Software Heritage

Towards A New Era for Software Engineering, Cybersecurity, and AI

Roberto Di Cosmo

Director, Software Heritage
Inria and Université Paris Cité

October 14, 2026
Seminario SISTEDES, online



Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

Software Heritage

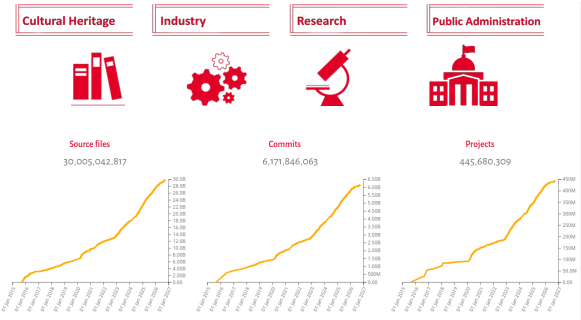
THE GREAT LIBRARY OF SOURCE CODE

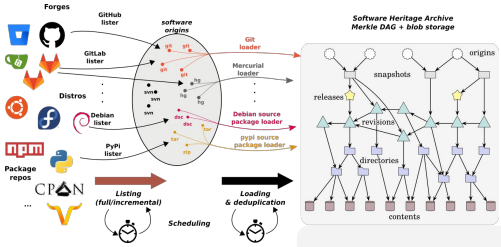
Inria with unesco

The largest open source code archive: one infrastructure, open, shared, non profit

Unique digital common good *built in France since 2015*







5000+ platforms

All versions, all history development in a single graph

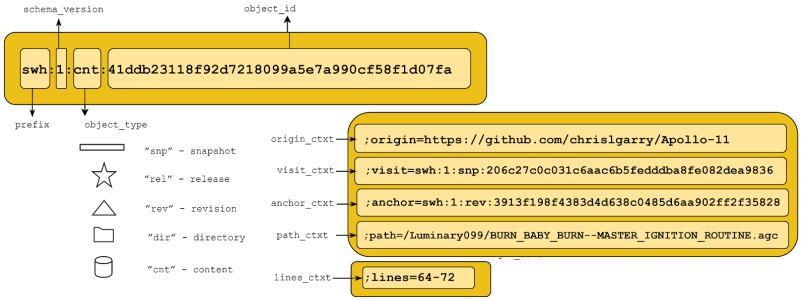
- **50 × 10⁹** nodes
- **1000 × 10⁹** edges
- ~ **3 PB** of storage

A revolutionary **infrastructure** ensures **availability** guarantees **integrity** enables **traceability**

The Software Hash identifier (SWHID)

Software Heritage Identifiers (SWHID)

see swhid.org



50+B
intrinsic,
decentralised,
cryptographic

Full fledged *source code references* for traceability, integrity and reproducibility

- Linux Foundation SPDX 2.2
- IANA-registered `"swh:"`
- WikiData property P6138

Examples: Apollo 11 AGC excerpt, Quake III rsqrt
Guidelines available, see the HOWTO

ISO/IEC 18670, see swhid.org

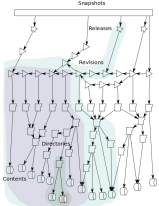
Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

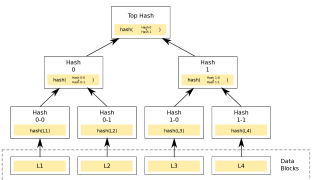
From archive to revolutionary infrastructure

Modern "Library of Alexandria", *international, non profit, long term* initiative
addressing the needs of *industry, research, culture and society as a whole*

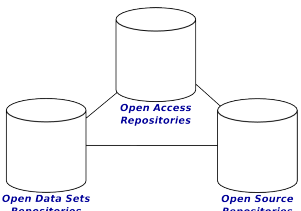
Software Graph



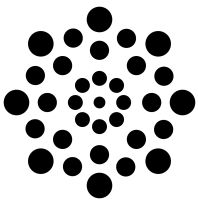
Software Blockchain



Open Science pillar



Big Code



Today we focus on

Open Science • Large-scale analysis • Cybersecurity • AI

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive — ACAD_ADOPTION
4. Research on software: datasets from the archive — DATASETS
5. Selected highlight: Impact on software studies — ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) — SECURITY
8. Selected highlight: AI and transparent LLMs — CODECOMMONS
9. One last thing
10. Conclusion

