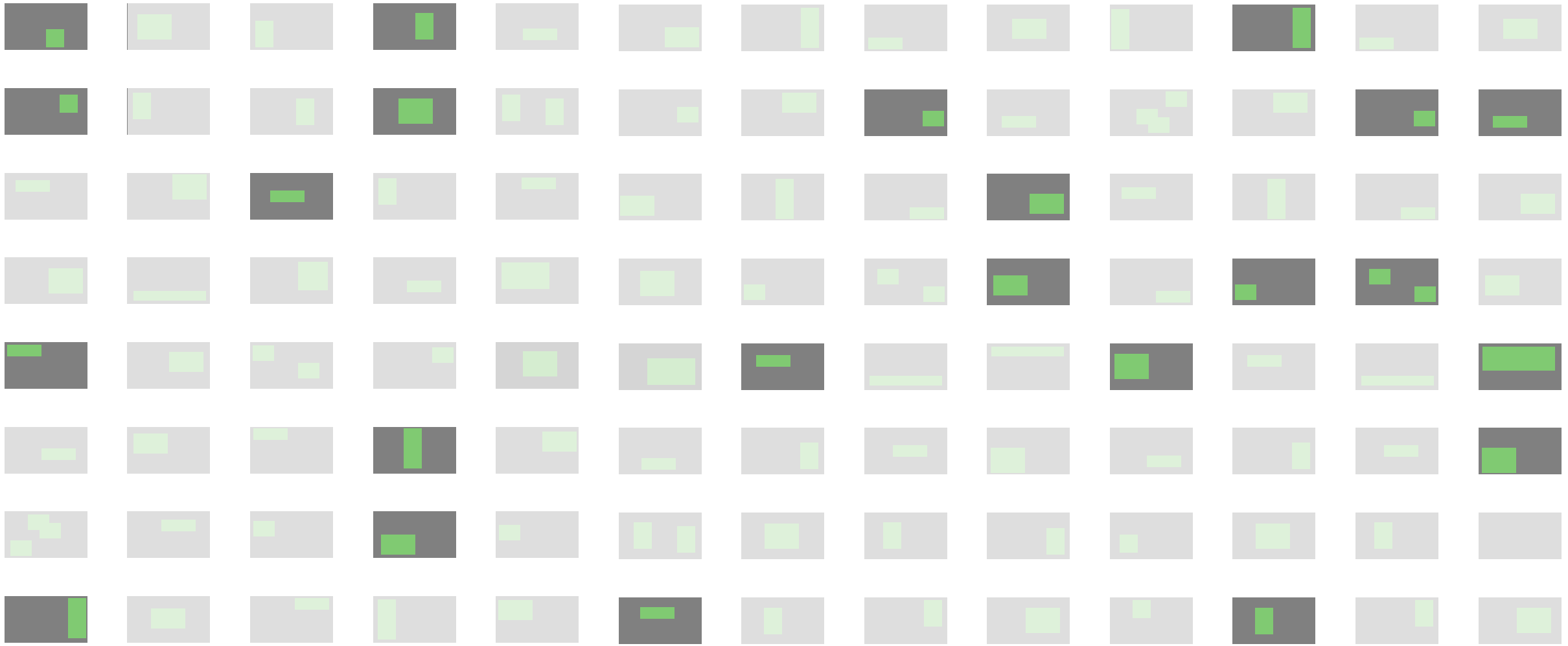




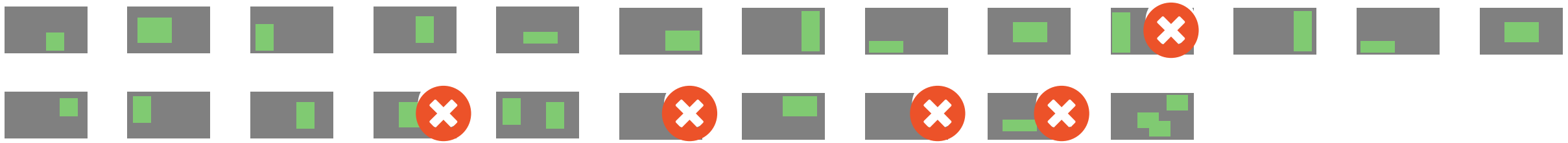




