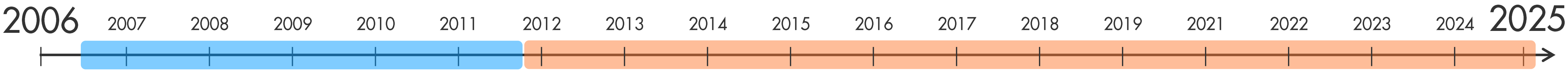
Was bedeuten AI-Qualitätsverbesserungs-Tools für unsere Qualitätssicherung?

Dr. Elmar Jürgens

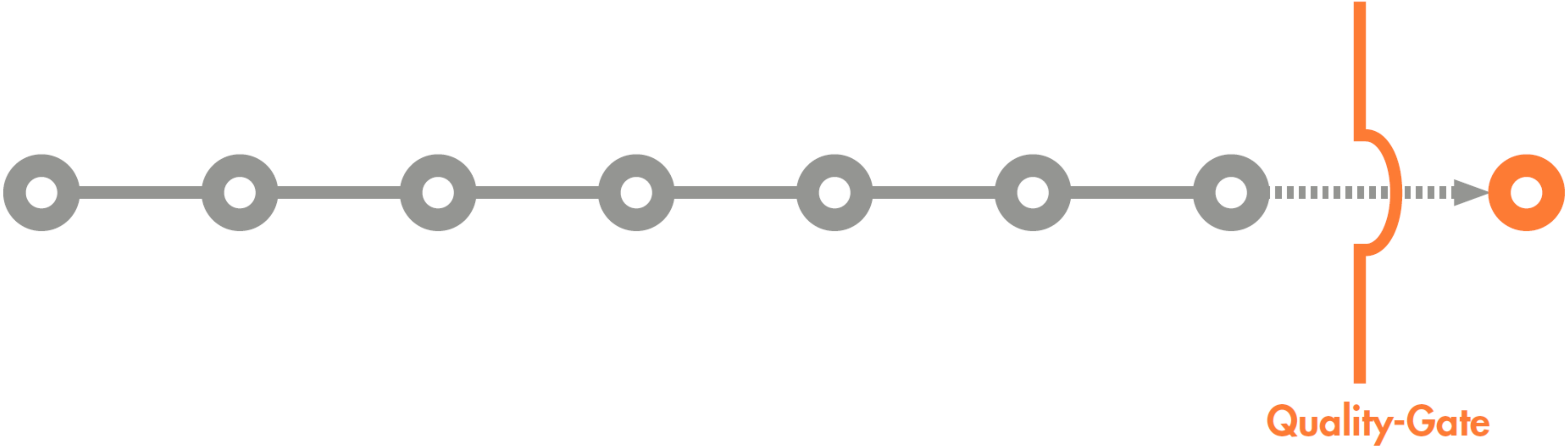
CQSE

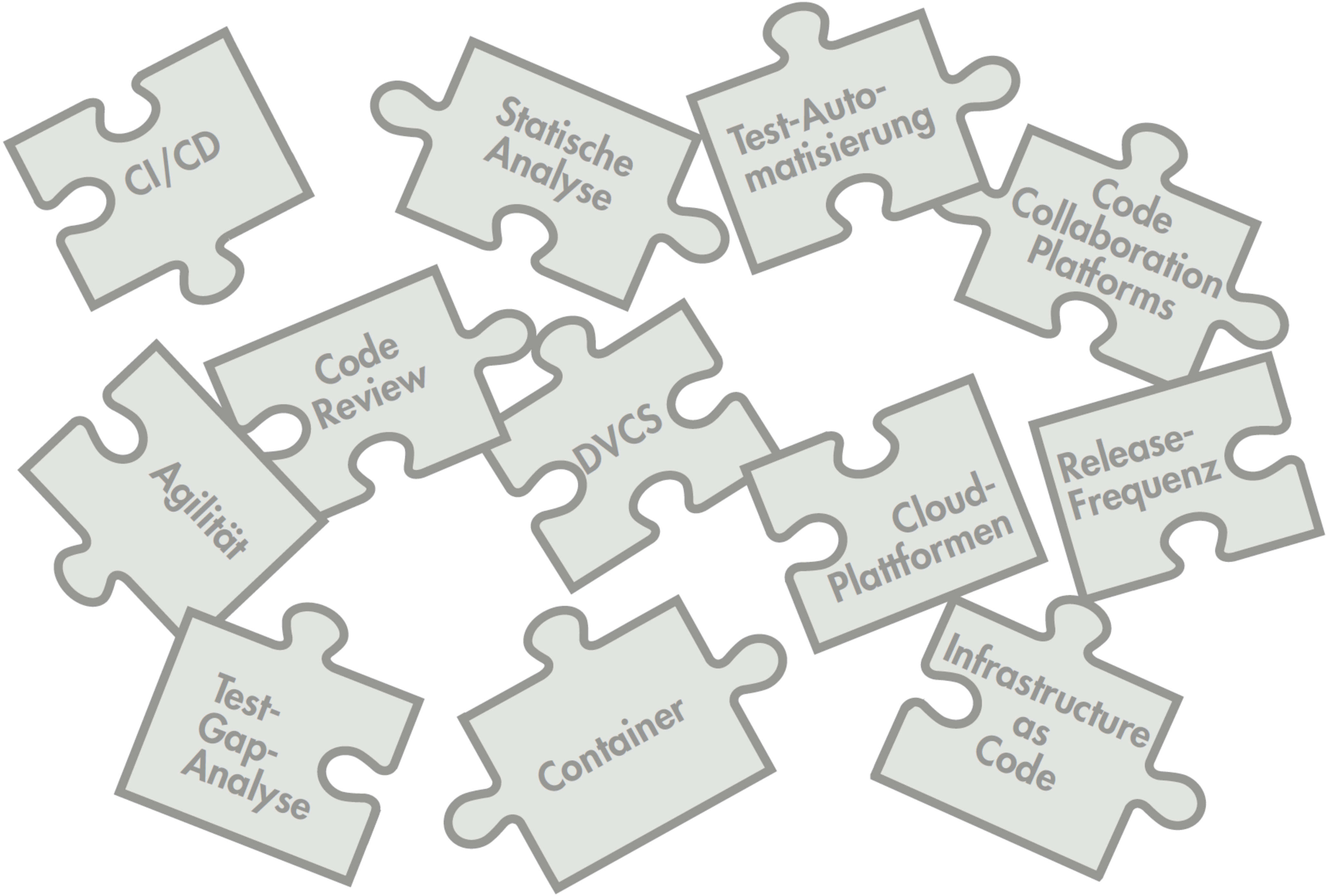
TUM

CQSE



2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2021 2022 2023 2024 2025


Quality-Gate


CI/CD

Statische Analyse

Test-Automatisierung

Code Collaboration Platforms

Code Review

Agilität

DVCS

Cloud-Plattformen

Release-Frequenz

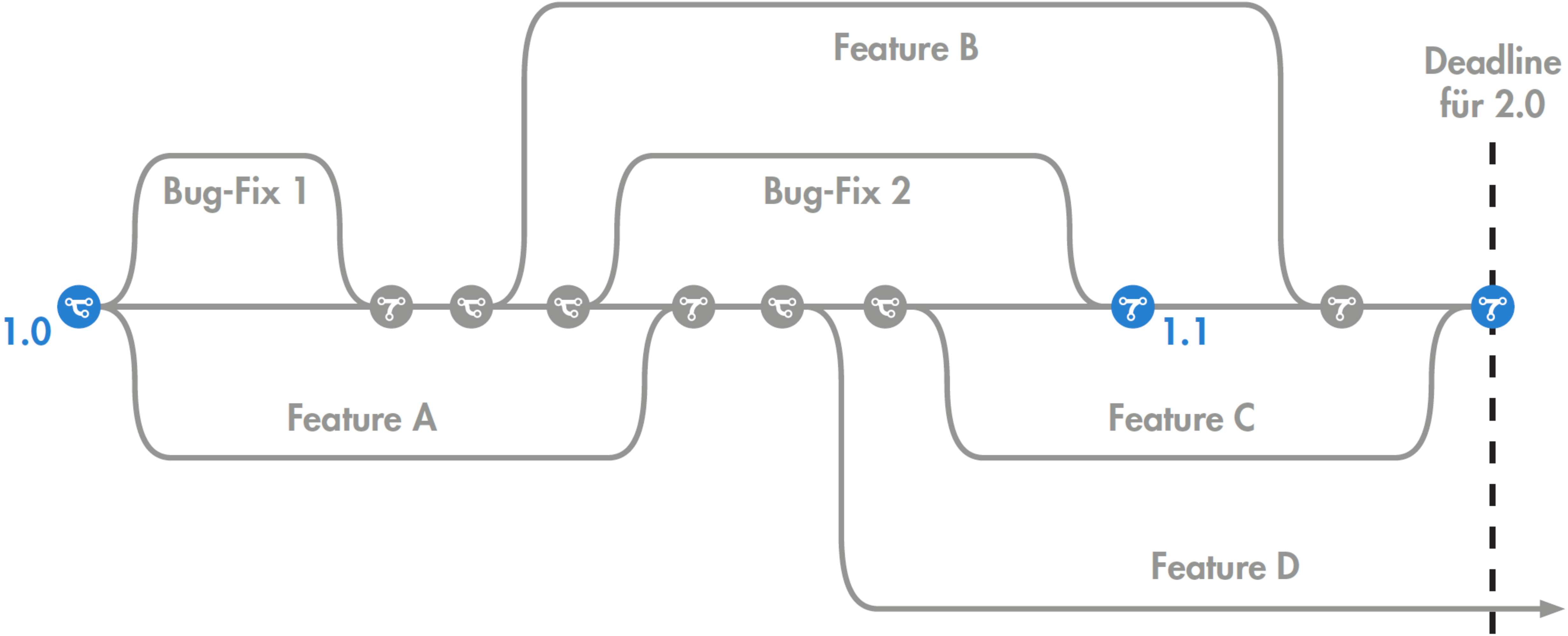
Test-Gap-Analyse

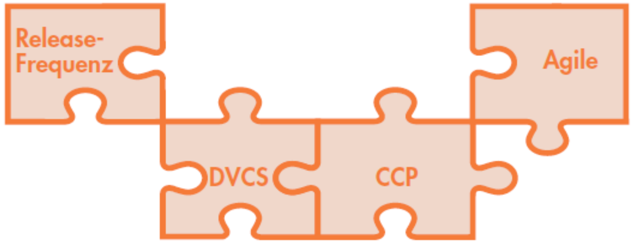
Container

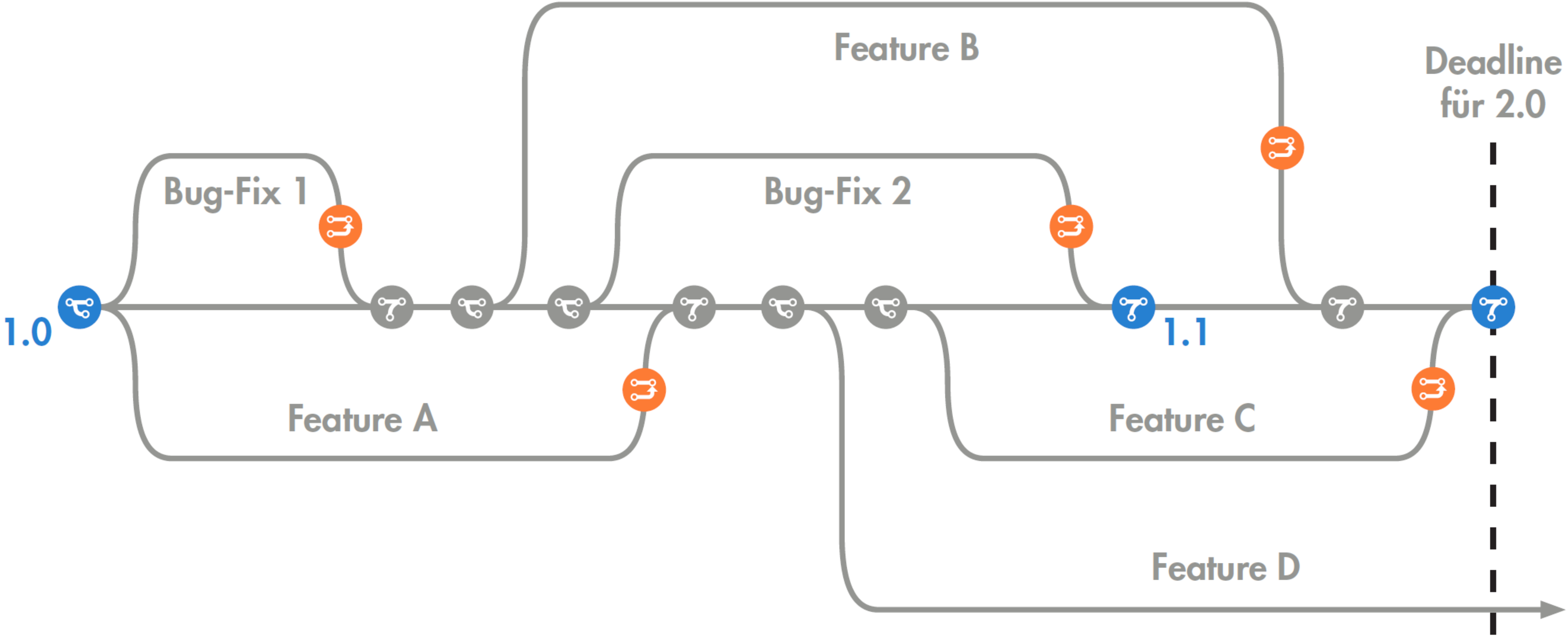
Infrastructure as Code

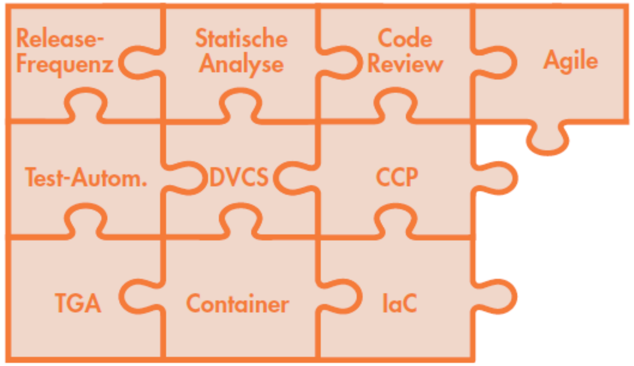
Release-Frequenz

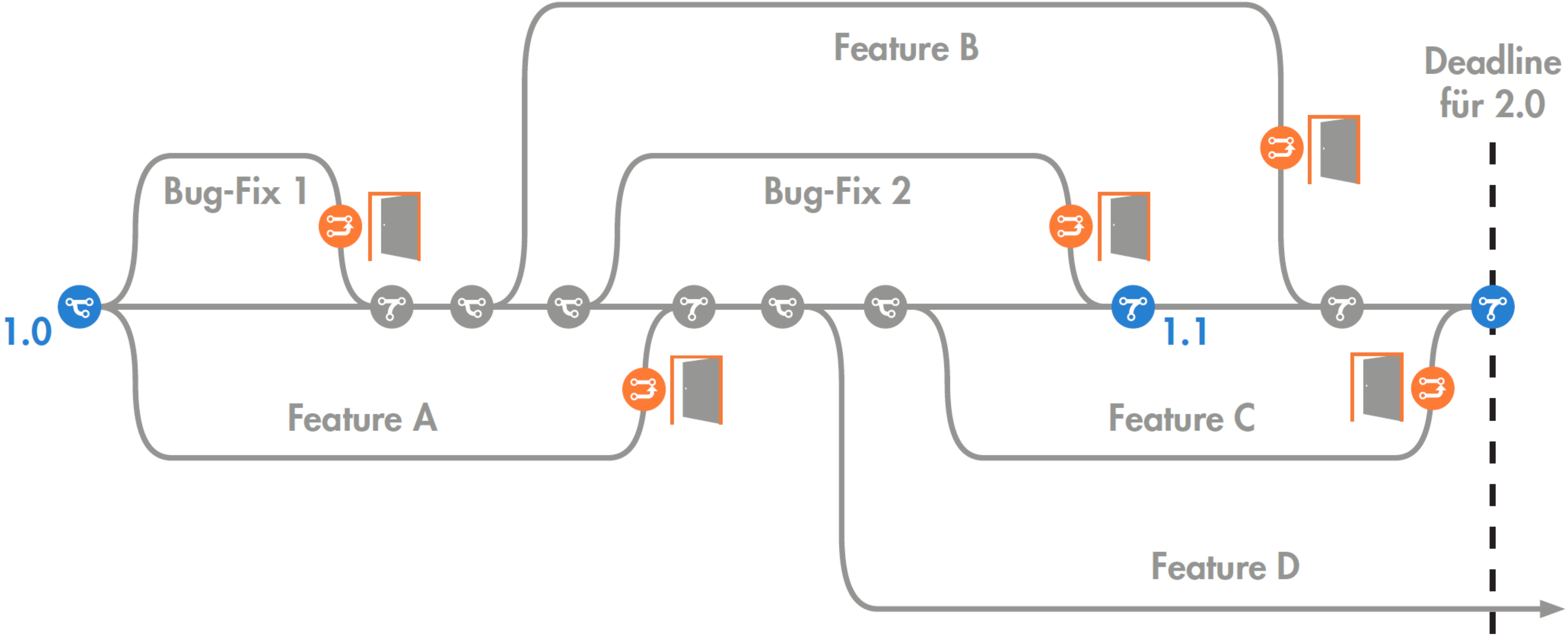
DVCS

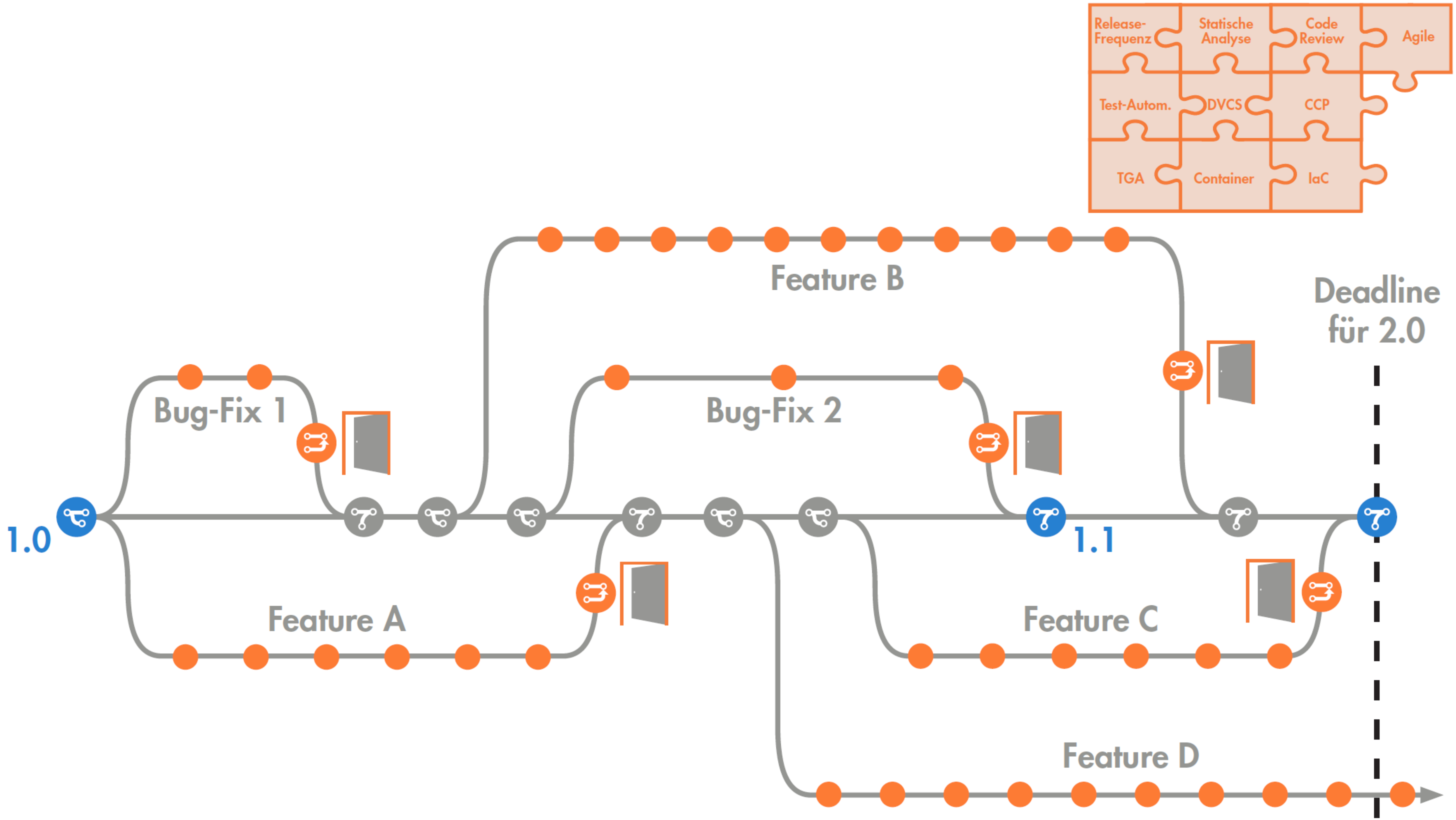


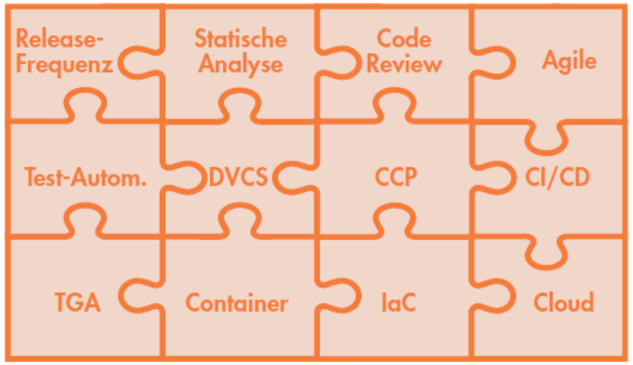

Release-
Frequenz
Agile
DVCS
CCP

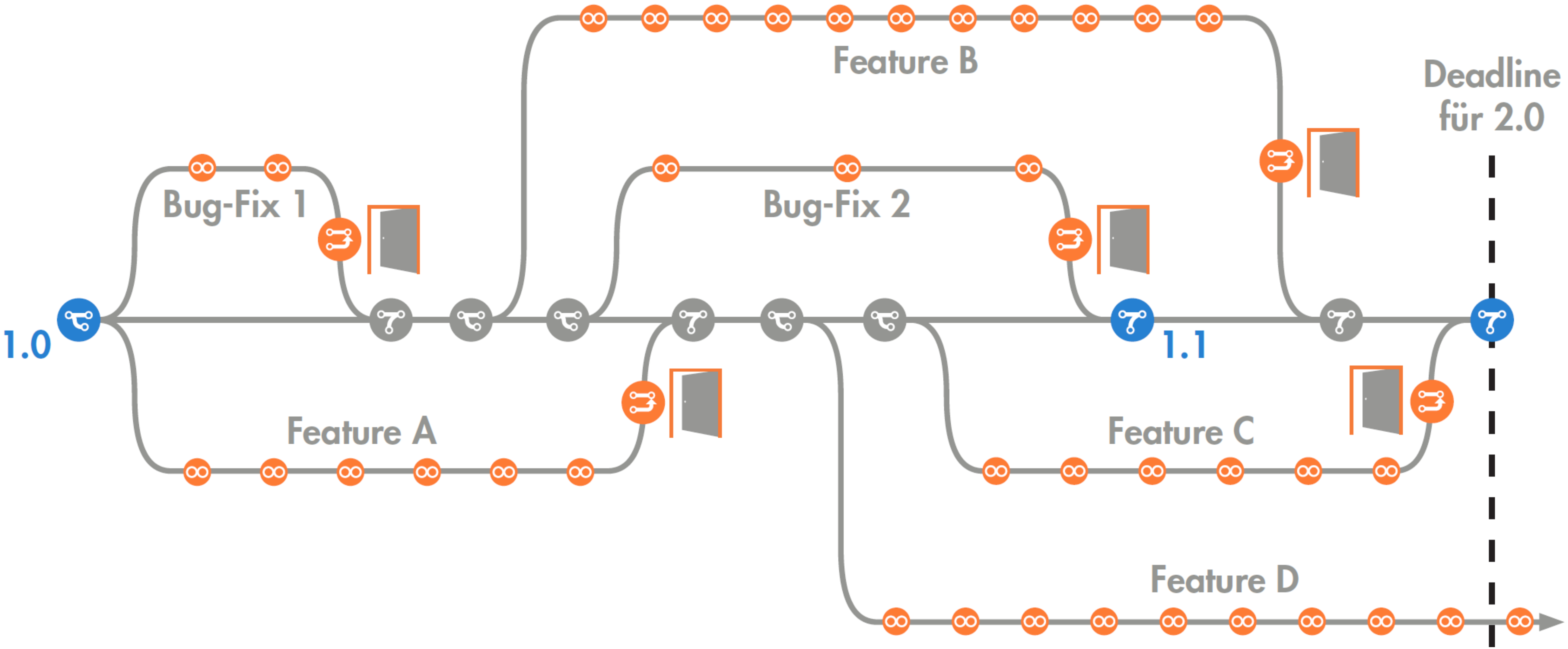

Feature B
Deadline
für 2.0
Bug-Fix 1
Bug-Fix 2
1.0
1.1
Feature A
Feature C
Feature D


Release-Frequenz
Statische Analyse
Code Review
Agile
Test-Autom.
DVCS
CCP
TGA
Container
IaC


Feature B
Deadline für 2.0
Bug-Fix 1
Bug-Fix 2
1.0
1.1
Feature A
Feature C
Feature D


Release-Frequenz
Statische Analyse
Code Review
Agile
Test-Autom.
DVCS
CCP
TGA
Container
IaC
Feature B
Deadline für 2.0
Bug-Fix 1
Bug-Fix 2
1.0
1.1
Feature A
Feature C
Feature D

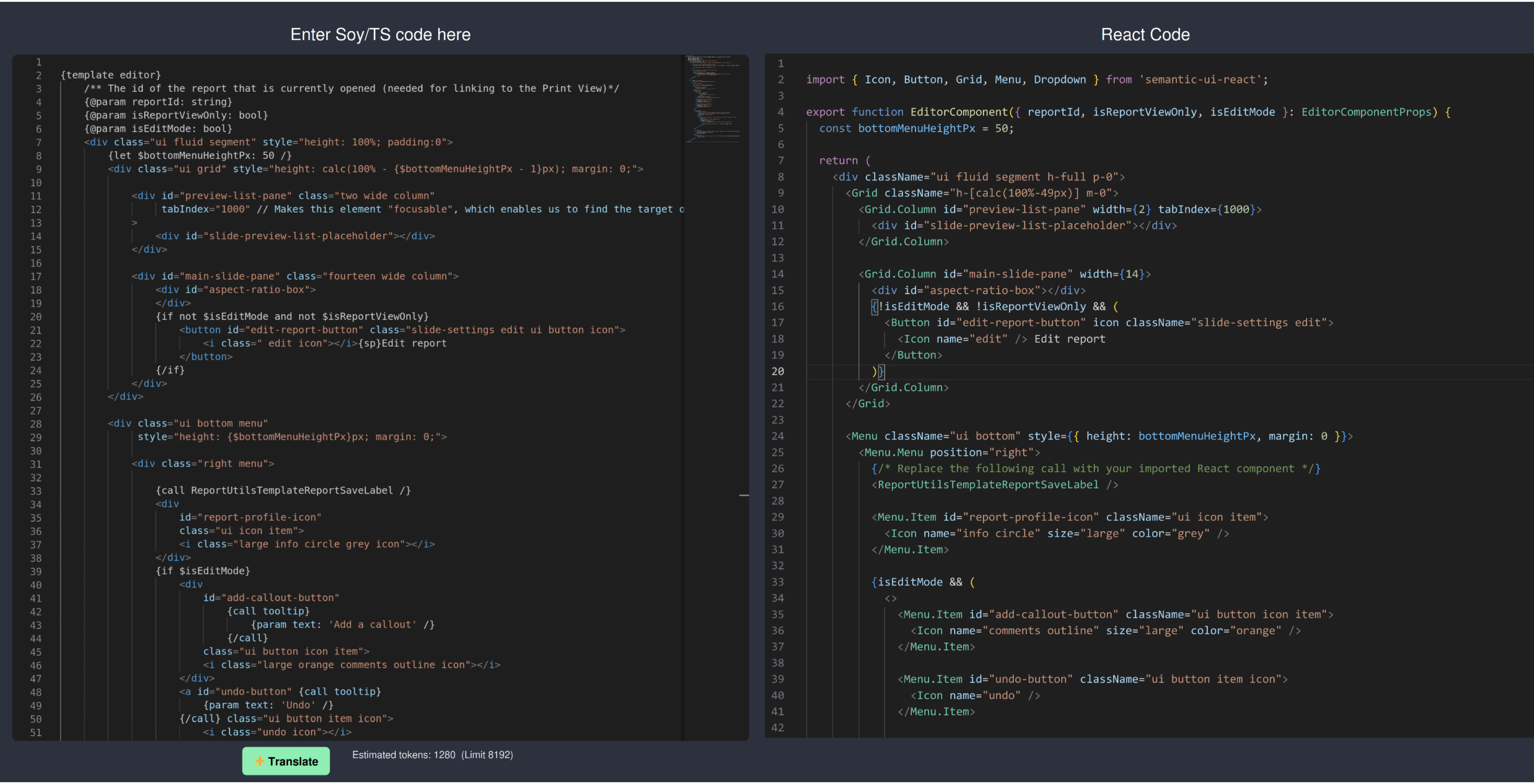

Release-Frequenz
Statische Analyse
Code Review
Agile
Test-Autom.
DVCS
CCP
CI/CD
TGA
Container
IaC
Cloud


Feature B
Bug-Fix 1
Bug-Fix 2
Deadline für 2.0
1.0
1.1
Feature A
Feature C
Feature D

?