An example is worth a thousand words

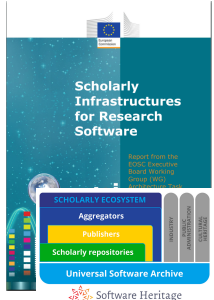
- Browse (e.g. Apollo 11 [excerpt], your work may be already there !)
- Trigger archival, use the updateswh browser extension, configure the webhooks
- Get and use SWHIDs (full specification available online)
- Cite software with biblatex-software package from CTAN
  - Overleaf ACMART template available
- Example in journals: article from IPOL
- Example with Parmap: devel on Github, archive in SWH, curated deposit in HAL
- Extracting all the software products for Inria, for CNRS, for CNES, for LIRMM or for Rémi Gribonval using HalTools
- Curated deposit in SWH via HAL, see for example: LinBox, SLALOM, Givaro, NS2DDV, SumGra, Coq proof, ...
- Example use in research articles:
  - compare Fig. 1 and conclusions in the 2012 version and the updated version
  - SWHID in a replication experiment

A few adoption indicators

Policy







- Recommendations in ANR 2023 guidelines (p. 17)
- HAL+SWH in the Open Science software booklet

Projects

eosc | FAIRCORE4EOSC
Core Components Supporting a FAIR EOSC

The CodeMeta Project

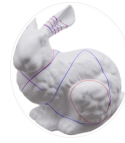
eosc | FAIR-IMPACT
Expanding FAIR solutions across EOSC

Users and collaborations

What are they “referencing”?

source	n	percentage
Not available	2868	46.22
GitHub	1151	18.55
software heritage	387	6.24
zenodo	142	2.29
r package	70	1.13
cran	56	0.90
r package version	54	0.87
gitlab	35	0.56

Graphics Replicability Stamp Initiative



b/Surf: Interactive Bézier Splines on Surface Meshes

Claudio Mancinelli, Giacomo Nazzaro, Fabio Pellacini, Enrico Puppo
IEEE Transactions on Visualization and Computer Graphics (TVCG)

doi | Repository

Best practices to disseminate

Artifact Evaluation Committees

current status

- many guidelines developed in the 2010's ignore Software Heritage
- ACM DL or Zenodo deposit of a zip + DOI still required

state of the art

- archive on Software Heritage
- use the qualified SWHID

Software citation

state of the art

- include codemeta.json (or citation.cff, better if generated from codemeta.json)
- generate biblatex entry, and use biblatex-software (CTAN, acmart.cls, etc.)



Want the full story? The course notes ***The Source Code of Science***

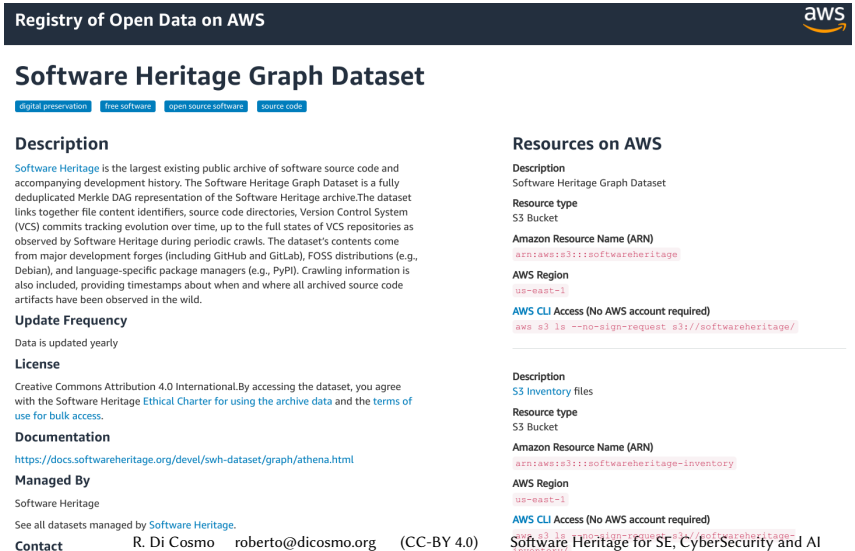
Illustration: AI-generated (Google Gemini), author-directed; provenance record in the book's appendix.