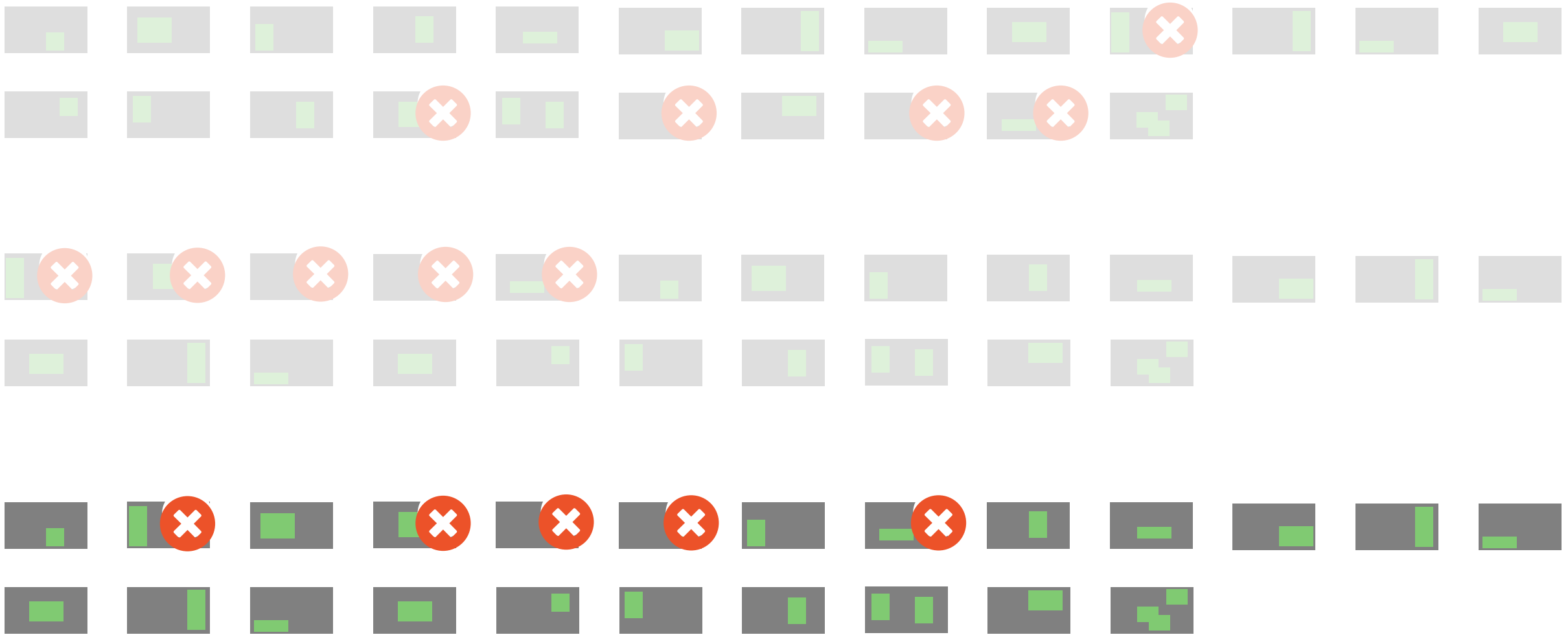
Schritt 1: Selektion betroffener Testfälle