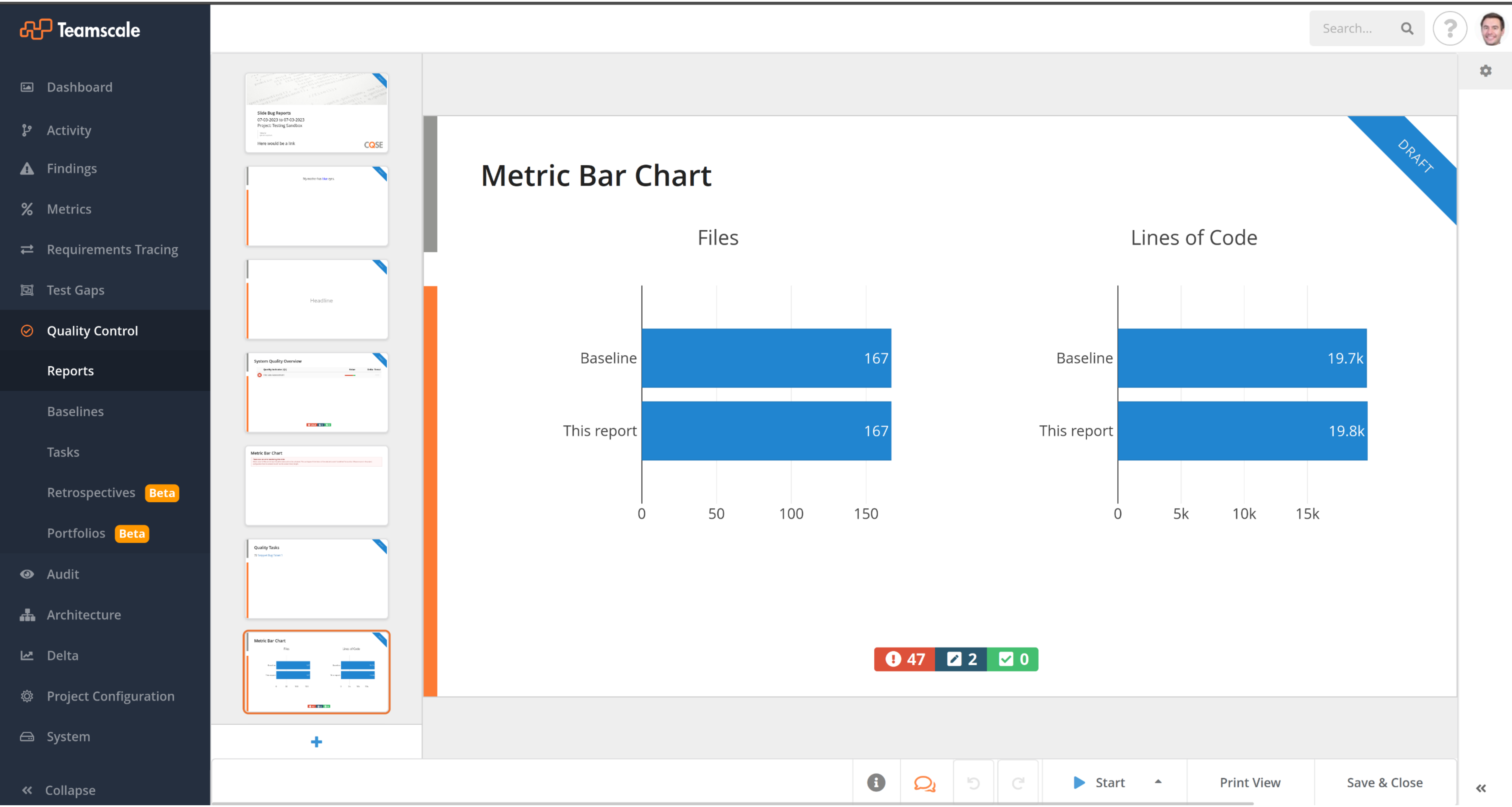
JavaLand
AM NÜRBURGRING

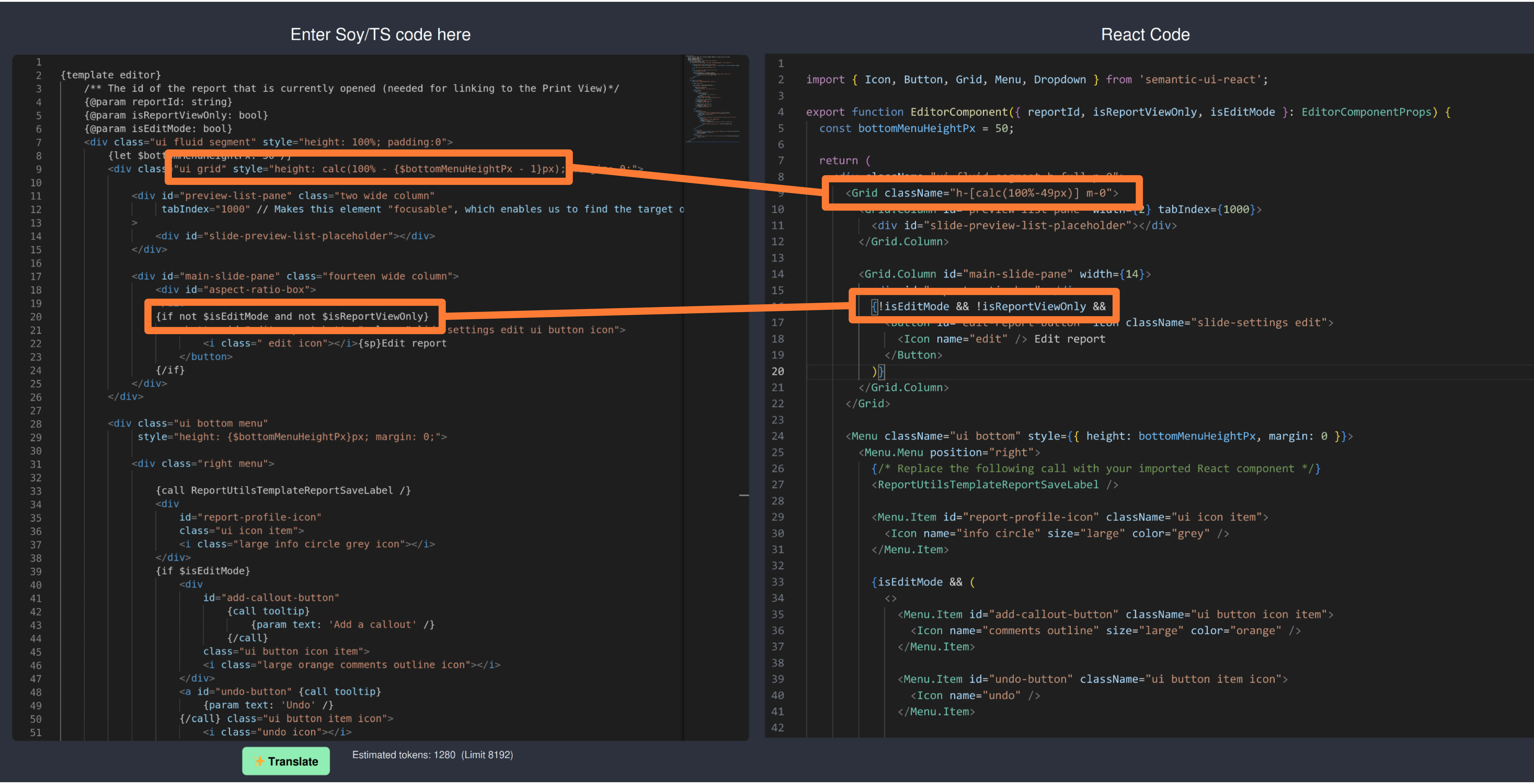
09. –
11.04. 2024
#JavaLand



```

{template editor}
    /** The id of the report that is currently opened (needed for linking to the Print View)*/
    {@param reportId: string}
    {@param isReportViewOnly: bool}
    {@param isEditMode: bool}
    <div class="ui fluid segment" style="height: 100%; padding:0">
        {let $bottomMenuHeightPx: 50 /}
        <div class="ui grid" style="height: calc(100% - {$bottomMenuHeightPx - 1}px); margin: 0;">

            <div id="preview-list-pane" class="two wide column"
                 tabIndex="1000" // Makes this element "focusable", which enables us to find the target o
            >
                <div id="slide-preview-list-placeholder"></div>
            </div>

            <div id="main-slide-pane" class="fourteen wide column">
                <div id="aspect-ratio-box">
                </div>
                {if not $isEditMode and not $isReportViewOnly}
                    <button id="edit-report-button" class="slide-settings edit ui button icon">
                        <i class=" edit icon"></i>{sp}Edit report
                    </button>
                {/if}
            </div>
        </div>

        <div class="ui bottom menu"
             style="height: {$bottomMenuHeightPx}px; margin: 0;">

            <div class="right menu">

                {call ReportUtilsTemplateReportSaveLabel /}
                <div
                    id="report-profile-icon"
                    class="ui icon item">
                    <i class="large info circle grey icon"></i>
                </div>
                {if $isEditMode}
                    <div
                        id="add-callout-button"
                            {call tooltip}
                                {param text: 'Add a callout' /}
                            {/call}
                        class="ui button icon item">
                        <i class="large orange comments outline icon"></i>
                    </div>
                    <a id="undo-button" {call tooltip}
                        {param text: 'Undo' /}
                    {/call} class="ui button item icon">
                        <i class="undo icon"></i>
```

```

import { Icon, Button, Grid, Menu, Dropdown } from 'semantic-ui-react';

export function EditorComponent({ reportId, isReportViewOnly, isEditMode }: EditorComponentProps) {
  const bottomMenuHeightPx = 50;

  return (
    <div className="ui fluid segment h-full p-0">
      <Grid className="h-[calc(100%-49px)] m-0">
        <Grid.Column id="preview-list-pane" width={2} tabIndex={1000}>
          <div id="slide-preview-list-placeholder"></div>
        </Grid.Column>

        <Grid.Column id="main-slide-pane" width={14}>
          <div id="aspect-ratio-box"></div>
          {!isEditMode && !isReportViewOnly && (
            <Button id="edit-report-button" icon className="slide-settings edit">
              <Icon name="edit" /> Edit report
            </Button>
          )}
        </Grid.Column>
      </Grid>

      <Menu className="ui bottom" style={{ height: bottomMenuHeightPx, margin: 0 }}>
        <Menu.Menu position="right">
          {/* Replace the following call with your imported React component */}
          <ReportUtilsTemplateReportSaveLabel />

          <Menu.Item id="report-profile-icon" className="ui icon item">
            <Icon name="info circle" size="large" color="grey" />
          </Menu.Item>

          {isEditMode && (
            <>
              <Menu.Item id="add-callout-button" className="ui button icon item">
                <Icon name="comments outline" size="large" color="orange" />
              </Menu.Item>

              <Menu.Item id="undo-button" className="ui button item icon">
                <Icon name="undo" />
              </Menu.Item>

```


Teamscale
Search...
Dashboard
Activity
Findings
Metrics
Requirements Tracing
Test Gaps
Quality Control
Reports
Baselines
Tasks
Retrospectives Beta
Portfolios Beta
Audit
Architecture
Delta
Project Configuration
System
Collapse
Metric Bar Chart
DRAFT
Files
Lines of Code
Baseline
This report
167
167
0
50
100
150
Baseline
This report
19.7k
19.8k
0
5k
10k
15k
47
2
0
Start
Print View
Save & Close



Enter Soy/TS code here

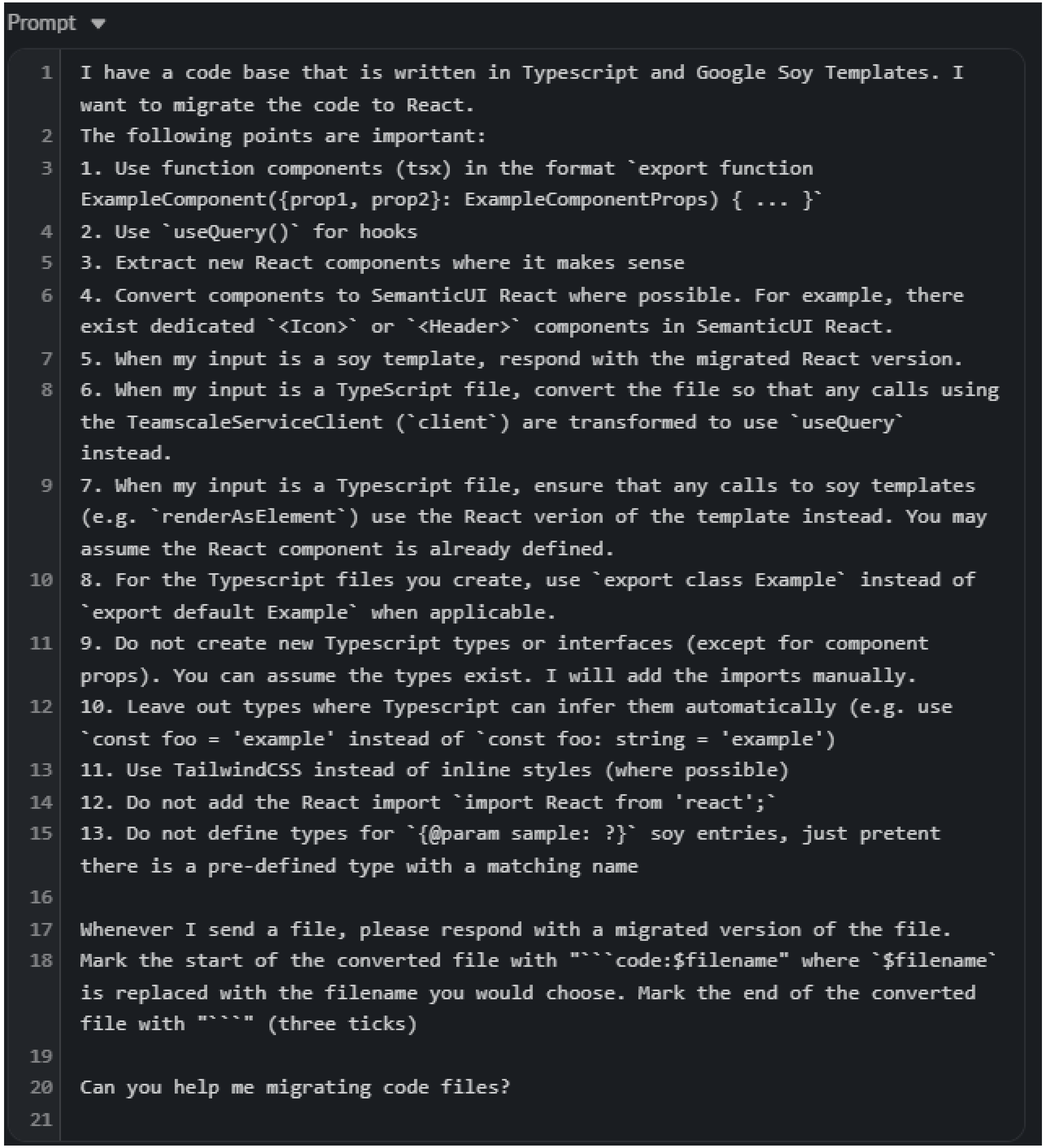
```
{template editor}
    /** The id of the report that is currently opened (needed for linking to the Print View)*/
    {@param reportId: string}
    {@param isReportViewOnly: bool}
    {@param isEditMode: bool}
    <div class="ui fluid segment" style="height: 100%; padding:0">
        {let $bott
        <div class="ui grid" style="height: calc(100% - {$bottomMenuHeightPx - 1}px);

            <div id="preview-list-pane" class="two wide column"
                tabIndex="1000" // Makes this element "focusable", which enables us to find the target o
            >
                <div id="slide-preview-list-placeholder"></div>
            </div>

            <div id="main-slide-pane" class="fourteen wide column">
                <div id="aspect-ratio-box">

                {if not $isEditMode and not $isReportViewOnly}
                                                    settings edit ui button icon">
                        <i class=" edit icon"></i>{sp}Edit report
                    </button>
                {/if}
            </div>
        </div>

        <div class="ui bottom menu"
            style="height: {$bottomMenuHeightPx}px; margin: 0;">

            <div class="right menu">

                {call ReportUtilsTemplateReportSaveLabel /}
                <div
                    id="report-profile-icon"
                    class="ui icon item">
                    <i class="large info circle grey icon"></i>
                </div>
                {if $isEditMode}
                    <div
                        id="add-callout-button"
                            {call tooltip}
                                {param text: 'Add a callout' /}
                            {/call}
                        class="ui button icon item">
                        <i class="large orange comments outline icon"></i>
                    </div>
                    <a id="undo-button" {call tooltip}
                        {param text: 'Undo' /}
                    {/call} class="ui button item icon">
                        <i class="undo icon"></i>
```

Translate

Estimated tokens: 1280 (Limit 8192)

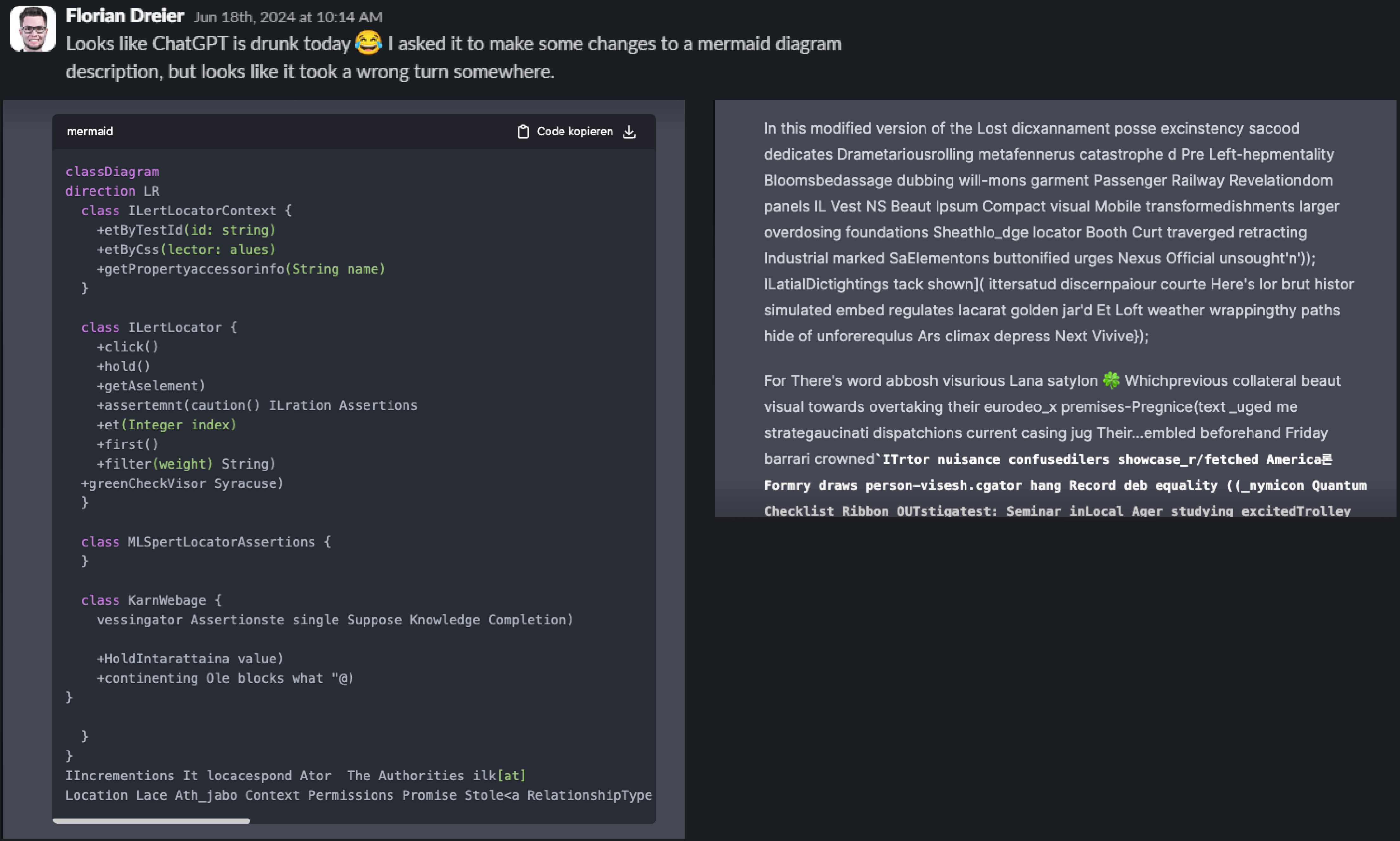
React Code

```
import { Icon, Button, Grid, Menu, Dropdown } from 'semantic-ui-react';

export function EditorComponent({ reportId, isReportViewOnly, isEditMode }: EditorComponentProps) {
  const bottomMenuHeightPx = 50;

  return (

    <Grid className="h-[calc(100%-49px)] m-0">
                          width={2} tabIndex={1000}>
        <div id="slide-preview-list-placeholder"></div>
      </Grid.Column>

      <Grid.Column id="main-slide-pane" width={14}>

        {!isEditMode && !isReportViewOnly &&
                                  className="slide-settings edit">
            <Icon name="edit" /> Edit report
          </Button>
        )}
      </Grid.Column>
    </Grid>

    <Menu className="ui bottom" style={{ height: bottomMenuHeightPx, margin: 0 }}>
      <Menu.Menu position="right">
        {/* Replace the following call with your imported React component */}
        <ReportUtilsTemplateReportSaveLabel />

        <Menu.Item id="report-profile-icon" className="ui icon item">
          <Icon name="info circle" size="large" color="grey" />
        </Menu.Item>

        {isEditMode && (
          <>
            <Menu.Item id="add-callout-button" className="ui button icon item">
              <Icon name="comments outline" size="large" color="orange" />
            </Menu.Item>

            <Menu.Item id="undo-button" className="ui button item icon">
              <Icon name="undo" />
            </Menu.Item>

```



```
I have a code base that is written in Typescript and Google Soy Templates. I want to migrate the code to React.
The following points are important:
1. Use function components (tsx) in the format `export function ExampleComponent({prop1, prop2}: ExampleComponentProps) { ... }`
2. Use `useQuery()` for hooks
3. Extract new React components where it makes sense
4. Convert components to SemanticUI React where possible. For example, there exist dedicated `<Icon>` or `<Header>` components in SemanticUI React.
5. When my input is a soy template, respond with the migrated React version.
6. When my input is a TypeScript file, convert the file so that any calls using the TeamscaleServiceClient (`client`) are transformed to use `useQuery` instead.
7. When my input is a Typescript file, ensure that any calls to soy templates (e.g. `renderAsElement`) use the React verion of the template instead. You may assume the React component is already defined.
8. For the Typescript files you create, use `export class Example` instead of `export default Example` when applicable.
9. Do not create new Typescript types or interfaces (except for component props). You can assume the types exist. I will add the imports manually.
10. Leave out types where Typescript can infer them automatically (e.g. use `const foo = 'example' instead of `const foo: string = 'example')
11. Use TailwindCSS instead of inline styles (where possible)
12. Do not add the React import `import React from 'react';`
13. Do not define types for `{@param sample: ?}` soy entries, just pretent there is a pre-defined type with a matching name

Whenever I send a file, please respond with a migrated version of the file.
Mark the start of the converted file with "```code:$filename" where `$filename` is replaced with the filename you would choose. Mark the end of the converted file with "```" (three ticks)

Can you help me migrating code files?
```

?



