Treating Run-time Execution History as a First-Class Citizen: Co-Versioning Run-time Behavior alongside Code (BeCoV)

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Marcus Kessel marcus.kessel@uni-mannheim.de (University of Mannheim, Germany, SE Group)

FSE'26 IVR @ Montreal, Canada – Ideas, Visions and Reflections , July 9th, 2026

Data is Becoming a Strategic Asset ("*Data is Gold*")

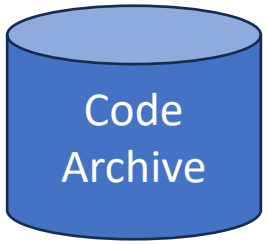
UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Code Changes

- preserved in Git (i.e., persisted)

Code changes

git GitHub



Run-time Behavior (Observations)

- "disappears" (i.e., ephemeral)

xUnit

```
import org.junit.Test;
import static org.junit.Assert.*;
  import org.junit.Test;
  import static org.junit.Assert.*;
    import org.junit.Test;
    import static org.junit.Assert.*;

    public class ArrayStackTest {

        ...

        @Test
        public void test_ti()  throws Throwable  {
            ArrayStack arrayStack = new ArrayStack();
            arrayStack.push("hello world");
            int int0 = arrayStack.size();
            assertEquals(1, int0);

    }
```

Run-time Evidence

Data is Becoming a Strategic Asset ("*Data is Gold*")

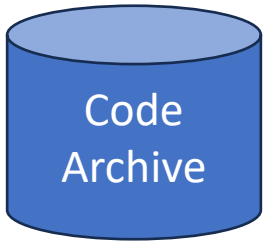
UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Code Changes

- preserved in Git (i.e., persisted)

Code changes

git GitHub



Run-time Behavior (Observations)

- "disappears" (i.e., ephemeral)

xUnit

Lots of computational resources already spent to obtain ...

Run-time Evidence

!

Practice (CI/Testing)

- often reduced to **PASS/FAIL signals** (or unstructured, aggregated, etc.)
- Unsystematic, scattered landscape

One of the richest data sources - **run-time behavior -** is still disposable

Version History: Code Has Git, Behavior Has ?

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Git versions Code, not Behavior

- Git: how code changed, but not whether behavior changed (cf. Rice's Theorem)

Did this method's behavior changed between commit X and commit Y?

Code	Run-time Behavior
persistent	~~persistent~~
queryable	~~queryable~~
versioned	~~versioned~~
diffable	~~diffable~~

What if Behavior had Git-like Powers?

- Before: manually re-run tests, inspect logs, or reconstruct past executions
- After: query behavioral archive