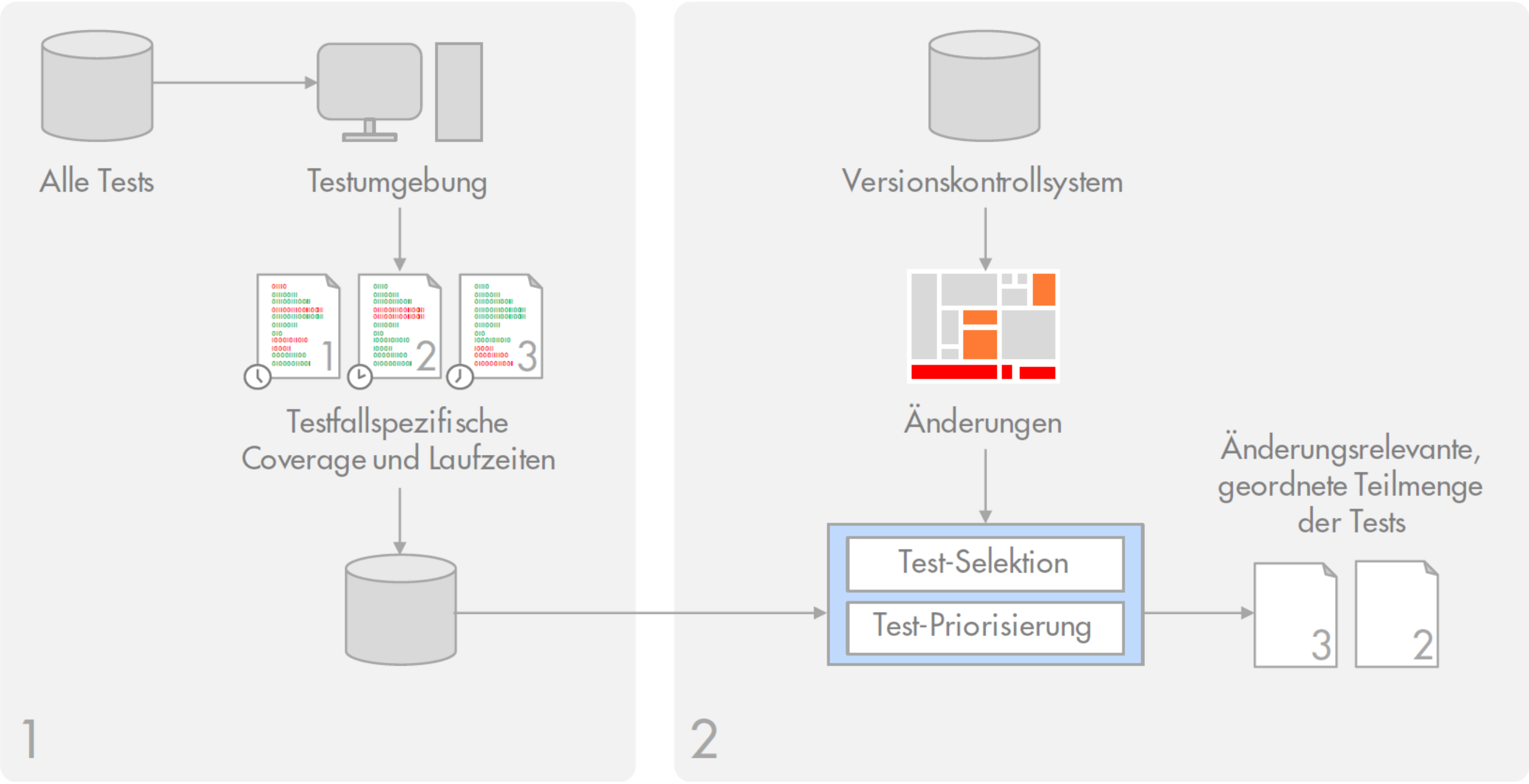


Schritt 2: Priorisierung selektierter Testfälle

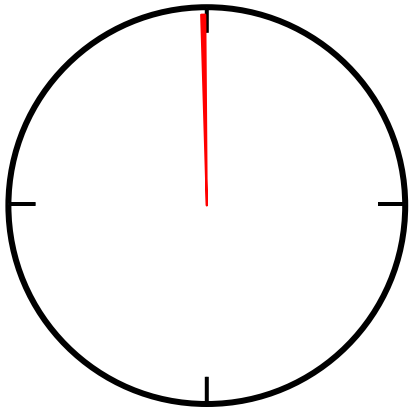


Schritt 2: Priorisierung selektierter Testfälle

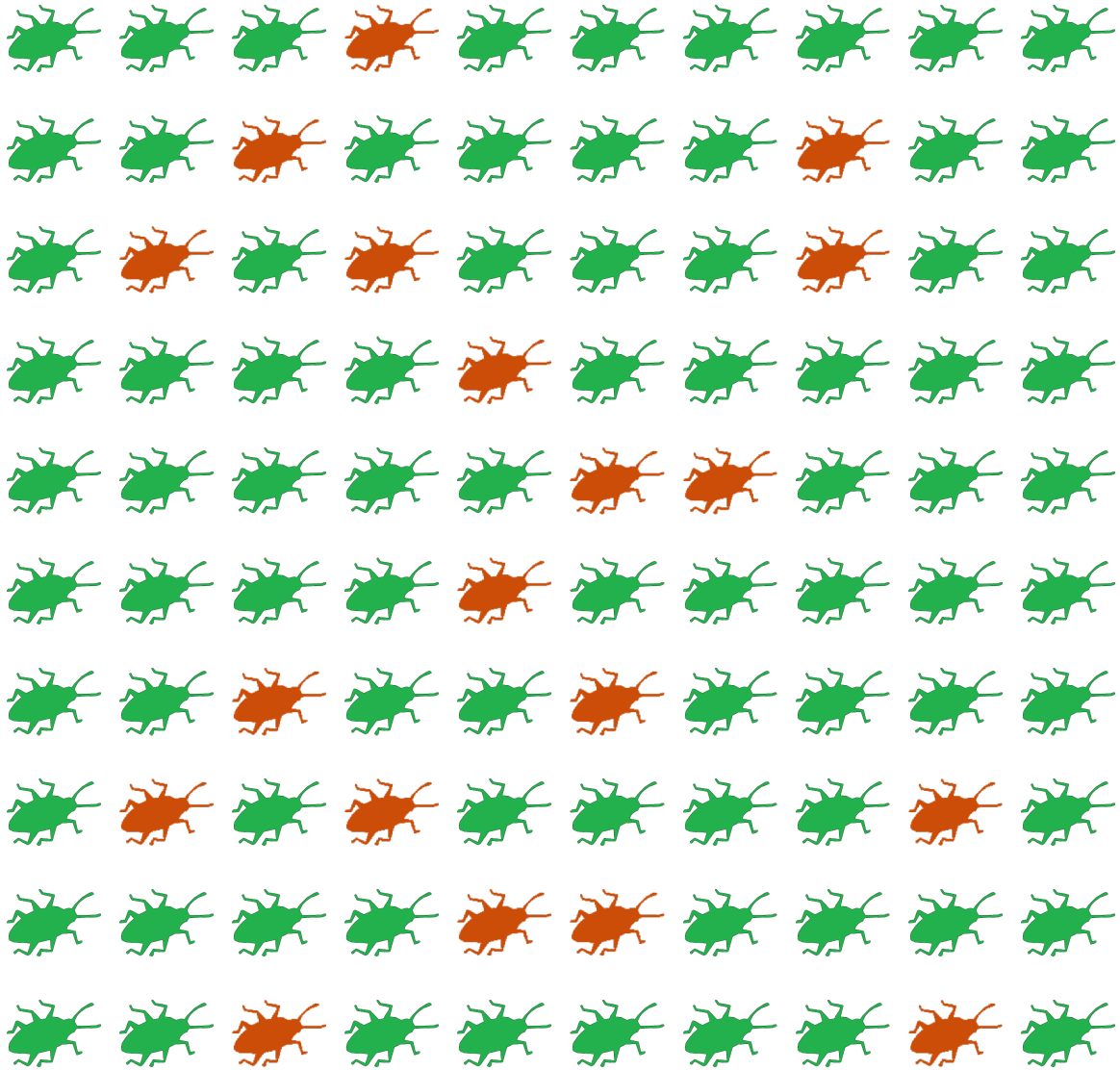



Alle Tests
Testumgebung
1
2
3
Testfallspezifische
Coverage und Laufzeiten
1
Versionskontrollsystem
Änderungen
Test-Selektion
Test-Priorisierung
Änderungsrelevante,
geordnete Teilmenge
der Tests
3
2
2