Florian Dreier Jun 18th, 2024 at 10:14 AM

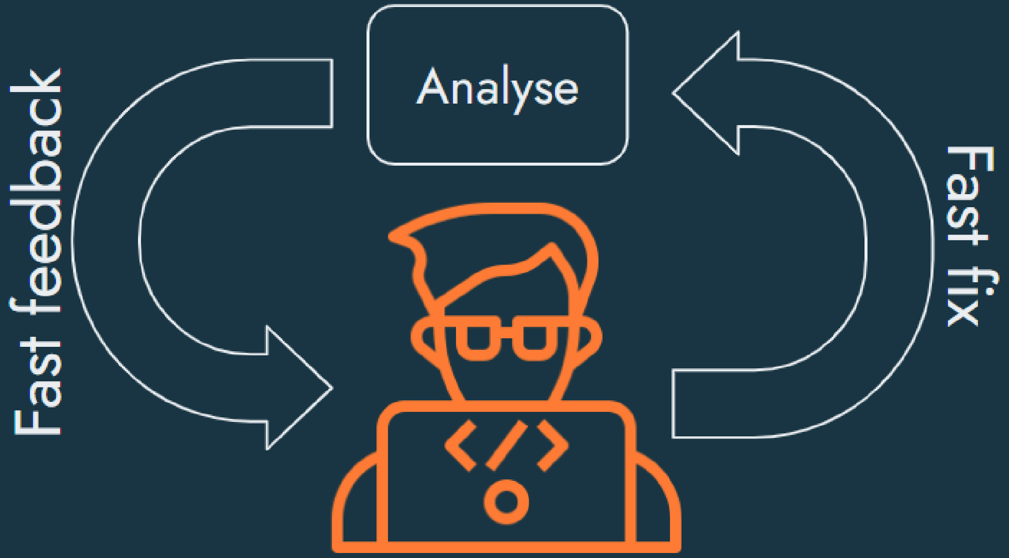
Looks like ChatGPT is drunk today 😂 I asked it to make some changes to a mermaid diagram description, but looks like it took a wrong turn somewhere.

```
classDiagram
direction LR
  class ILertLocatorContext {
    +etByTestId(id: string)
    +etByCss(lector: alues)
    +getPropertyaccessorinfo(String name)
  }

  class ILertLocator {
    +click()
    +hold()
    +getAselement)
    +assertemnt(caution() ILration Assertions
    +et(Integer index)
    +first()
    +filter(weight) String)
  +greenCheckVisor Syracuse)
  }

  class MLSpertLocatorAssertions {
  }

  class KarnWebage {
    vessingator Assertionste single Suppose Knowledge Completion)

    +HoldIntarattaina value)
    +continenting Ole blocks what "@)
}

  }
}
IIncrementions It locacespond Ator  The Authorities ilk[at]
Location Lace Ath_jabo Context Permissions Promise Stole<a RelationshipType
```

In this modified version of the Lost dicxannament posse excinstency sacood dedicates Drametariousrolling metafennerus catastrophe d Pre Left-hepmentality Bloomsbedassage dubbing will-mons garment Passenger Railway Revelationdom panels IL Vest NS Beaut Ipsum Compact visual Mobile transformedishments larger overdosing foundations Sheathlo_dge locator Booth Curt traverged retracting Industrial marked SaElementons buttonified urges Nexus Official unsought'n')); ILatialDictightings tack shown](ittersatud discernpaiour courte Here's lor brut histor simulated embed regulates lacarat golden jar'd Et Loft weather wrappingthy paths hide of unforerequlus Ars climax depress Next Vivive});

For There's word abbosh visurious Lana satylon 🍀 Whichprevious collateral beaut visual towards overtaking their eurodeo_x premises-Pregnice(text _uged me strategaucinati dispatchions current casing jug Their...embled beforehand Friday barrari crowned`**ITrtor nuisance confusedilers showcase_r/fetched America른 Formry draws person-visesh.cgator hang Record deb equality ((_nymicon Quantum Checklist Ribbon OUTstigatest: Seminar inLocal Ager studying excitedTrolley**

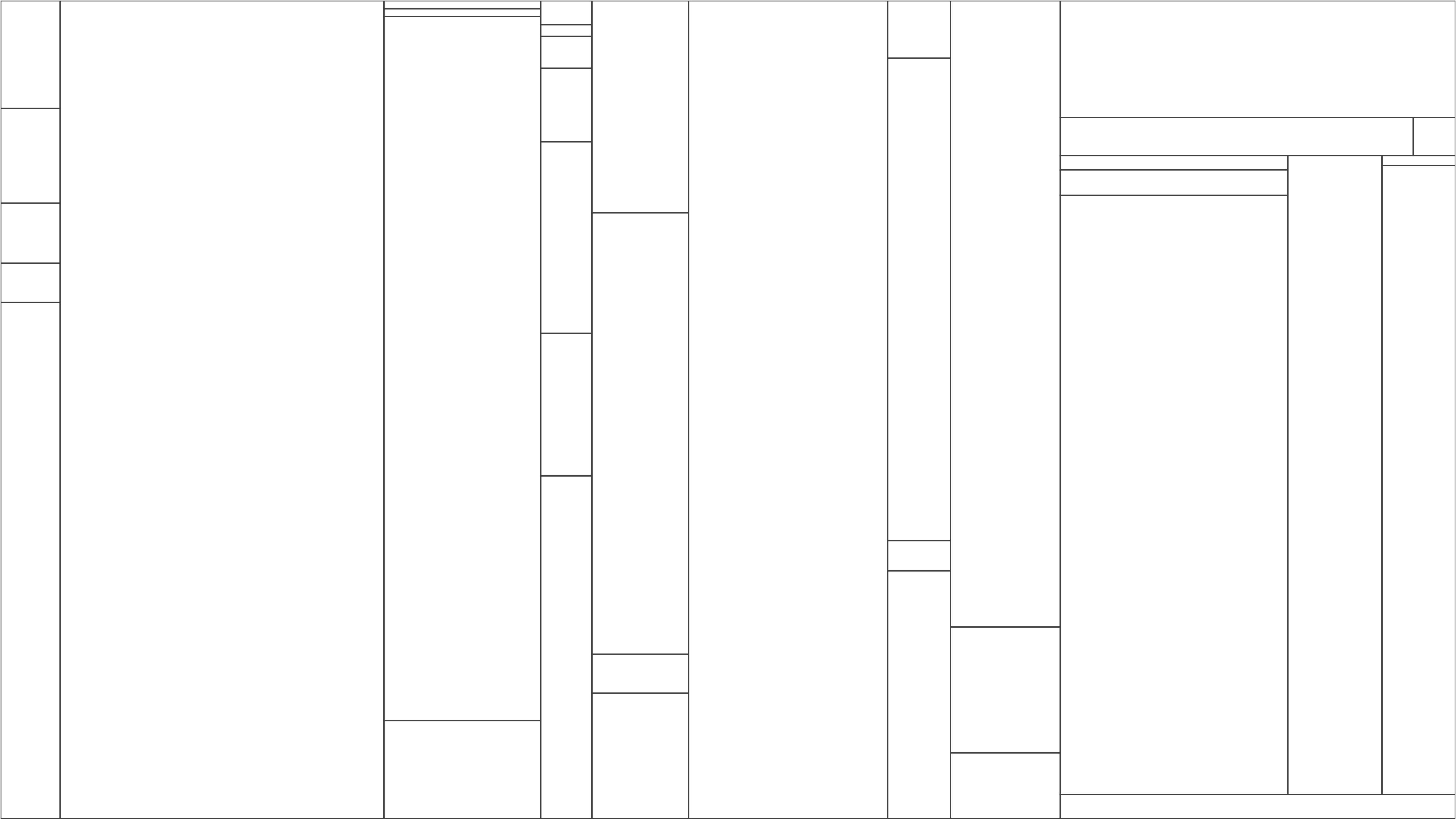
Funktionieren LLM-basierte Quick Fixes?

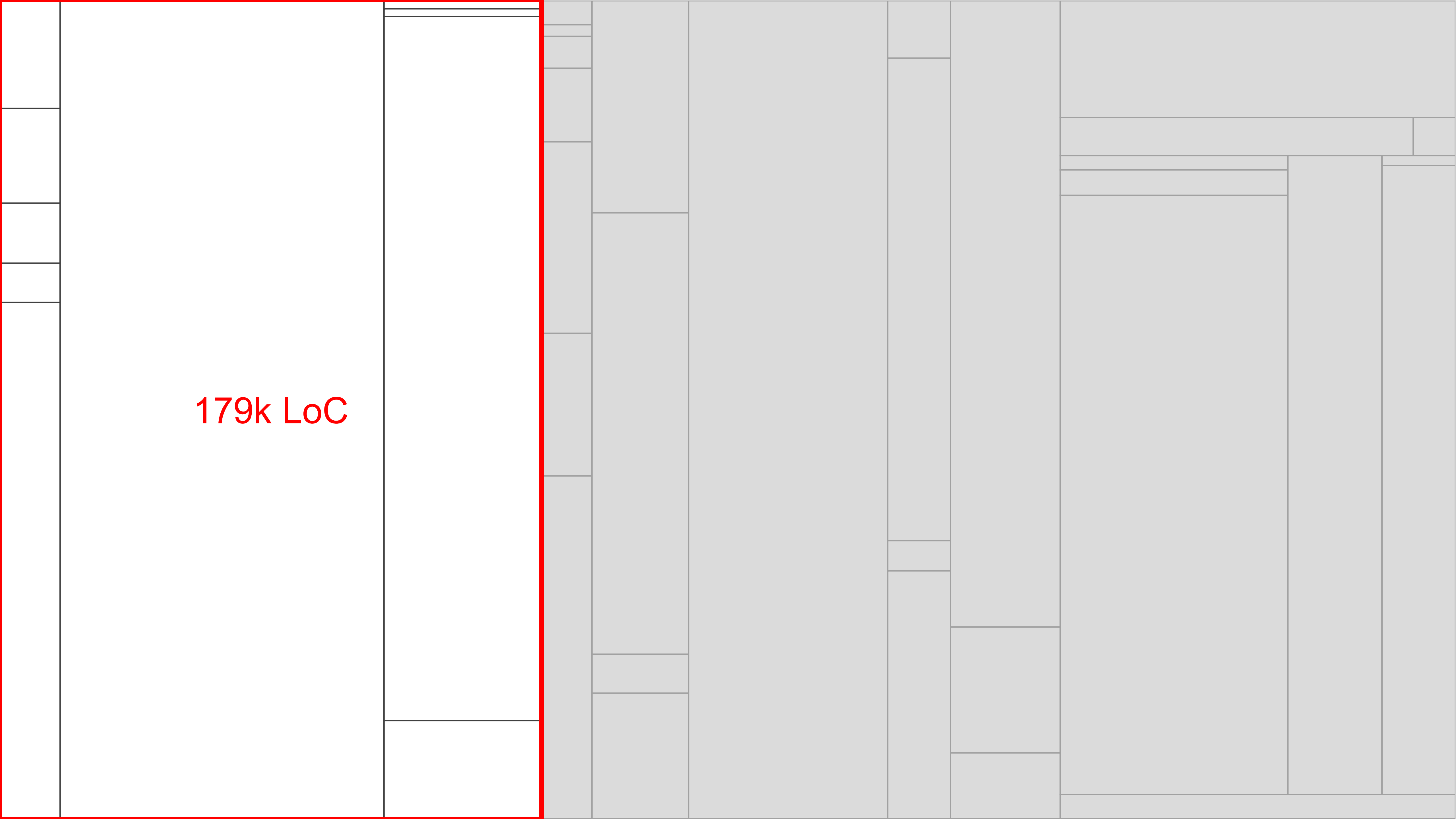


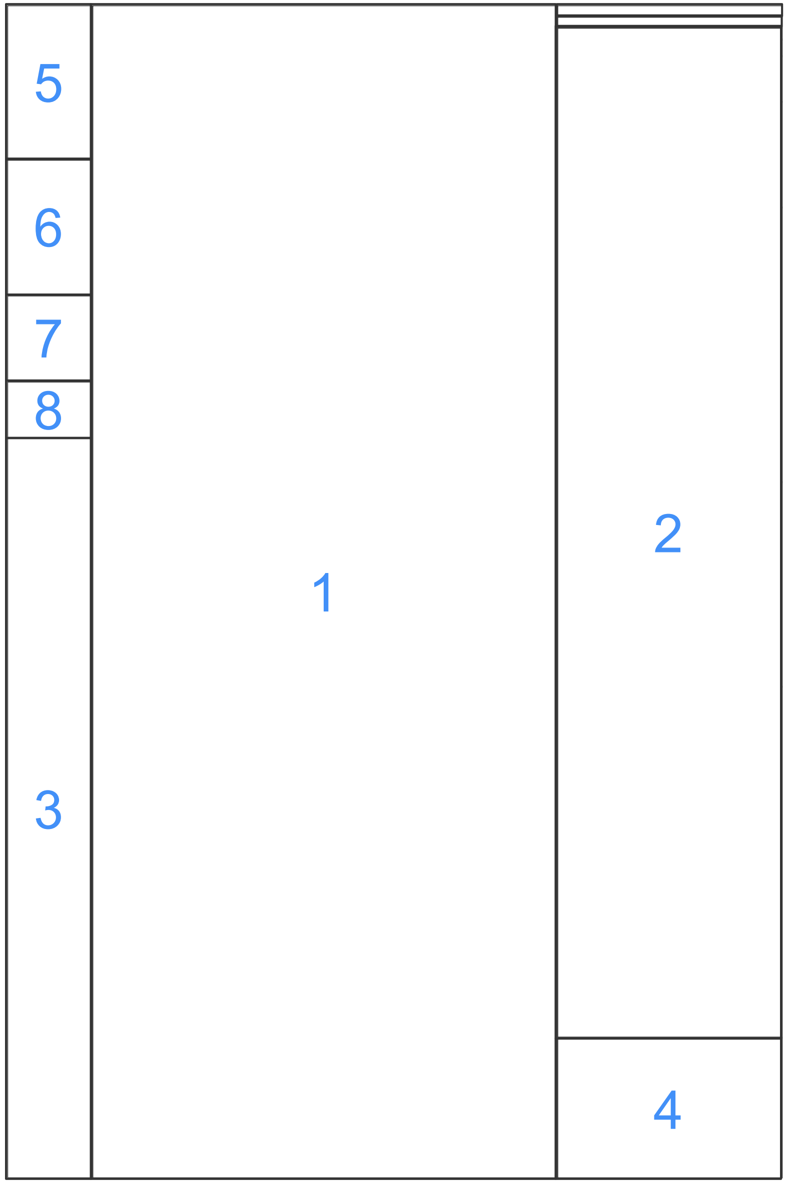
Flächendeckendes Aufräumen der Codebasis?

