

What Public Software Can Teach Us

Lessons for Open Technology Researchers from Studying Software Heritage

Stefano Zacchiroli

Télécom Paris, Institut Polytechnique de Paris
`stefano.zacchiroli@telecom-paris.fr`

26 October 2026 — Open Technology Research Symposium 2026
University of Barcelona



Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

- 
- 1 Software Heritage
 - 2 What the archive tells us
 - 3 Takeaways for open technology research

About the speaker

- Professor of Computer Science, Télécom Paris, Institut Polytechnique de Paris
- Software Heritage co-founder & Chief Scientific Officer (CSO)
- Free/Open Source Software activist (25+ years); Debian Developer & former 3x Debian Project Leader; former OSI director

Today's agenda

- 1 A brief introduction to **Software Heritage**
- 2 **Research results** that use the archive as a lens on public software
- 3 **Takeaways**: why open source, open data, and European/public platforms matter for *doing* this kind of research



Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Reference catalog



find and **reference** all
software source code

Universal archive



preserve and **share** all
software source code

Research infrastructure



enable analysis of all
software source code

One infrastructure
open and shared



The largest archive ever built



Digital preservation of software source code, with benefits

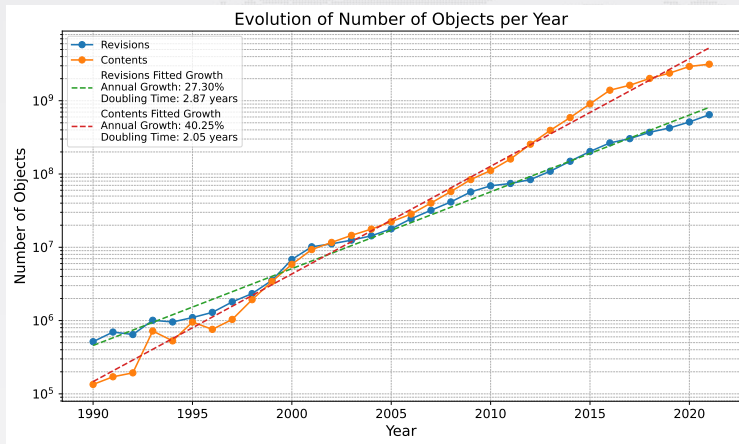
For the first time, *all* public software is a single object of study

- **Uniform**: one data model (a Merkle DAG), forge and VCS details abstracted away: git, svn, hg, tarballs, packages all look alike
- **Deduplicated**: every file, directory, commit stored once, across ~500 M projects: so *"where did this come from?"*, *"where is it used?"* become answerable
- **Cross-platform**: not only GitHub — also GitLab instances, public sector and university forges, Linux distros, package managers, rescued legacy code
- **Historical**: what was deleted, rewritten, or moved at the origin is still there
- **Open**: the archive, the code, and all curated/derived datasets are available datasets.softwareheritage.org

From preservation to research

An archive that *preserves* the Software Commons for future generations is also a **scientific observatory** to *study* it.

- 
- 1 Software Heritage
 - 2 What the archive tells us
 - 3 Takeaways for open technology research



Key findings

- Public code grows **exponentially** and has been for 30 years: original files double every ~2 years, original commits every ~3 years
- We can **trace the provenance of source code** at this scale in a compact relational model via the notion of isochrone graphs

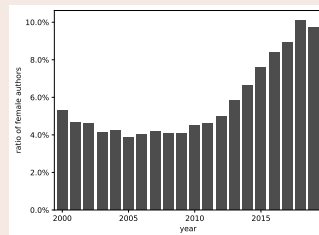
Rousseau, Di Cosmo, Zacchiroli. *Software Provenance Tracking at the Scale of Public Source Code*. Empir. Softw. Eng. 25(4), 2020

