A4Q WORLD CONGRESS

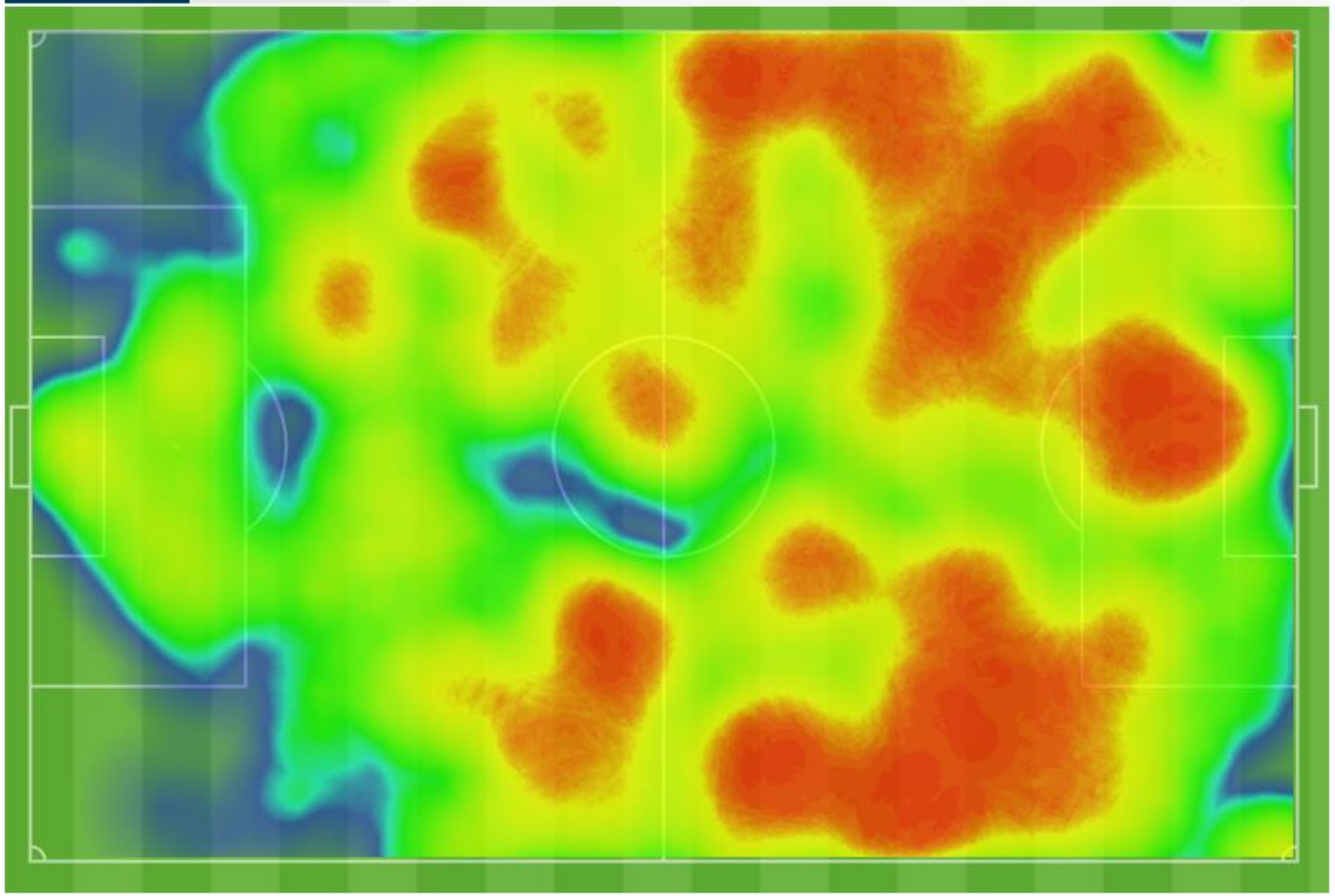
Test Intelligence

Lücken schließen und überflüssige Tests entfernen, um mehr Fehler in weniger Zeit zu finden

Dr. Elmar Juergens

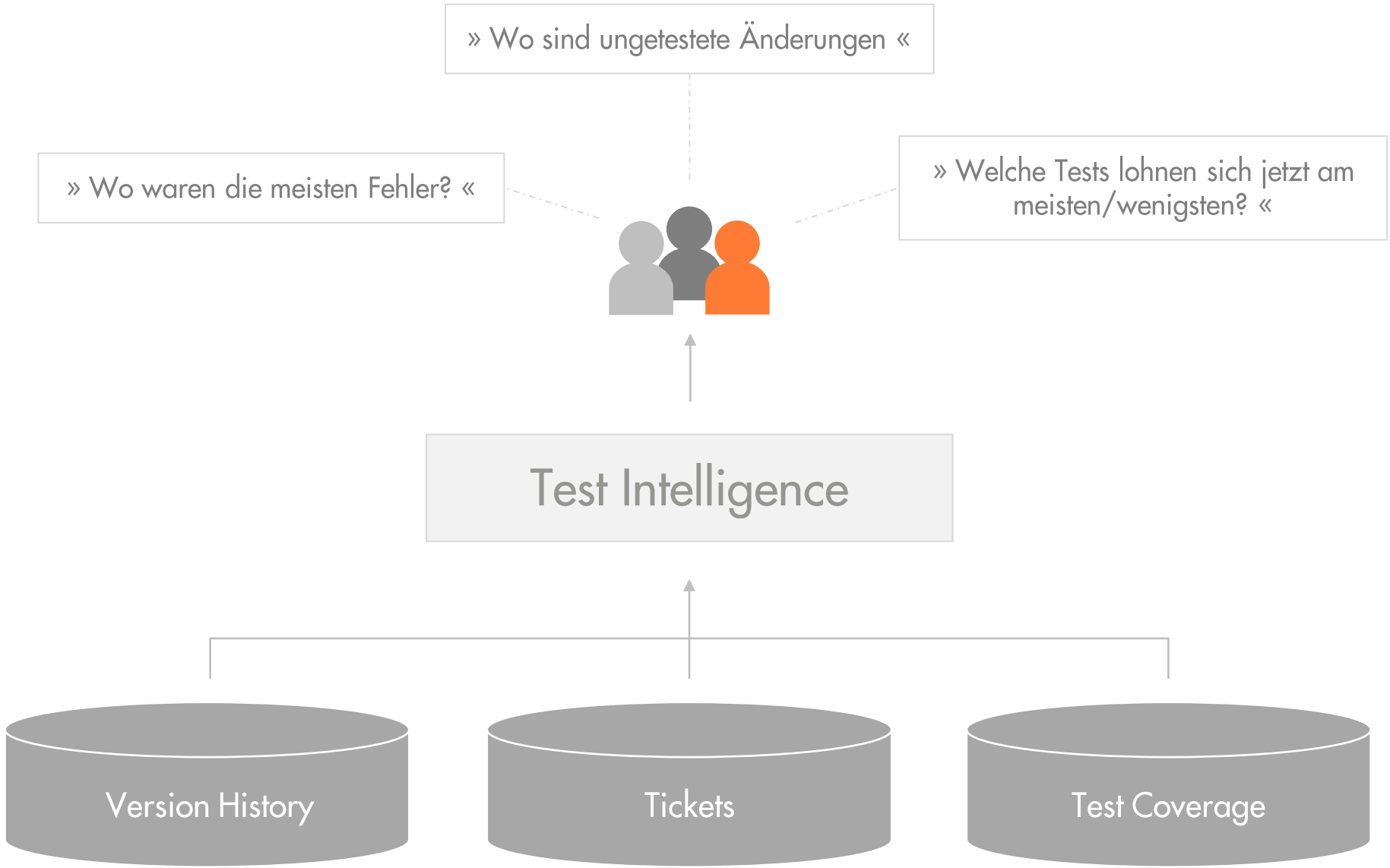
FC Bayern München		
FC Bayern München		
TW	1	Manuel Neuer
AW	21	Lucas Hernández
AW	5	Benjamin Pavard
AW	4	Niklas Süle
AW	27	David Alaba
MF	29	Kingsley Coman
MF	10	Leroy Sané
MF	25	Thomas Müller
MF	6	Joshua Kimmich
MF	18	Leon Goretzka
ST	13	Eric Maxim Choupo-Moting
Einwechselspieler		
17		Jérôme Boateng
19		Alphonso Davies

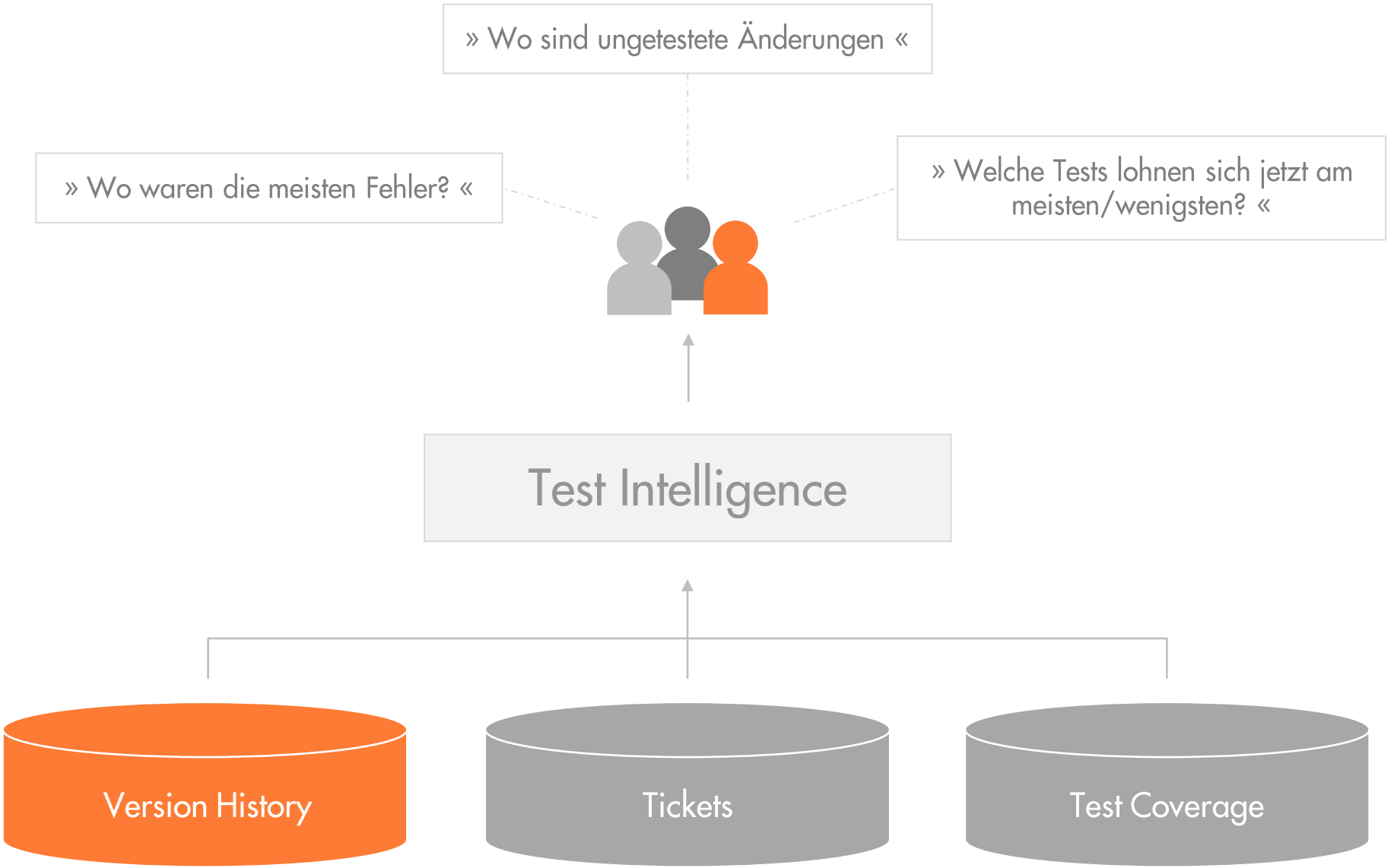
Heatmap | Touchmap



Paris Saint-Germain		
Paris Saint-Germain		
TW	1	Keylor Navas
AW	5	Marquinhos
AW	31	Colin Dagba
AW	22	Abdou Diallo
AW	3	Presnel Kimpembe
MF	23	Julian Draxler
MF	27	Idrissa Gueye
MF	10	Neymar
MF	15	Danilo Pereira
MF	11	Ángel Di María
ST	7	Kylian Mbappé
Einwechselspieler		
25		Mitchel Bakker
18		Moise Kean
21		Ander Herrera
12		Rafinha