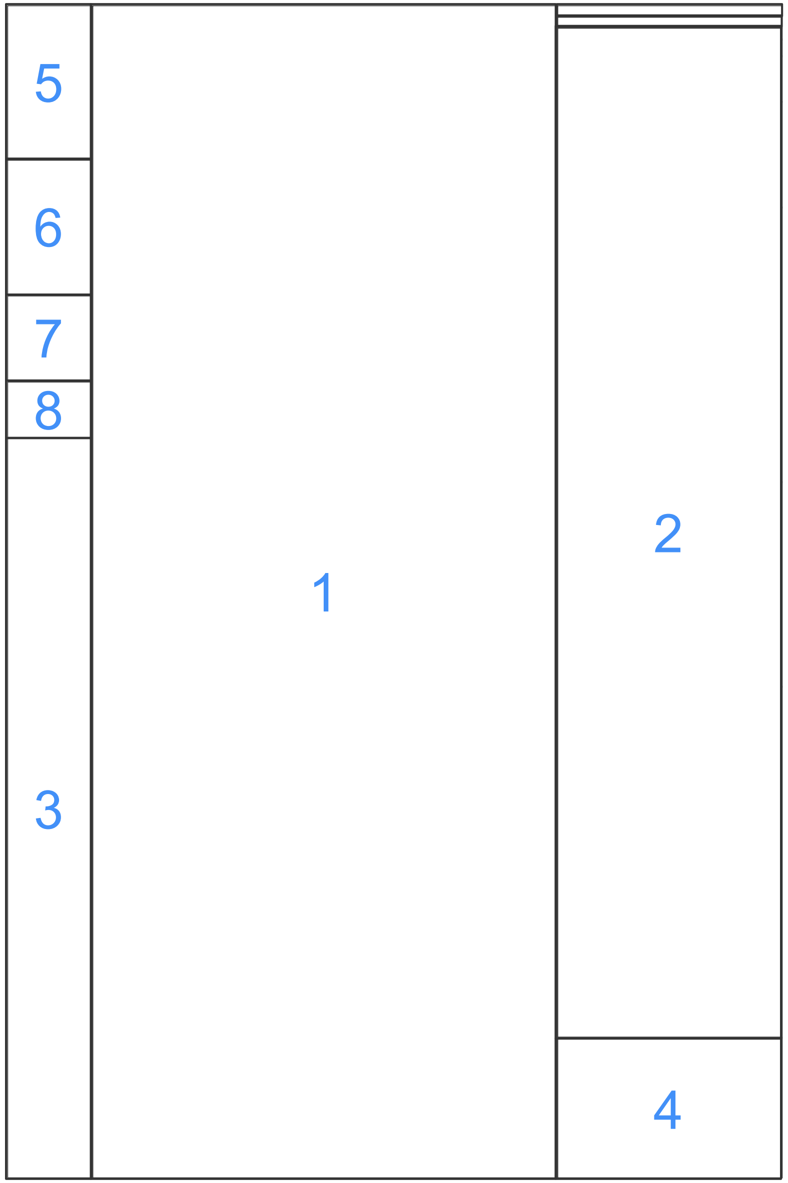
Ja, ca. ⅔ der Fixes sind nützlich

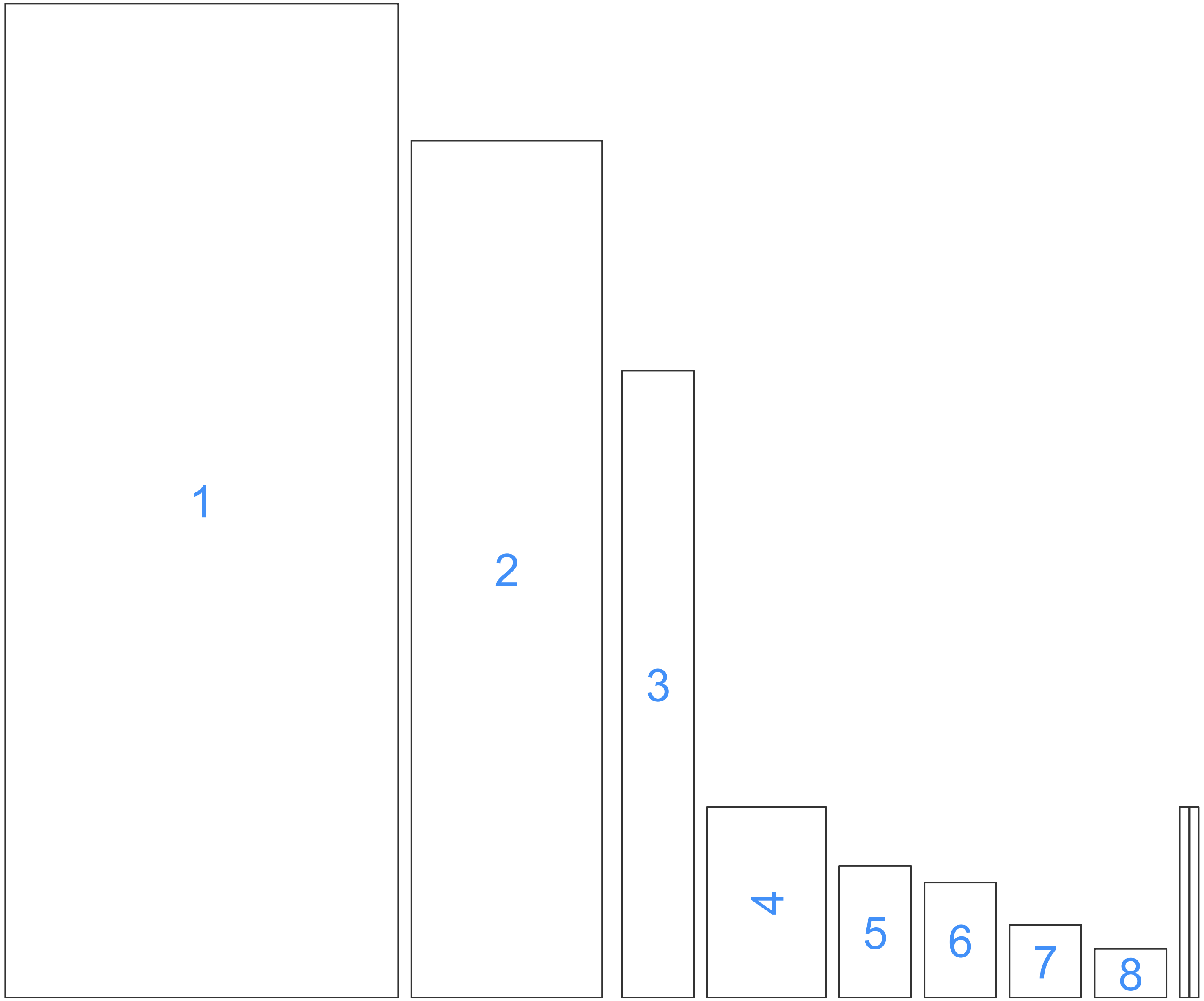
Nein, ca. ⅓ ist unbrauchbar

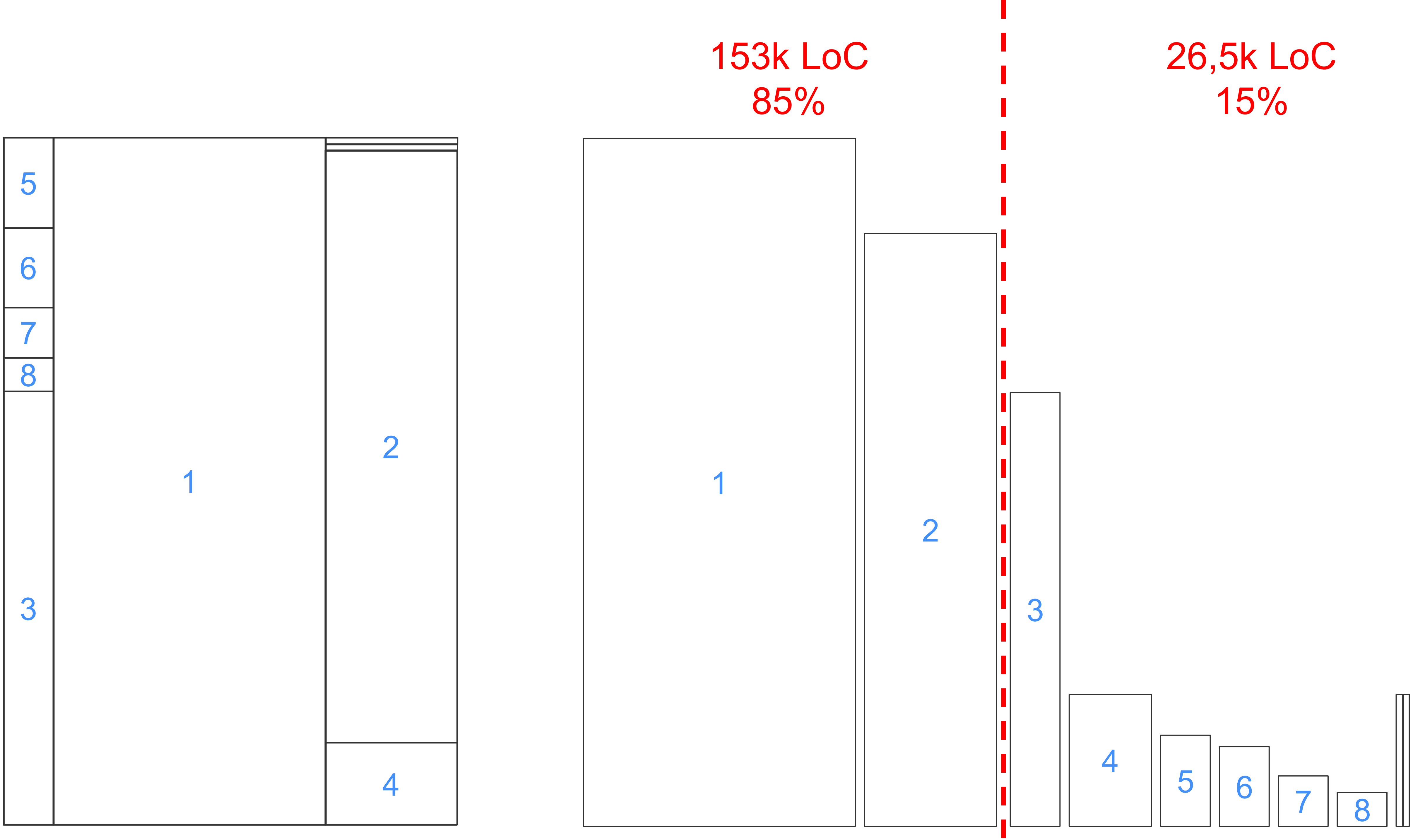


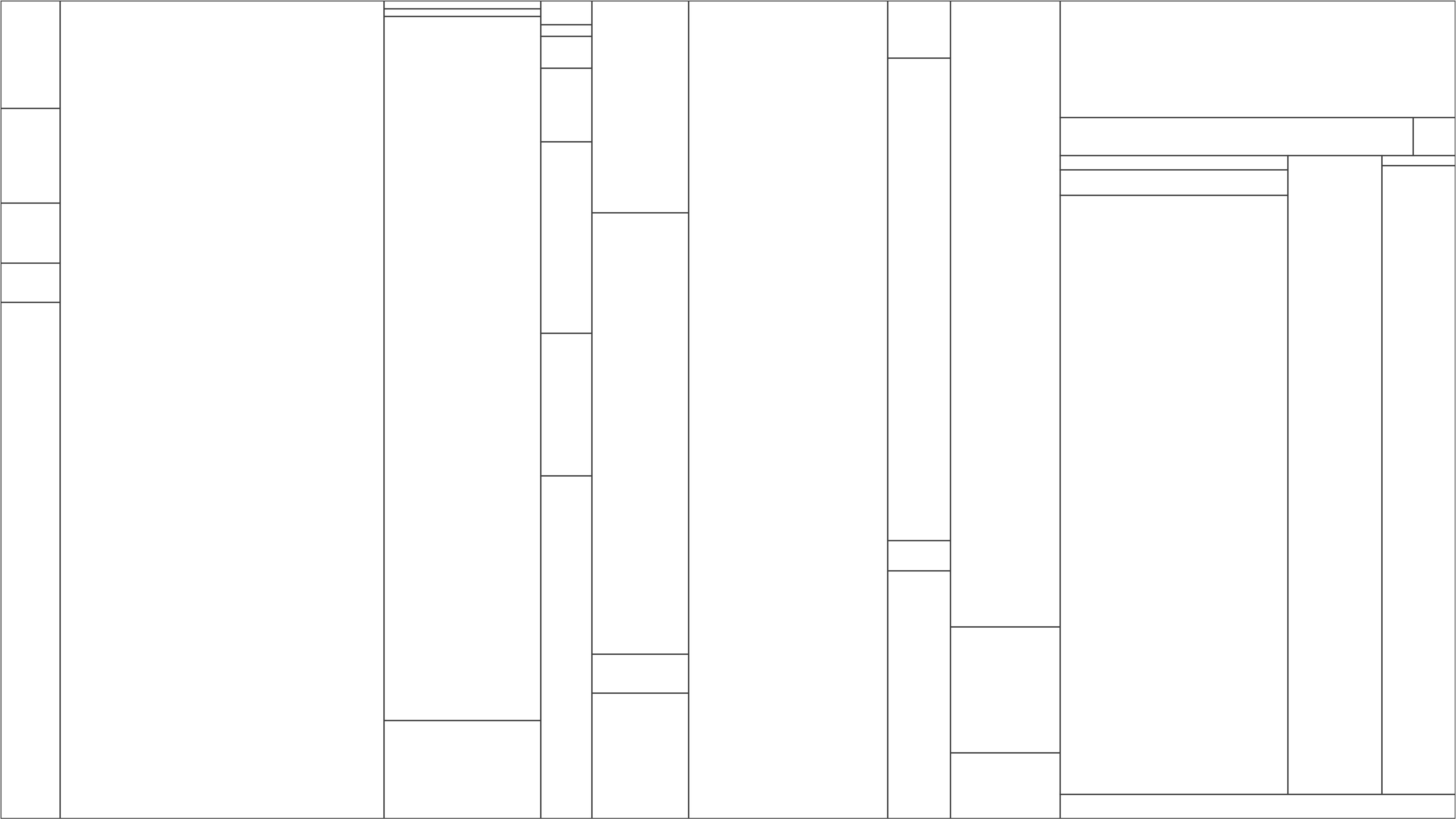

179k LoC

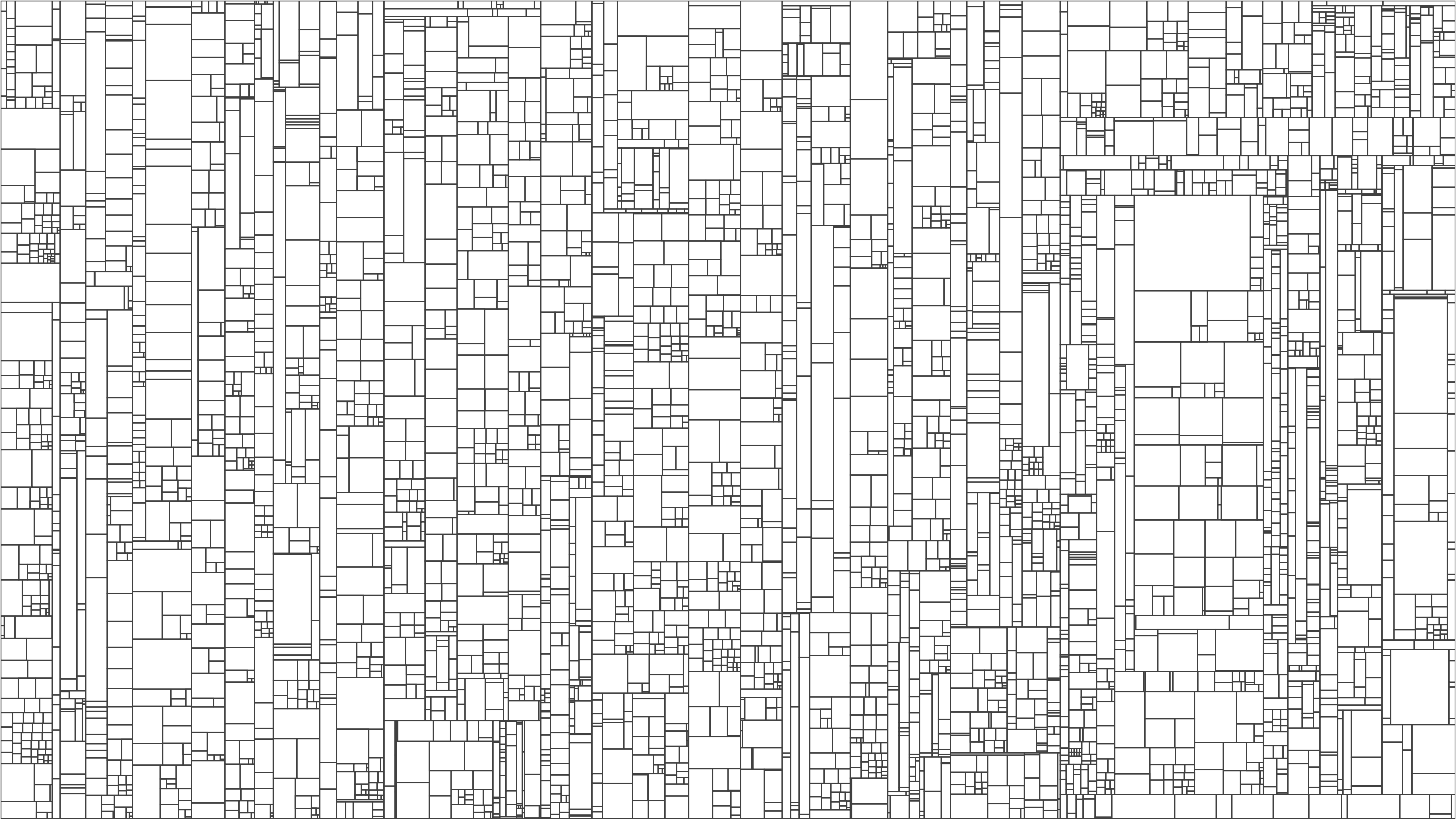

5
6
7
8
1
2
3
4

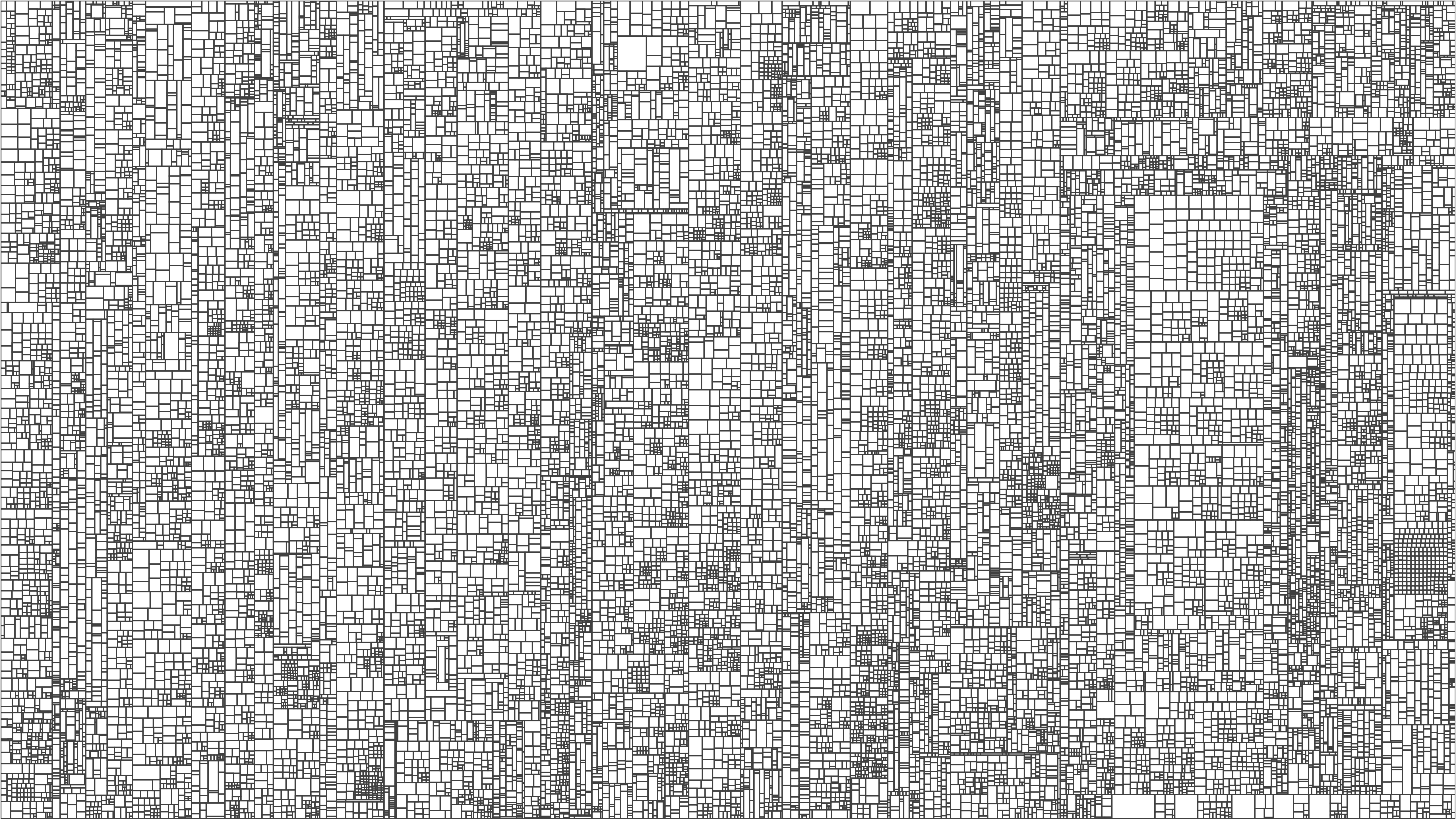

5
6
7
8
1
2
3
4

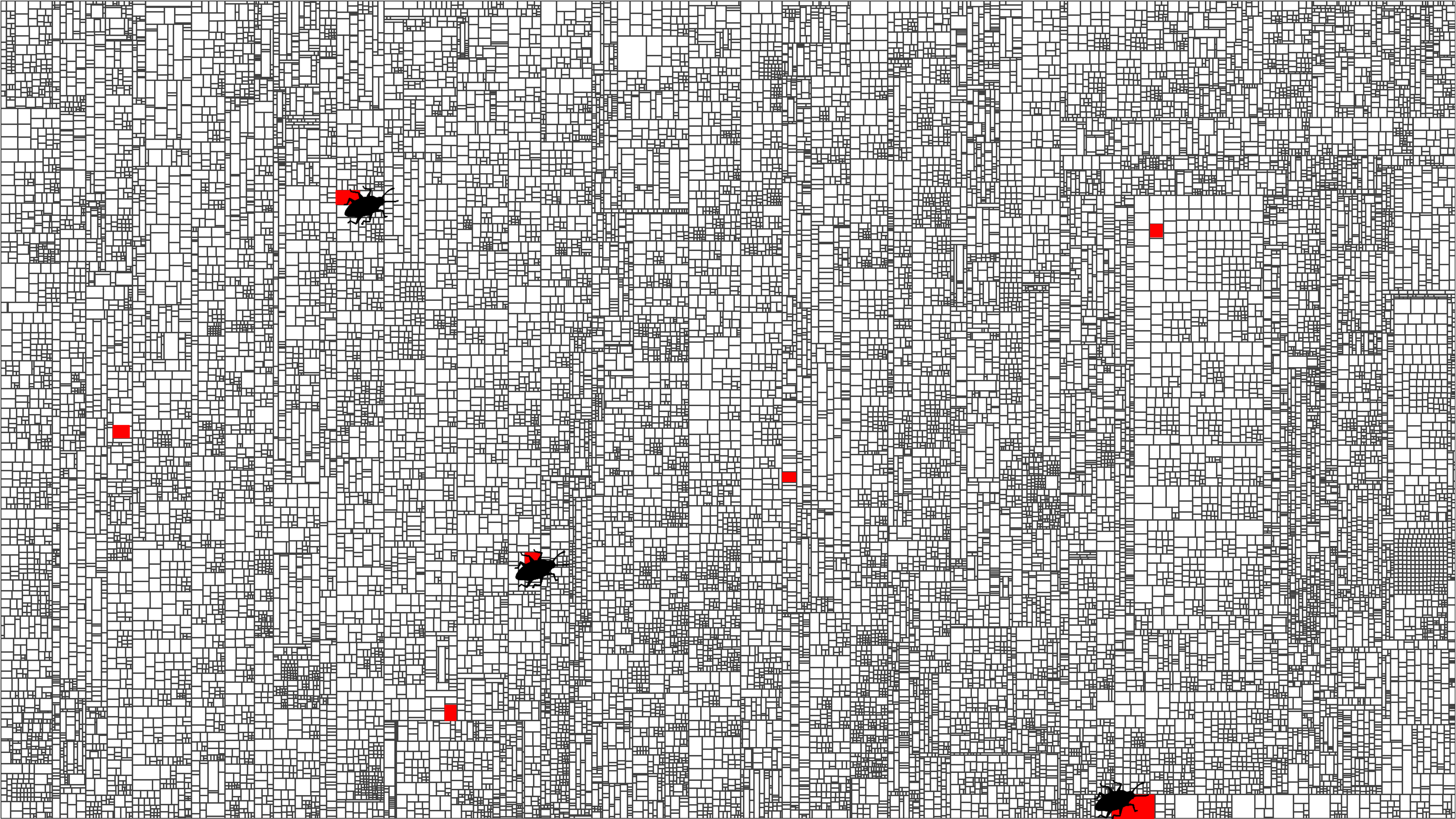

1
2
3
4
5
6
7
8

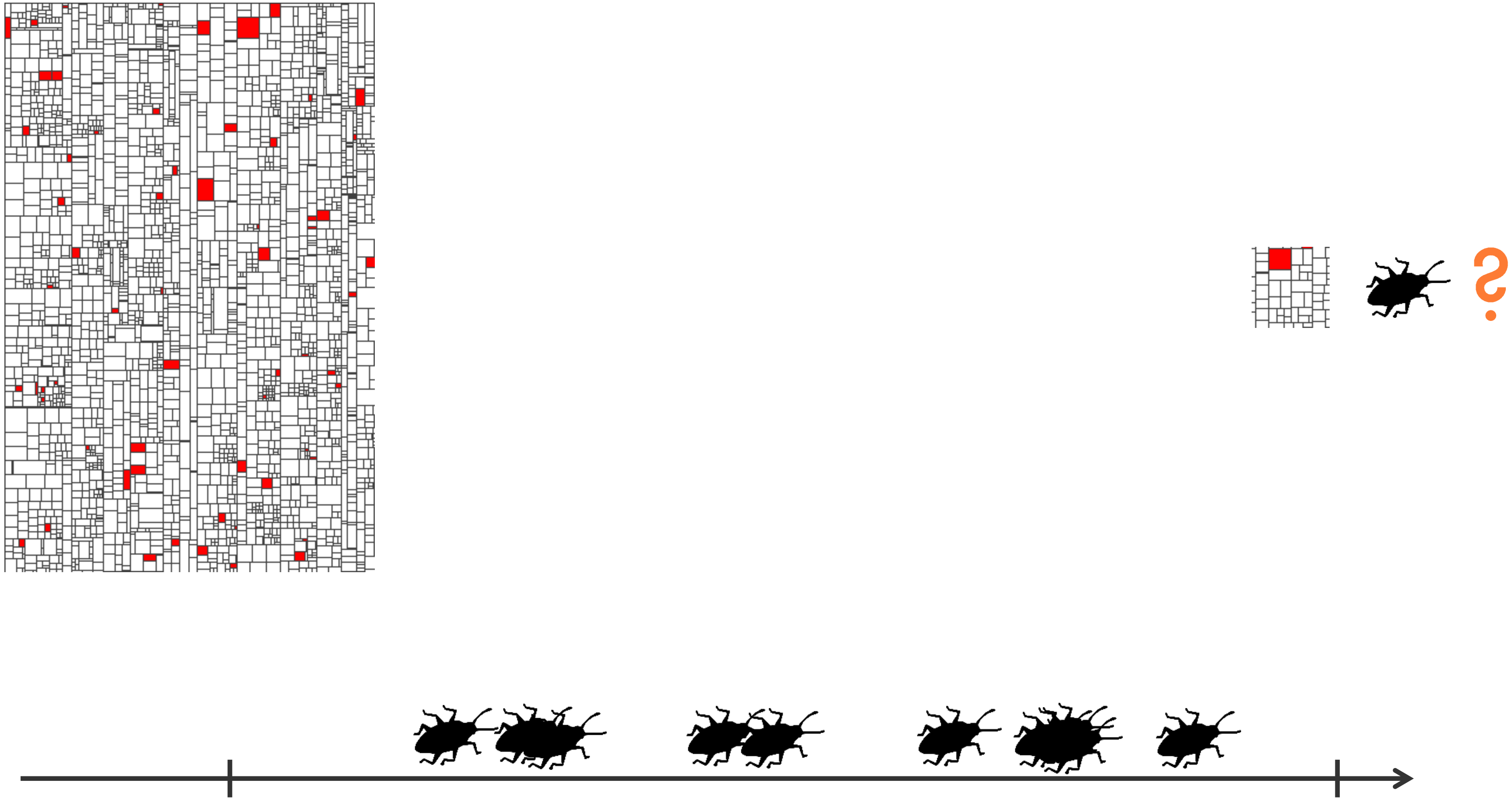

5
6
7
8
1
2
3
4
153k LoC
85%
26,5k LoC
15%
1
2
3
4
5
6
7
8





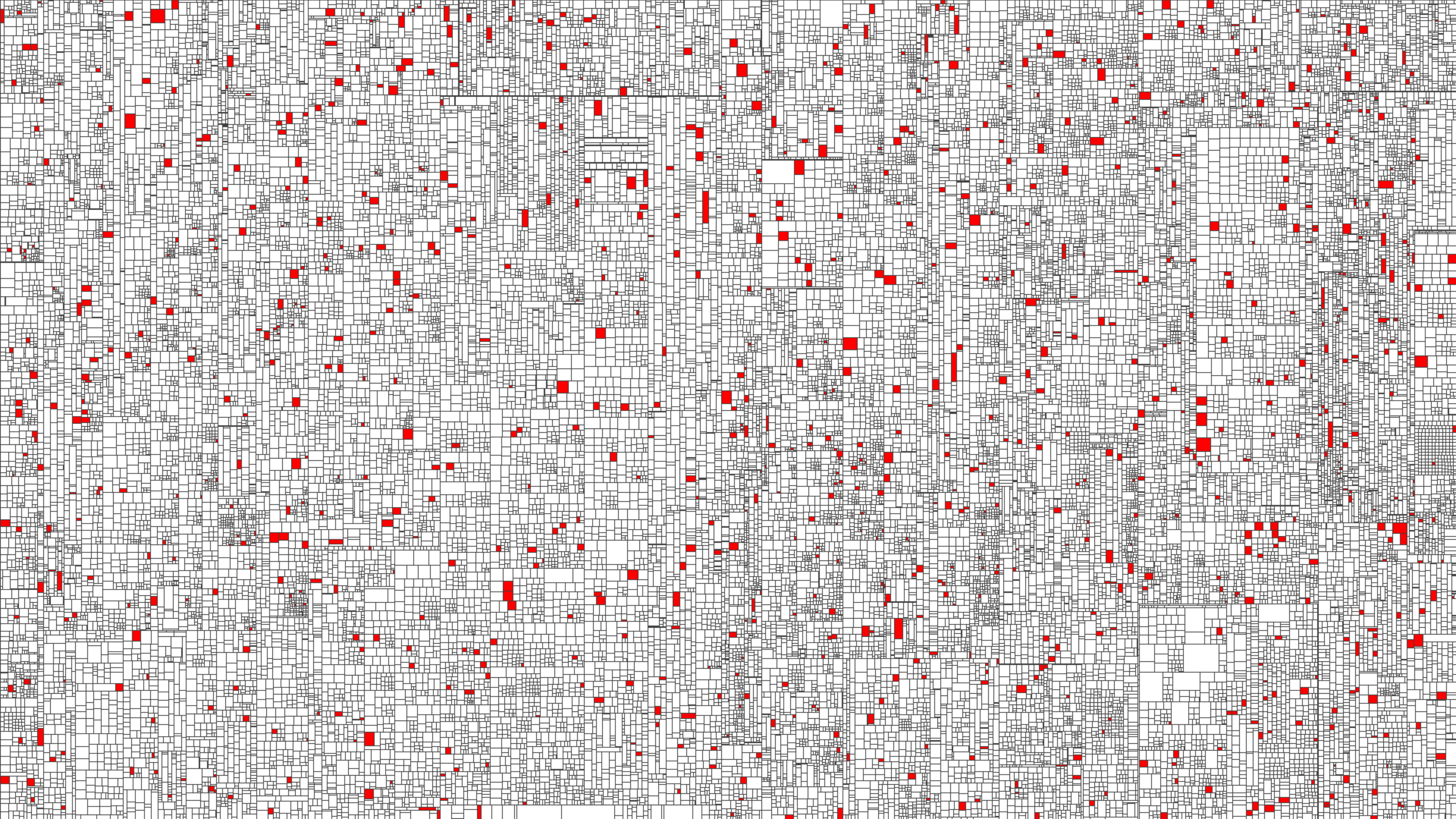


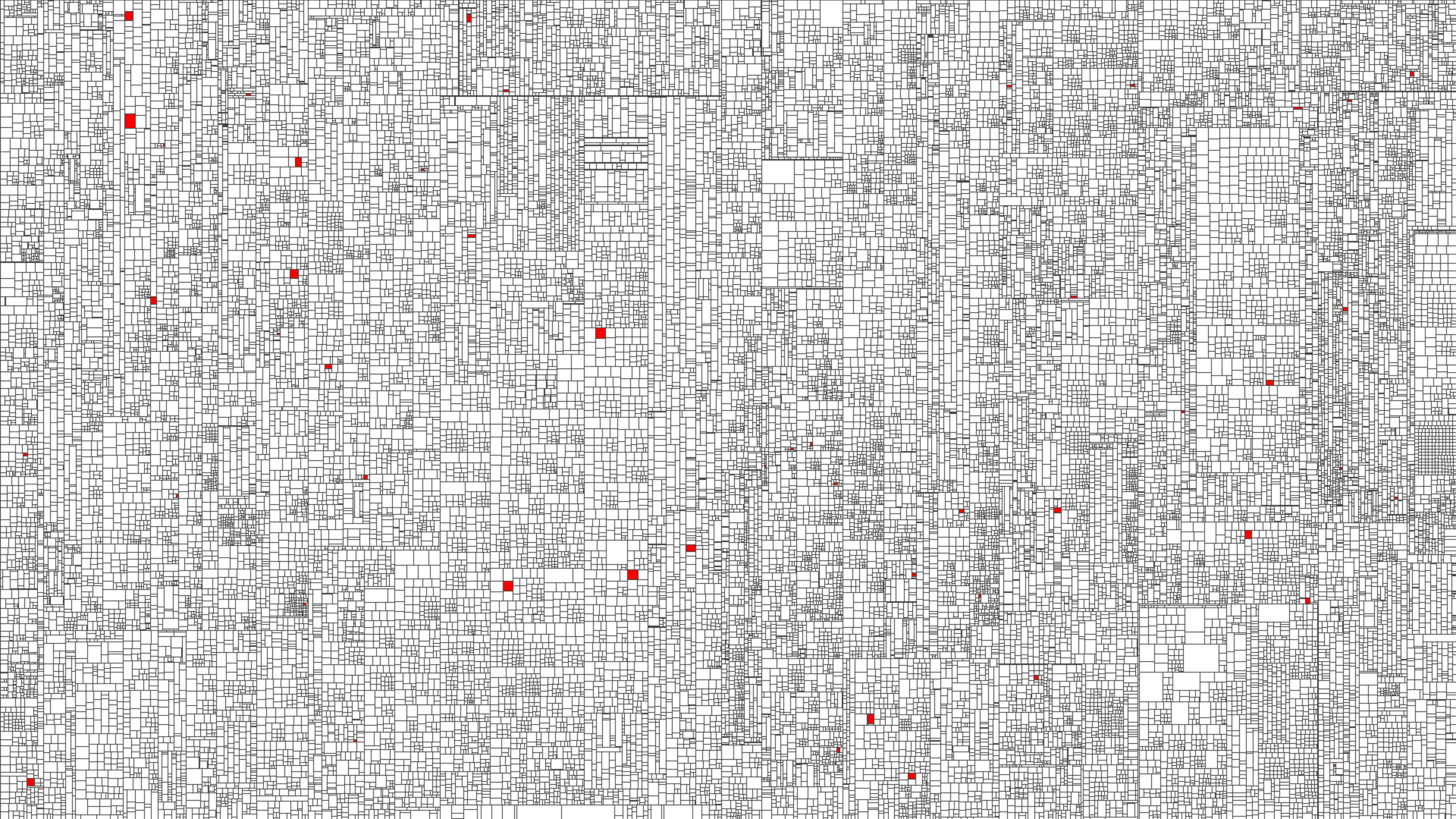




Release

Time of Study



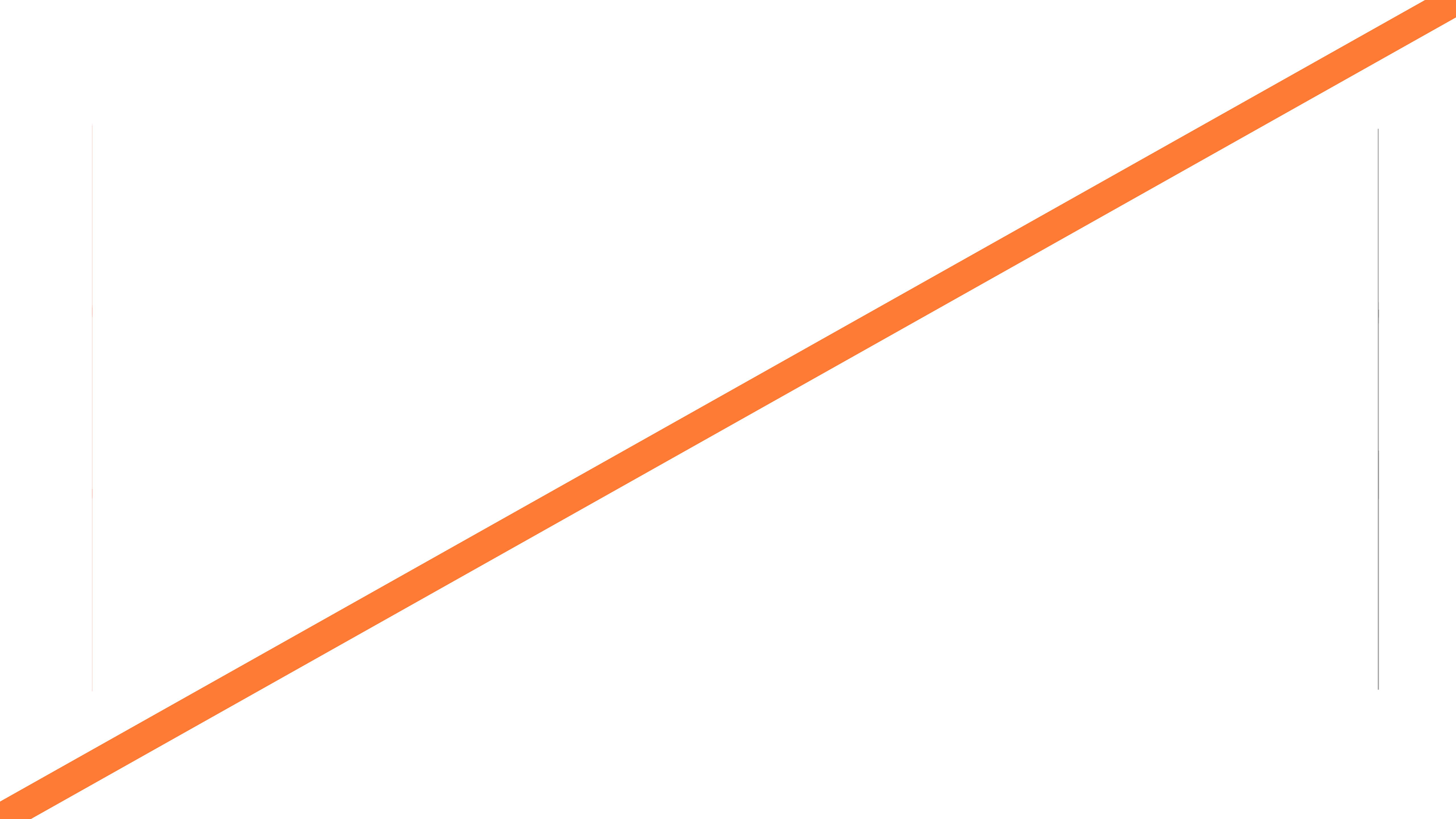


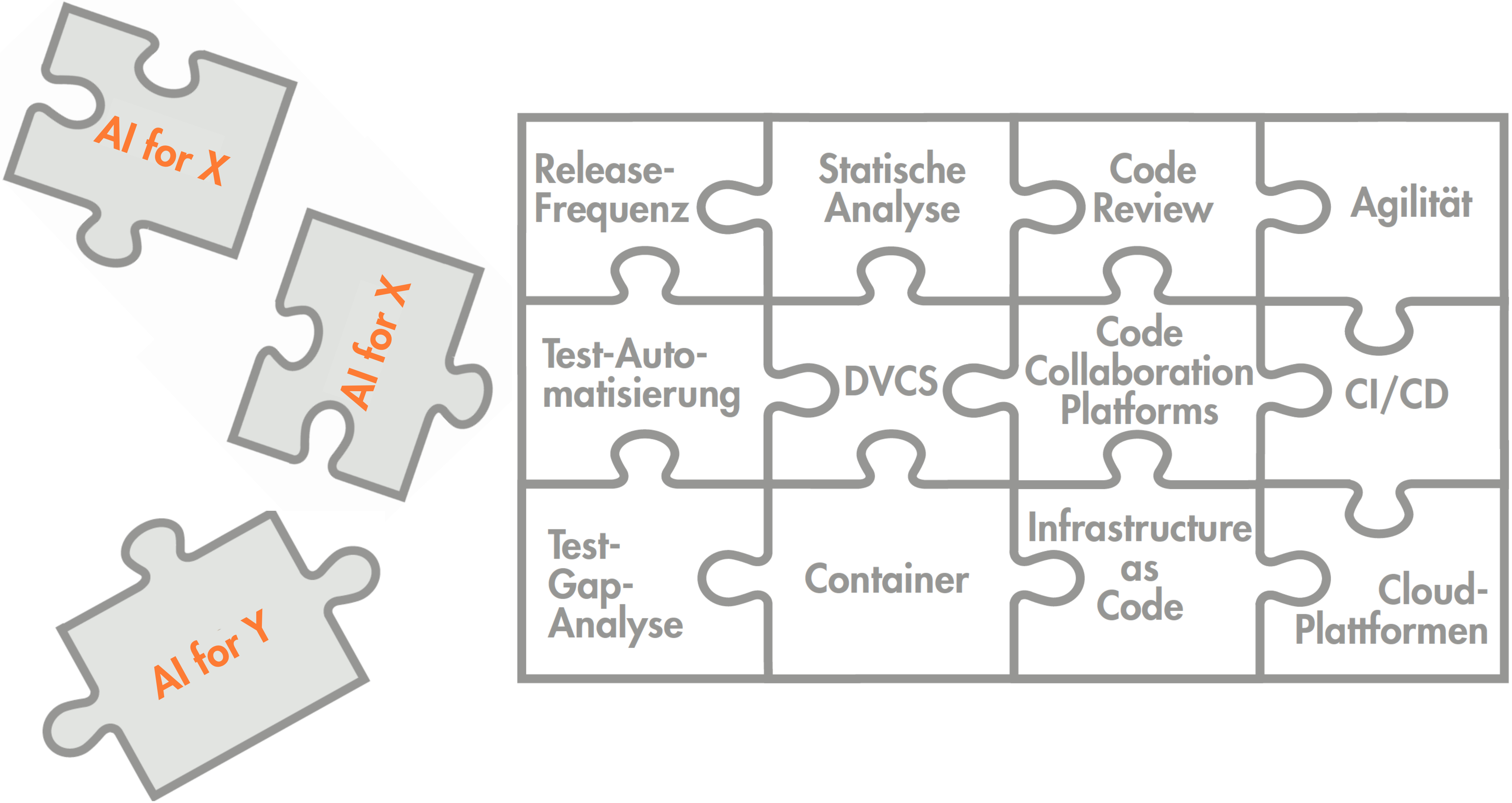
Evaluation

Release	# „defect prone“ Methods	# Bugs (Top 50)
1.4:	1127	
2.0:	1176	

Pascarella, Palomba, Bacchelli, *Re-evaluating Method-Level Bug Prediction*, 2018:
Prediction nicht besser als zufällige Klassifikation.