1%



80%

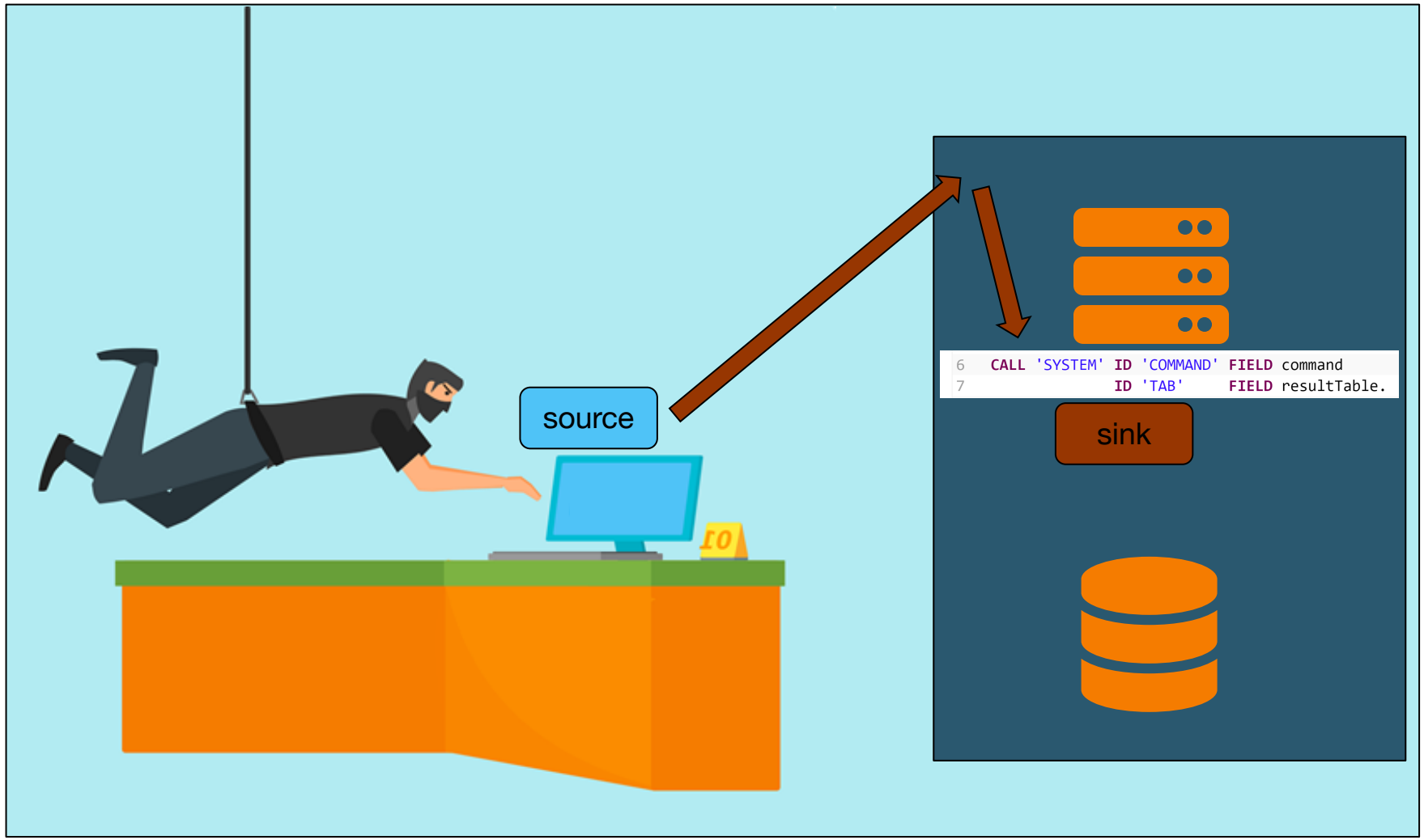


Presenting Findings of Complex Analyses

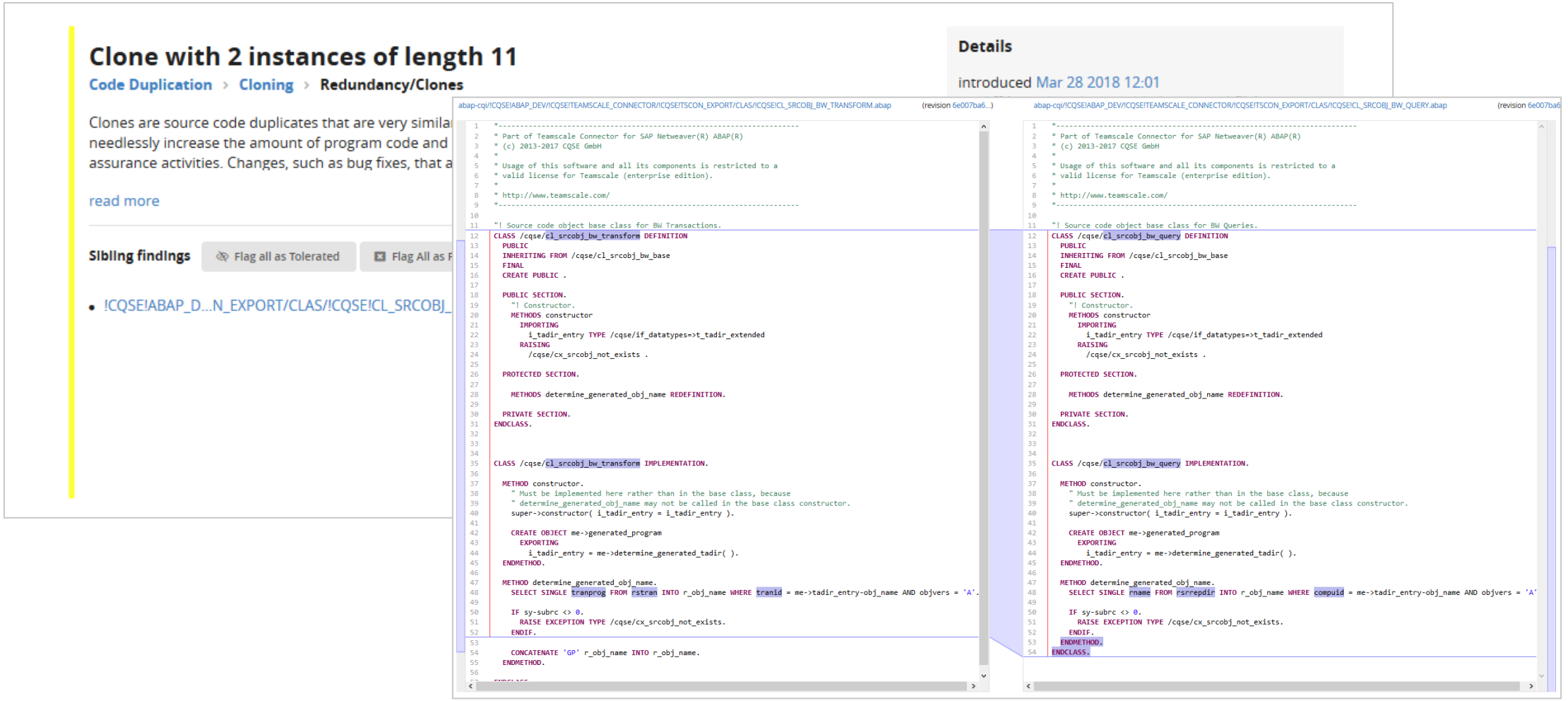
Open Topic

CQSE
Continuous Quality in Software Engineering

Security Threat: System Command Injection



Interpretation of Findings



Interpretation of Findings

Execution of OS command: `filename rea`
Z_CALLER.abap line 31.