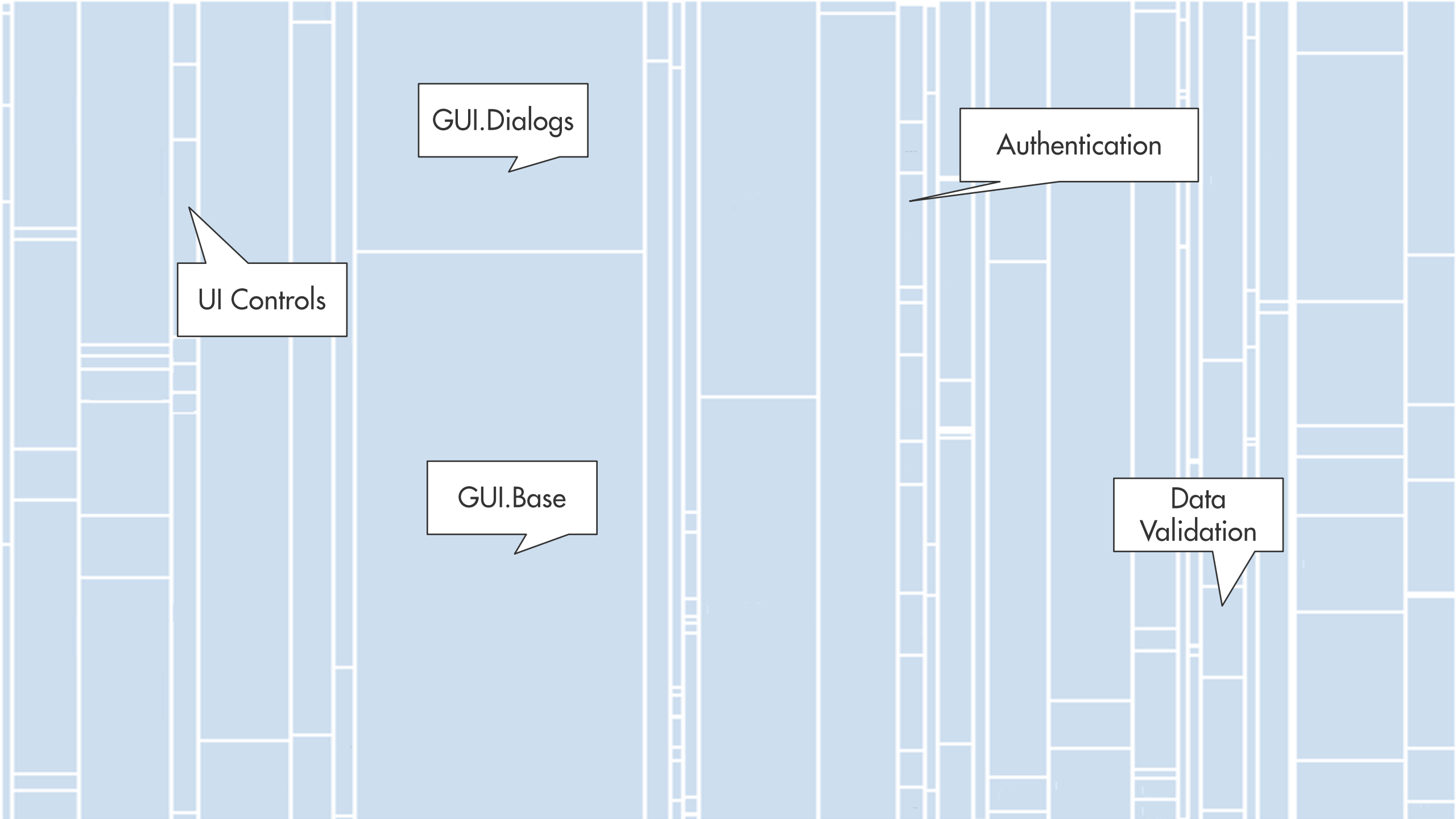
https://sportdaten.welt.de/fussball/champions-league/bayern-muenchen/paris-st-germain/2021-04-07/clvfh/2021/142784/heatmap