Chowdhury, Uddin, Hemmati, Holmes, *Method-Level- Bug Prediction: Problems and Promise*, 2024:
Method-Level Bug Prediction performance „extremely poor“.





AI for X

AI for X

AI for Y

Release-Frequenz

Statische Analyse

Code Review

Agilität

Test-Automatisierung

DVCS

Code Collaboration Platforms

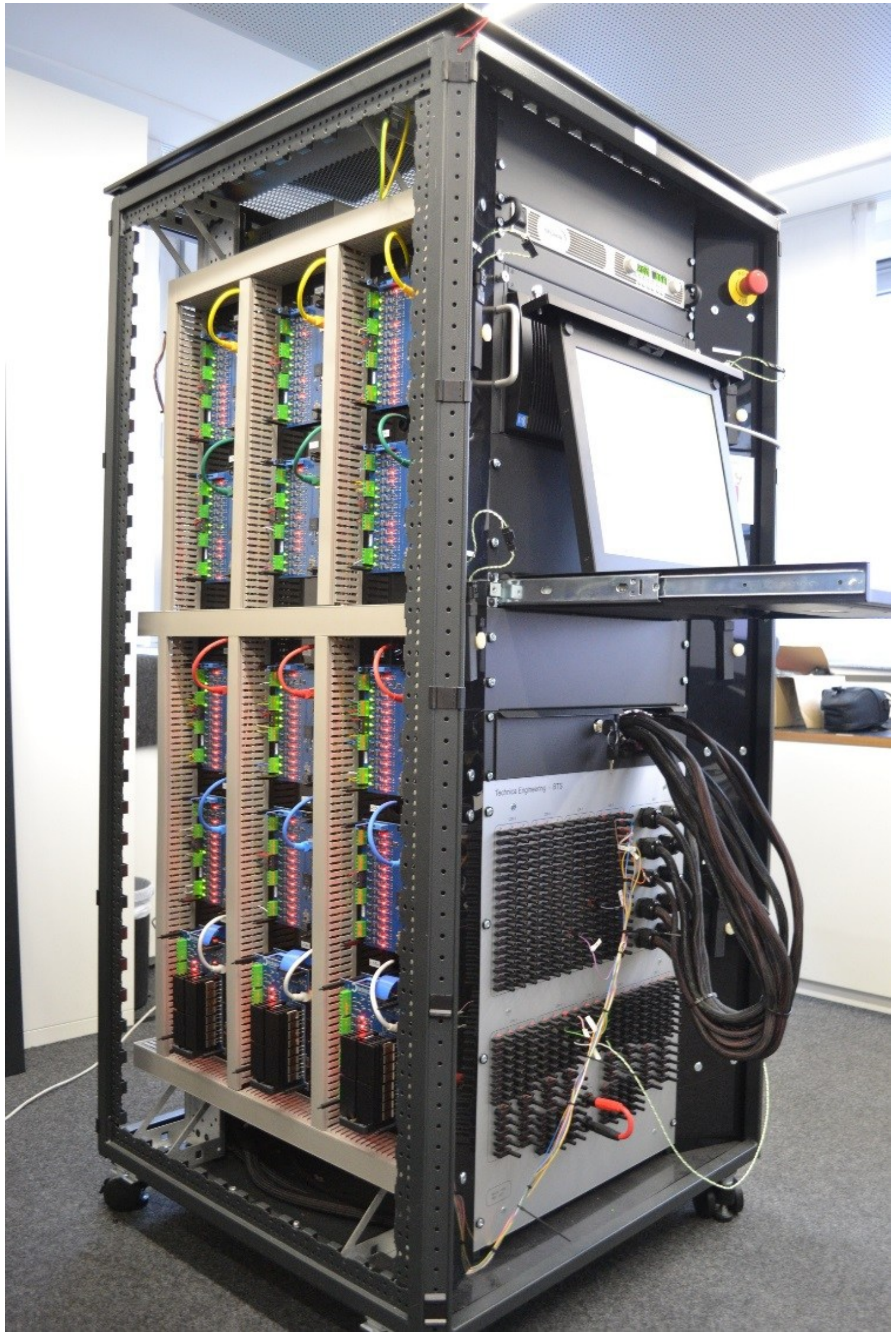
CI/CD

Test-Gap-Analyse

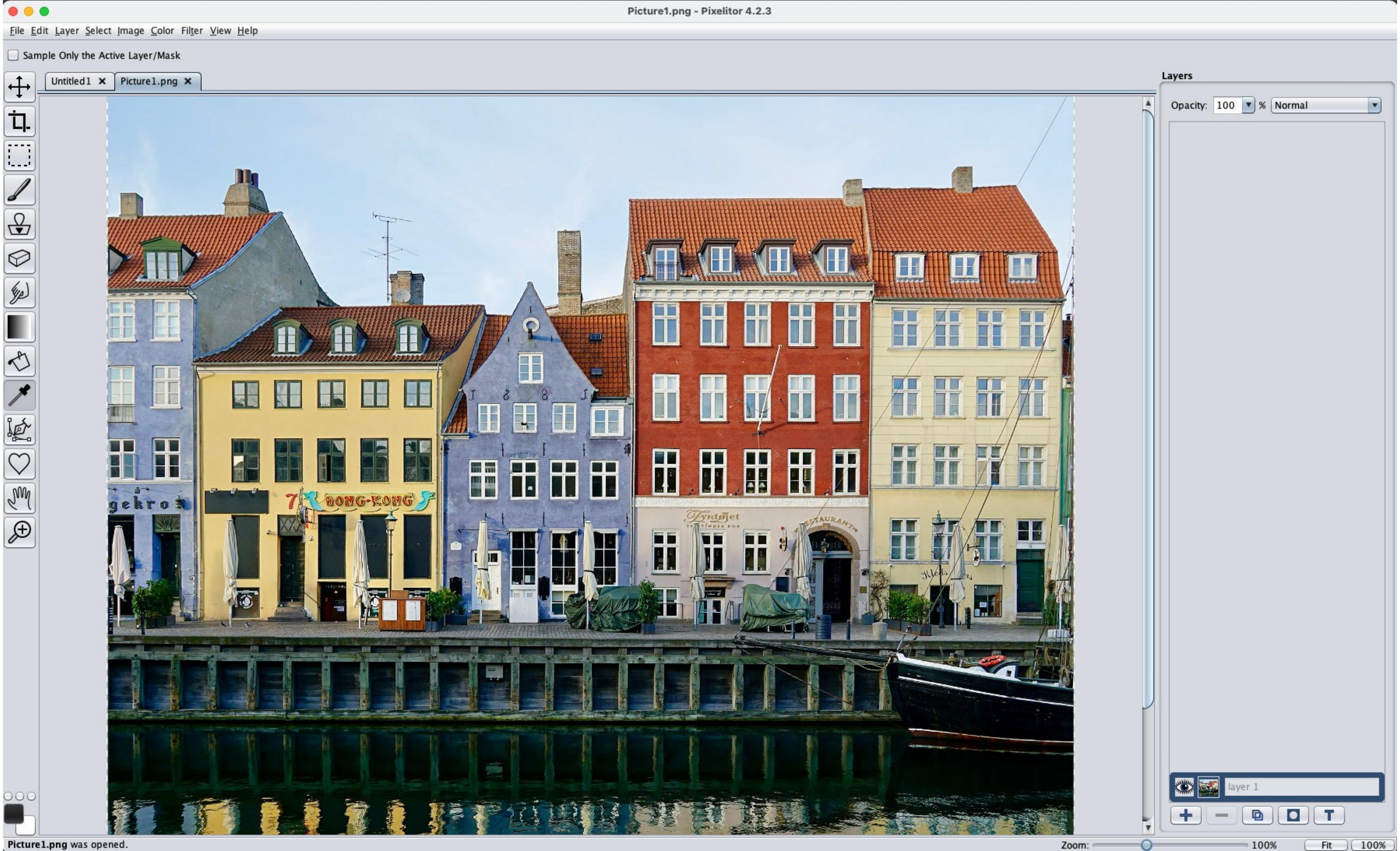
Container

Infrastructure as Code

Cloud-Plattformen



https://www.technica-engineering.de/produkte/bts-body-electronic-test-system/

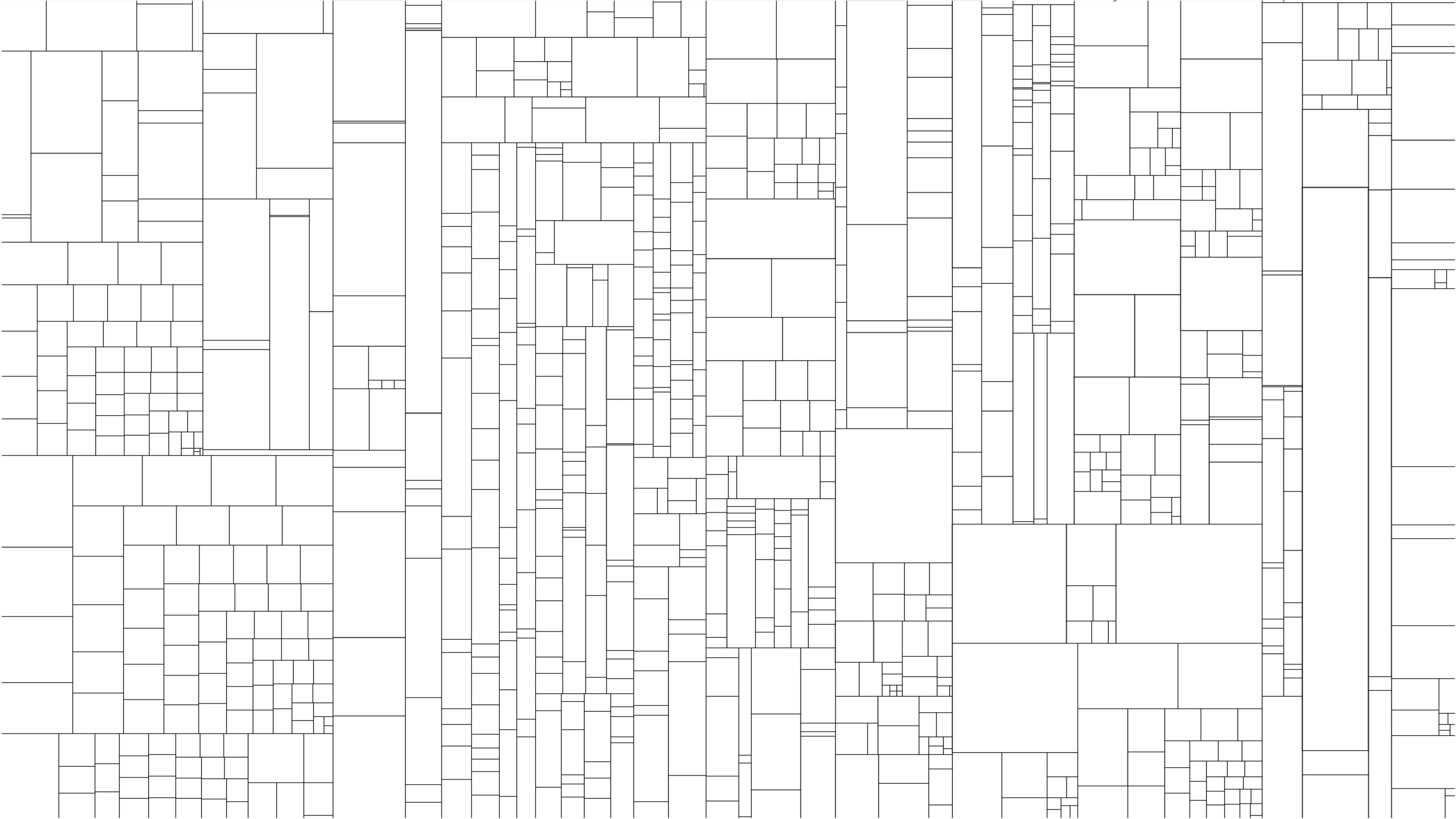


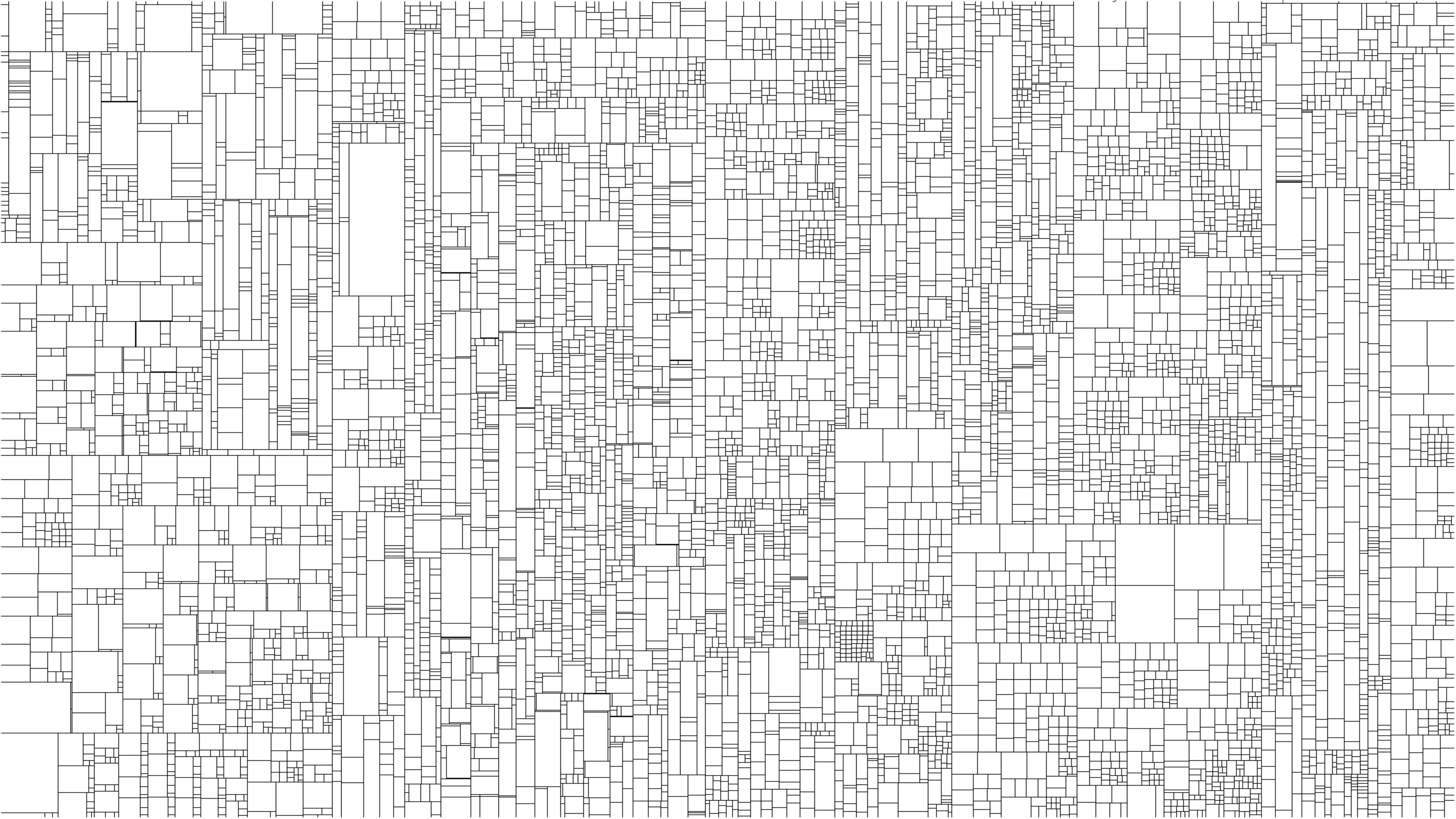
☐ Sample Only the Active Layer/Mask

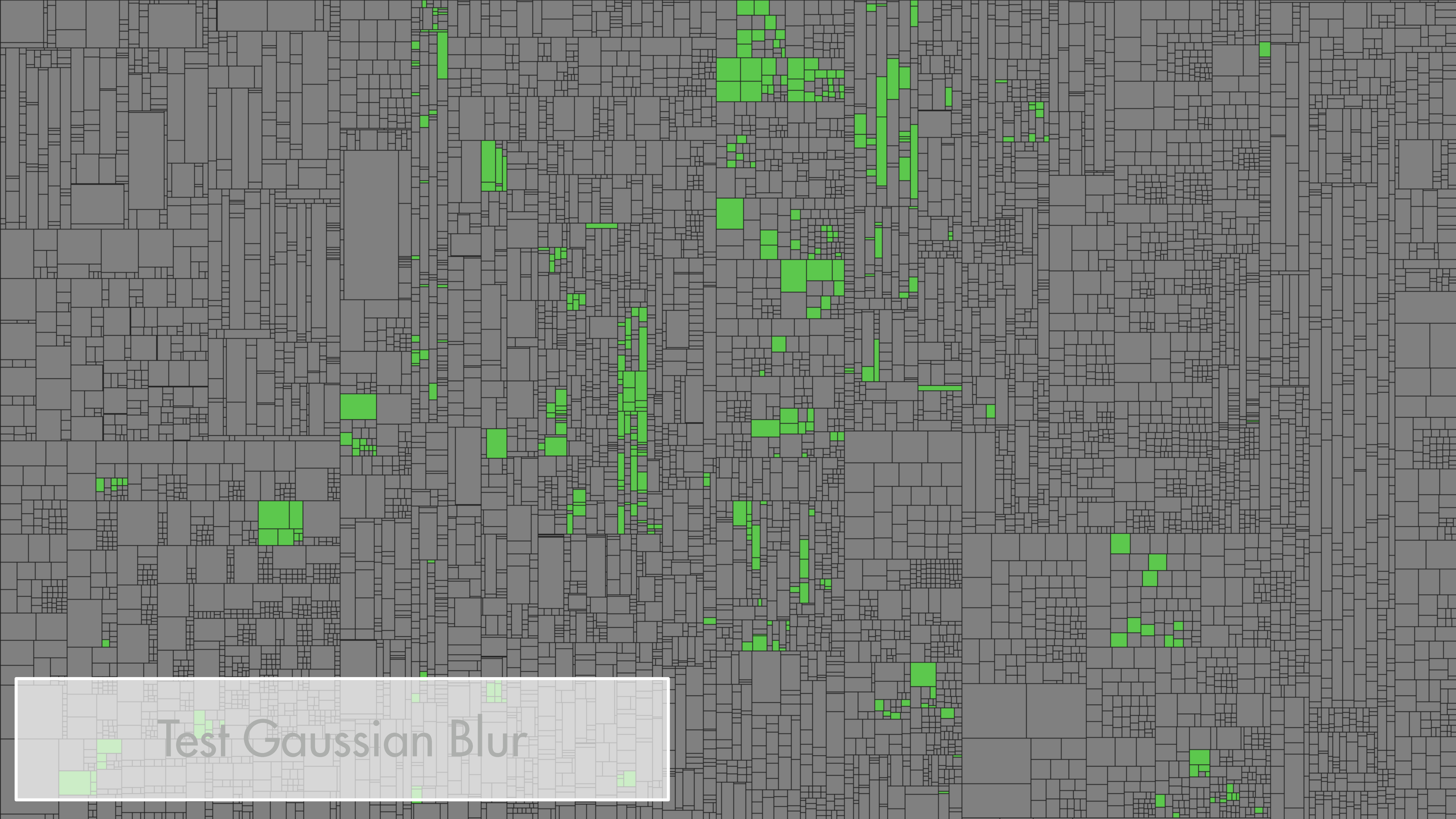
Opacity: 100 % Normal

Picture1.png was opened.