Updated data & code: gitlab.softwareheritage.org/rdicosmo/archive-growth

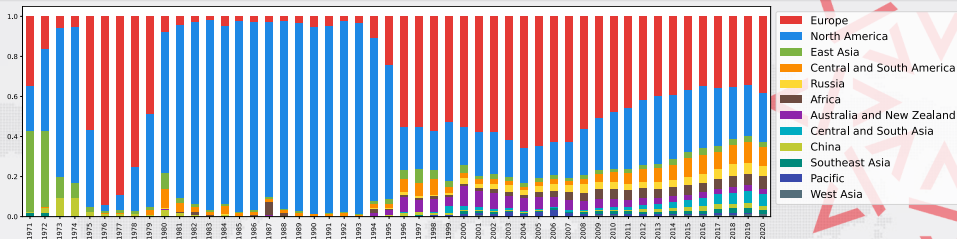
Archived commit metadata contains public information that can be mined to study diversity, equity, inclusion (DEI) traits of the global population of public code authors.

Gender gap — key findings

- Male authors contributed 92% of public code commits up to 2019.
- Female authors (and their commits) have grown stably for 15 years reaching 10% of yearly commits in 2019.
- The COVID-19 pandemic has *caused* a trend inversion (it is not just correlation!)



- Zacchiroli. *Gender differences in public code contributions: a 50-year perspective*. IEEE Software, 2021
- Rossi and Zacchiroli. *Worldwide gender differences in public code contributions [...]*. ICSE SEIS, 2022
- Casanueva et al. *The Impact of the COVID-19 Pandemic on Women's Contribution to Public Code*. Empir. Softw. Eng. 30(25), 2025



Geographic gap — key findings

- Early decades of public code dominated by contributions from North America, followed by a period of alternating dominance between North America and Europe.
- Since then geographic diversity has increased constantly, with raising importance of contributions from Central and South America.
- *Geo and Gender gap*: the trend of increased female contributions is global, with the exception of some regions in Asia where it is either slower or flat.

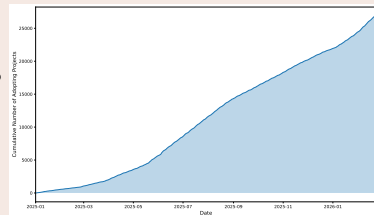
Rossi and Zacchiroli. *Geographic diversity in public code contributions*. MSR 2022

Coding agents went from nothing to a quarter of GitHub in a year

128k projects mined for traces left by coding agents:

CLAUDE.md, Co-authored-by:, copilot/* branches

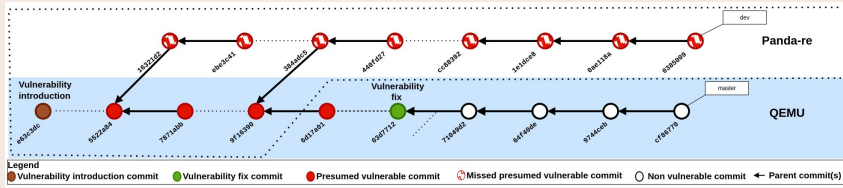
- 22–29% of projects in Jan 2025 to Feb 2026
- Broad: every topic, every language, young and old, big and small
- AI commits are $3\times$ larger; $\frac{2}{3}$ are features and bug fixes
- Follow-up replication study: projects *born after August 2025*: 72–76%, Claude Code leading



- Robbes, Matricon, Degueule, Hora, Zacchiroli. *Agentic Much? Adoption of Coding Agents on GitHub*. ACM TOSEM, 2026. Preprint: [arXiv:2601.18341](#)
- Robbes, Matricon, Degueule, Hora, Zacchiroli. *Agentic Very Much!*. Preprint: [arXiv:2606.07448](#), 2026
- Robbes, Matricon, Degueule, Hora, Zacchiroli. *Promises, Perils, and (Timely) Heuristics for Mining Coding Agent Activity*. MSR 2026

1.7 M forks left behind when a security fix lands

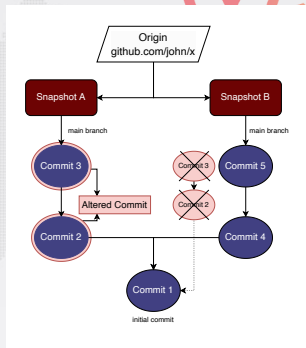
Fork between the *introduction* and the *fix* of a vulnerability, and you inherit it — silently.