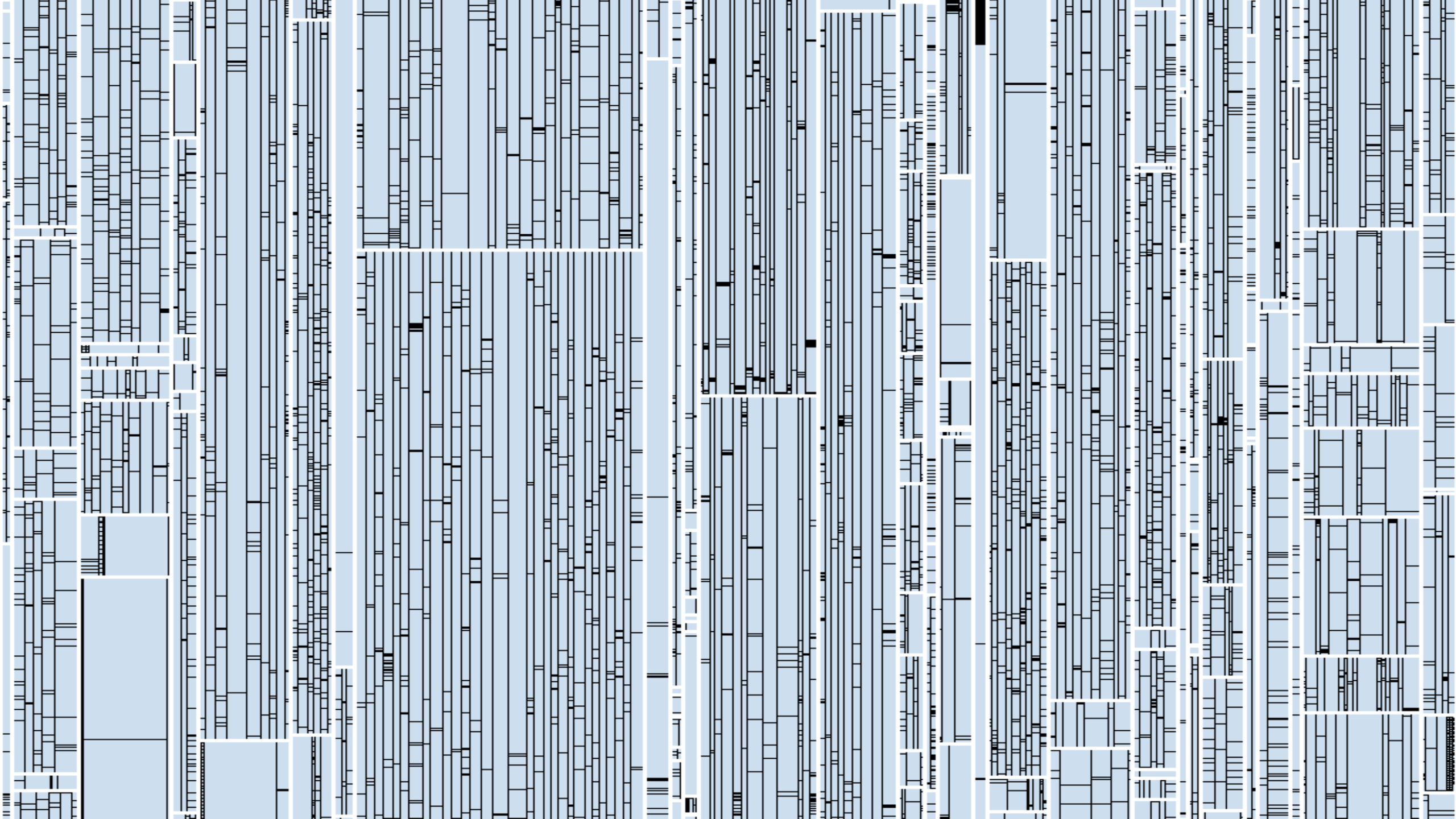

» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage

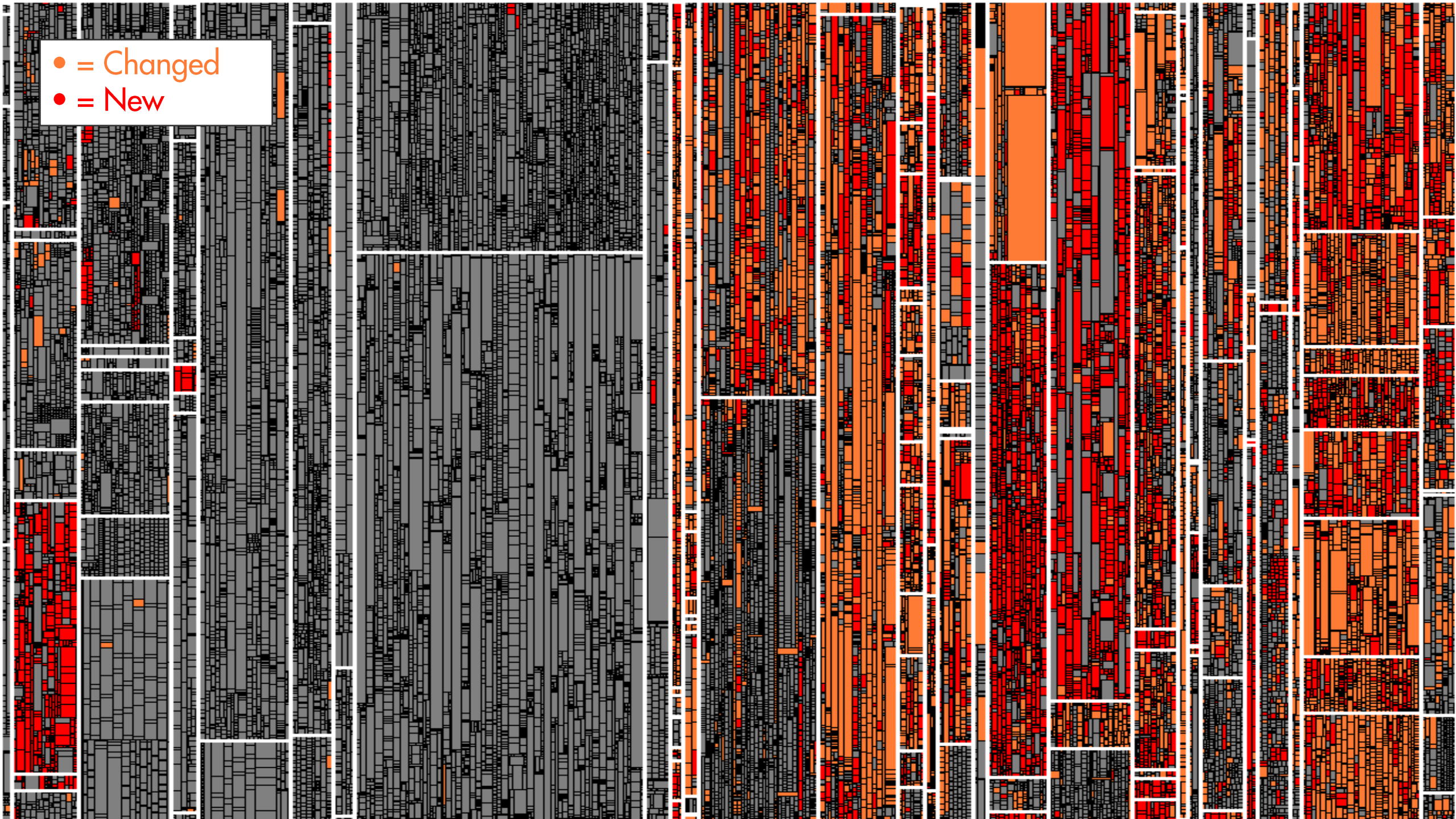

» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage

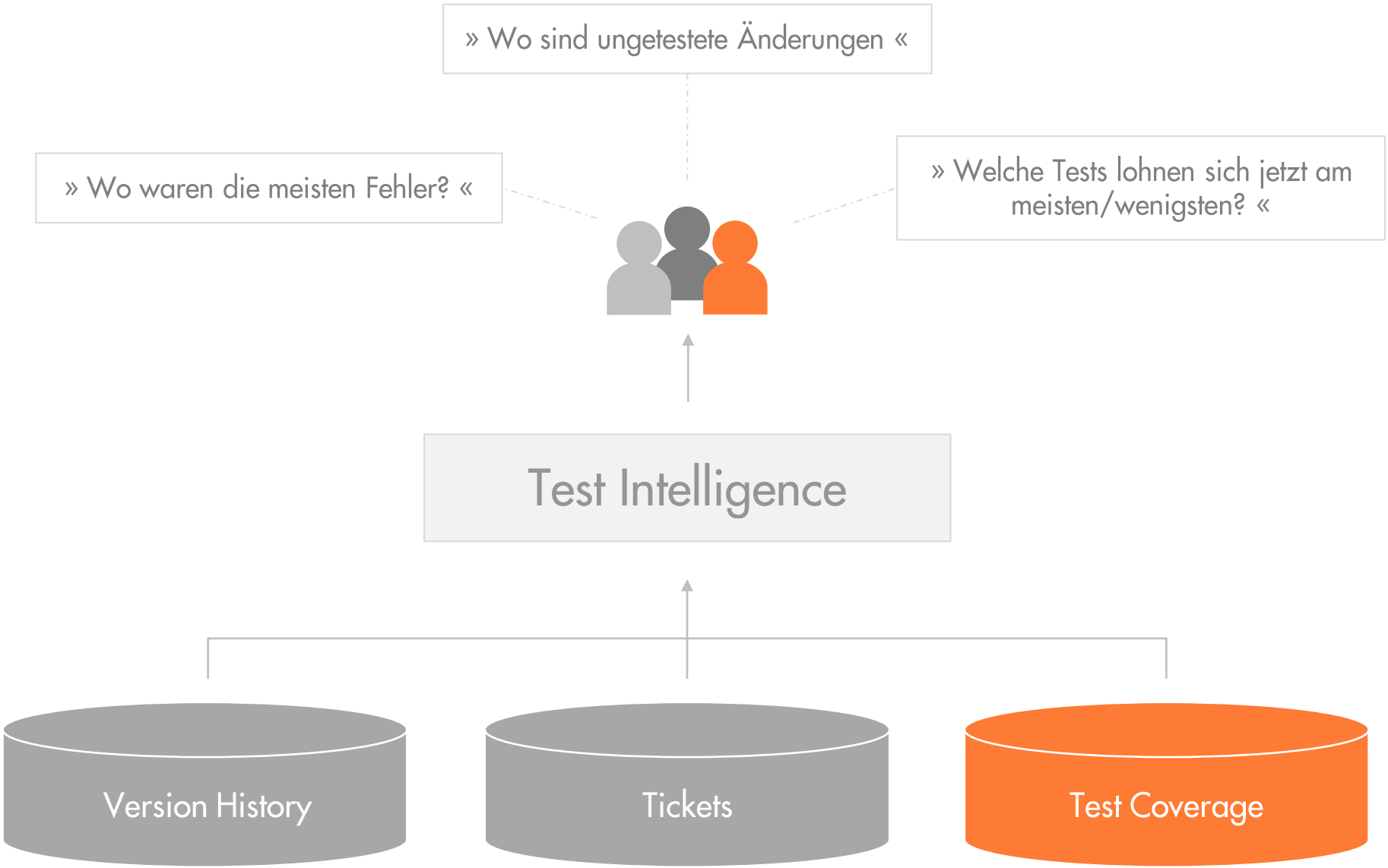
```
/* package */class NamespaceRenames {

    /** Maps from old name fragment to new name fragment */
    public Set<ImmutablePair<String, String>> namespaceRenames = new HashSet<

    /**
     * Computes a rename rule from an old and a new namespace name based on a
     * type name correspondence. <code>
     * For name1 = a.oldnamespace.b.c.D and name2 = a.newnamespace.b.c.D, res
     * </code>
     * */
    public static ImmutablePair<String, String> computeRenameRule(
            String fqTypeName1, String fqTypeName2, String separator) {

        if (fqTypeName1.equals(fqTypeName2)) {
            return null;
        }

        String commonSuffix = StringUtils.longestCommonSuffix(fqTypeName1,
                fqTypeName2);
        if (StringUtils.isEmpty(commonSuffix)) {
            return null;
        }

        int separatorPosition = commonSuffix.indexOf(separator);
        if (separatorPosition != -1) {
            commonSuffix = commonSuffix.substring(separatorPosition);
        }

        String from = StringUtils.stripSuffix(commonSuffix, fqTypeName1);
        String to = StringUtils.stripSuffix(commonSuffix, fqTypeName2);

        return new ImmutablePair<String, String>(from, to);
    }
```

```
/* package */class NamespaceRenames {

    /** Maps from old name fragment to new name fragment */
    public Set<ImmutablePair<String, String>> namespaceRenames = new Has

    // TODO (LH) Doc vs. method name : 'Compute' vs. 'find'
    /**
     * Computes a rename rule from an old and a new namespace name based
     * type name correspondence. <code>
     * For name1 = a.oldnamespace.b.c.D and name2 = a.newnamespace.b.c.D
     * </code>
     * */
    // TODO (LH) PLease reflect in identifier names that they refert to
    // names
    public static ImmutablePair<String, String> findRenameRule(String na
            String name2, String separator) {

        if (name1.equals(name2)) {
            return null;
        }

        // TODO (LH) Looks like StringUtils is missing a 'longestCommons
        String commonSuffix = reverse(StringUtils.longestCommonPrefix(
                reverse(name1), reverse(name2)));
        if (StringUtils.isEmpty(commonSuffix)) {
            return null;
        }

        int separatorPosition = commonSuffix.indexOf(separator);
        if (separatorPosition != -1) {
            commonSuffix = commonSuffix.substring(separatorPosition);
        }

        // TODO (LH) Please reflect in identifier names that these are
        // namespaces
        String from = StringUtils.stripSuffix(commonSuffix, name1);
        String to = StringUtils.stripSuffix(commonSuffix, name2);

        return new ImmutablePair<String, String>(from, to);
    }
```

