adesso | insurance solutions

| Stefan Krecher

METRICS OVER MAYHEM | MAKE COMPLEXITY VISIBLE

A SIMPLE APPROACH TO ANALYSE LEGACY VISUALWORKS SYSTEMS

ESUG 2025

AGENDA

adesso | insurance solutions

1. About
2. The Problem
3. The Solution
4. The Assumption
5. Core Queries
6. Core Metrics
7. Analytics & Visualization
8. Conclusion & Outlook

ABOUT

adesso | insurance solutions

Me

> I'm a Principal Software Engineer who fell in love with Smalltalk back in the 90s.
> After years of Java development and architecture—and a detour into cloud engineering—I finally returned to Smalltalk two years ago at adesso insurance solutions.

The System

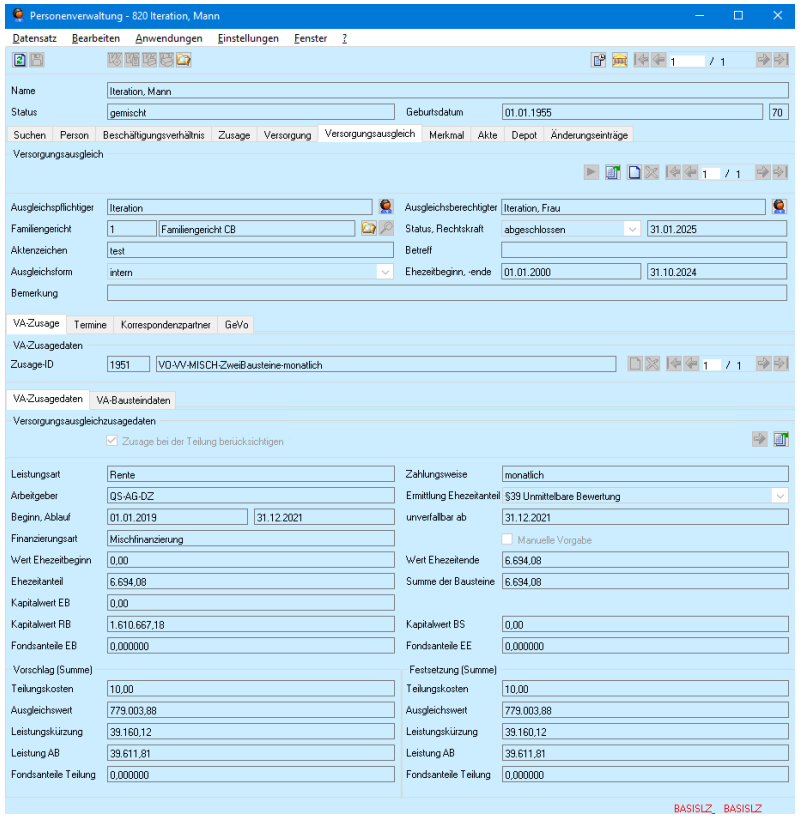
> The application is called *CollPhir* and has been continuously developed and used in production for 20+ years.
> It is a highly complex system for managing company pension schemes (*betriebliche Altersvorsorge*), consisting of a Windows-based desktop client, a headless server version and a Java-based web frontend. We currently use VisualWorks 9.2.