- "Color" the graph with [OSV.dev](https://osv.dev) data: 7162 repos → **1.7 M potentially vulnerable forks**
- **86.3 M potentially vulnerable commits** specific to forks, *invisible to dependency-based tooling*
- 152 cases checked by hand: **135 high-severity vulnerabilities**, 9 confirmed by maintainers
- Only doable with a global, deduplicated data model: all forks, all platforms, one graph

- Lefeuvre, Reux, Zacchiroli, Barais, Combemale. *Chasing One-day Vulnerabilities Across Open Source Forks*. SCORED 2026. Preprint: [arXiv:2511.05097](https://arxiv.org/abs/2511.05097)

- Git lets you **rewrite published history** (via force push), and forges keep no trace of it
- SWH compares successive snapshots of **111 M repos: 1.22 M repos** (~1.1%) with altered histories, **8.7 M altered commits**, 11.4% of them on *main/master* branches
- ~800k **license files retroactively altered** on main branches (32k repos)
- 13 M removals of **secrets** (keys, passwords): rewriting history is not security, copies persist
- Prototype tool (*GitHistorian*): vet a repository's track record of history alterations

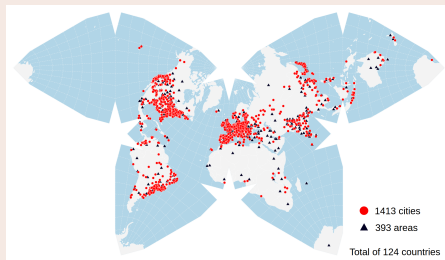


Solal Rapaport, Laurent Pautet, Samuel Tardieu, Stefano Zacchiroli

Altered Histories in Version Control System Repositories: Evidence from the Trenches

ASE 2025 <https://arxiv.org/abs/2509.09294>

1.61 M art-related repositories mined from the archive

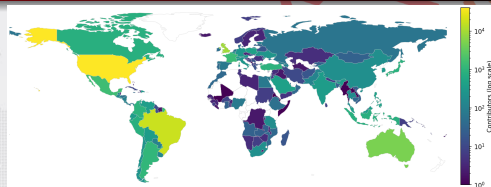


Found by scanning *every* archived origin for filesystem traces of 7 creative coding environments (Processing, p5.js, openFrameworks, SuperCollider, ...)

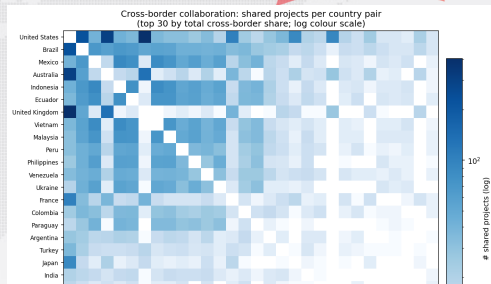
- 84,175 repositories (5.5%) are already offline, but preserved in Software Heritage
- Contributors in 124 countries, on every continent
- 96% of p5.js sketches synthesize new material, 50% use randomness
- Three purposes: CS education, art education, and art itself

Baudry, Gamage, Gonzalez Fernandez, MK, Pena Mendieta, Souza Pinter, Zacchiroli. *Art in Humanity's Code*. IEEE Trans. Softw. Eng., to appear. Open access preprint: [arXiv.org:2608.05174](https://arxiv.org/abs/2608.05174)

- 193 UN member states, 288 K projects *with* government contributors (not owned by governments), 93 K government contributors
- Public code contribution tracks **e-government development** (UN EGDI, $\rho = 0.52$) twice as strongly as **GDP per capita** ($\rho = 0.26$)
- 75% of projects have **no detectable licence**; 94.5% sit on a **single commercial platform** (GitHub); citation metadata nearly absent
- Method: filter on commit authors/committers by **email domain**; swap in *your* institution, sector, or country



government contributors per country (log scale)



cross-border collaboration: shared projects per country pair

- 
- 1 Software Heritage
 - 2 What the archive tells us
 - 3 Takeaways for open technology research