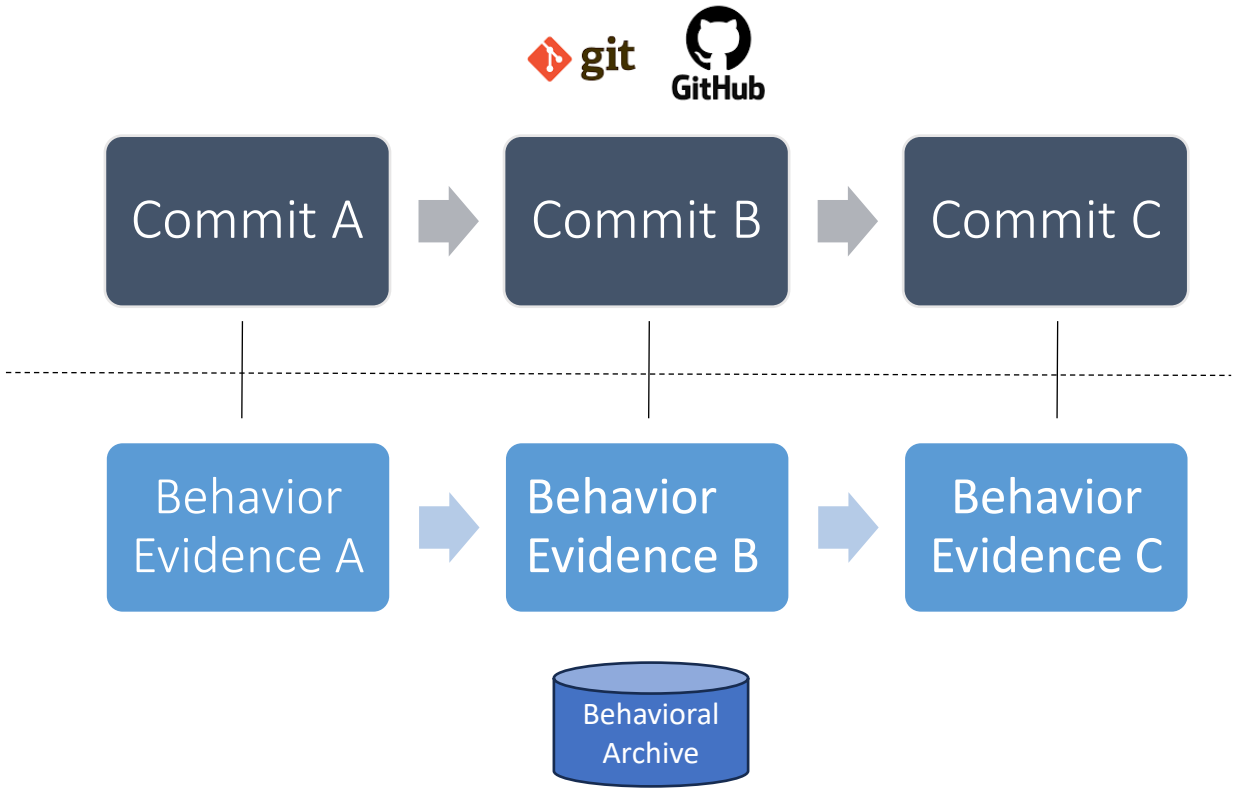
Code	Run-time Behavior
commit history	behavior history
git diff	behavior diff
code review	behavior review
repository mining	behavior analytics

+ Co-Versioning Changes

Idea: Behavioral Co-Versioning (BeCoV)

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Git History + Behavioral Archive



Augment CI/Testing

assumption lead to weak assertions and partial oracles ...

Run-time Execution - **proactive (decide upfront)**

execute → pass/fail → "discard"

augment (i.e., obtain as part of execution)