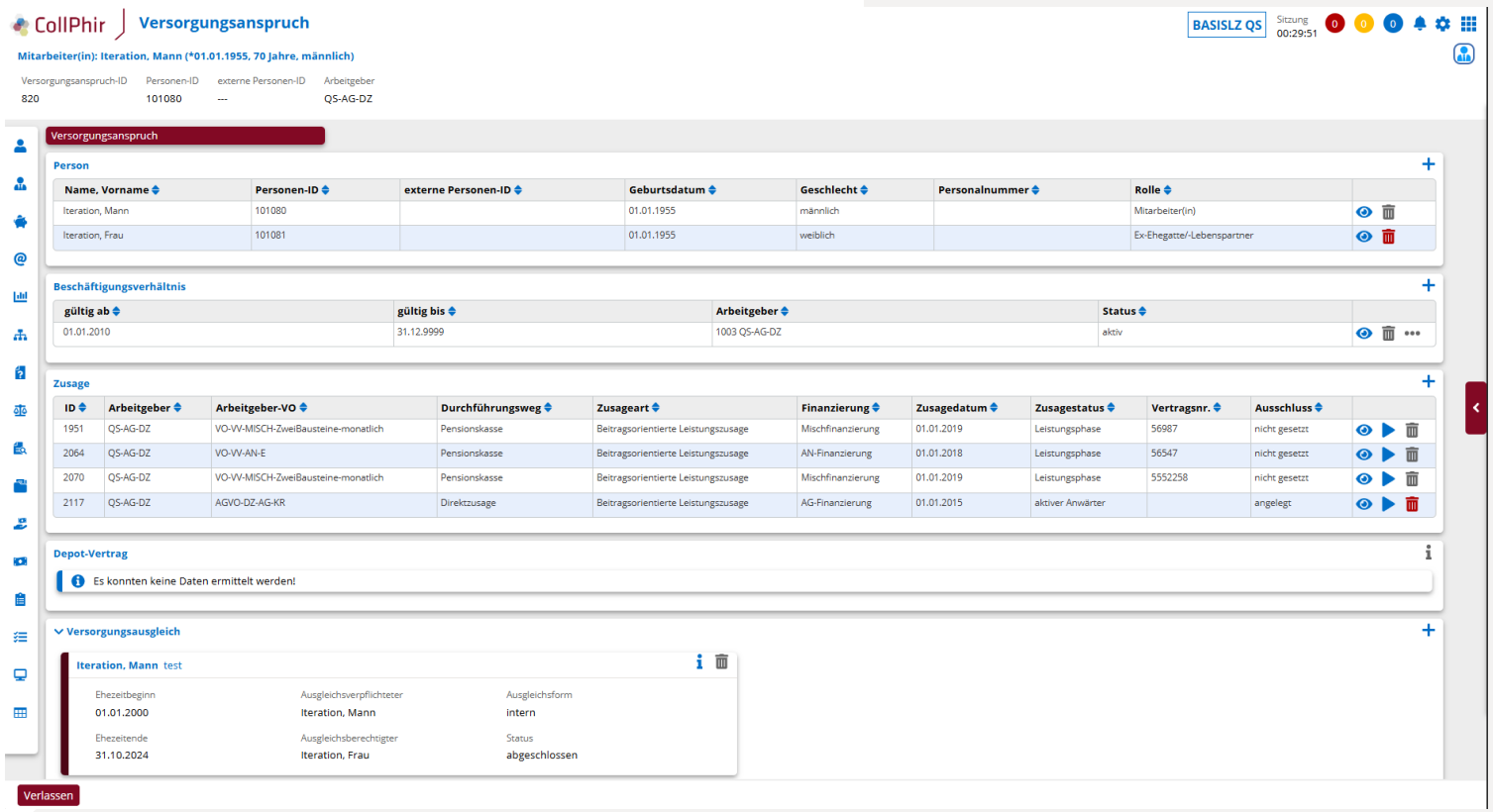
The Team

> The core dev-team consist of 5 Smalltalk developers, with more Smalltalkers in Professional Services, and multiple project-specific subteams.