Lesson 1: study the commons, not a platform

Platform-specific data has blind spots

- Up to **16% of forks** are invisible if you only trust GitHub's "fork" button (Pietri et al., MSR 2020)
- **5.5%** of art-related repositories are already gone at their origin; alive in SWH
- **1.22 M** repositories had their history rewritten; the forges kept no trace
- **94.5%** of projects with government contributors on one platform; GitHub studies go blind on the rest

What you cannot see, you cannot study

- **Survivorship bias** is the default when you sample what is online *today*
- Any research question about the *whole* ecosystem needs a **platform-agnostic, historical, reproducible** corpus

Lesson 2: open source + open data + archival = reproducible research

Every result in this talk can be re-run, extended, or refuted

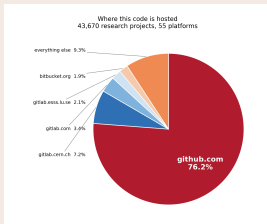
- **Open data:** corpora are public datasets (archive export, compressed graph, curated subsets) datasets.softwareheritage.org
- **Open source:** the archive stack and the analysis tooling are free software
- **Intrinsic identifiers:** pin code you study with **SWHIDs** (ISO/IEC 18670:2025), not URLs that rot

Research access granted at a platform's discretion is not research infrastructure

- Social media researchers learned it the hard way: free academic API access to Twitter/X closed in 2023, Reddit's API changed the same year
- Software researchers too: widely used GitHub mirrors (e.g., GHTorrent) are no longer maintained; rate limits and terms of service shape what can be asked
- Europe had to *legislate* researcher access to platform data (DSA, Art. 40); for public code, Software Heritage is a commons-based alternative that *exists today*

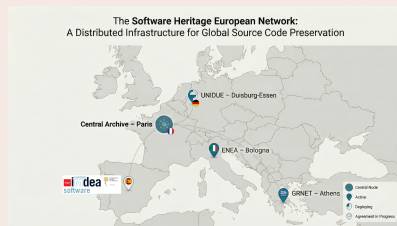
Lesson 3: public, European-rooted, globally shared platforms

Dependence, measured



Research software in SWH from the 196 institutions of the five EOSC Science Clusters (43,670 projects): **76% on one platform**, one jurisdiction; **5.4%** already gone at origin

Resilience, built



A non-profit, open, international archive with **independent mirrors** run by other organisations, in their own jurisdictions, on their own stacks

Sovereignty is not isolation: **plural custody** of a shared commons, so that neither *our code* nor *our ability to study it* depends on a single vendor's decisions.

What this means for you

Open technology researchers

- **Build on the archive** instead of scraping platforms; cite your corpus with SWHIDs; give derived datasets back
- **Ask ecosystem-wide questions:** governance, economic value, procurement outcomes, security, adoption — the data is there

Public institutions & policy makers

- **Archive by default** the code you produce or procure; make it identifiable (machine-readable metadata, e.g., `publiccode.yml`, CodeMeta)
- Prefer **open, plural, public-interest platforms** for where code is developed, and observed
- **Fund shared research infrastructure:** archive, mirrors, ...and the observatory!

Conclusion

- Software Heritage preserves *all* public source code **and its full history** — uniform, deduplicated, platform-agnostic, ~500 M projects
- Preserving it made the Software Commons **observable and researchable**
- Doing this research well requires **open source, open data, and public platforms** that no single vendor can switch off

Learn more, and get in touch

- Papers: softwareheritage.org/publications
- Data: datasets.softwareheritage.org

[Stefano Zacchioli](#) / stefano.zacchioli@telecom-paris.fr / [@zacchiro@mastodon.xyz](https://mastodon.xyz/@zacchiro)