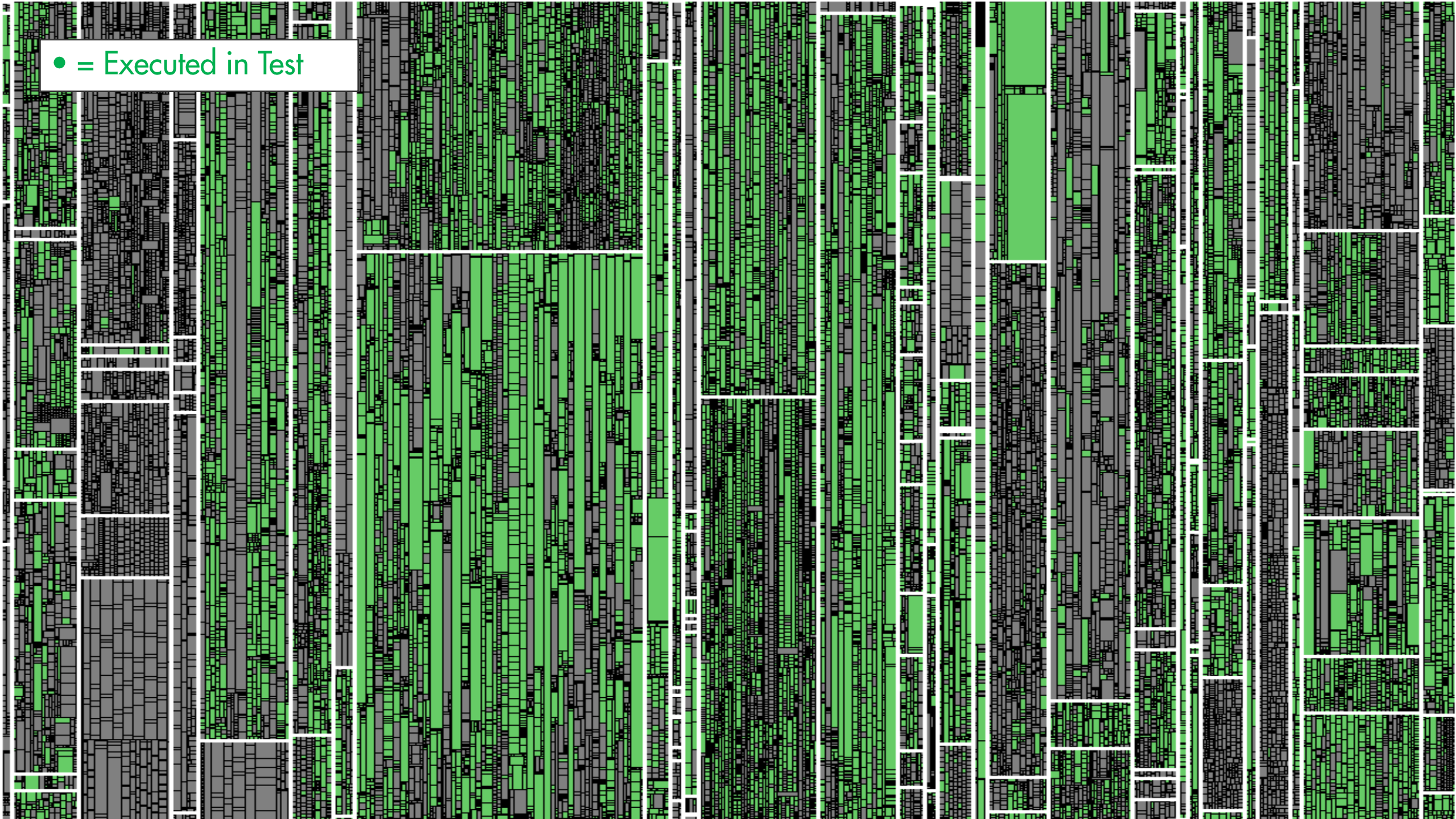

GUI.Dialogs
Authentication
UI Controls
GUI.Base
Data Validation

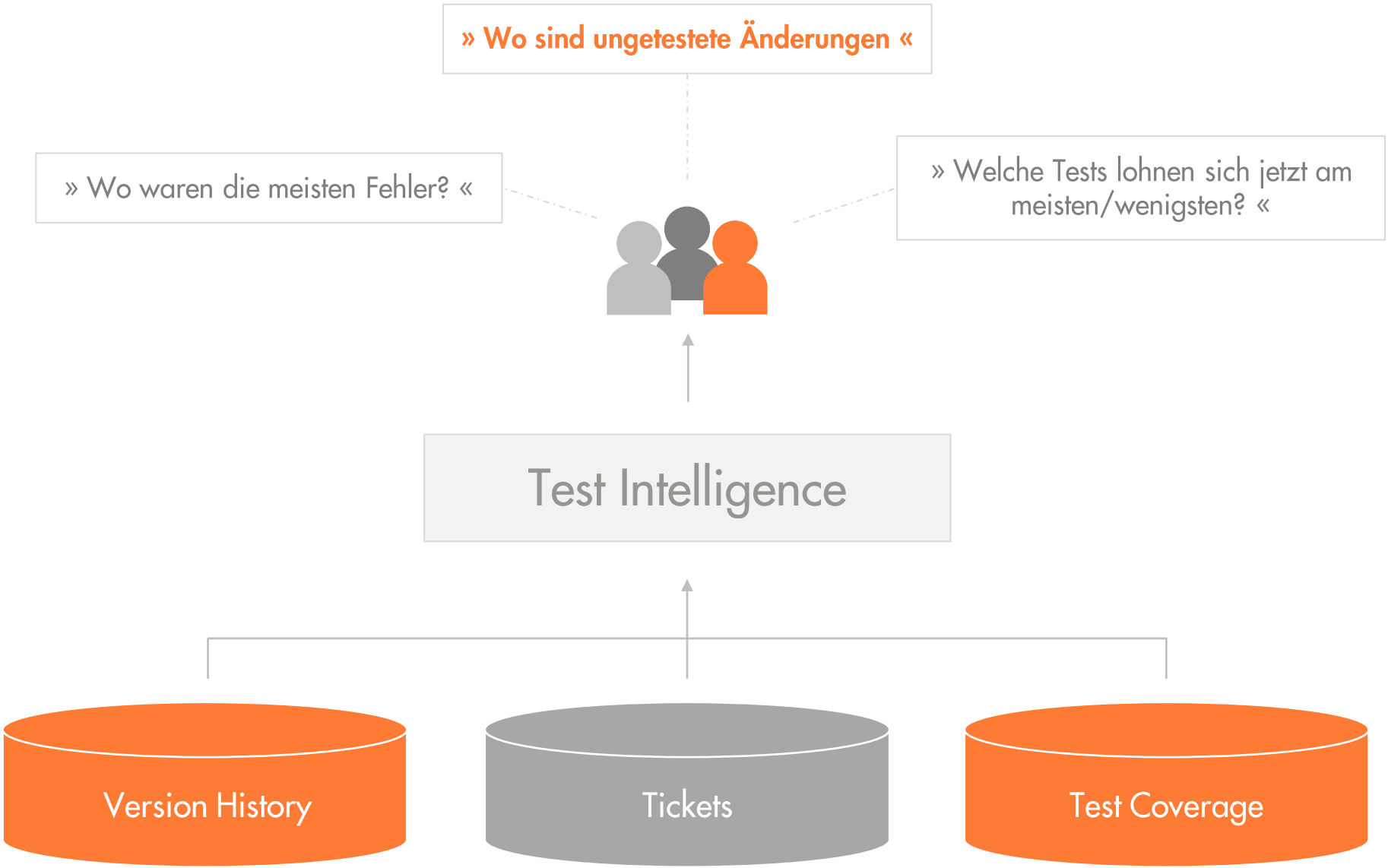


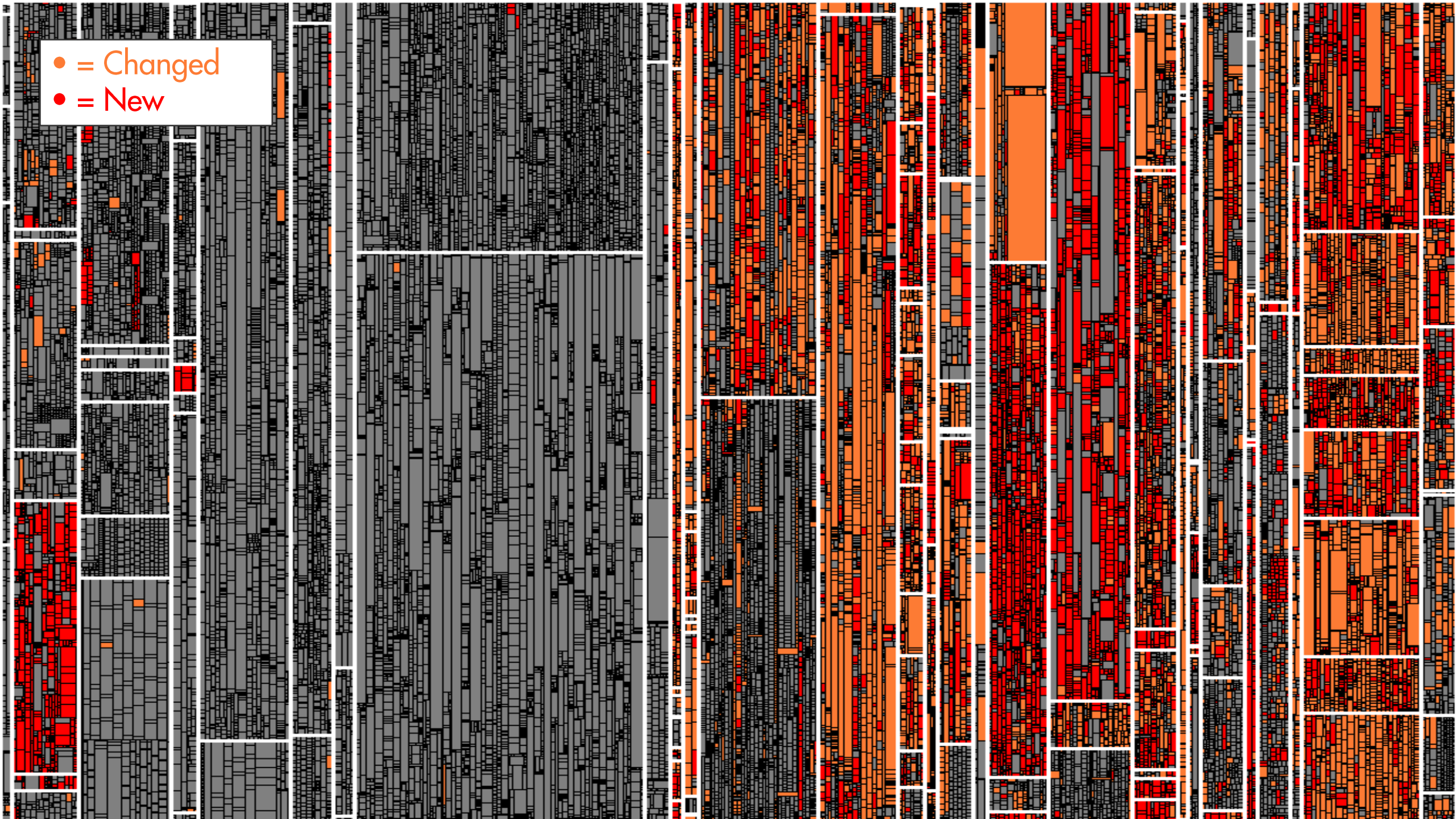

• = Changed
• = New

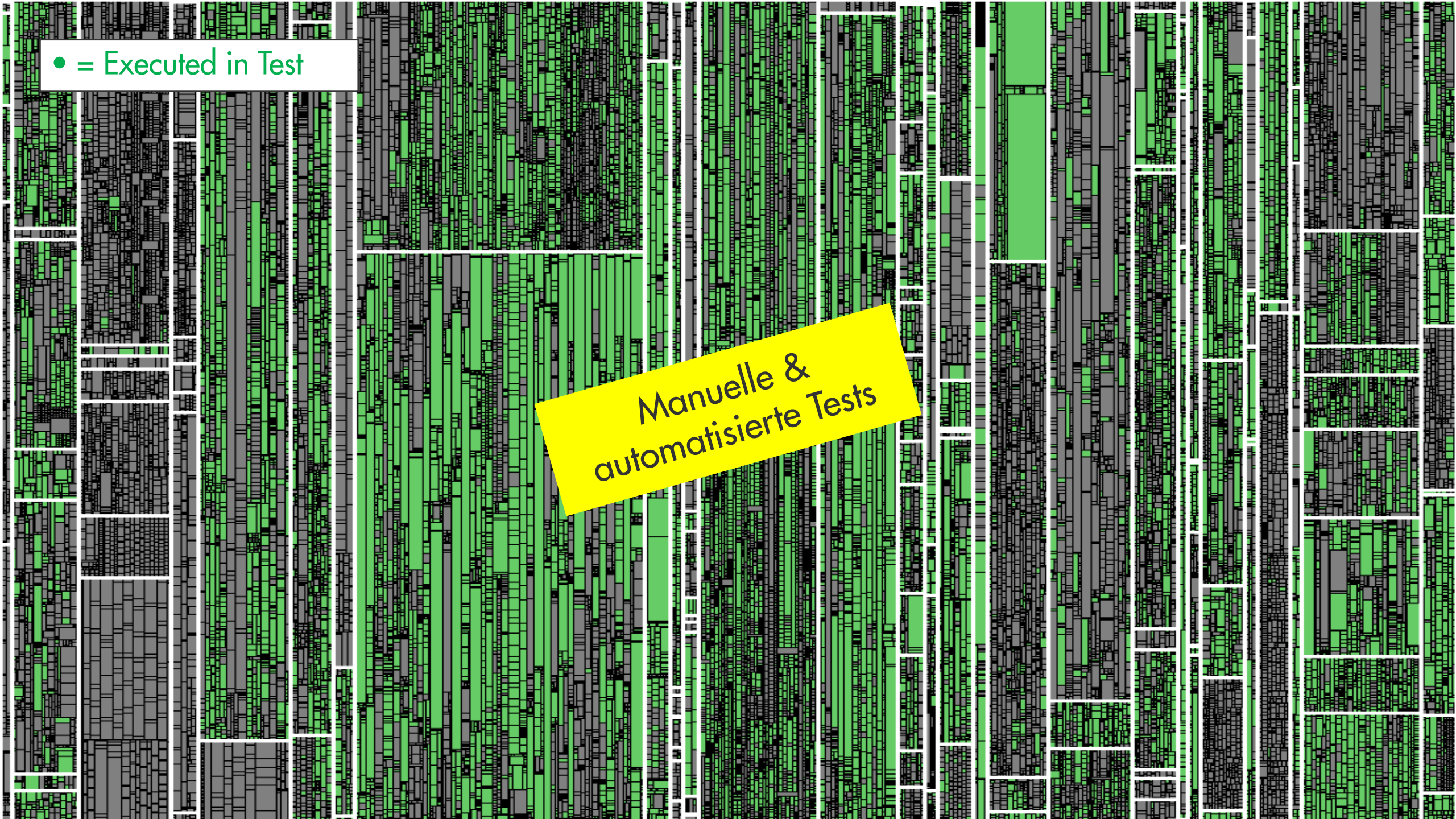

» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage

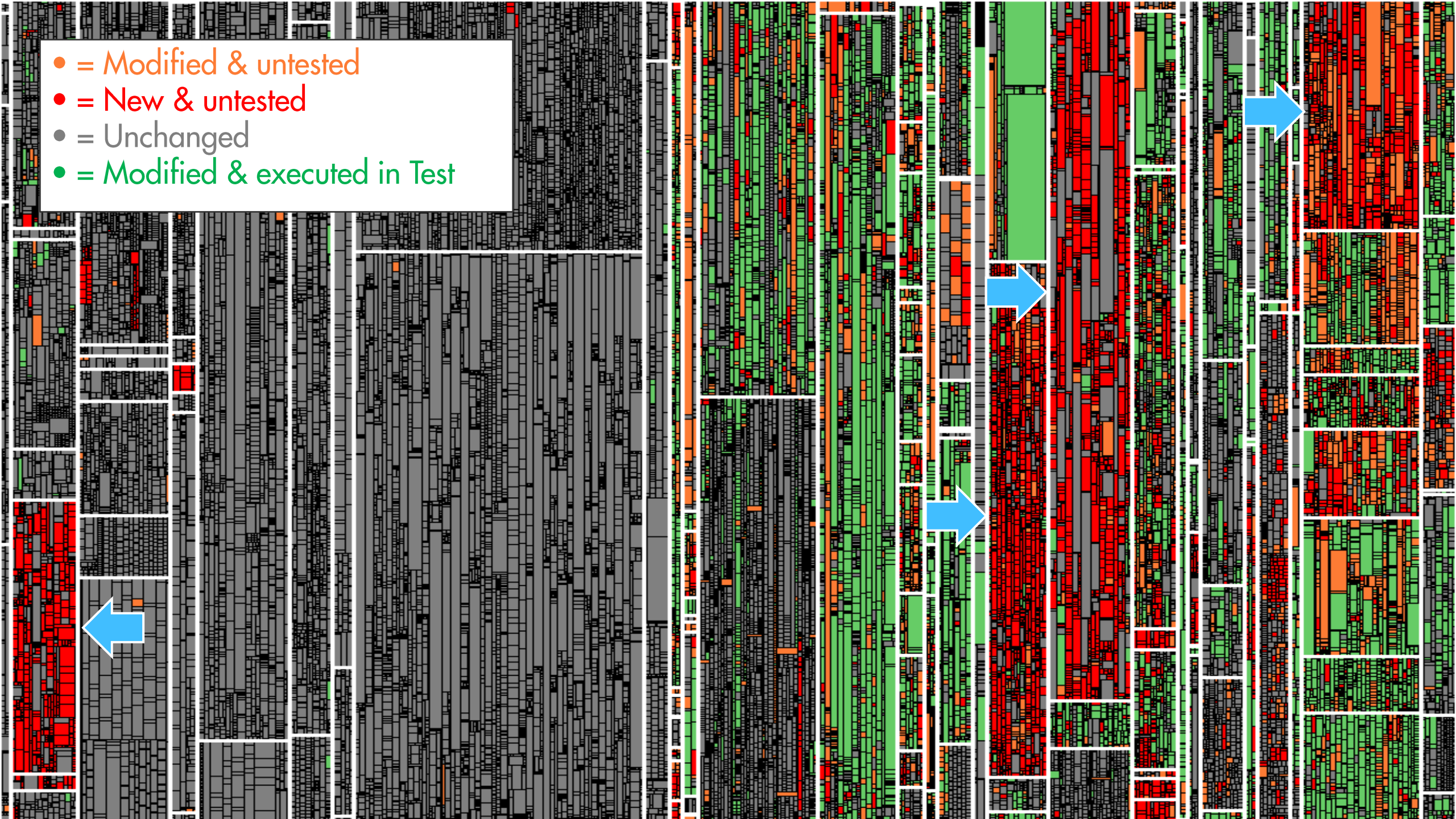
```
protected void calculateIndirectAmbiguities() {
    Map<NucleotideCompound, List<NucleotideCompound>> equivalentsMap = new HashMap<>>();

    List<NucleotideCompound> ambiguousCompounds = new ArrayList<NucleotideCompound>();
    for (NucleotideCompound compound : getAllCompounds()) {
        if (!compound.isAmbiguous()) {
            continue;
        }
        ambiguousCompounds.add(compound);
    }

    for (NucleotideCompound sourceCompound : ambiguousCompounds) {
        Set<NucleotideCompound> compoundConstituents = sourceCompound.getConstituents();
        for (NucleotideCompound targetCompound : ambiguousCompounds) {
            Set<NucleotideCompound> targetConstituents = targetCompound.getConstituents();
            if (targetConstituents.containsAll(compoundConstituents)) {
                NucleotideCompound lcSourceCompound = toLowerCase(sourceCompound);
                NucleotideCompound lcTargetCompound = toLowerCase(targetCompound);
                checkAdd(equivalentsMap, sourceCompound, targetCompound);
                checkAdd(equivalentsMap, sourceCompound, lcTargetCompound);
                checkAdd(equivalentsMap, targetCompound, sourceCompound);
                checkAdd(equivalentsMap, lcTargetCompound, sourceCompound);
                checkAdd(equivalentsMap, lcSourceCompound, targetCompound);
                checkAdd(equivalentsMap, lcSourceCompound, lcTargetCompound);
            }
        }
    }

    for (NucleotideCompound key : equivalentsMap.keySet()) {
        List<NucleotideCompound> vals = equivalentsMap.get(key);
        for (NucleotideCompound value : vals) {
            addEquivalent((C) key, (C) value);
            addEquivalent((C) value, (C) key);
        }
    }
}
```