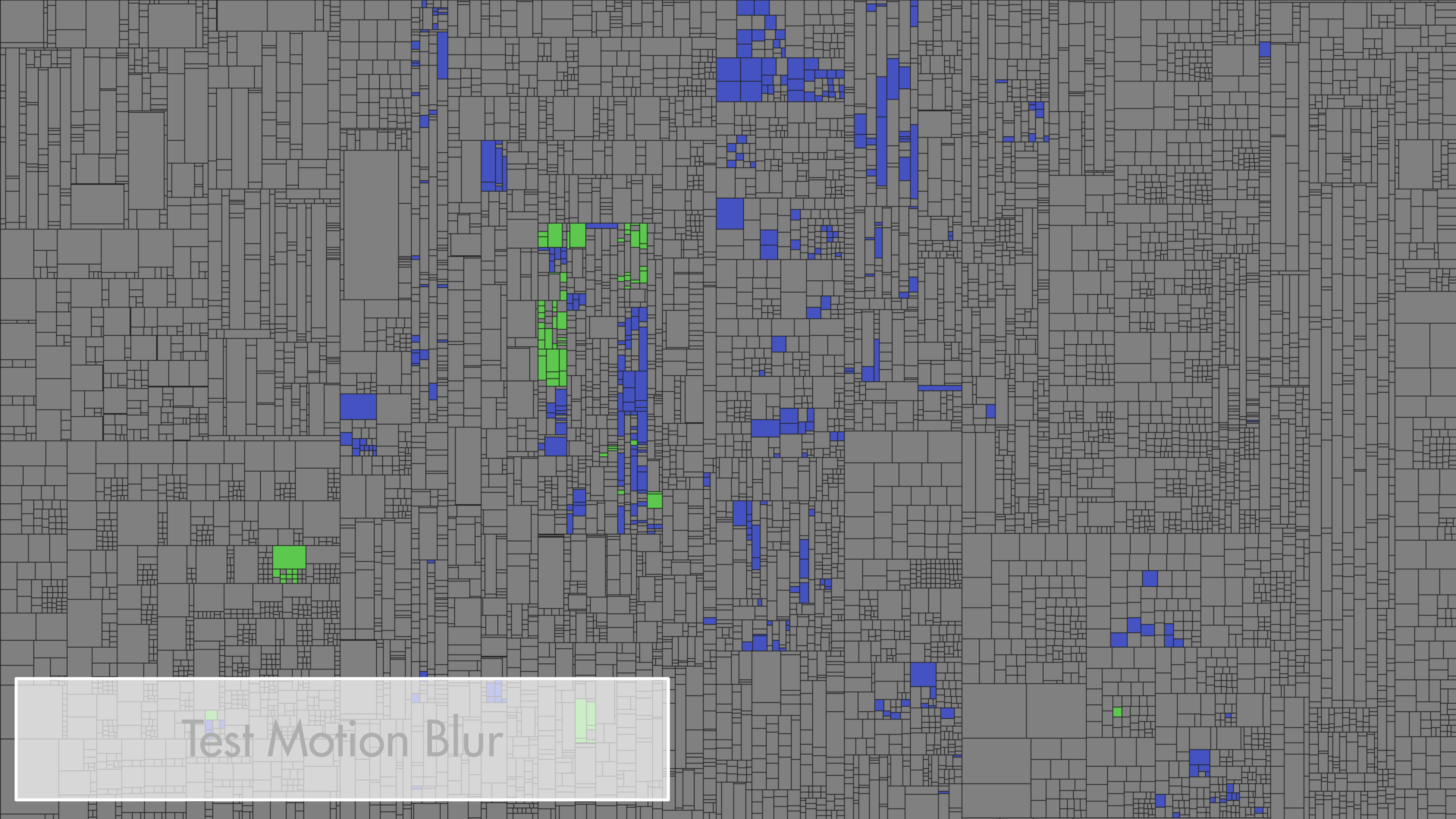
Zoom: 100%



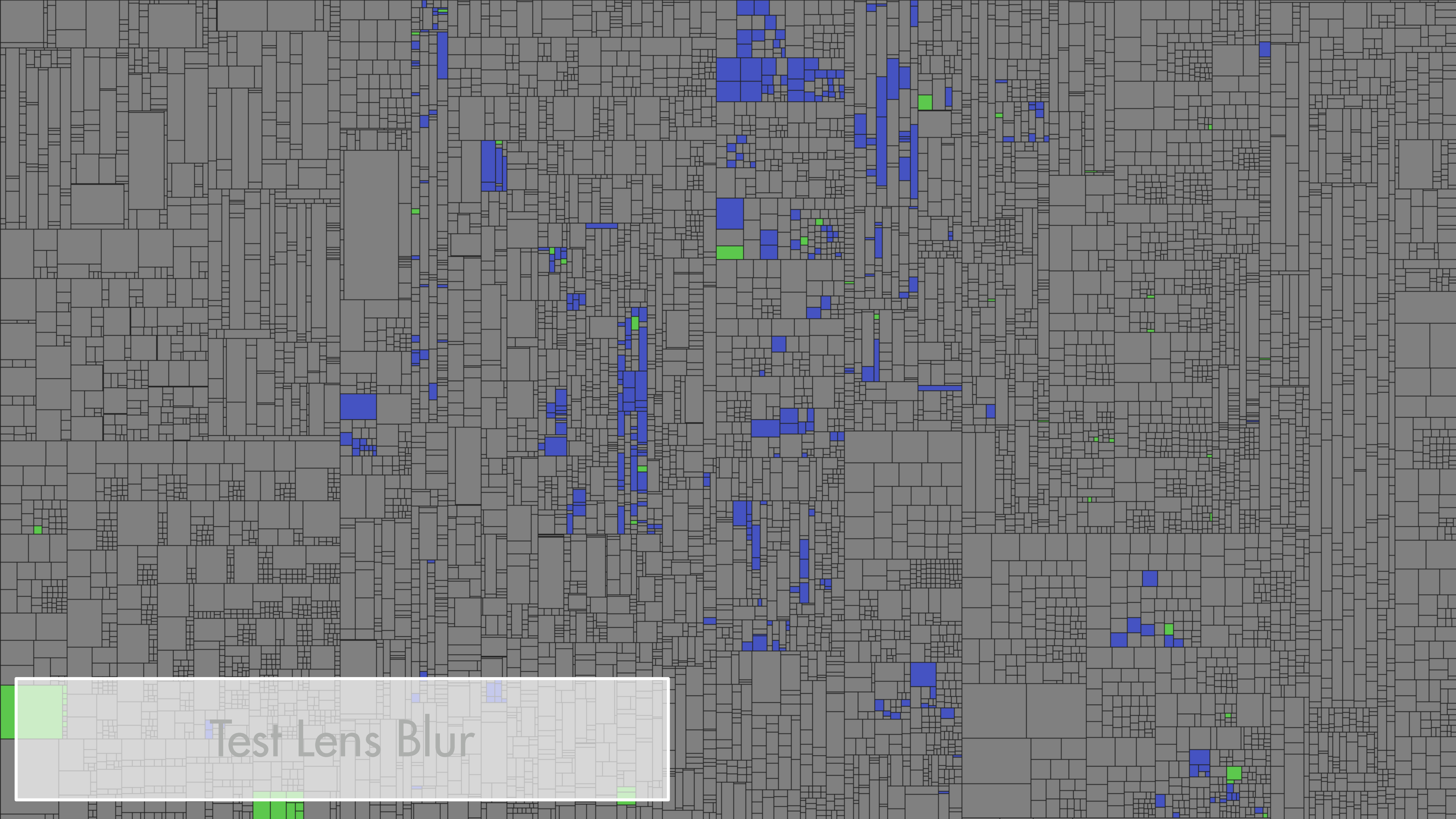




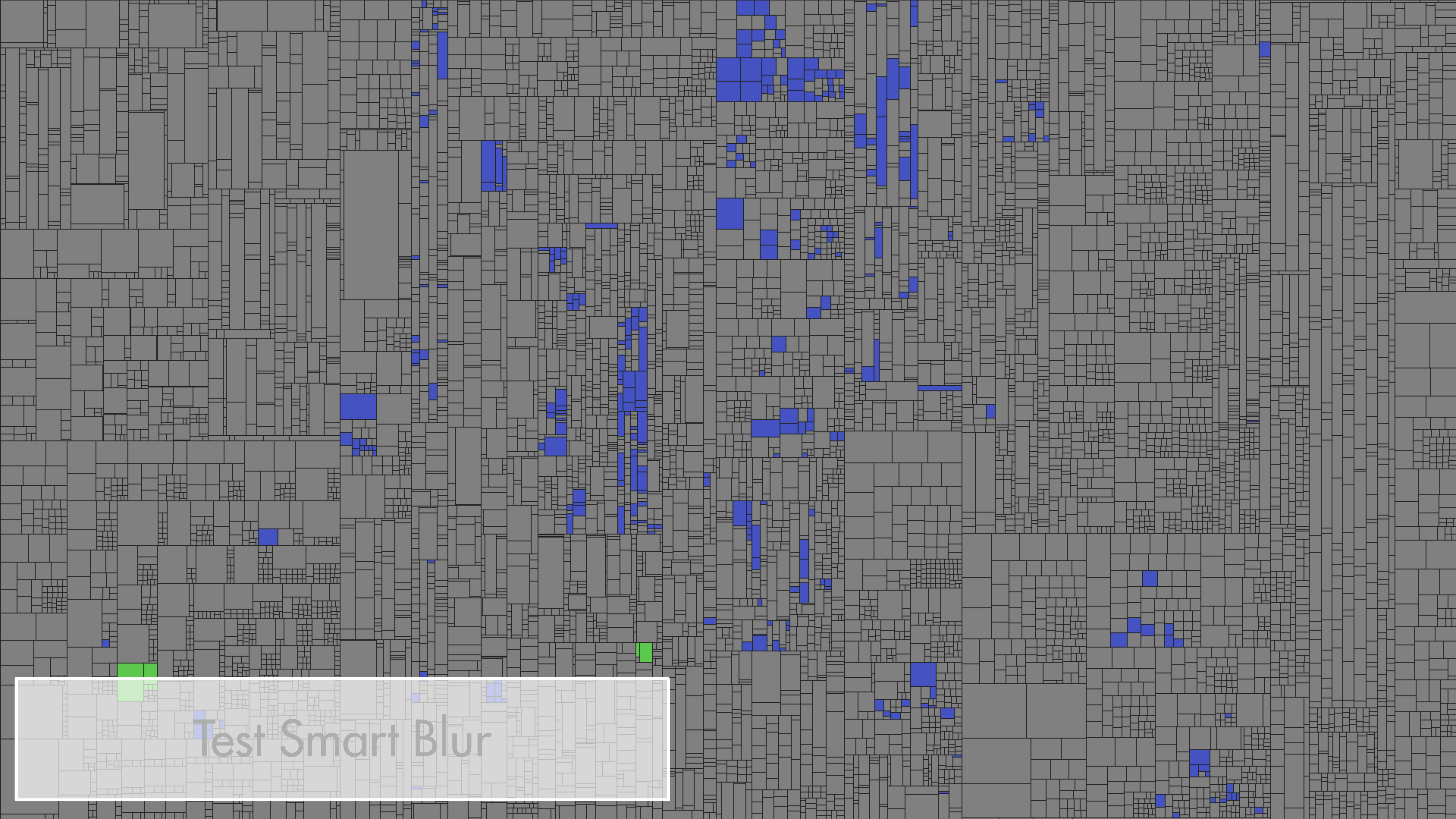
Test Gaussian Blur



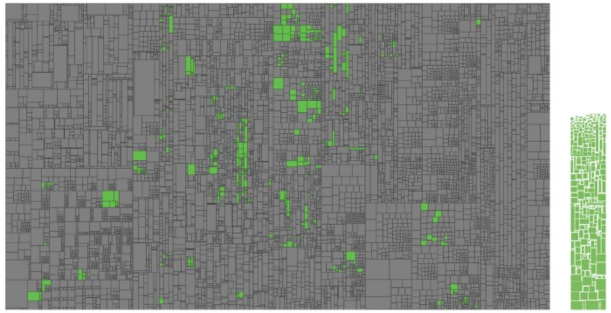
Test Motion Blur

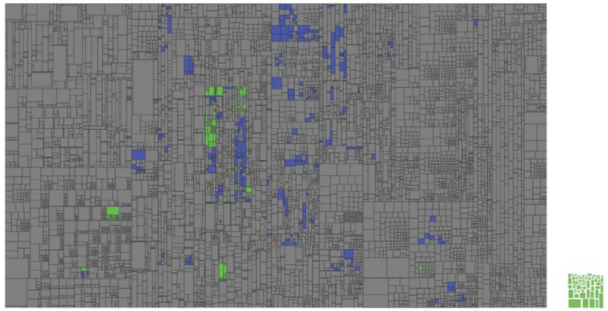


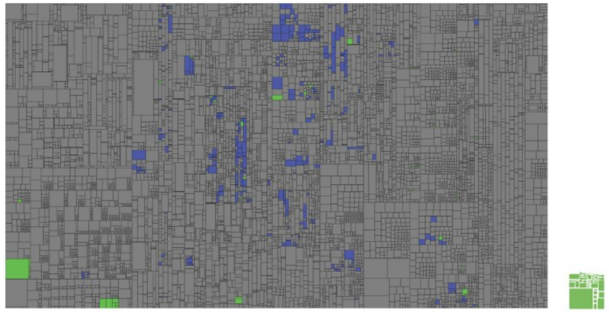
Test Lens Blur

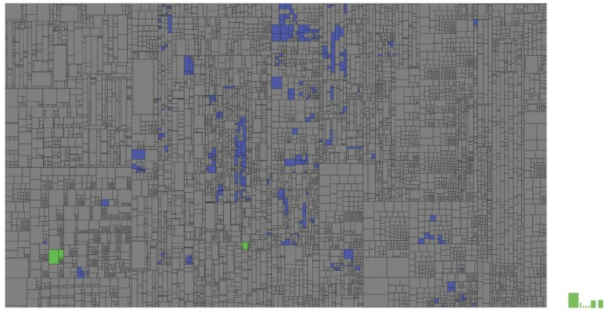


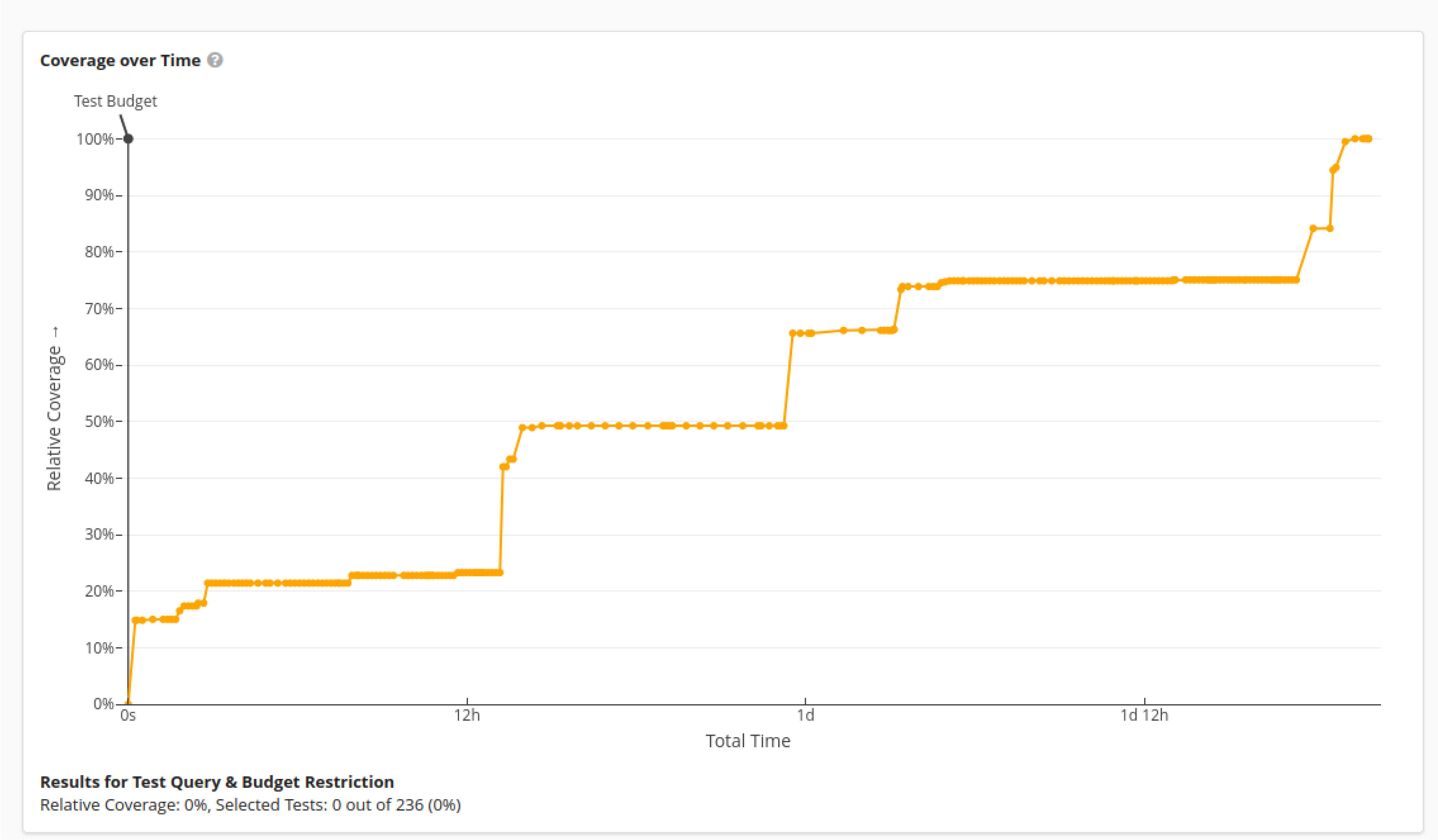
Test Smart Blur









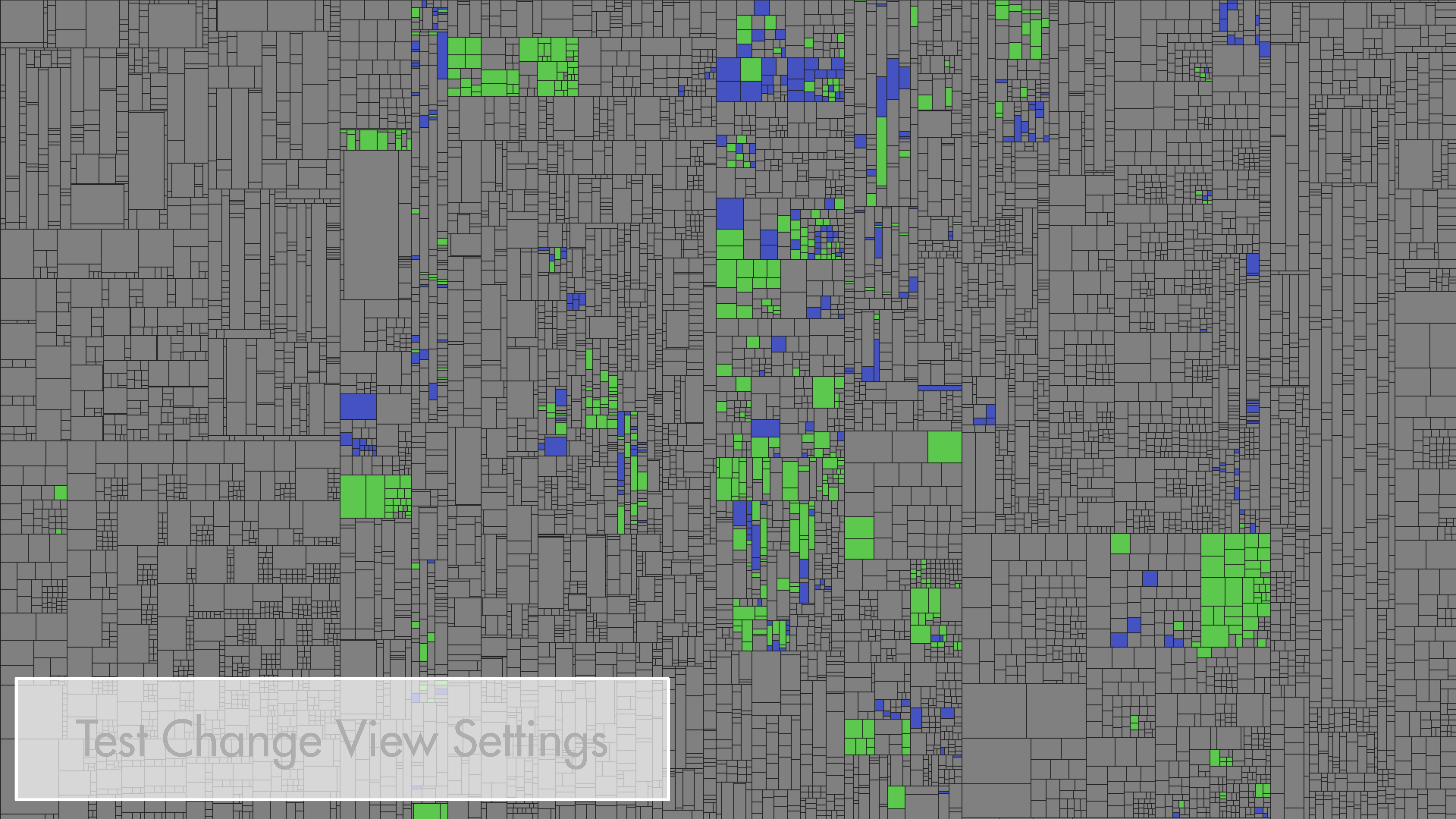


Coverage over Time

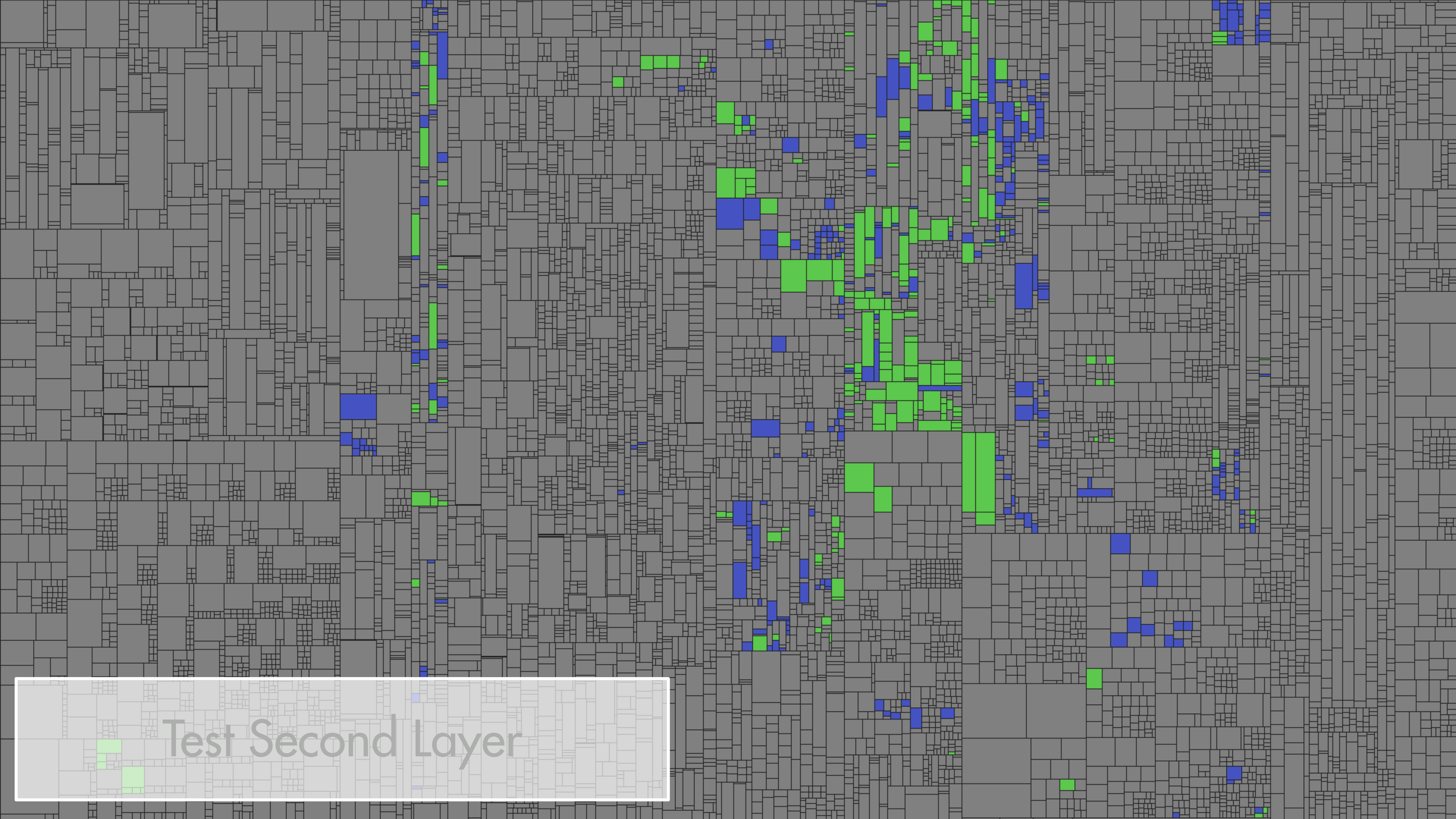
Results for Test Query & Budget Restriction

Relative Coverage: 0%, Selected Tests: 0 out of 236 (0%)