ABOUT
THE PRODUCT

adesso | insurance solutions





THE PROBLEM

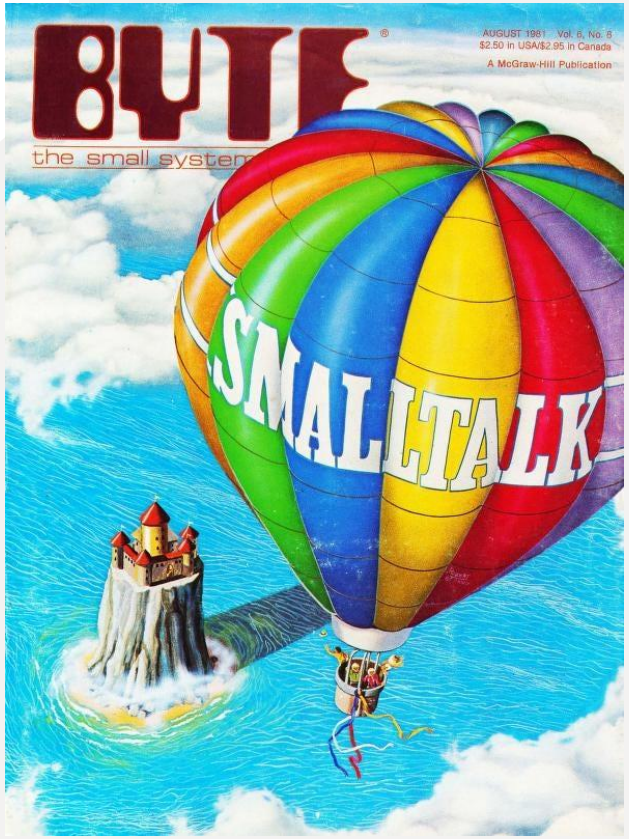
adesso insurance solutions

- A business-critical VisualWorks system, born long time ago.
- Scale: 6,800 classes, 1 million lines of code.
- Undocumented, highly complex, and difficult to maintain.
- Every change is a risk. High maintenance costs, slow feature development, and the constant threat of catastrophic side effects.

THE SOLUTION

adesso | insurance solutions

- The Challenge: "Tool Envy“
  - Felt jealousy towards the Java world with its sophisticated static analysis tools.
  - But Tools like Glamorous Toolkit or Moose were not an option for our VisualWorks project.
- The Smalltalk Way
  - We don't have to wait for tools; we can build our own.
  - The entire system is a live, queryable object
- Live, Interactive Analysis
- Visual Analysis & Drill-Down outside of VisualWorks



THE ASSUMPTION

adesso | insurance solutions

- Less is More: A small set of lightweight metrics (Instability, Surface Size, Cohesion, LOC, number of methods) already highlights hotspots and architectural risks.
- Even simple coupling metrics reveal critical architectural hotspots before deep code reviews happen.
- Packages with high Instability × Surface or classes with large code size tend to require more maintenance
- One linear scan of source code (≈ O (n)) is enough—no heavyweight tooling or runtime tracing required.
- Visual Power: scatter plots, bubble charts and heat maps turn those few numbers into clear stories for developers and stakeholders.

CORE QUERIES

adesso insurance solutions

Iterate over all classes

```
(Store.Glorp.StoreBundle pundleWithName: 'Collphir-Application' version: 'icp-2.2.0.655')
        allClasses values do: [:cls | ... ].
```

Iterate over methods

```
aClass instanceMethods do: [:selector | ... ]. aClass classMethods do: [:selector | ... ].
(aClass is of type StoreClassExtension when doing queries against the STORE)
```

Get the compiled method

```
compiledMethod := VAVertragsbaustein compiledMethodAt: #ueberpruefePlausibilitaet:in:
```