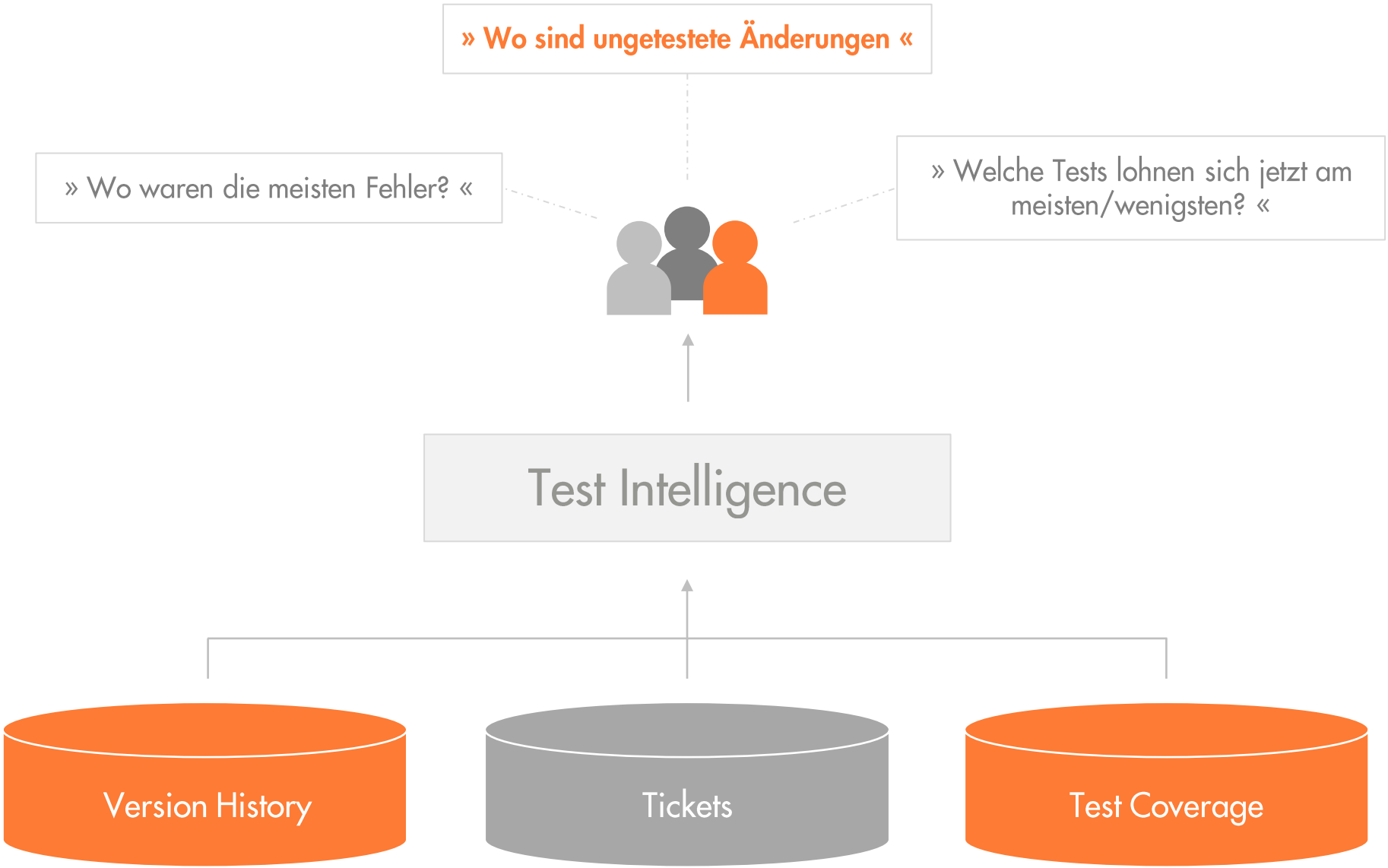

• = Executed in Test

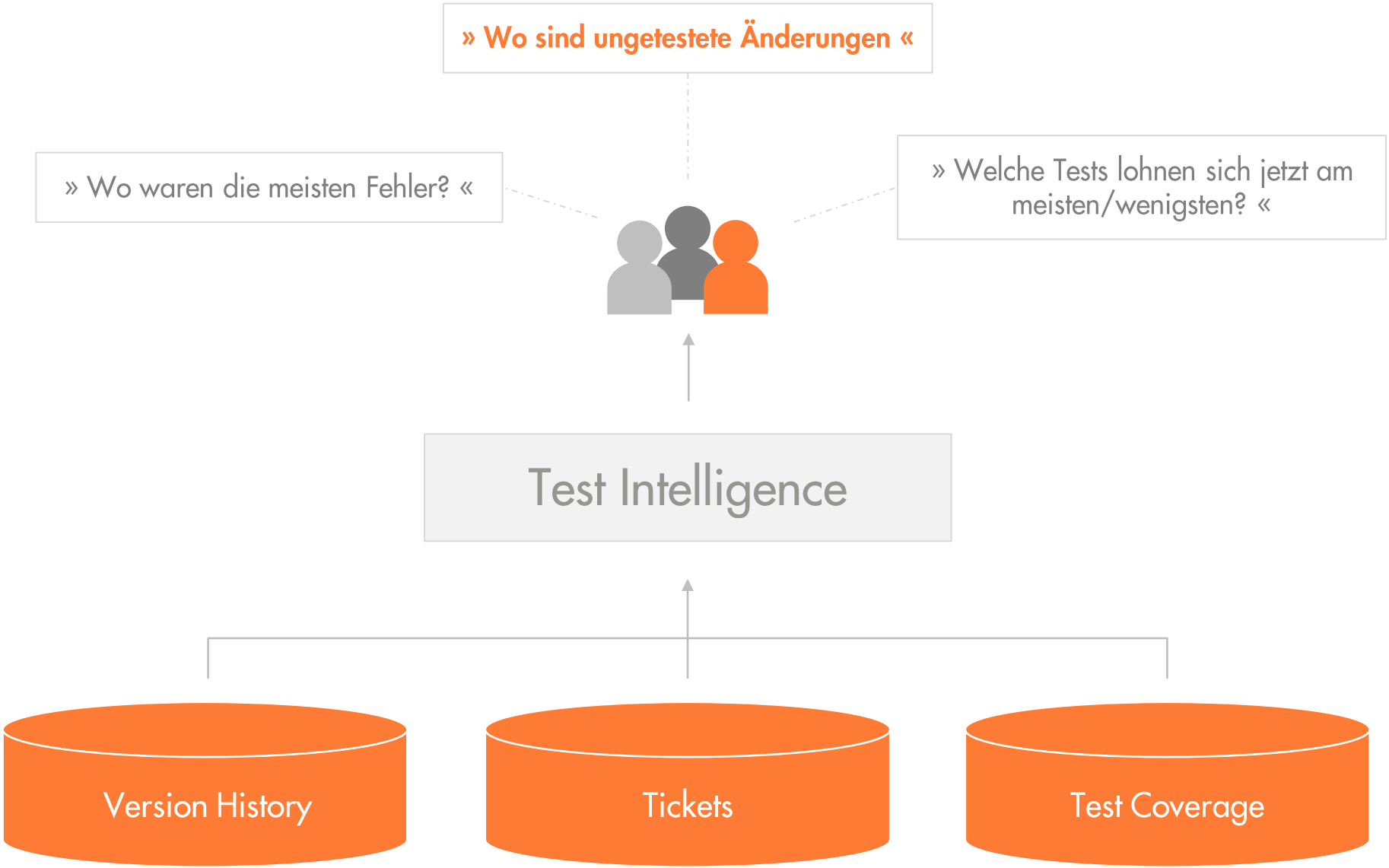

» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage


• = Changed
• = New


• = Executed in Test
Manuelle &
automatisierte Tests


• = Modified & untested
• = New & untested
• = Unchanged
• = Modified & executed in Test


» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage


» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage

Issue #	Subject			Test Gap
TS-10549	Undo/Redo for web-based architecture editor	Done		0%
TS-10784	Fix long method finding in TaintAnalysisRunner	Done		0%
TS-10923	Implement metric 'Nesting Depth' for Simulink	Done		29%
TS-11364	External findings are not registered during first upload	Done		14%
TS-11942	Manual test coverage upload during development	Done		43%
TS-12050	Tool for transferring findings blacklists and tasks	Done		50%
TS-12262	Cannot set or alter alias without reanalysis	Done		0%
TS-13151	Fetch parent relationship of TFS work items	Done		0%

Issue #	Subject		Test Gap
TS-14421	Get rid of TestGapSynchronizer block	Done	0%
TS-14733	Remove Dataflow blocks	Done	22%

Done **Issue TS-14733 - Remove Dataflow blocks**

Creator: (on Apr 06 2018 19:44)

Last update: Aug 24 2018 09:32

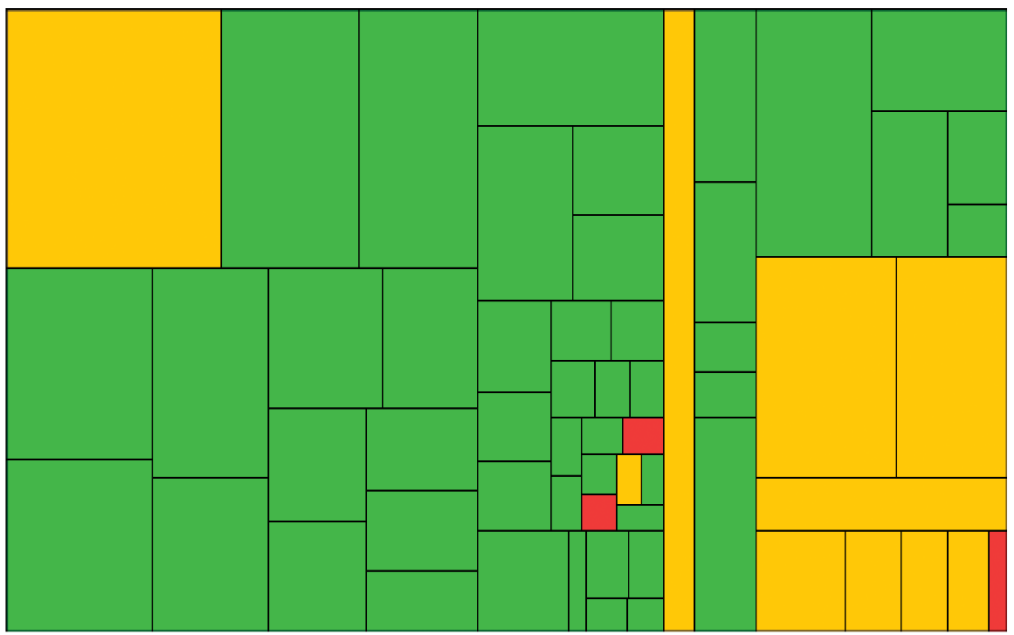
Assignee:

Project	Type	Priority	Resolution	Fix Version
TS	Maintenance	Normal	Green	Teamscale 4.5

Component	Labels	Affected Version	Customer	Customer Issue
Backend	Performance			

Epic Name	Freshdesk URL	Merge Request
		https://git.cqse.eu/cqse/teamscale/3621

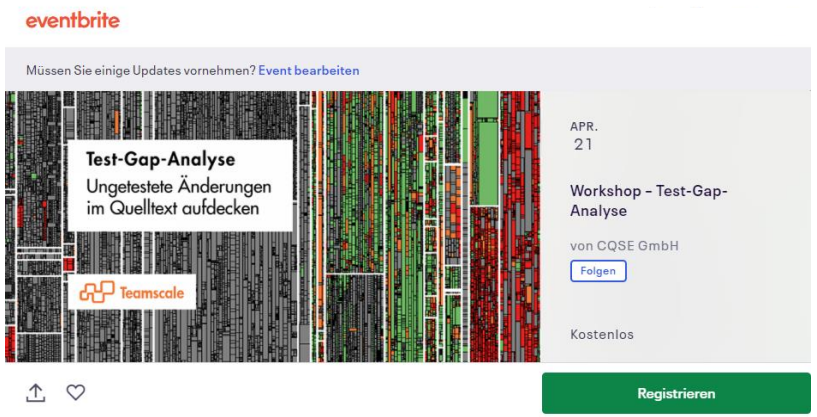
Aug 15 2018 12:37–Now | Test Gap: 22%



Virtueller Workshop

21. April 21, 10:30 – 12:00 Uhr

Anmeldung:
http://cqse.eu/tga-workshop-gtb



Ablauf

- Test-Gap-Analyse (TGA) in Forschung und Praxis
- Live-Demonstration
- Nutzen-Analyse zum Einsatz bei der Munich Re