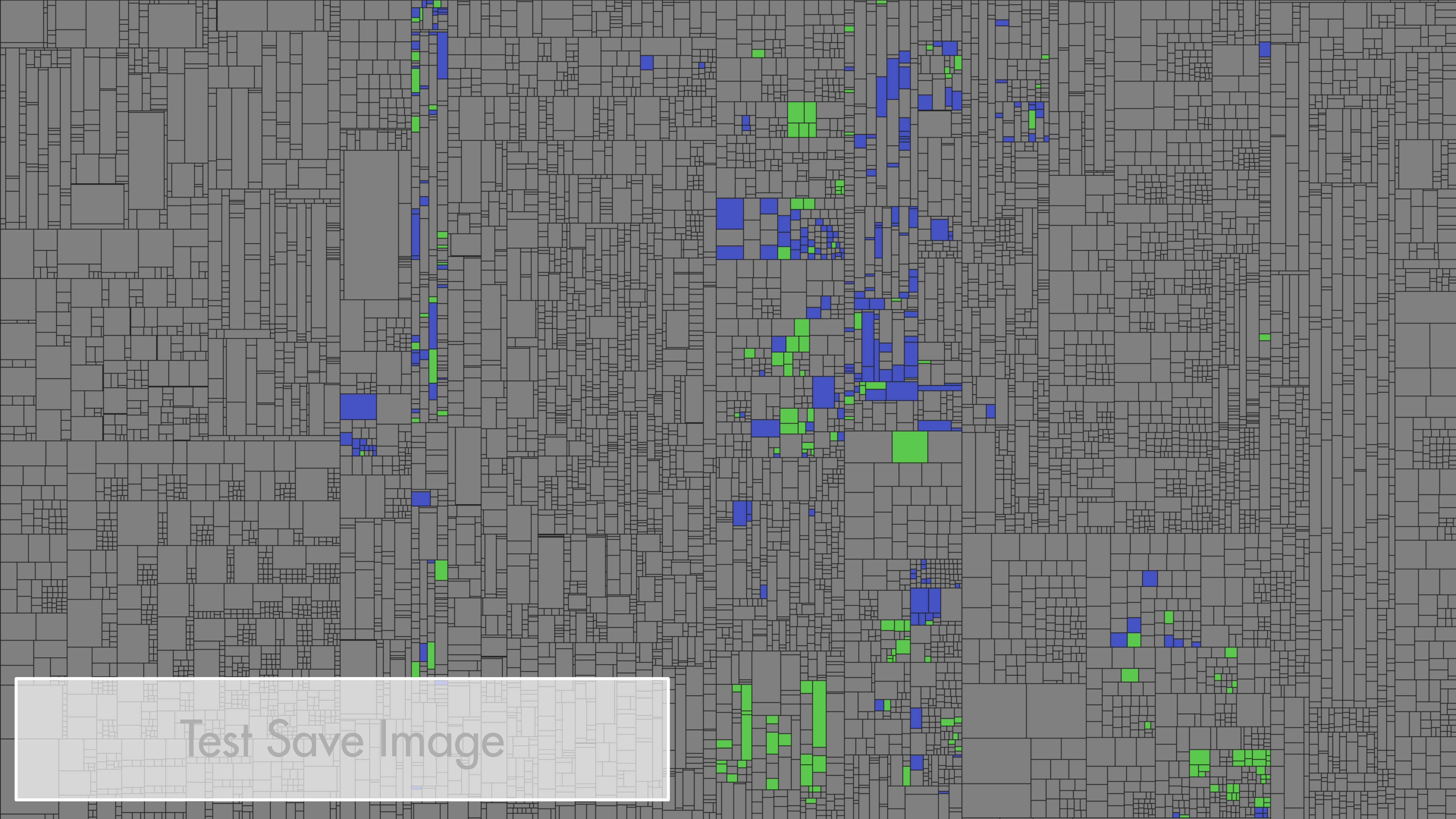
Test Create and Modify Selection

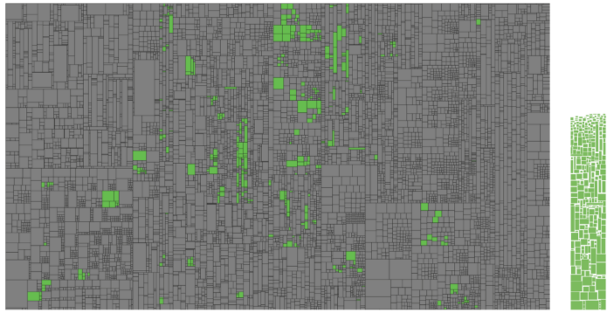


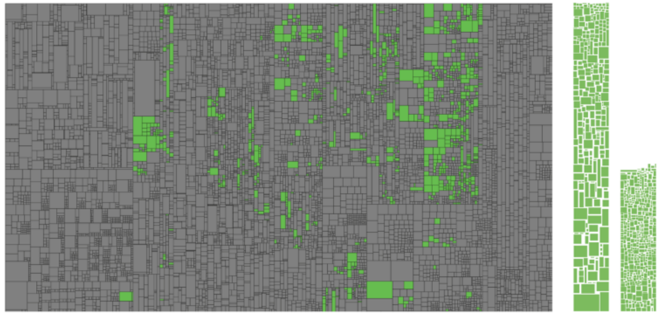
Test Change View Settings

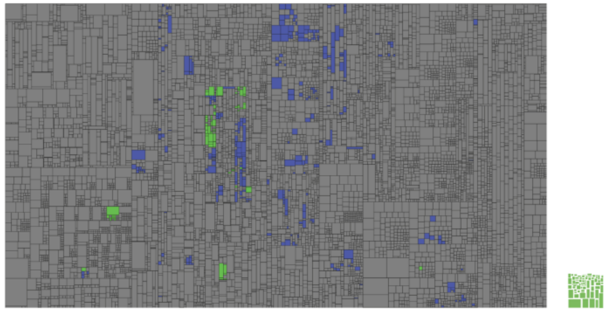


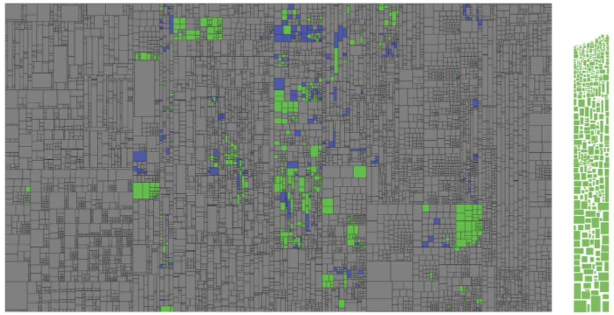
Test Second Layer

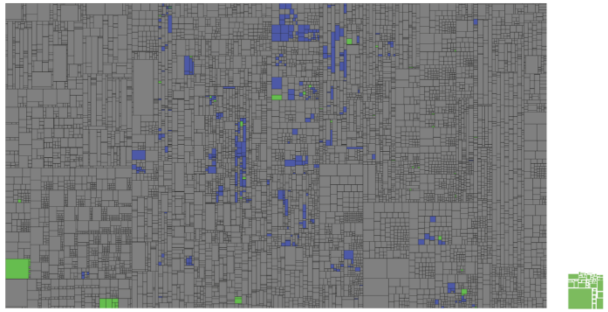

Test Save Image

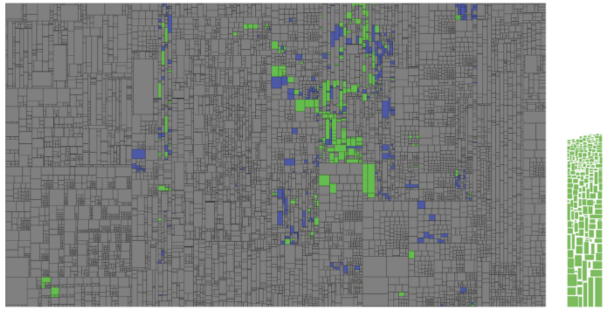


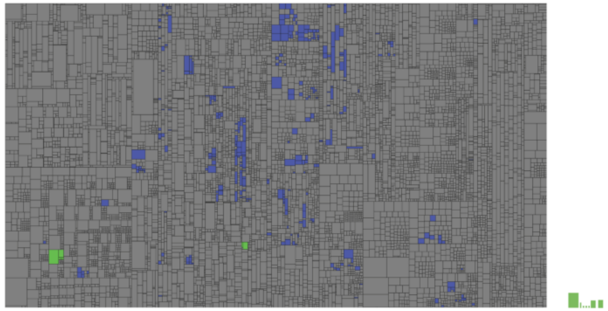


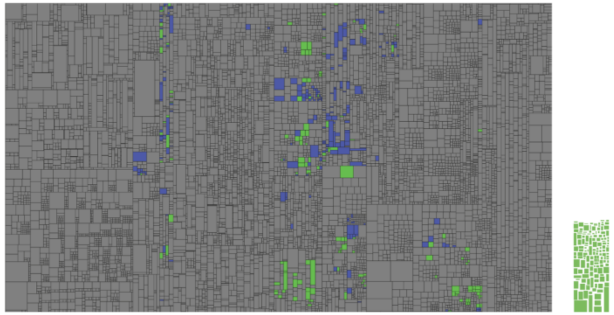


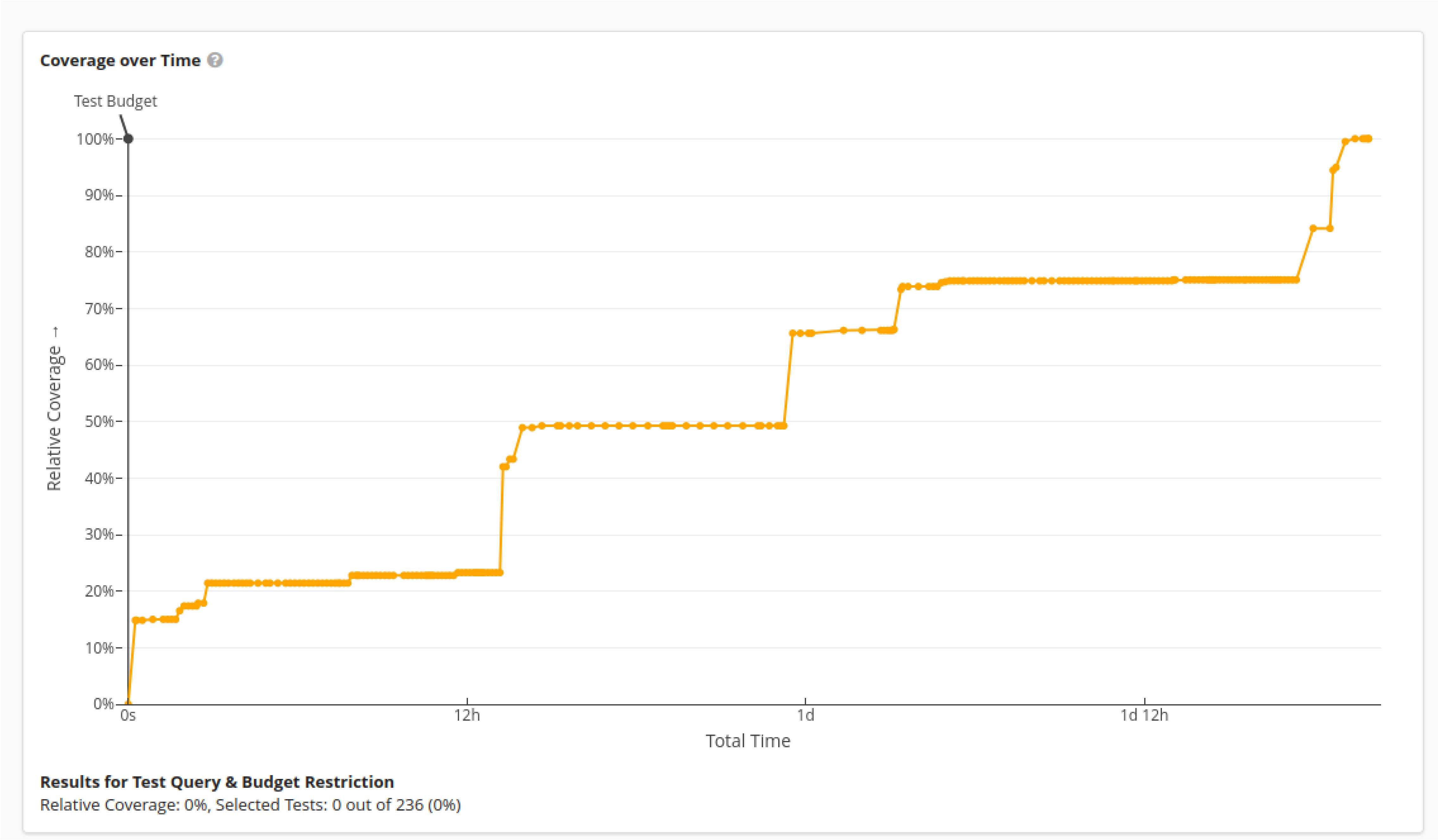






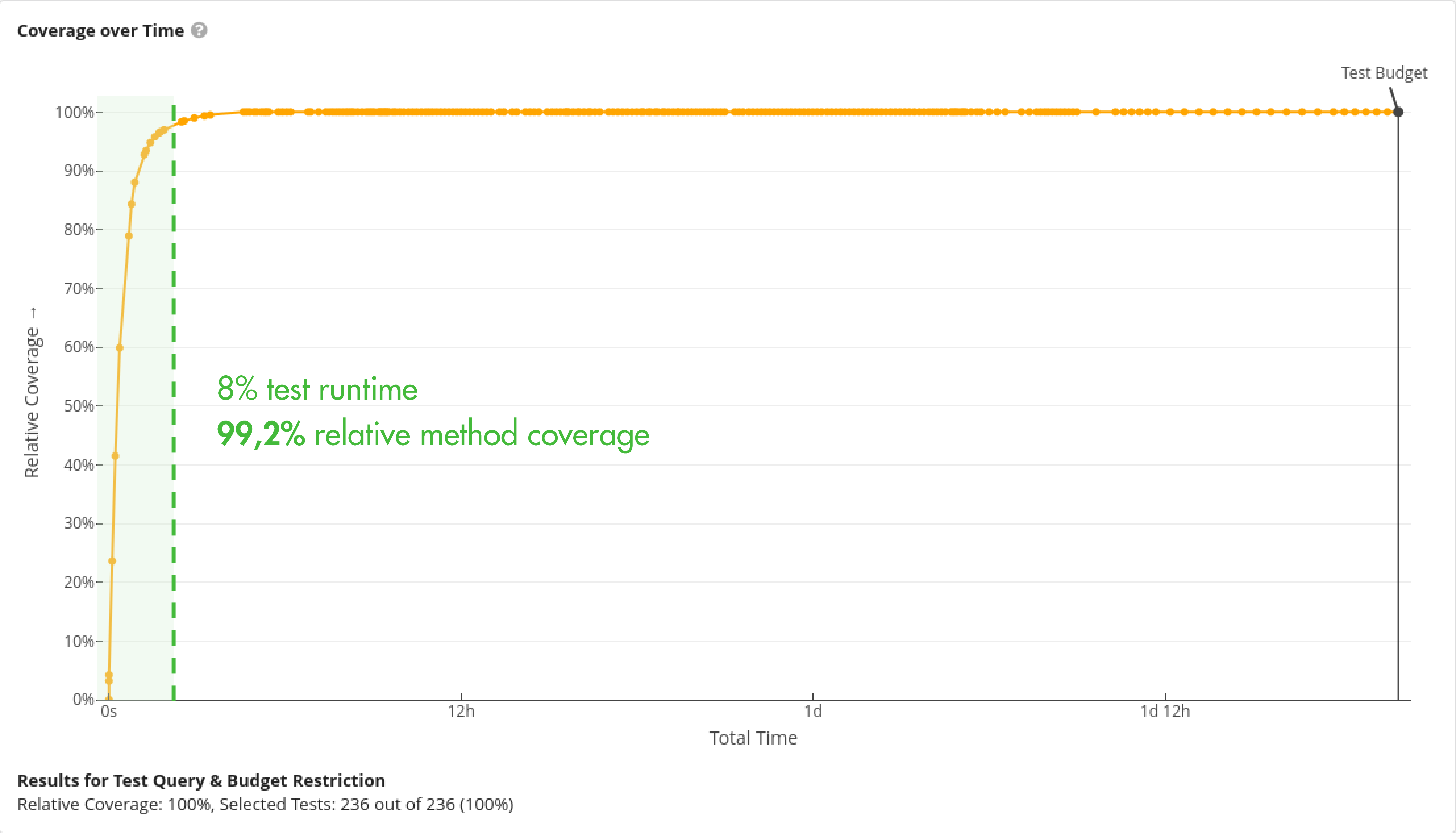






Results for Test Query & Budget Restriction

Relative Coverage: 0%, Selected Tests: 0 out of 236 (0%)

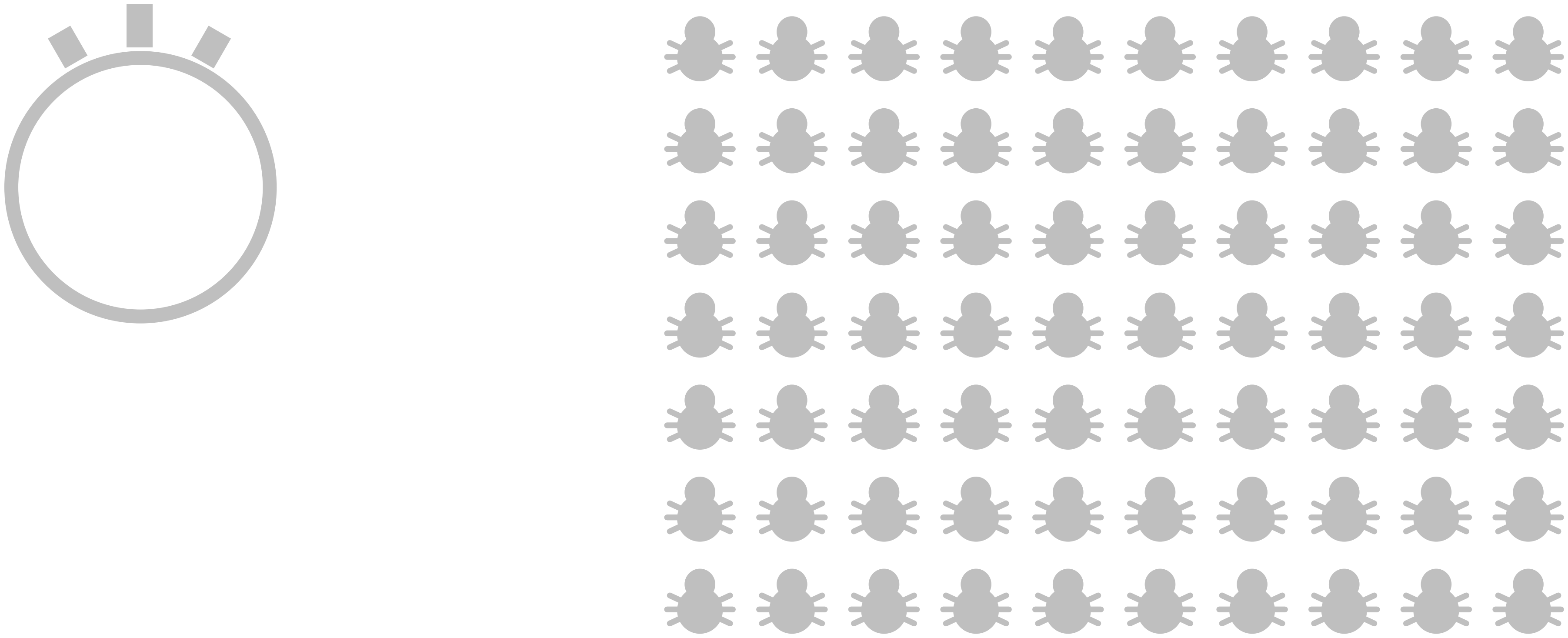


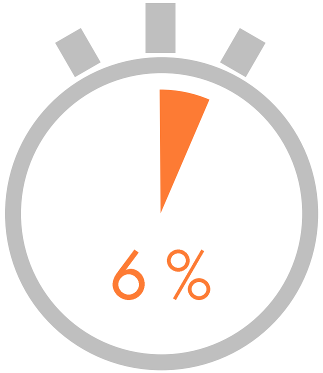
8% test runtime

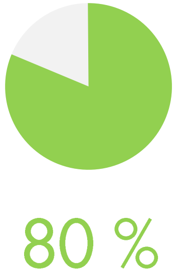
99,2% relative method coverage

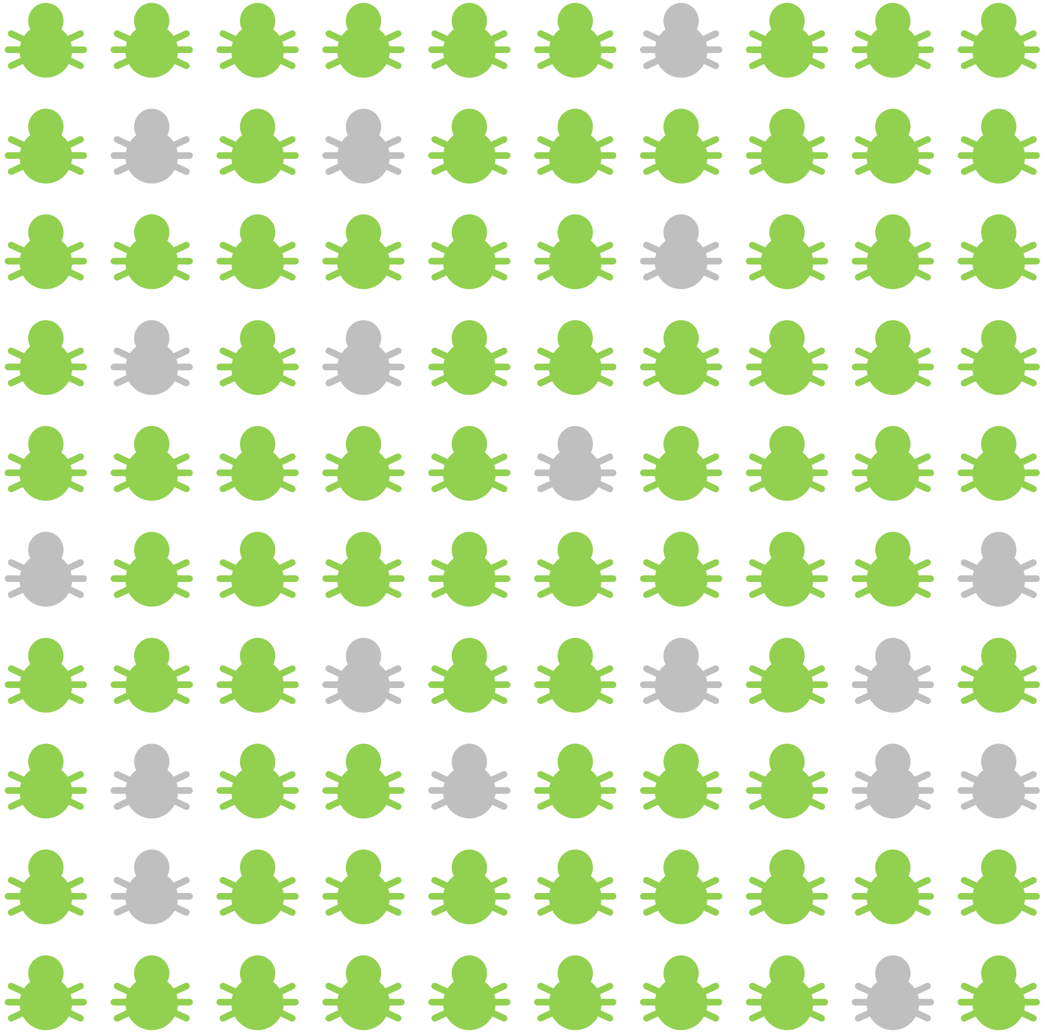
Results for Test Query & Budget Restriction

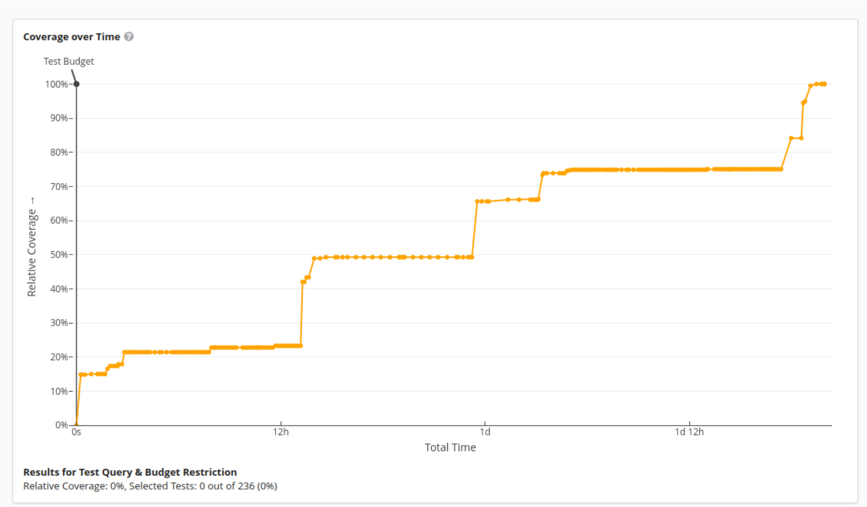
Relative Coverage: 100%, Selected Tests: 236 out of 236 (100%)



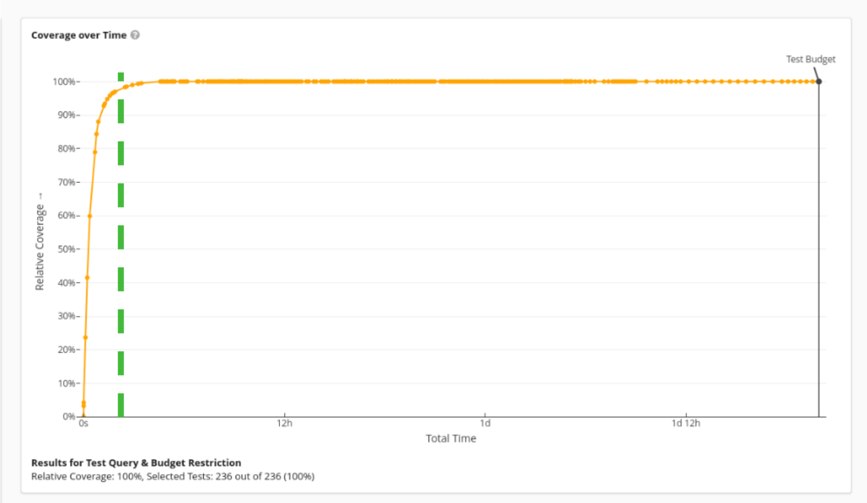

6 %


80 %



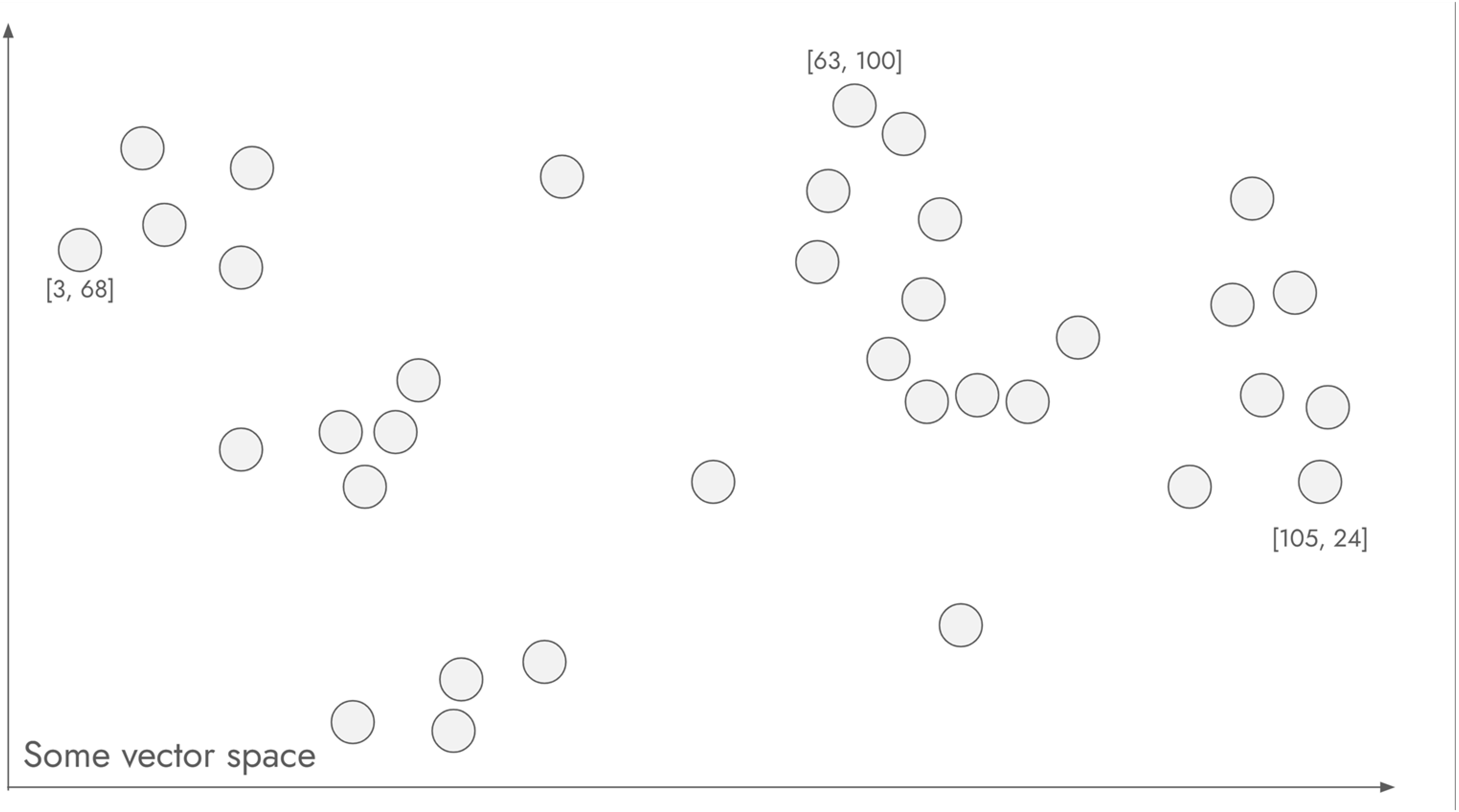


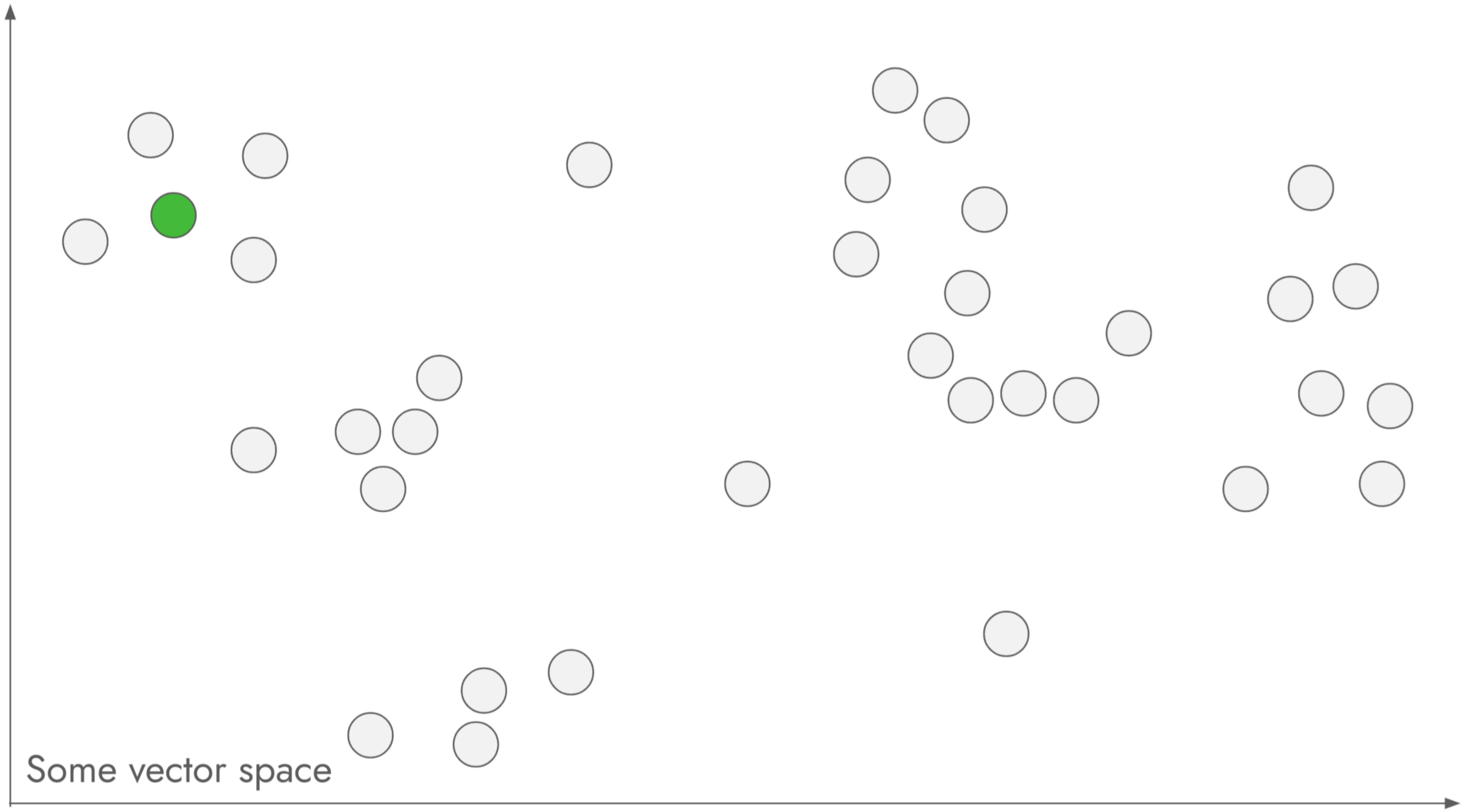
Results for Test Query & Budget Restriction
Relative Coverage: 0%, Selected Tests: 0 out of 236 (0%)