Deep dives on selected results

- Coding agents on GitHub
- One-day vulnerabilities across forks
- Altered histories in Git repositories
- Art in humanity's code

result #4

result #5

result #6

result #7

Software Heritage

- A peek under the hood
- Software Hash IDentifiers (SWHIDs)

A new kind of tool, in the space of one year

- *Completion* LLMs finish a line; *coding agents* take a whole task — find the bug, write the feature, open the pull request
- Claude Code, Copilot, Cursor, Codex, Gemini, Jules, ... nearly all shipped in the first half of 2025
- Unlike IDE completion, they are **observable in the wild**

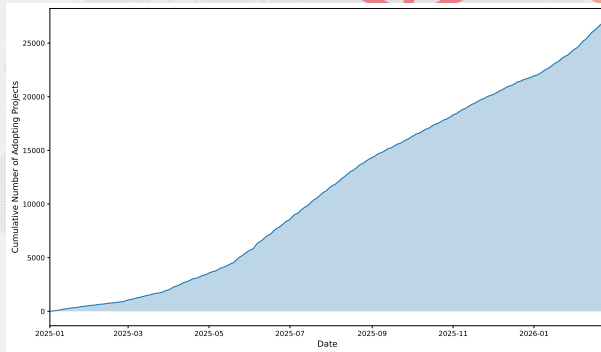
Four families of traces, 63 agents, 196 heuristics

- *Files* (93): CLAUDE.md, AGENTS.md, .cursorrules
- *Authors* (79): Co-authored-by: Claude
- *Branches* (20): copilot/fix-... — *Labels* (4): PRs tagged codex
- Hand-validated on 1,200 artifacts: 0.5–1% false positives
- **128,018 GitHub projects** swept, 2025-01-01 to 2026-02-21

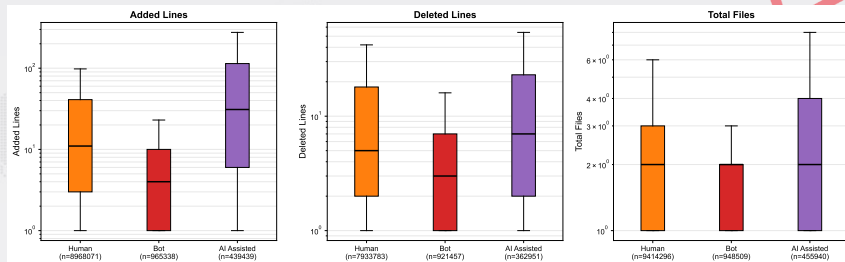
How many projects have adopted coding agents?

RQ1 — key findings

- 12.1% of projects show agent files, a further 11.5% show agent commits — the two barely overlap
- Overall estimate: **22–29% of GitHub projects**, for a tool category barely one year old
- Inflection points track the releases: Claude Code (Feb 2025), the wave of May 2025, and a fresh acceleration in early 2026
- Projects *born after August 2025* tell a rather different story — shortly



What do coding agents actually commit?



RQ5 & RQ6 — key findings

- AI-assisted commits are **3× larger** than human ones (median +31 vs. +11 lines) — and they delete more too: churn goes up
- 790 Claude Code commits classified by hand: **feat 36% + fix 30%**, two thirds of the sample — agents build and repair
- **chore** is only 7% (31% for humans): the boring maintenance stays human, or stays with the old bots