Vortragende

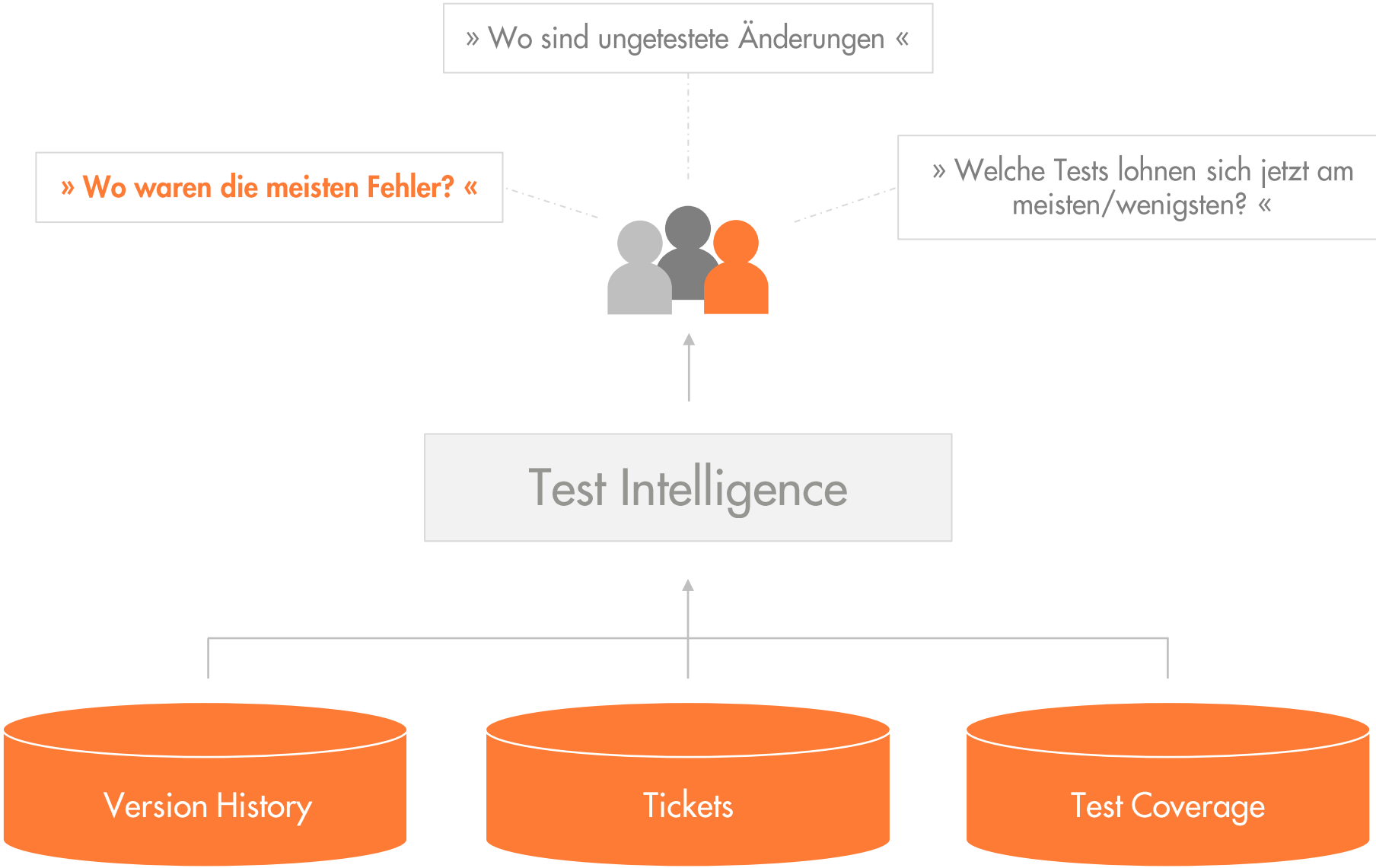
Dr. ***Elmar*** *Jürgens*. Mitgründer der CQSE und beschäftigt sich sein ganzes Berufsleben mit Qualitätsanalysen von Software.

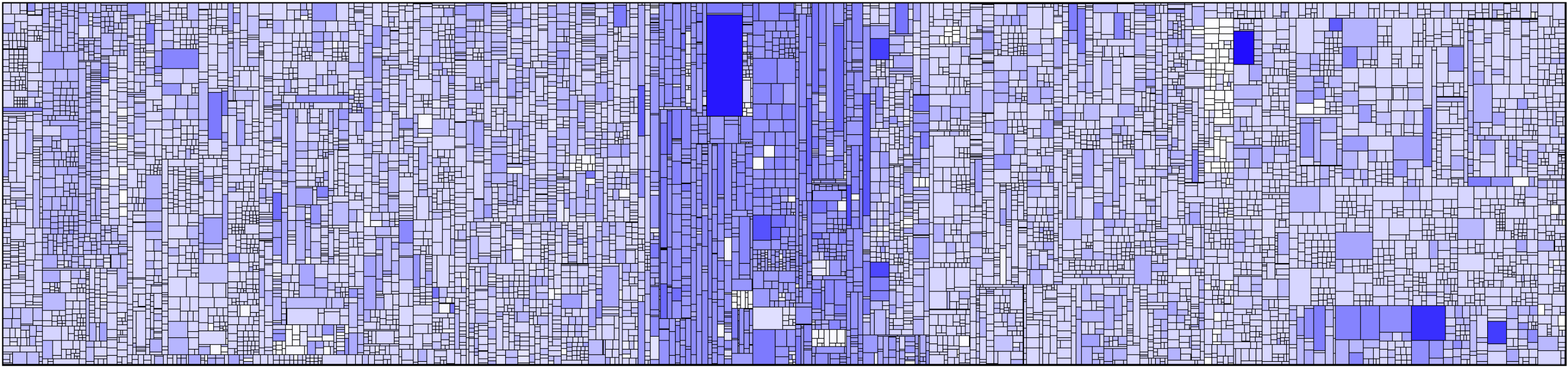
Fabian *Streitel*. Hat als Leiter des Piloten-Teams die TGA bei vielen Kunden erfolgreich eingeführt.

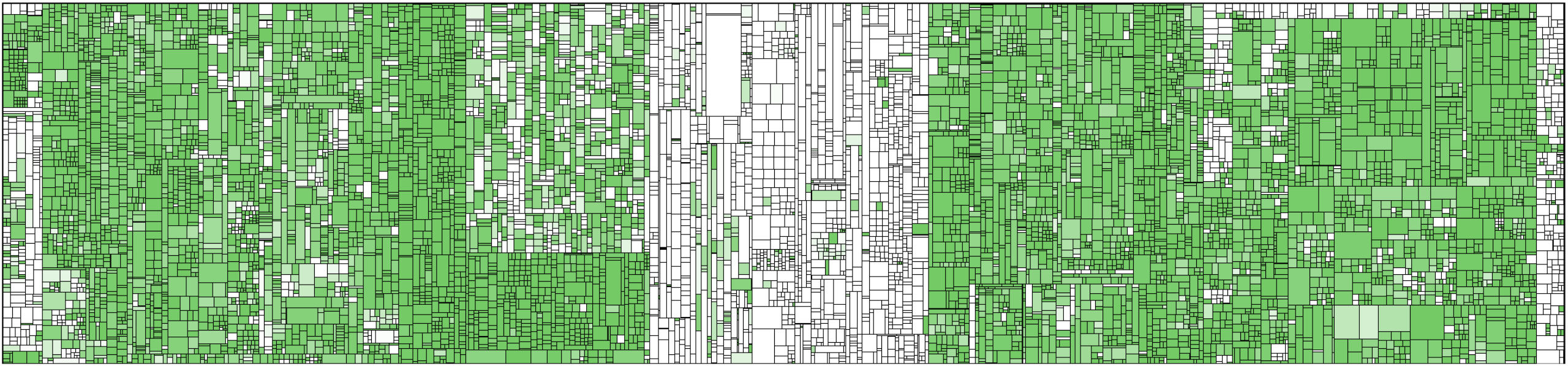
Jakob *Rott*. Setzt die TGA im Kundenumfeld täglich ein und hat viele Ausreden gehört, warum Sachen nicht getestet wurden.