Get references to other classes via compiled methods's literals

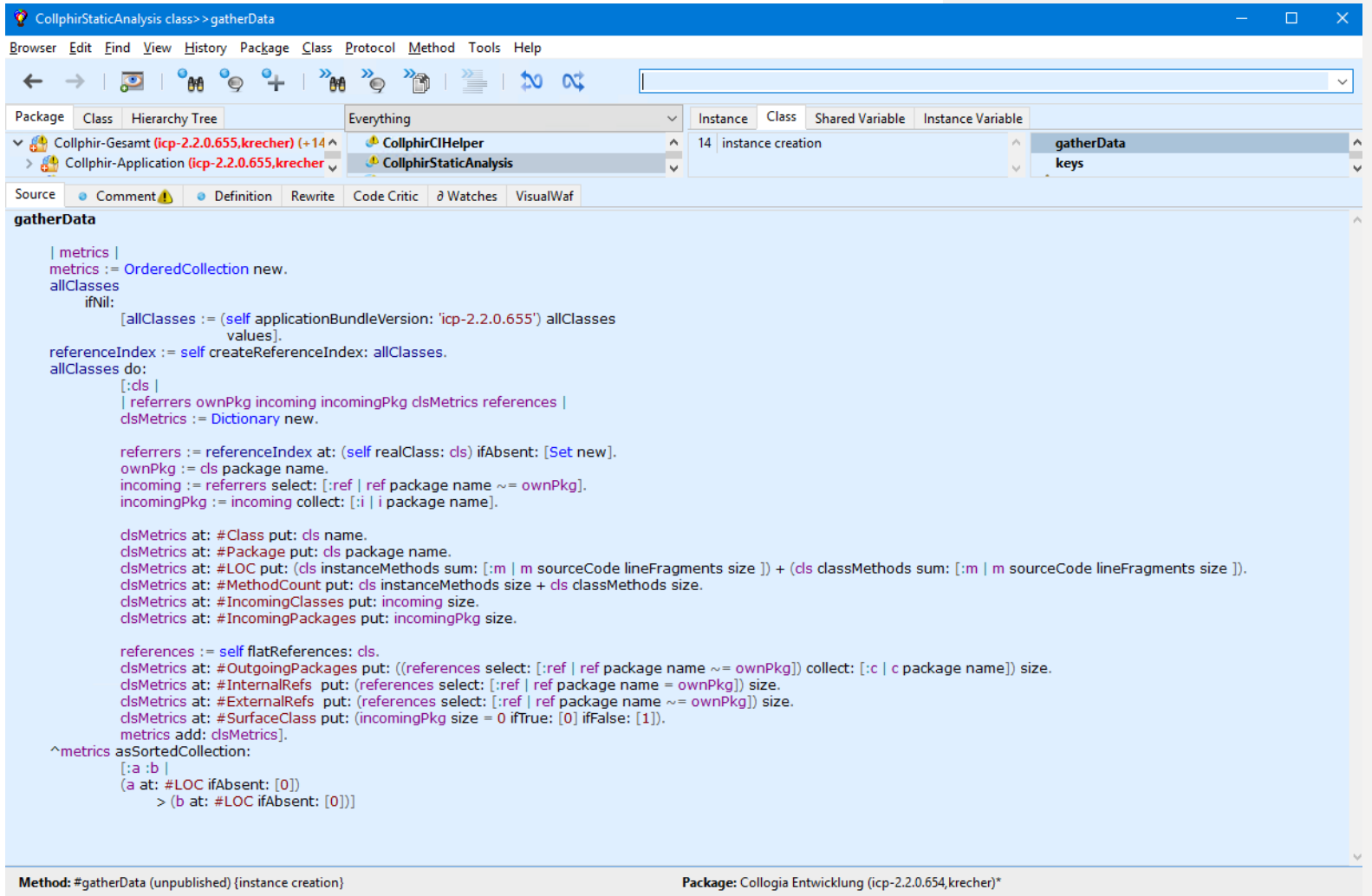
```
compiledMethod literals
"a literal can be of type class -> this is reference to another class. But this can also be a BlockClosure/
CompiledBlock so we need to recursively check for references"
```

LOC

```
aClass instanceMethods sum: [:m | m sourceCode lineFragments size ]) + (cls classMethods sum: [:m | m
sourceCode lineFragments size ])
```

CORE METRICS

- LOC
- Methods per Class
- Incoming Classes
- Incoming Packages
- Outgoing Packages
- Internal References
- External References
- Surface Class (0/1)