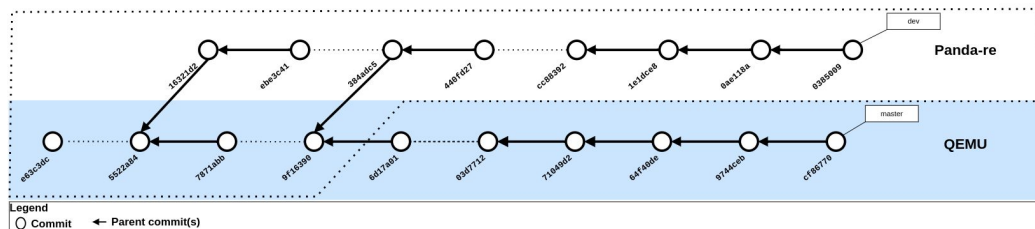
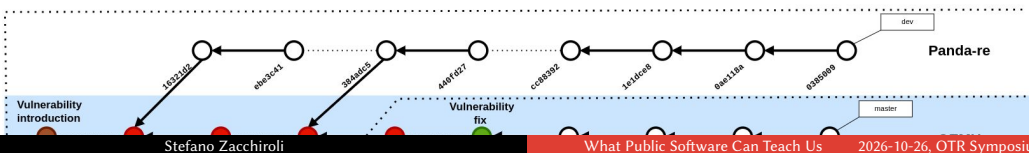
... and the newest projects are already past two thirds

A follow-up on 12,794 projects *created after* August 2025

- Conservative adoption **71.8%** (high estimate 76.2%), against 26.5% for the older cohort measured at the same time — more than double
- File-level traces: 60.0% vs. 14.7%, a four-fold difference
- Use is also more *intensive*: 41.2% of new adopters are “pervasive” (>20% AI-assisted commits), vs. 21.1% among older projects
- **Claude Code leads** in new projects (6,443), three times its closest competitor; Copilot falls to a distant third
- Human-authored commits grow too — a strong hint that a lot of agentic activity still goes undetected

- Robbes, Matricon, Degueule, Hora, Zacchiroli. *Agentic Much? Adoption of Coding Agents on GitHub*. ACM TOSEM, 2026. Preprint: [arXiv:2601.18341](https://arxiv.org/abs/2601.18341)
- Robbes, Matricon, Degueule, Hora, Zacchiroli. *Agentic Very Much!*. Preprint: [arXiv:2606.07448](https://arxiv.org/abs/2606.07448), 2026
- Robbes, Matricon, Degueule, Hora, Zacchiroli. *Promises, Perils, and (Timely) Heuristics for Mining Coding Agent Activity*. MSR 2026

- Any change to a piece of software (*commit*) can **introduce a new vulnerability**.
- Or it can **fix an existing vulnerability**.
- What happens if a project is forked **between introduction and fix** of a vulnerability?
- It inherits the vulnerability, ... until the change with the fix is integrated.



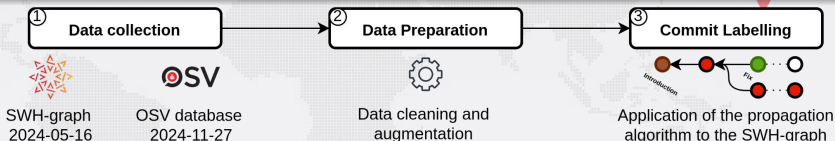
Chasing one-day vulnerabilities across forks, at archive scale

Approach

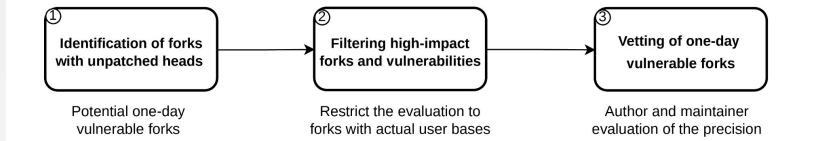
Start from a **public DB of vuln. introduced/fixed** in public commits (e.g., [OSV.dev](https://osv.dev)).

- 1 "Color" the entire graph of public code development history **with vulnerability info.**
 - Software Heritage is the only place where this can be done at the scale of all forks, across all public code, independently of specific development platforms.
- 2 **Inform maintainers** of vulnerable forks. (After validation.)