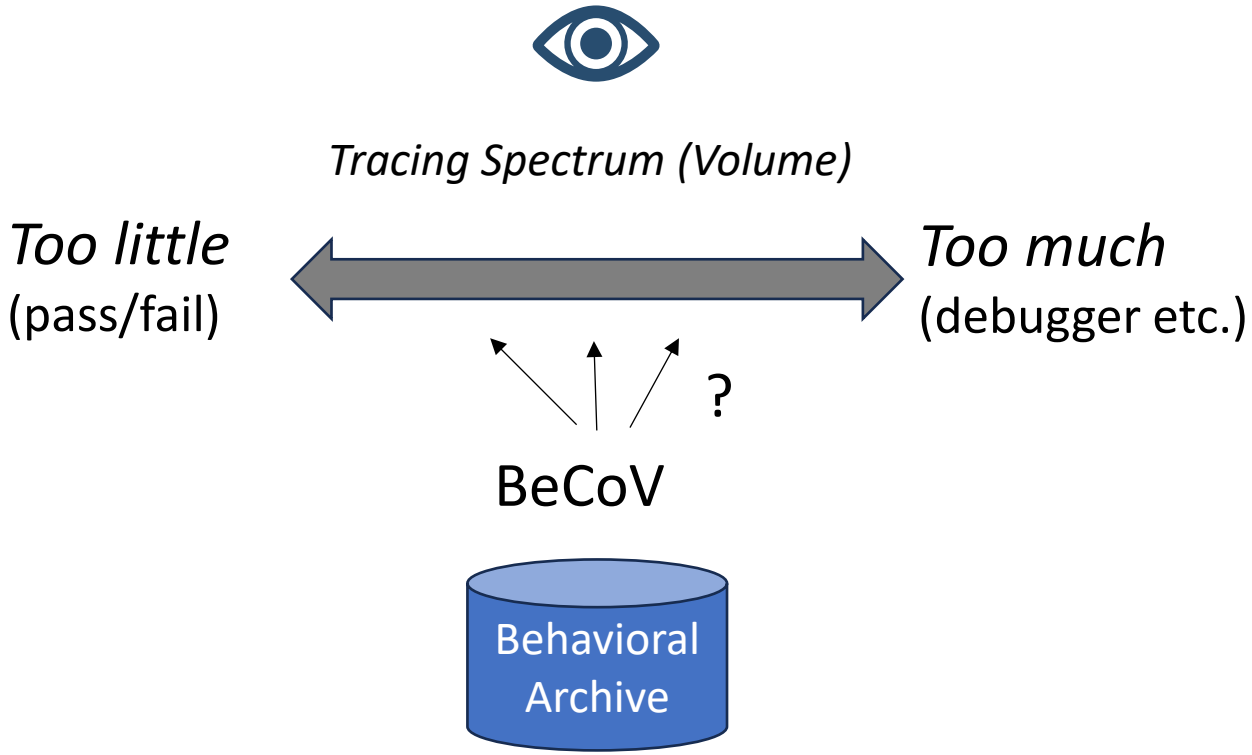
BeCoV – **retrospective (query later)**

capture → archive → query

Positioning: BeCoV as a Data Layer (Source)

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

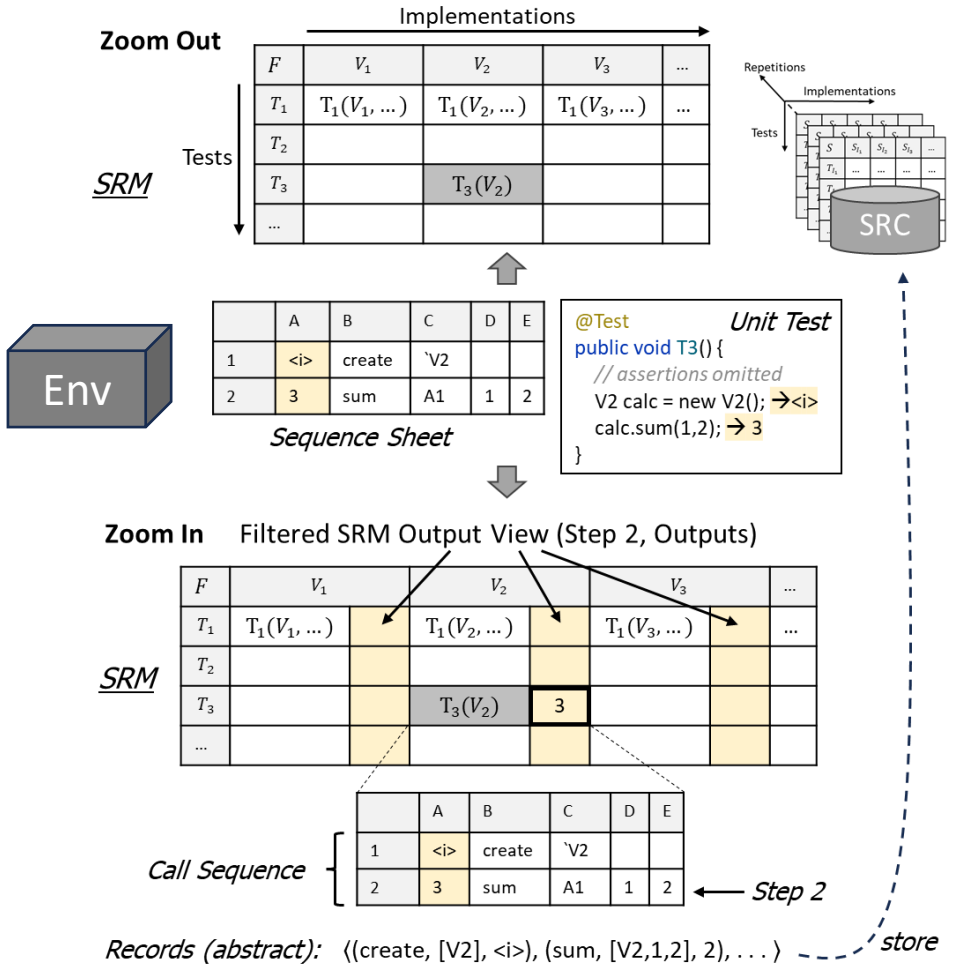
- Run-time evidence are facts
  - about what happened under specific inputs, tests, environments, and repetitions
  - NOT ground truth (!), but discrepancies (regressions)
- BeCoV complements existing tools/analyzers
  - preserving systematic run-time evidence
- "Goldlocks problem"
  - find a "sweet spot" to enable retrospective analysis (range of tasks)
  - … or adapt