ANALYTICS & VISUALIZATION OVERVIEW

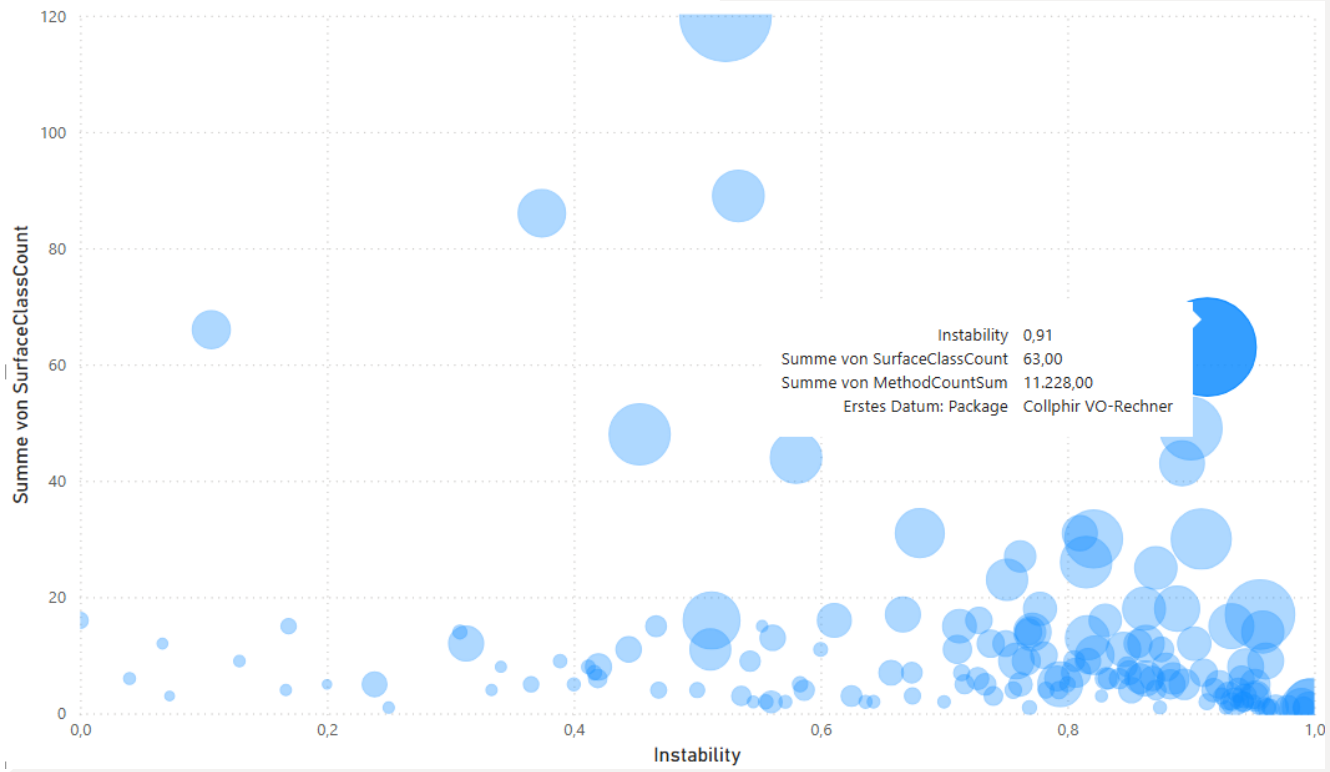
adesso | insurance solutions

> Static Metric Extraction in VisualWorks
  - We implemented custom scripts to analyze class references, method counts, and code metrics.
  - The result: a flat CSV file with one row per class and all relevant metrics (“Core Metrics”).
> Data Import in Power BI
  - The CSV file was loaded directly into Power BI as a data source.
  - Column types were set and class-level data was optionally grouped by package.
> Derived Metrics in Power BI
  - Additional metrics like Instability were computed using calculated columns.
  - Aggregations like Surface Size per Package or Total WMC were created via grouping in the Power Query editor.
> Visualization and Exploration
  - Using scatter plots, bubble charts, and heat maps, we visualized metric combinations.
  - This made it easy to spot architectural hotspots, God classes, unstable packages, and low-cohesion areas.

ANALYTICS & VISUALIZATION INSTABILITY

adesso | insurance solutions

- The concept of *Instability* was introduced by Robert C. Martin (Uncle Bob) in *"Design Principles and Design Patterns" (2003)* as part of his package design principles.
- Definition: **Instability = Ce / (Ce + Ca)**
  - Ce (Efferent Coupling): Number of packages a package depends on
  - Ca (Afferent Coupling): Number of packages that depend on it
  - Value Range: 0 (very stable) → 1 (very unstable)
- Helps to assess how likely a package is to change
- A stable package (low I) should not depend on unstable ones
- High instability is not always bad — it can be okay if the package is meant to change



- Bubble-Size: Method Count
- Top-right corner: Large, unstable, and widely used → critical maintenance risk
- Bottom-left: Stable and isolated → typically internal, low-risk packages
- Helps detect "leaky" or overexposed architecture zones