Phase 1: color the commit graph with vuln. info



Phase 2: find and vet vulnerable forks



Results (highlights)

- Starting from 7162 repos in OSV, we identified **1.7 M forks potentially vulnerable** in their most recent commit.
 - 86.3 M potentially vulnerable commits specific to forks, not findable with current tools.
- We manually verified 152 cases, confirming **135 high-severity vulnerabilities in popular forks**; 9 were further confirmed by maintainers.



panda-re/panda

Fork of QEMU, system emulator and virtualizer

CVE-2019-13164



go-nv/goenv

Fork of pyenv, for Go

CVE-2022-35861

XPERIA
Sony Smartphone

sonyperiadev/kernel

Fork of Linux, for the Xperia dev community

CVE-2021-45485

CVE-2021-4154

bitcoin-sv/bitcoin-sv

Fork of Bitcoin

CVE-2021-37492



Romain Lefeuvre, Charly Reux, Stefano Zacchiroli, Olivier Barais, Benoit Combemale

Chasing One-day Vulnerabilities Across Open Source Forks

SCORED 2026. Preprint: <https://arxiv.org/abs/2511.05097>

Git allows **rewriting history** (of the version control kind!)

```
$ git rebase --interactive <...>  
$ git push --force
```

- Useful feature! E.g., to clean your code before sharing it
- Also annoying and **risky** on public branches
 - **Hinders reproducibility** and **voids availability** of specific Git objects
 - **Supply chain concerns:** what was altered and why?

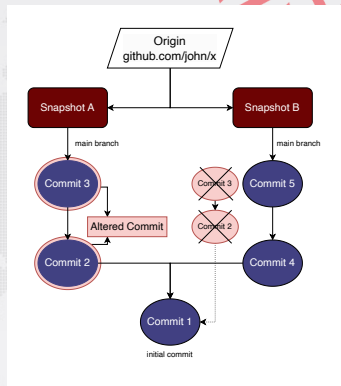


- 1 How often are public Git histories altered?
- 2 When this happens, what is changed and why?

Note: forges do not keep the history of these modifications. SWH is the only place where they can be analyzed globally.

A large-scale analysis of Git repository alterations

- 1 Retrieve from Software Heritage **111 M Git repositories** (1) archived at least twice, (2) with different states ("snapshots")
- 2 For each repository, compare snapshots 2-by-2 to detect **altered histories** and the **root cause** of each alteration
- 3 Classify **altered commits** by what changed before/after alteration



Solal Rapaport, Laurent Pautet, Samuel Tardieu, Stefano Zacchiroli

Altered Histories in Version Control System Repositories: Evidence from the Trenches

ASE 2025 <https://arxiv.org/abs/2509.09294>

Key findings on destructive repository alterations

How often?

- **1.22M repositories** contain altered histories (~1.1%)
- **8.7M altered commits** → Pro tip: make sure your important commits are in SWH!

Where?

- Pull request branches: 37.6% → Might be OK, depending on the workflow
- **main/master branches**: 11.4% → Concerning!

What?

- **Commit metadata** (13.3%): author, date, message, ... → Risky for provenance & IP
- **File/dir. changes** (76.8%): *retroactive* file modifications and/or deletions

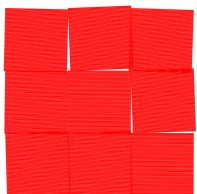
Case study #1: License changes

- ~800K **retroactively altered license files** on main branches
- Spanning **32k repositories** (76 with 1000+ stars)
- 79% version updates (e.g., GPL 2→3)
- 14% full changes (e.g., MIT→GPL)
- Serious concern: retroactive changes may *de facto* suppress previously granted rights (without an archival copy!)

Case study #2: Removing secrets

- 13M file removals involved files/paths referring to "secrets"
 - Examples: **private keys, certificates, passwords**
- Spanning 75k repositories
- Issue: **History alteration** $\nrightarrow$ **security** (archived copies persist)
- Keys must be rotated, not only purged from Git
- Might indicate **poor security practices**.