Run-time Evidence – Tabular Representations

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

- Invocation-centric
  - stimulus/response encoding
  - invocation-centric: **⟨(op1, in1, out1), (op2, in2, out2), . . .⟩**
  - + shape / structure (i.e., types)
  - + context (git, environment, build, …)
- SRMs (Stimulus-Response-Matrix)
  - tests × implementations comparison matrix
- SRCs (Stimulus-Response-Cube)
  - multi-dimensional SRMs across contexts (time, repetitions, envs etc.)
- "Living" (Continual) property
  - add rows/columns dynamically



Kessel, Marcus, and Colin Atkinson. "Morescient GAI for software engineering." ACM Transactions on Software Engineering and Methodology 34.5 (2025): 1-17.

Proof of Concept: Local Behavioral Archives

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Observation Lakehouse

- Lakehouse-style storage + DuckDB analytics
  - columnar storage (compression, RLE etc.)
  - analytics via SQL (embedded OLAP)
  - partitioning avoids full table scans
- Single invocations append-only table
- Initial Tracing Strategy
  - focal methods, controlled test depth (= test depth), simple canonicalization

Proof of Concept (Local Laptop)

- BeCoV prototype (pytest)
  - replay commits of *python dateutil repository* (replayed past 100 commits with 4.702 tests on "focal units") - **66.9MB**
- Observation Lakehouse (SANER'26)
  - Java implementations
  - runtime observations as analytical data (95.154 tests) - **51MB**

Kessel, Marcus. "Towards Observation Lakehouses: Living, Interactive Archives of Software Behavior." 2026 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). IEEE, 2026.

Behavior-aware Change Classification & Discrepancies

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

- Change Categories
  - Code Changed
    - $Test=, Code\Delta, Obs=$
  - Code and Observations changed
    - $Test=, Code\Delta, Obs\Delta$
  - Observations changed
    - $Test=, Code=, Obs\Delta$
  - Test and Code changed
    - $Test\Delta, Code\Delta$
  - Other: build/environment changes etc.
- Use cases
  - code review, regression localization, silent change/drift detection, auditability, explainability, MSR/evolution analysis, agent convergence etc.