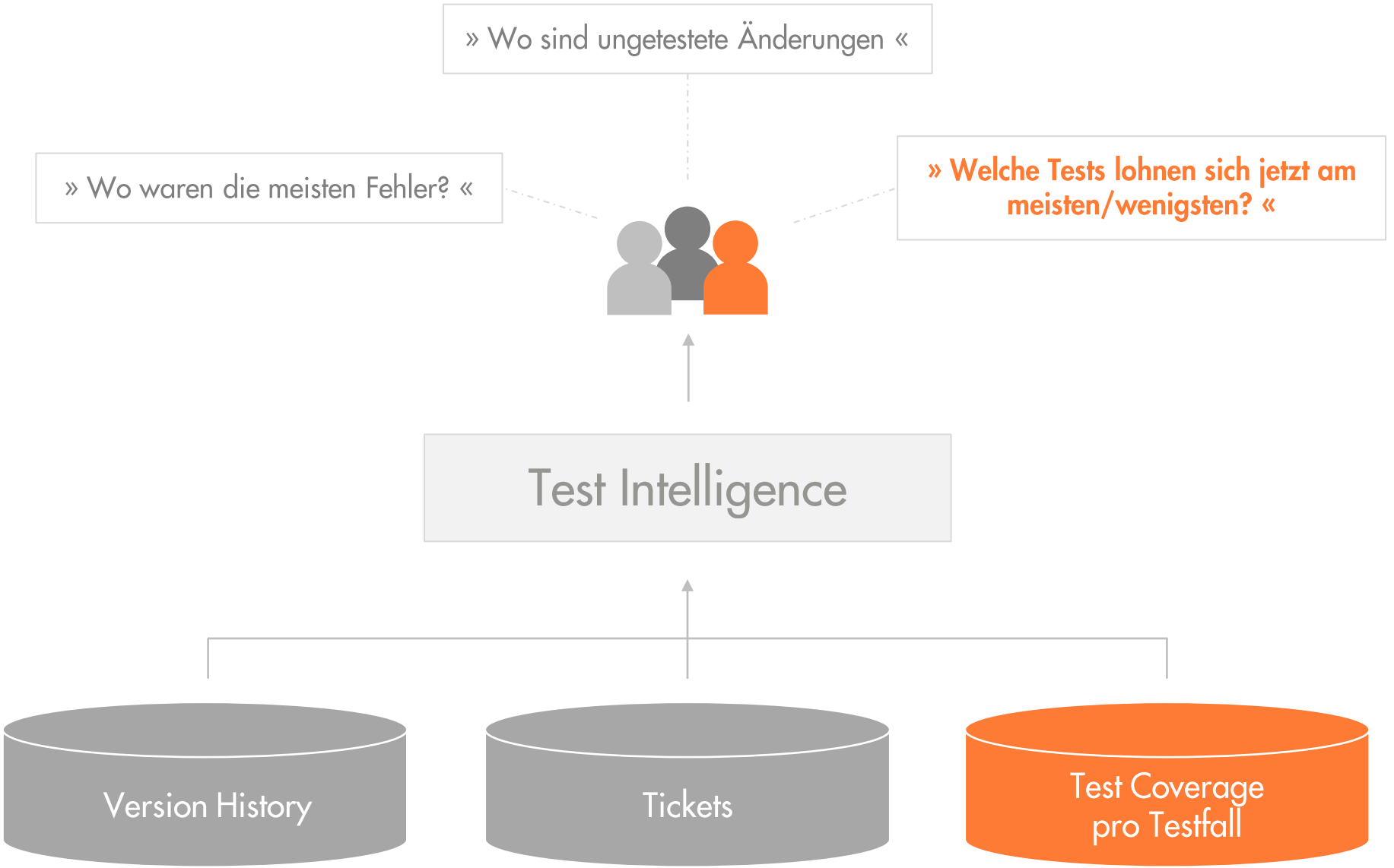
Einordnung

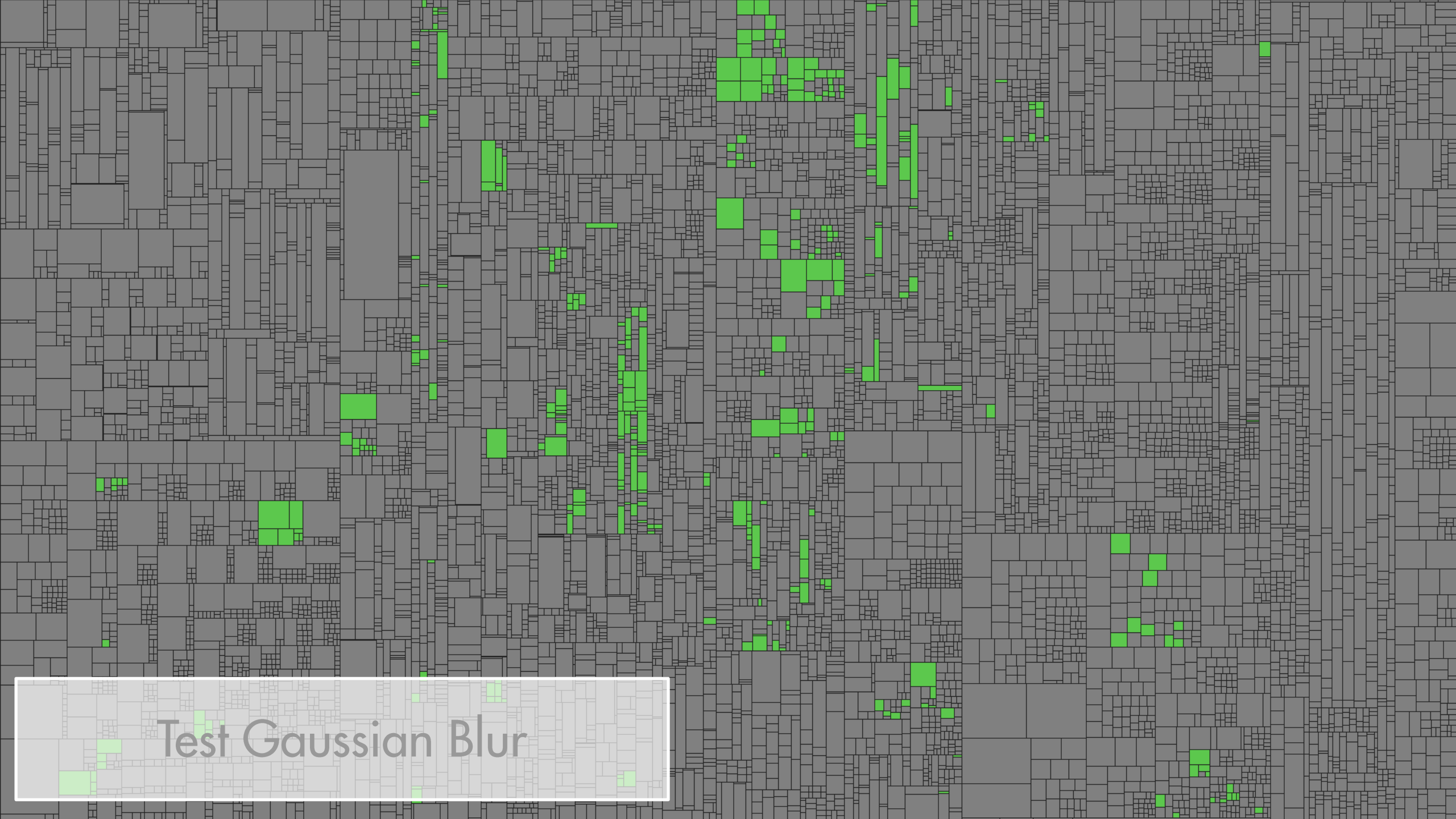
Der Workshop fasst mehrere Vorträge zusammen, die in den letzten Jahren auf dem *German Testing Day*, den *Software Quality Days* und dem *QS-Tag* mit Best Presentation Awards ausgezeichnet wurden. Anders als diese Einzelvorträge stellt er aber keine neuen Analysen oder neue Forschungsarbeiten vor, sondern gibt einen konsolidierten Überblick für diejenigen, die sich für das Thema *Test-Gap-Analyse* neu interessieren.


» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage

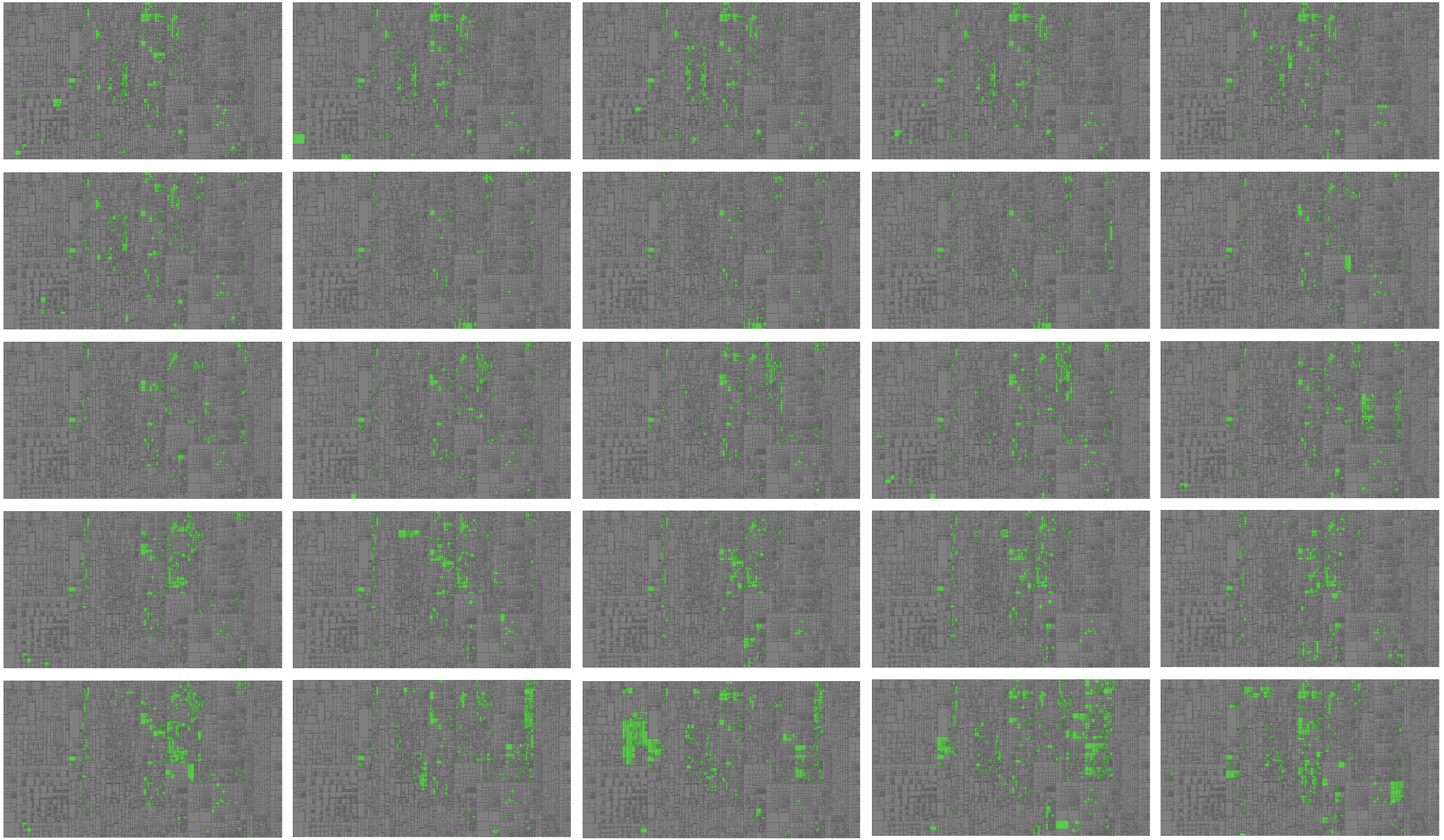


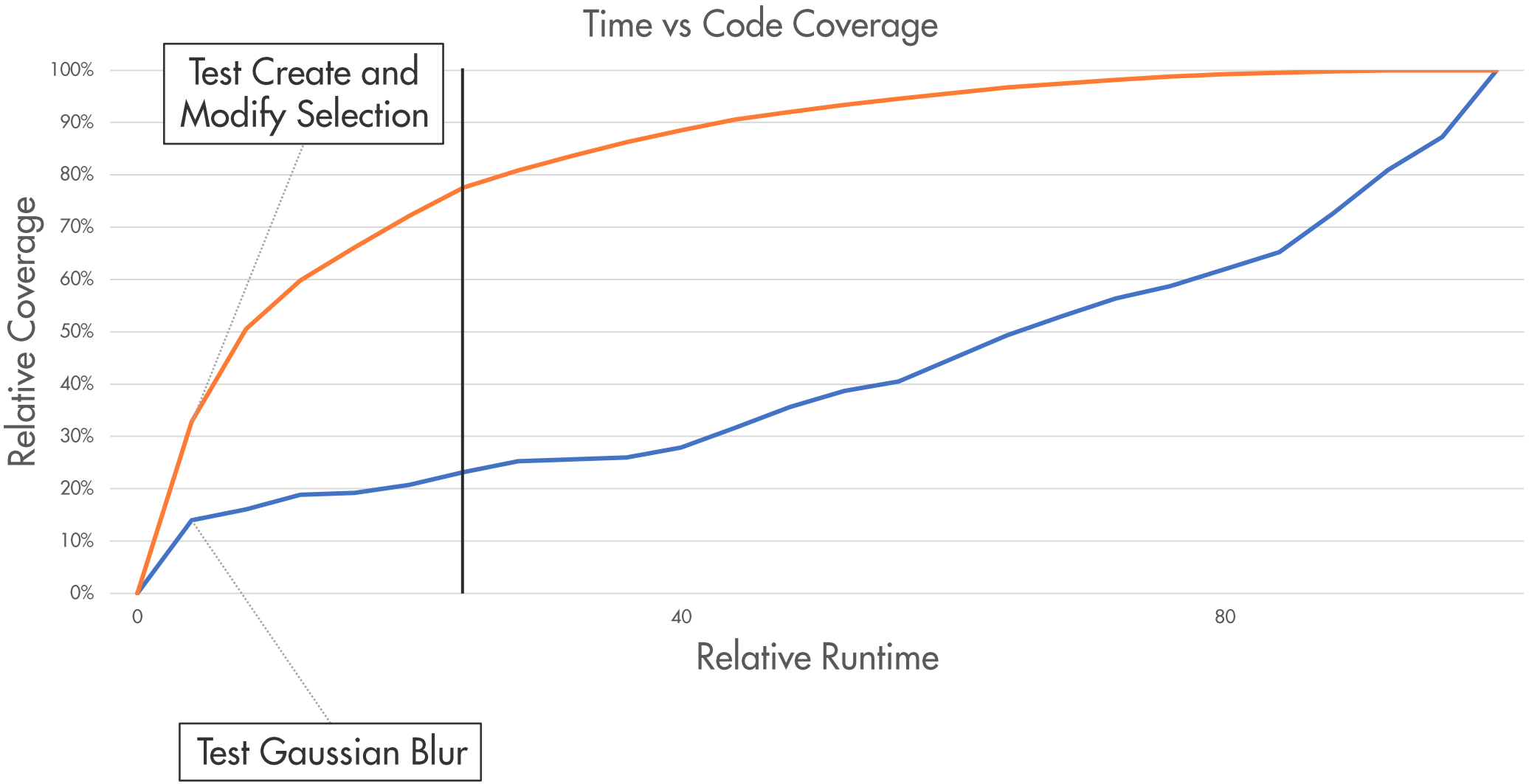


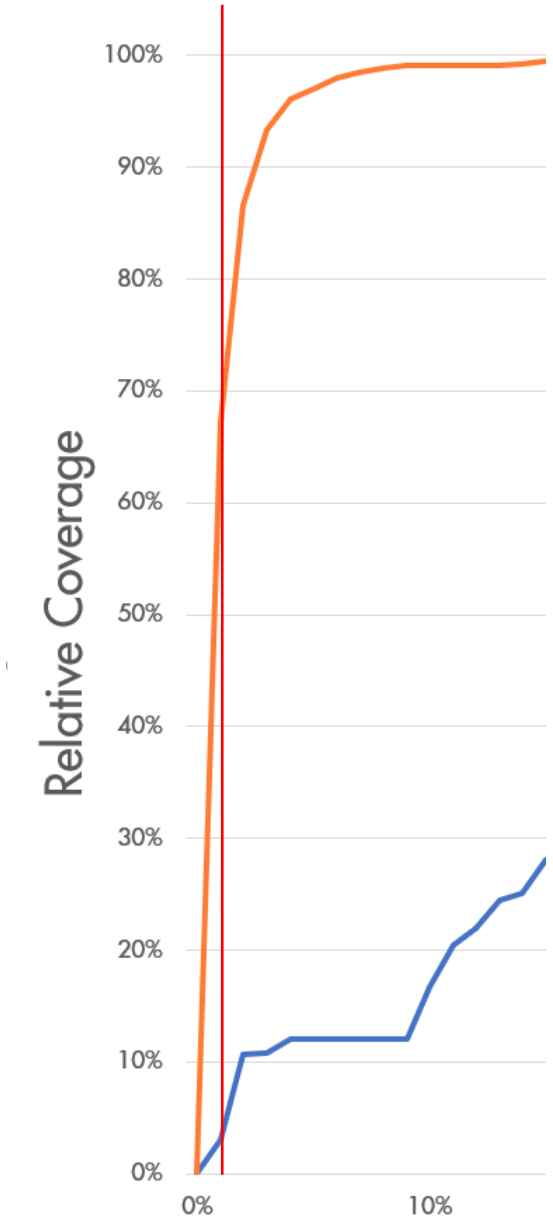

» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage pro Testfall

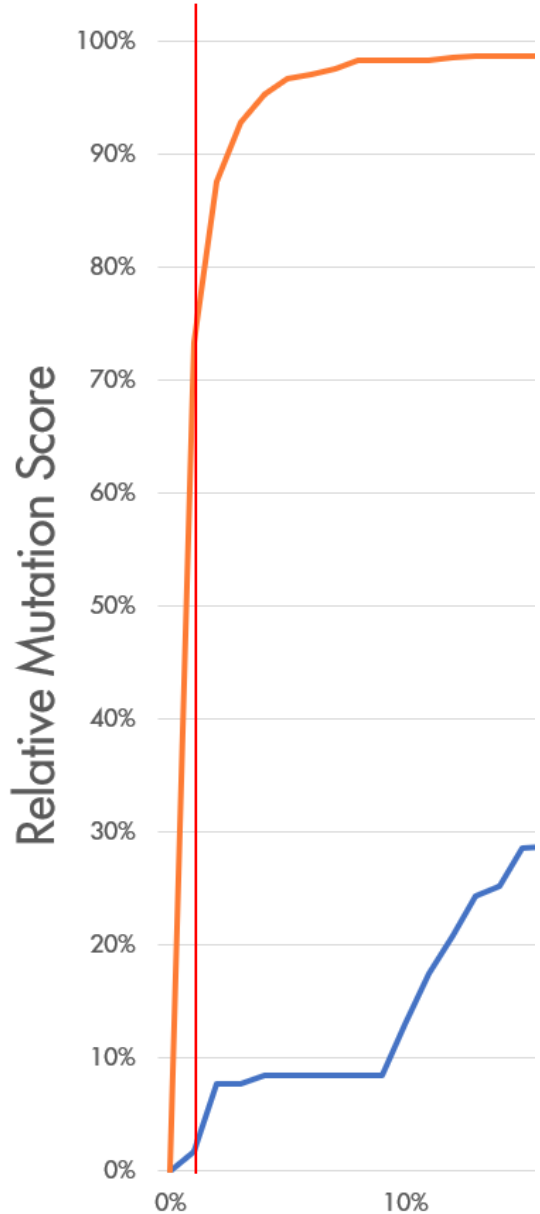

Test Gaussian Blur


Test Create and Modify
Selection

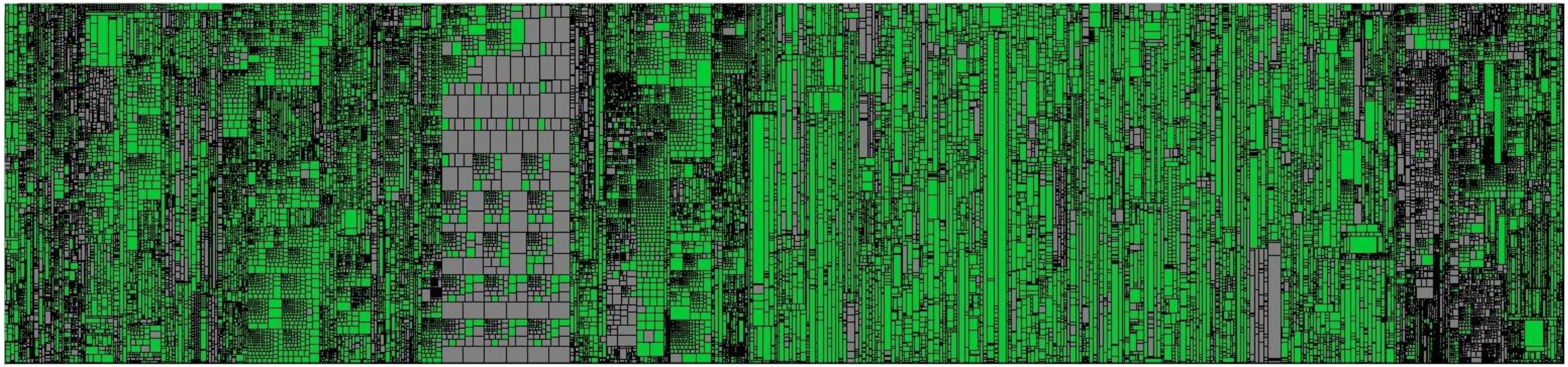



Time vs Code Coverage
Test Create and Modify Selection
Test Gaussian Blur
Relative Coverage
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
0
40
80
Relative Runtime

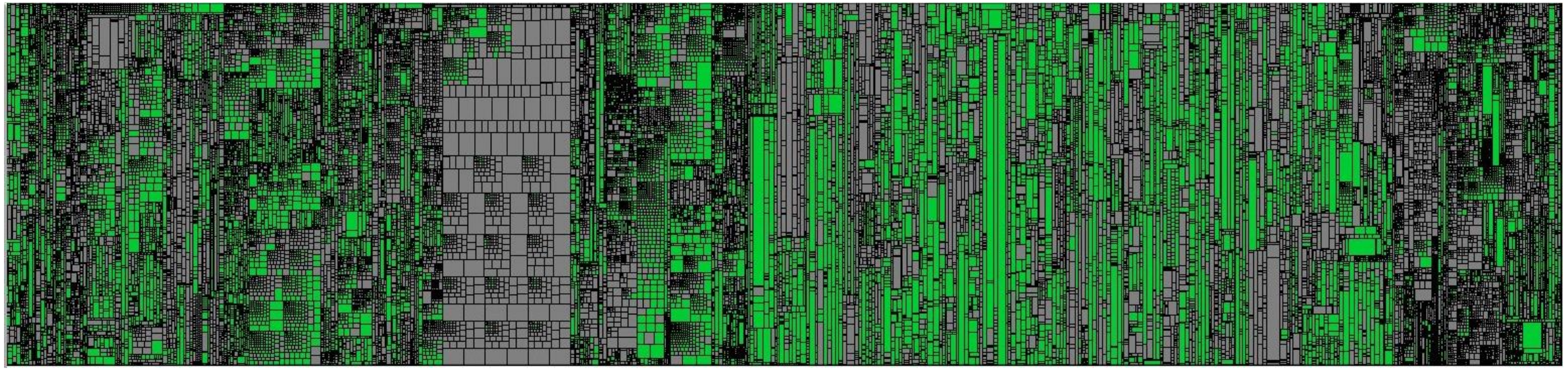

Relative Coverage
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
0%
10%


Relative Mutation Score
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
0%
10%

400+ Stunden Testausführung



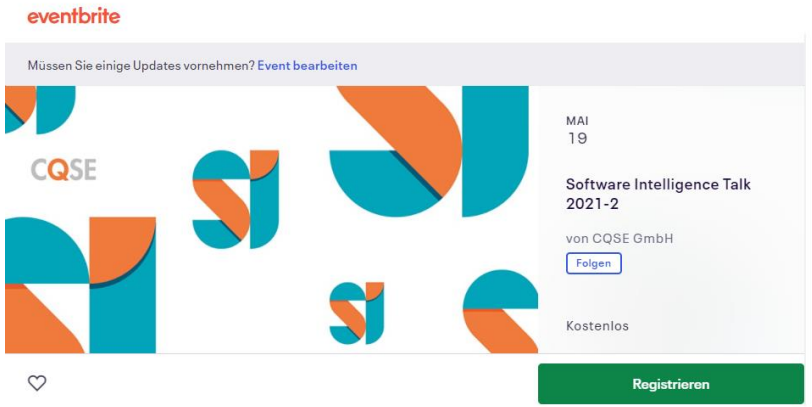
1 Stunde Testausführung: >50% Coverage



Virtueller Workshop

19. Mai 21, 10:30 – 12:00 Uhr

Anmeldung:
http://cqse.eu/si-talks/2gtb



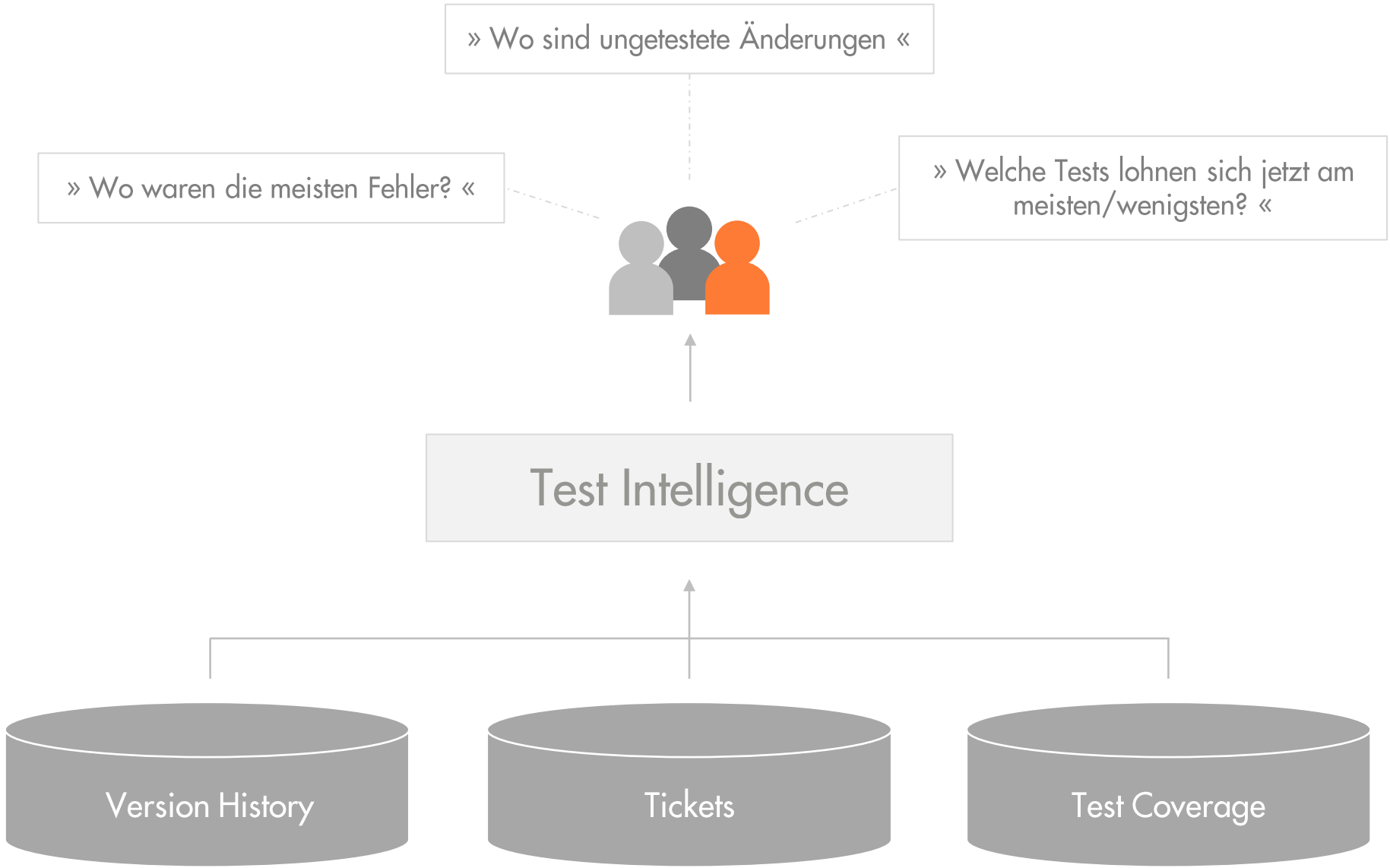
Zu den Sprechern

Jeannette Wernicke Jeannette Wernicke ist Innovation Managerin im Innovation Lab der Bayerische Versorgungskammer. Sie beschäftigt sich dort damit neue Technologien und Trends auf ihre Anwendbarkeit in der BVK zu untersuchen. Davor war sie als Software-Entwicklerin und -Architektin mit Schwerpunkt Software-Qualität tätig.

Raphael Nömmer hat seinen Master an der Technischen Universität München mit einem Fokus auf Software-Engineering und Software-Qualität abgeschlossen. Er hat seine Masterarbeit zu Test-Suite-Minimierung und Pareto-Testing erstellt und promoviert bei der CQSE in diesem Themenbereich.

Fabian Streitel hat als Leiter des CQSE Piloten-Teams die Test-Analysen bei vielen Kunden erfolgreich eingeführt.

Dr. Elmar Jürgens hat die CQSE mitgegründet und beschäftigt sich sein ganzes Berufsleben mit Qualitätsanalysen von Software.


» Wo sind ungetestete Änderungen «
» Wo waren die meisten Fehler? «
» Welche Tests lohnen sich jetzt am meisten/wenigsten? «
Test Intelligence
Version History
Tickets
Test Coverage

Kontakt – Ich freue mich auf Fragen ☺

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

CQSE GmbH
Centa-Hafenbrädl-Str 59
81249 München
www.cqse.eu