ANALYTICS & VISUALIZATION
PACKAGE-LEVEL COHESION

adesso | insurance solutions

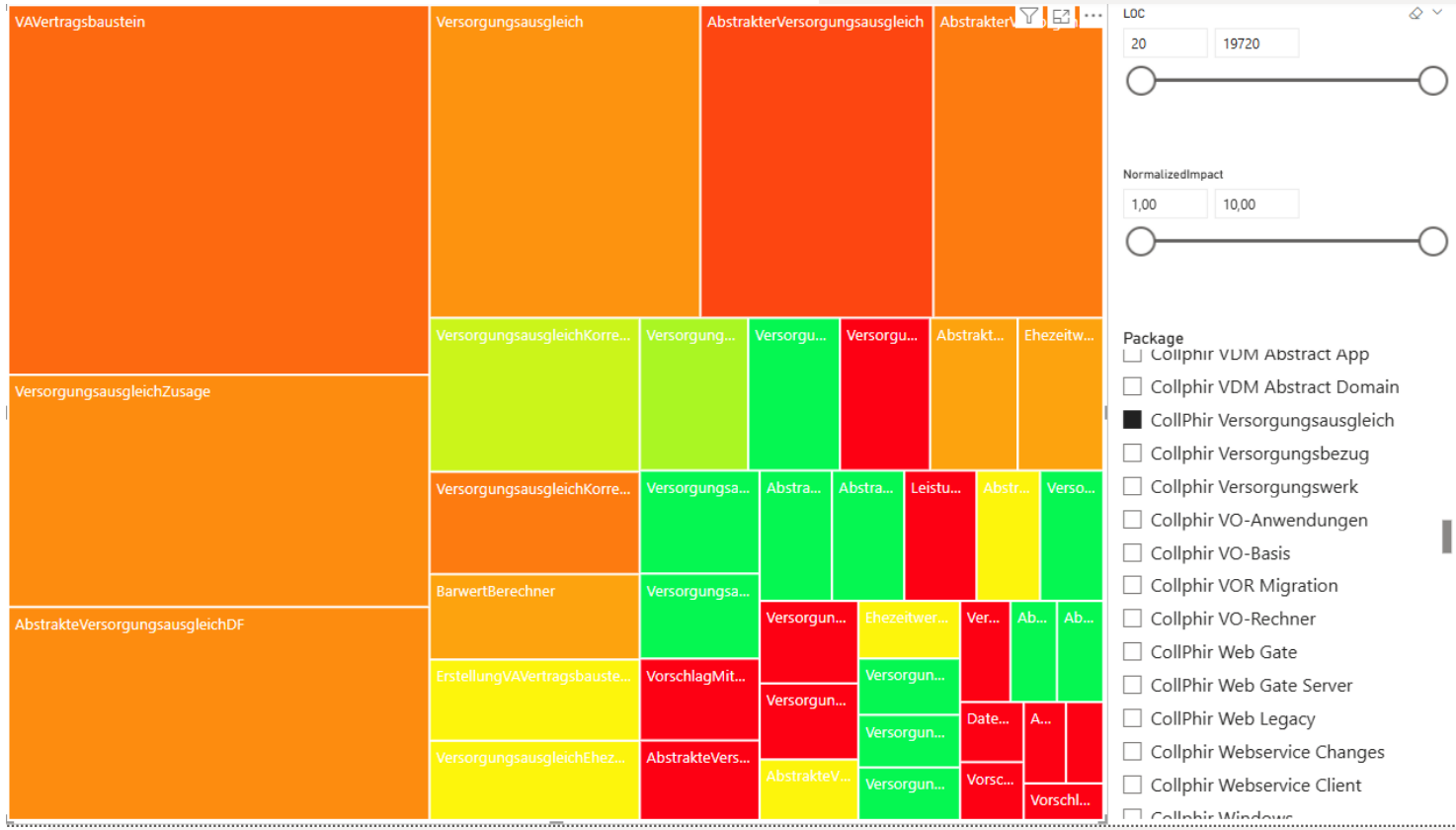
- Measures how closely classes within a package relate to each other
- High cohesion = well-focused, modular code
- High cohesion improves maintainability, testability and understandability
- Low cohesion often signals God packages or poor modularization
- **Cohesion = InternalRefs / (InternalRefs + ExternalRefs)**
- **Impact = LOC x (1 – Cohesion)**
  - Normalized to scale 1-10
  - Highlights packages that are large and poorly structured
  - Helps prioritize refactoring targets