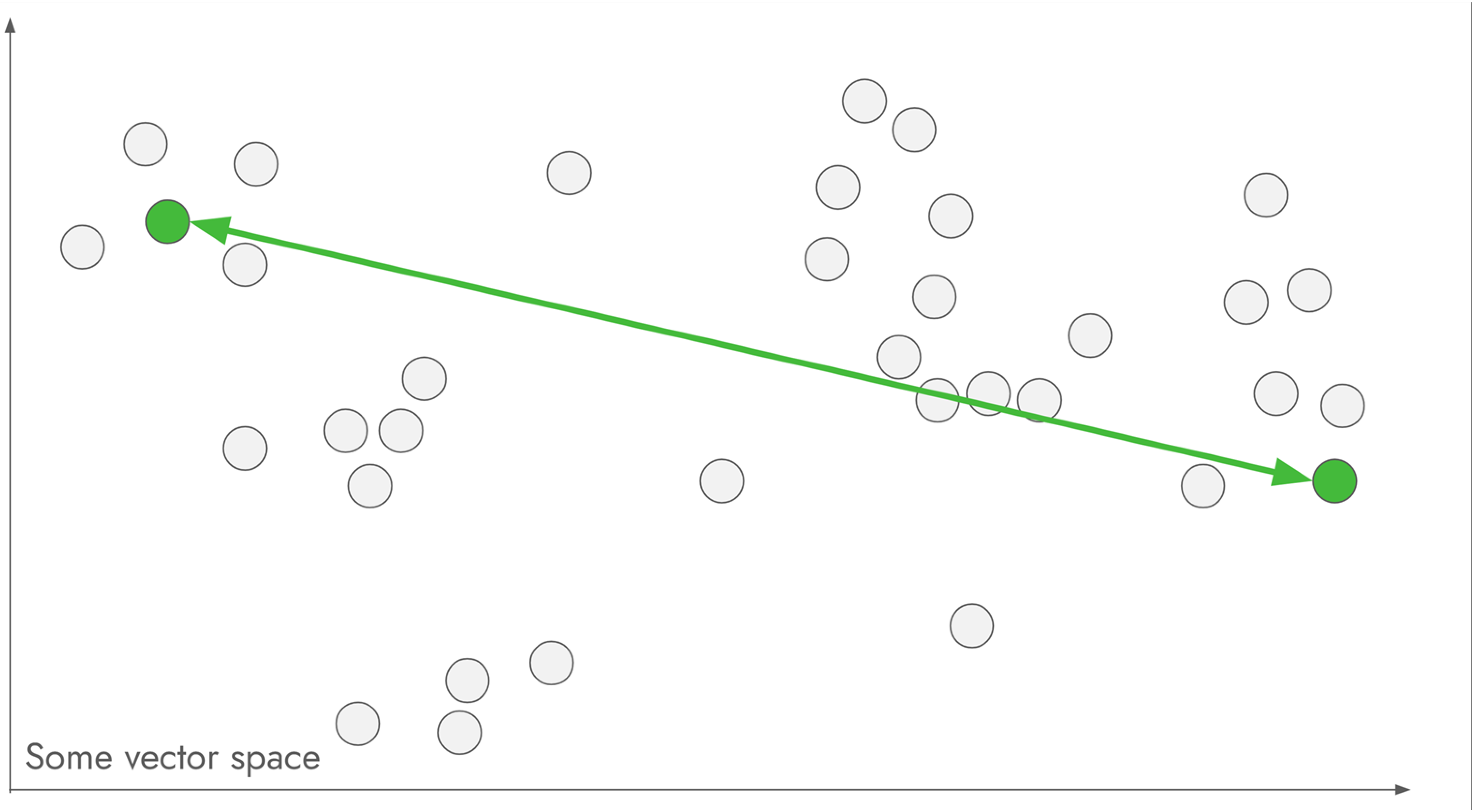


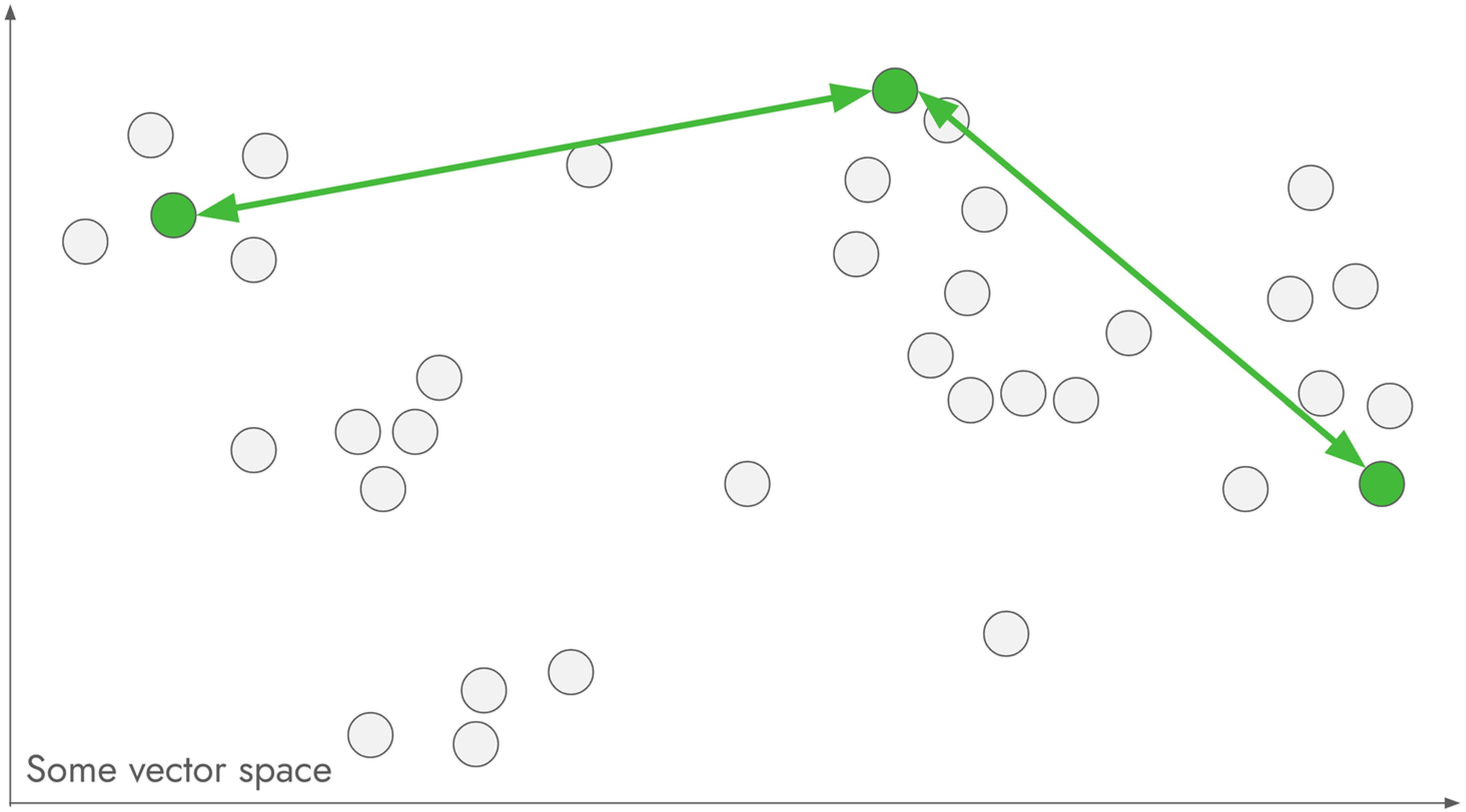
Results for Test Query & Budget Restriction
Relative Coverage: 100%, Selected Tests: 236 out of 236 (100%)

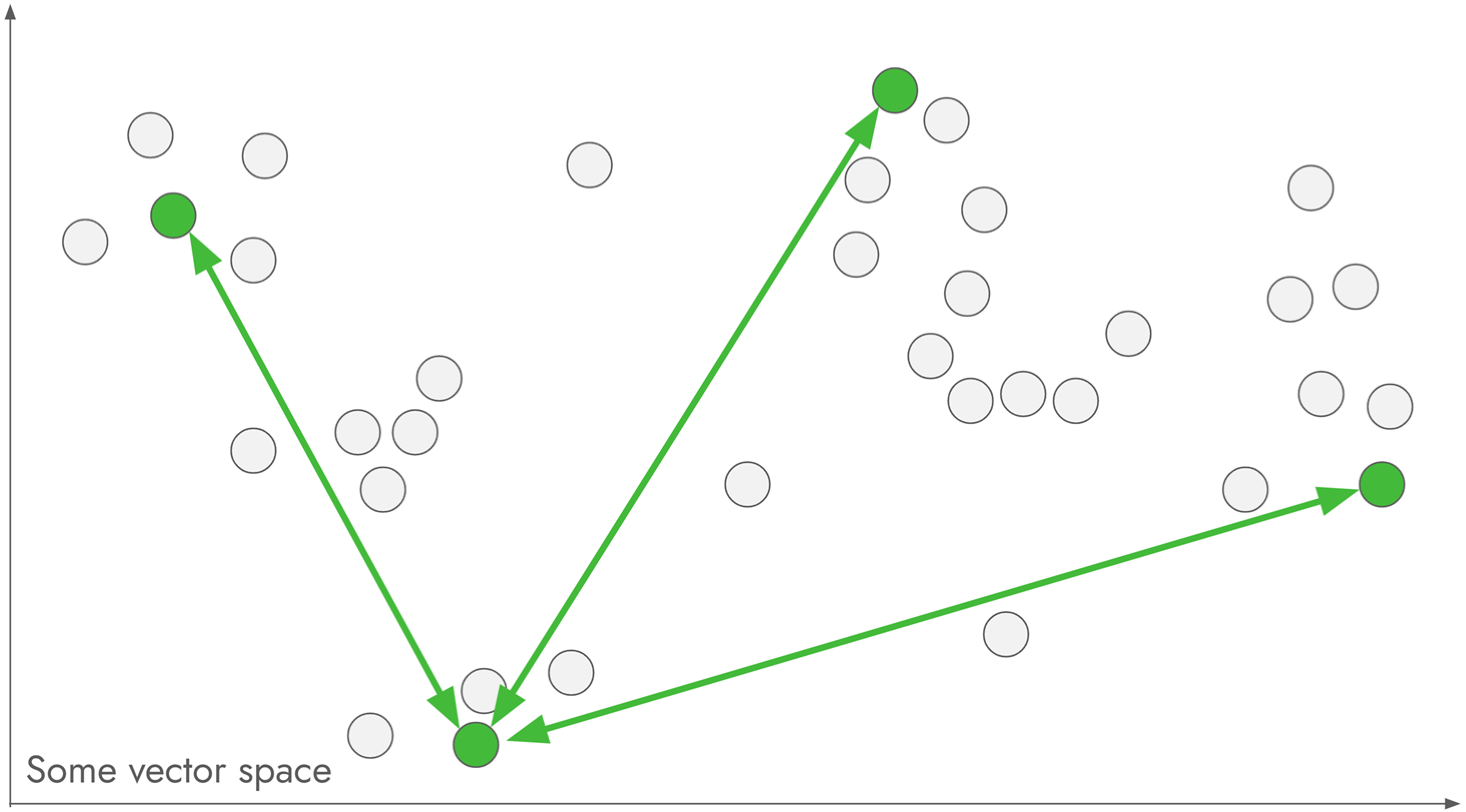
Sortieren von Tests nach “Unähnlichkeit”

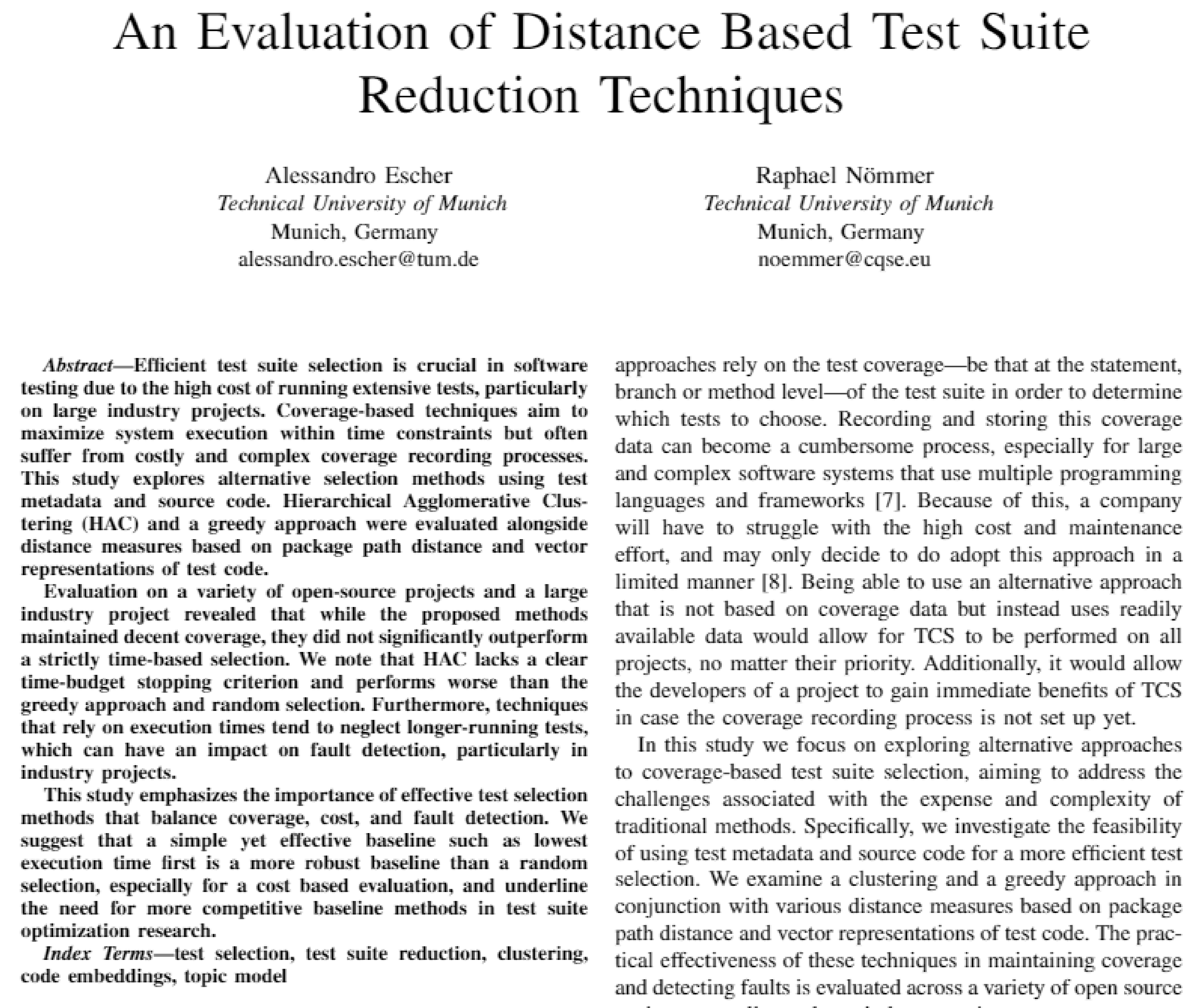

[63, 100]
[3, 68]
[105, 24]
Some vector space


Some vector space


Some vector space


Some vector space


Some vector space



An Evaluation of Distance Based Test Suite Reduction Techniques

Alessandro Escher
Technical University of Munich
Munich, Germany
alessandro.escher@tum.de

Raphael Nömmer
Technical University of Munich
Munich, Germany
noemmer@cqse.eu

***Abstract*—Efficient test suite selection is crucial in software testing due to the high cost of running extensive tests, particularly on large industry projects. Coverage-based techniques aim to maximize system execution within time constraints but often suffer from costly and complex coverage recording processes. This study explores alternative selection methods using test metadata and source code. Hierarchical Agglomerative Clustering (HAC) and a greedy approach were evaluated alongside distance measures based on package path distance and vector representations of test code.**

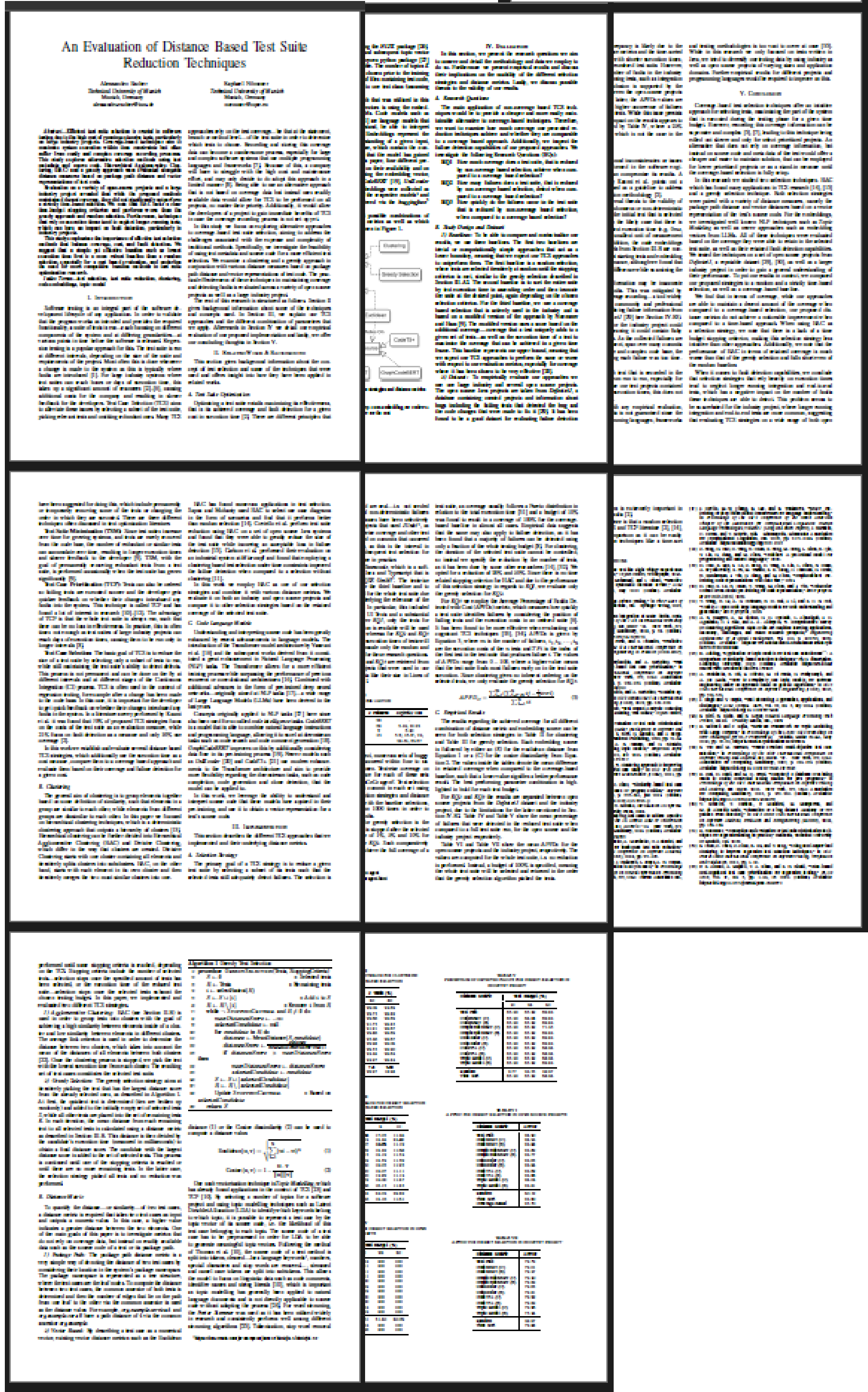
Evaluation on a variety of open-source projects and a large industry project revealed that while the proposed methods maintained decent coverage, they did not significantly outperform a strictly time-based selection. We note that HAC lacks a clear time-budget stopping criterion and performs worse than the greedy approach and random selection. Furthermore, techniques that rely on execution times tend to neglect longer-running tests, which can have an impact on fault detection, particularly in industry projects.

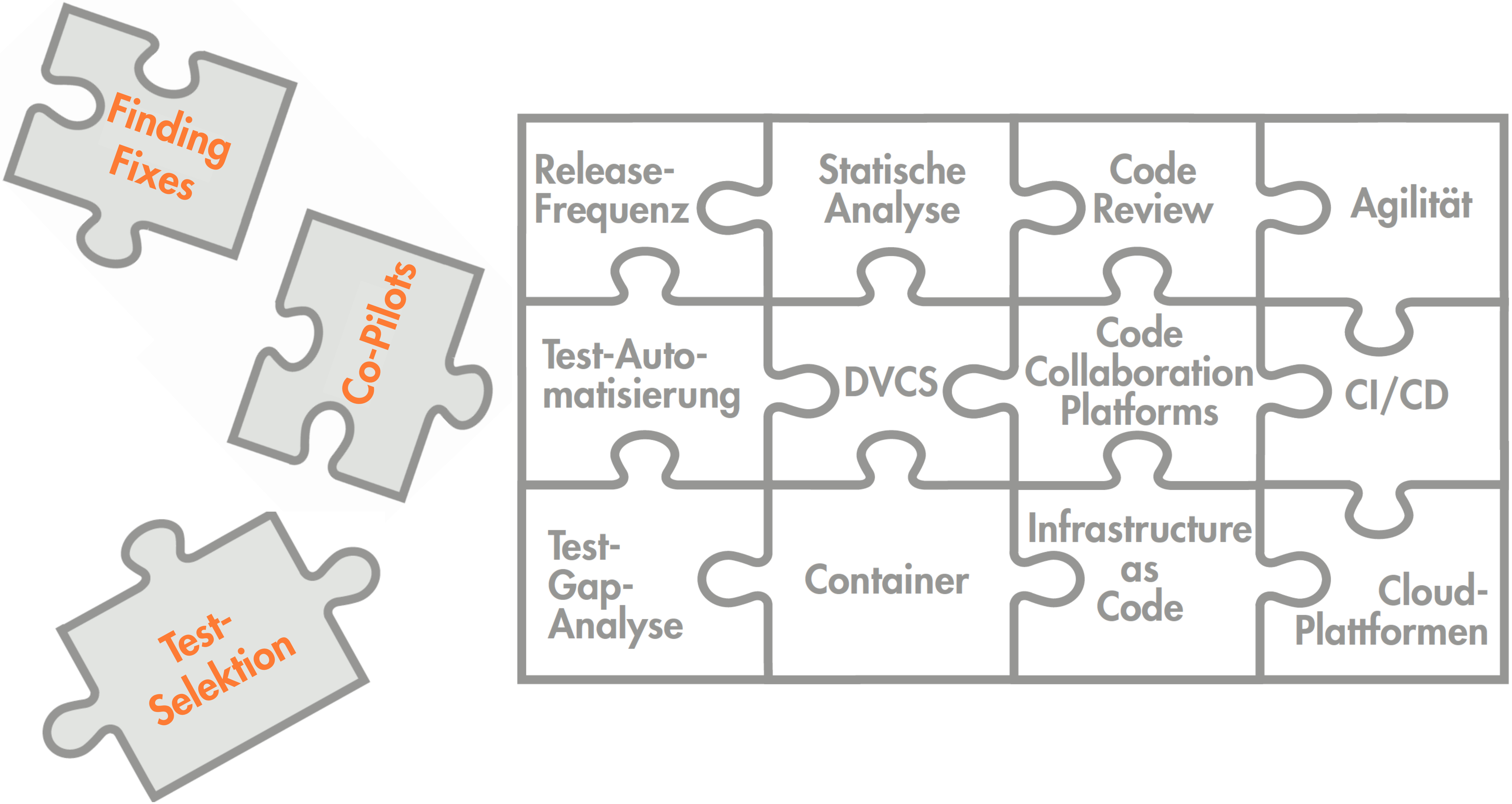
This study emphasizes the importance of effective test selection methods that balance coverage, cost, and fault detection. We suggest that a simple yet effective baseline such as lowest execution time first is a more robust baseline than a random selection, especially for a cost based evaluation, and underline the need for more competitive baseline methods in test suite optimization research.

***Index Terms*—test selection, test suite reduction, clustering, code embeddings, topic model**

approaches rely on the test coverage—be that at the statement, branch or method level—of the test suite in order to determine which tests to choose. Recording and storing this coverage data can become a cumbersome process, especially for large and complex software systems that use multiple programming languages and frameworks [7]. Because of this, a company will have to struggle with the high cost and maintenance effort, and may only decide to do adopt this approach in a limited manner [8]. Being able to use an alternative approach that is not based on coverage data but instead uses readily available data would allow for TCS to be performed on all projects, no matter their priority. Additionally, it would allow the developers of a project to gain immediate benefits of TCS in case the coverage recording process is not set up yet.

In this study we focus on exploring alternative approaches to coverage-based test suite selection, aiming to address the challenges associated with the expense and complexity of traditional methods. Specifically, we investigate the feasibility of using test metadata and source code for a more efficient test selection. We examine a clustering and a greedy approach in conjunction with various distance measures based on package path distance and vector representations of test code. The practical effectiveness of these techniques in maintaining coverage and detecting faults is evaluated across a variety of open source





Finding Fixes

Co-Pilots

Test-Selektion

Release-Frequenz	Statische Analyse	Code Review	Agilität
Test-Automatisierung	DVCS	Code Collaboration Platforms	CI/CD
Test-Gap-Analyse	Container	Infrastructure as Code	Cloud-Plattformen

Fazit

Wir müssen uns darauf einstellen, dass Gen-AI in allen Phasen des Entwicklungsprozesses Einzug halten wird.

Ich sehe **nicht**, dass unser QS-Weltbild dadurch revolutioniert wird.

Es gibt viele Ansätze, die einzelne Tätigkeiten effizienter machen oder erste ermöglichen, wie bei der Optimierung von Smoke-Tests oder der Selektion von Testfällen. Entscheidend sollte hier aber sein, wie nützlich sie sind, nicht wieviel AI drinsteckt ;-)

Kontakt – Ich freue mich auf Diskussionen ☺

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

Link zu Folien

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

CQSE