taint analysis › Taint Propagation › Dataflow/Taint Propagation ·

A data value is read from user input and passed to a statement that exec
For further information see SAP help page on system command injectior

- Source: PROG/Z_CALLER.abap:3
- Sink: PROG/Z_CALLER.abap:31

in PROG/Z_CALLER.abap

```
1  REPORT  ztestsrc_df_class.
2
3   PARAMETERS:       filename  TYPE String DEFAULT 'default_filename'.
   0 This source taints the variable `filename`
4
5  *---------------------------------------------------------------------*
6  *       CLASS zlcl_main DEFINITION
7  *---------------------------------------------------------------------*
8  *
9  *---------------------------------------------------------------------*
10 CLASS zlcl_main DEFINITION.
11   PUBLIC SECTION.
12     CLASS-METHODS main.
13     CLASS-METHODS foo
14         IMPORTING info TYPE String
15         RETURNING VALUE(returnValue) TYPE String.
16 ENDCLASS.                    "lcl_main DEFINITION
17
18 *---------------------------------------------------------------------*
19 *       CLASS zlcl_main IMPLEMENTATION
20 *---------------------------------------------------------------------*
21 *
22 *---------------------------------------------------------------------*
23 CLASS zlcl_main IMPLEMENTATION.
24
25    METHOD main.
   2 Field `filename` may contain tainted data
26     DATA: secret TYPE string.
27
28     secret = foo( filename ).
   3 Possibly tainted parameter at index 0 can be modified in method call
   6 Possibly tainted return value in assignee `MCR_foo^2_ASSIGNEE_`
29     secret = secret + 'postfix'.
30
31     CALL 'SYSTEM' ID 'ls' FIELD secret.
   7 OS command injection with tainted variable `secret`
32   ENDMETHOD.                    "main
33   METHOD foo.
34     returnValue = 'CLEAN'.
35   ENDMETHOD.                    "main
36 ENDCLASS.                    "lcl_main IMPLEMENTATION
37
38 START-OF-SELECTION.
39    zlcl_main=>main( ).
   1 Possibly tainted field `PROG/Z_CALLER.abap##filename` can be modified in method call
```

Agenda

- 11:40 Introduction
- 11:45 Static Analysis for Quality Improvement
- 12:05 Continuous Quality Improvement
- 12:50 Innovation & Improvement
- **13:10 Q&A**
- 13:20 End of Lecture

Additional Literature

- **»Containing the Findings Flood«** (Blog)
  https://cqse.eu/en/blog/containing-the-findings-flood/
- **»Does Quality Control Really Make Your Code Better?«** (Blog)
  https://cqse.eu/en/blog/effect-of-quality-control/
- **»McCabe's Cyclomatic Complexity and Why We Don't Use It«** (Blog)
  https://cqse.eu/en/blog/mccabe-cyclomatic-complexity/
- **»Haben wir das Richtige getestet?«** und **»Immer kürzere Testphasen?«** (OBJEKTspektrum 2018)
  https://cqse.eu/tga-ticket-coverage-objektspektrum
- **»Fehler früh erkennen trotz großer, langlaufender Test-Suites«** (OBJEKTspektrum 2018)
  https://cqse.eu/test-impact-analyse-objektspektrum
- **»Teamscale – The Official User Reference«**
  https://cqse.eu/userguide

Contact

Dr. Sven Amann · amann@cqse.eu · +49 172 1860063

CQSE GmbH
Hilpertstraße 31 (HUB31)
64289 Darmstadt

CQSE
Continuous Quality in Software Engineering