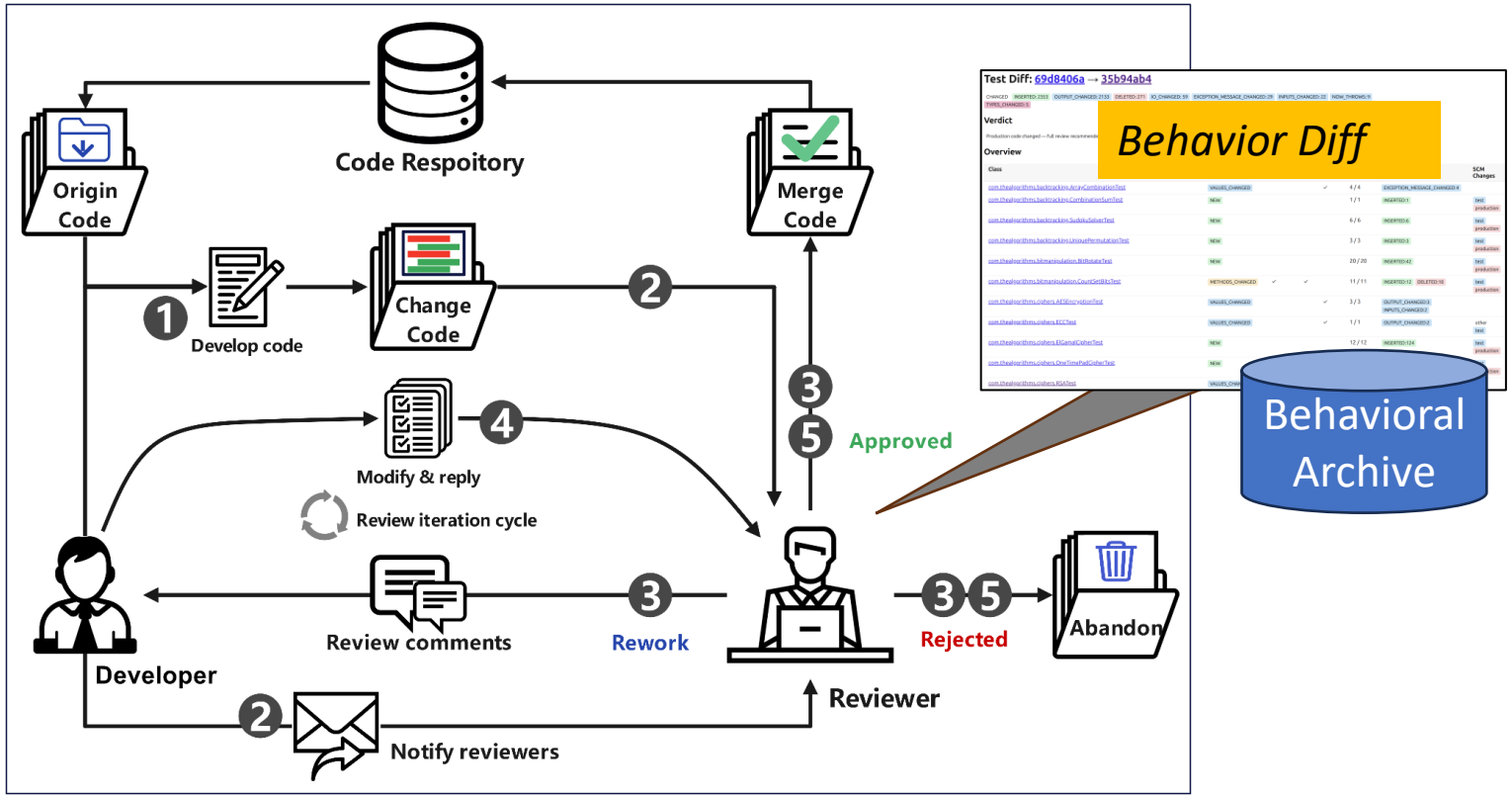


Image Credit: Yang, Zezhou, et al. "A Roadmap for Modern Code Review: Challenges and Opportunities." ACM Transactions on Software Engineering and Methodology (2026). https://arxiv.org/pdf/2405.18216

Research Agenda & Conclusion

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Research Directions

- Behavioral diffing & equivalence
  - Serialization/Canonicalization
- Archive contents - strategic tracing
- Behavioral identity
  - How do we track behavior (functional abstractions) under evolution?
- Code-test-behavior co-evolution
  - How do code, tests, and observed behavior co-evolve?
- AI/agent behavioral memory
  - Improved convergence, assertions, ...
- Federated behavioral archives
  - Project-level and cross-project archives
  - support prediction, MSR, GAI

Fundamental Challenges

- Remain
  - Oracle problem (ground truth)
  - Tracing (volume) and non-determinism
  - No tests mean no run-time evidence
- Behavioral archives change strategy
  - **decouple run-time verification from analysis**
  - offline, retrospective analysis

Thank you!

preprint

Literature

UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

- *Kessel, Marcus. "Treating Run-time Execution History as a First-Class Citizen: Co-Versioning Run-time Behavior alongside Code.“ In 34th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (FSE Companion ’26), July 05–09, 2026, Montreal, QC, Canada. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3803437.3805589*
  - *https://arxiv.org/abs/2604.16933*
- *Kessel, Marcus. "Towards Observation Lakehouses: Living, Interactive Archives of Software Behavior." 2026 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). IEEE, 2026.*
  - *https://arxiv.org/abs/2512.02795*
- *Kessel, Marcus, and Colin Atkinson. "Morescient GAI for software engineering." ACM Transactions on Software Engineering and Methodology 34.5 (2025): 1-17.*
  - *https://arxiv.org/abs/2406.04710*