Package	CohesionScore	MethodCount	LOC	Impact
Collogia Framework	0,23	9705	78720	10,00
Collphir VO-Rechner	0,37	11228	81315	8,70
Collphir DemoLZ	0,25	5288	55286	7,20
Collphir Bav-Standardmodell	0,21	3827	33436	4,90
CollPhir Web Legacy	0,21	4192	29691	4,50
Collphir EGecko	0,30	3524	32915	4,40
Collphir VO-Anwendungen	0,33	4231	30727	4,10
CollPhirBasisDatenmodell	0,21	3989	26213	4,10
COLImageLoading	0,07	1077	16901	3,40
Collphir Versorgungswerk	0,36	2680	22102	3,10
CollPhirFondsDatenmodell	0,28	2701	19709	3,10
CollphirBaustein	0,02	684	12957	2,90
CollPhirZeitwertkontoDatenmodell	0,22	2405	16491	2,90
CollPhirZeitwertkontoApplikationen	0,26	1139	15681	2,70
CollphirZeitwertkontoVerarbeiter	0,16	1944	13287	2,70
Collphir Handelsprozess	0,16	1713	12209	2,50
Collphir InstallationsWerkzeuge	0,22	1319	12310	2,40
CollPhirKontodefinition	0,37	1598	14901	2,40
BaseVisualWorksChanges	0,02	1323	9061	2,30
Collogia Framework Domain	0,30	2238	12543	2,30
Collphir Fonds Unterstuetzung	0,18	817	10521	2,30
Collphir Konsistenzpruefung	0,23	1426	10990	2,30
CollPhir Web Gate Server	0,31	1922	13051	2,30
CollPhirRiesterDatenmodell	0,40	2716	14233	2,30
CollPhirStrukturierteAkte	0,22	1364	11306	2,30
Wave-Server	0,35	1962	13389	2,30
CollPhirFondsApplikationen	0,24	960	10173	2,20
CollPhirRiesterApplikationen	0,48	889	14958	2,20
JNIPort-Java-Base	0,41	1862	13391	2,20
Colllphir-PSLife-WSDL-Generiert-Vertragsservice	0,42	3374	12718	2,10

ANALYTICS & VISUALIZATION
CLASS-LEVEL COHESION

adesso | insurance solutions

- Each rectangle represents a single class in the system
- Size = Lines of Code (LOC): larger rectangles indicate larger classes
  - Color = Cohesion score:
- Interactive filters (Slicers):
  - LOC to focus on significant classes
  - Package to isolate and inspect specific modules
- Use cases:
  - Spot oversized or poorly structured classes
  - Detect refactoring candidates
  - Compare design consistency within a package
  - Combine visual and metric-based exploration



CONCLUSION & OUTLOOK

adesso | insurance solutions

Conclusion

> Even simple static metrics provide powerful insights into system structure and design quality
> Visualizations like heat maps and tree maps help uncover hidden complexity and refactoring hotspots
> The analysis revealed clear risk areas, low cohesion classes, and large, critical components
> Whole new level of transparency

Outlook

> Automate metric extraction and CSV export in the CI/CD pipeline
> Build a living system dashboard with Power BI and scheduled data updates
> Integrate code churn, test coverage, or bug density, dependency graphs, runtime data, or maybe team-oriented metrics such as contribution diversity
> Use insights for architecture discussions, roadmap planning, onboarding, or code reviews

adesso | insurance solutions

THANK YOU!

CONTACT: STEFAN.KRECHER@ADESSO-INSURANCE-SOLUTIONS.DE

adesso insurance solutions GmbH
Adessoplatz 1
44269 Dortmund
T +49 231 7000-8000
F +49 231 7000-1000
www.adesso-insure.de