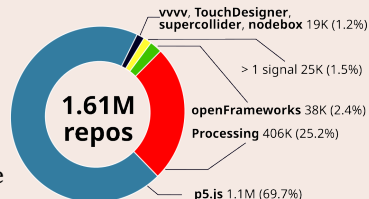
Artists have been coding for 60 years



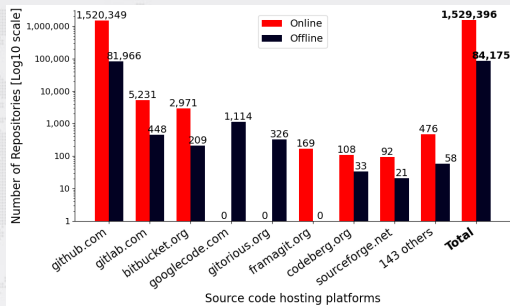
- *Generative art* goes back to Molnár (1975), Sýkora, Cornock, Edmonds
- Artist-led *open source* libraries made it mainstream: [Processing](#), [p5.js](#), [openFrameworks](#), [SuperCollider](#), [TouchDesigner](#), [nodebox](#), [vvvv](#)

Finding art-related code at archive scale

- *Signal*: filenames of 7 art environments (p5.js, ofApp.cpp, .pde, .scd, .toe, .ndbx, .v4p)
- Swept *every* archived origin, via the compressed graph (2025-05-18)
- **1.61 M repositories**: the first large-scale study of art code



Where art-related code lives — and what is already gone

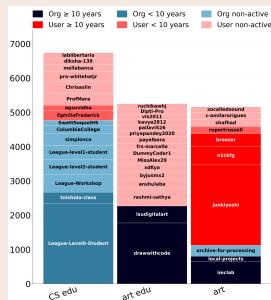


RQ1 — key findings

- GitHub holds 99.3% of it; artists, makers and schools also publish on GitLab, Bitbucket, Codeberg, SourceHut and 143 niche forges
- **84,175 repositories (5.5%) are no longer online:** only the archive has them
- Google Code and Gitorious are 100% gone — e.g. 1,114 Processing-era repositories

Why people publish art-related code

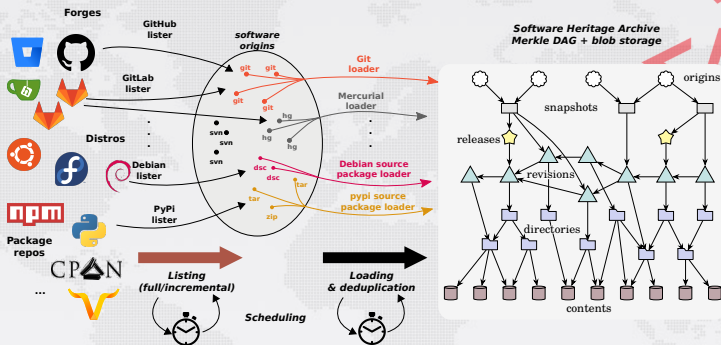
Top 42 accounts, 17,156 repositories: three purposes



- **CS education**: art as a way into programming — the League of Amazing Programmers alone accounts for 3,309 repositories
- **Art education**: code as a design medium — e.g. drawwithcode, the *Creative Coding* course at Politecnico di Milano
- **Art**: one repository per artwork, installation or tool — e.g. junkiyoshi, one openFrameworks piece a day since 2016 (2,344 repositories)
- Art accounts keep going; educational ones peak in 2020–2022, then stop

Baudry, Gamage, Gonzalez Fernandez, MK, Pena Mendieta, Souza Pinter, Zacchiroli. *Art in Humanity's Code*. IEEE Trans. Softw. Eng., to appear. Open access preprint: [arXiv.org:2608.05174](https://arxiv.org/abs/2608.05174)

A peek under the hood: a universal archive



Global development history permanently archived in a uniform data model

- over 30 billion unique source files from over 445 million software projects
- ~3PB (compressed) blobs, ~60 B nodes, over 1 trillion edges



An emerging standard

- Adoption: [SPDX 2.2](#), IANA-registered "swh:" URI prefix, WikiData [P6138](#), ...
- Breaking news: [standard ISO/IEC 18670:2025](#)

Examples

- [Apollo 11 AGC excerpt](#)
- [Quake III rsqrt](#)