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

The full graph in the AWS Open Data collection

https://registry.opendata.aws/software-heritage/



Registry of Open Data on AWS

aws

Software Heritage Graph Dataset

digital preservation | free software | open source software | source code

Description

Software Heritage is the largest existing public archive of software source code and accompanying development history. The Software Heritage Graph Dataset is a fully deduplicated Merkle DAG representation of the Software Heritage archive.The dataset links together file content identifiers, source code directories, Version Control System (VCS) commits tracking evolution over time, up to the full states of VCS repositories as observed by Software Heritage during periodic crawls. The dataset's contents come from major development forges (including GitHub and GitLab), FOSS distributions (e.g., Debian), and language-specific package managers (e.g., PyPI). Crawling information is also included, providing timestamps about when and where all archived source code artifacts have been observed in the wild.

Update Frequency

Data is updated yearly

License

Creative Commons Attribution 4.0 International.By accessing the dataset, you agree with the Software Heritage Ethical Charter for using the archive data and the terms of use for bulk access.

Documentation

https://docs.softwareheritage.org/devel/swh-dataset/graph/athena.html

Managed By

Software Heritage

See all datasets managed by Software Heritage.

Contact

Resources on AWS

Description
Software Heritage Graph Dataset

Resource type
S3 Bucket

Amazon Resource Name (ARN)
`arn:aws:s3:::softwareheritage`

AWS Region
`us-east-1`

AWS CLI Access (No AWS account required)
`aws s3 ls --no-sign-request s3://softwareheritage/`

Description
S3 Inventory files

Resource type
S3 Bucket

Amazon Resource Name (ARN)
`arn:aws:s3:::softwareheritage-inventory`

AWS Region
`us-east-1`

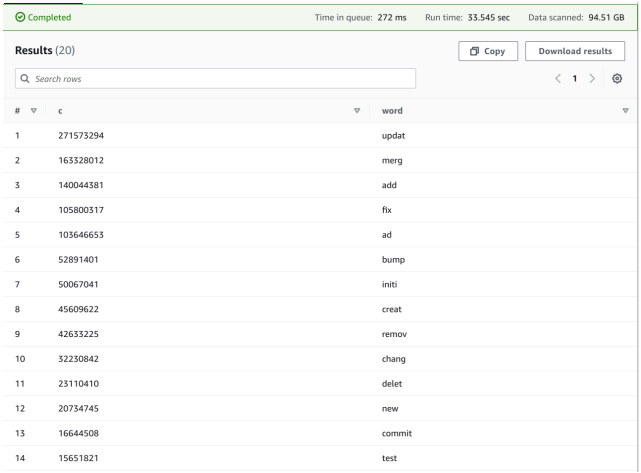
AWS CLI Access (No AWS account required)
`aws s3 ls --no-sign-request s3://softwareheritage-inventory`

Example: most popular commit verbs (stemmed)

Query using Amazon Athena

```
SELECT COUNT(*) AS C, word FROM (
        SELECT word_stem(lower(split_part(
         trim(from_utf8(message)),' ', 1)))
        AS word FROM revision
   WHERE length(message) < 1000000)
WHERE word != ''
GROUP BY word
ORDER BY C
DESC LIMIT 20;
```

Results



Completed — Time in queue: 272 ms — Run time: 33.545 sec — Data scanned: 94.51 GB

Results (20) — Copy — Download results

Search rows — 1

#	c	word
1	271573294	updat
2	163328012	merg
3	140044381	add
4	105800317	fix
5	103646653	ad
6	52891401	bump
7	50067041	initi
8	45609622	creat
9	42633225	remov
10	32230842	chang
11	23110410	delet
12	20734745	new
13	16644508	commit
14	15651821	test

Going beyond SQL

State-of-the-art graph compression from social networks



Paolo Boldi, Antoine Pietri, Sebastiano Vigna, Stefano Zacchiroli

Ultra-Large-Scale Repository Analysis via Graph Compression

SANER 2020, 27th Intl. Conf. on Software Analysis, Evolution and Reengineering. IEEE

Results

Full graph structure (58.7 B nodes, 1.04 T edges) in 400 GiB RAM

- traversal time is tens of ns per edge
- bidirectional traversals implemented
- beware: metadata access is still *off RAM*

~~Java~~ Rust and gRPC APIs available …

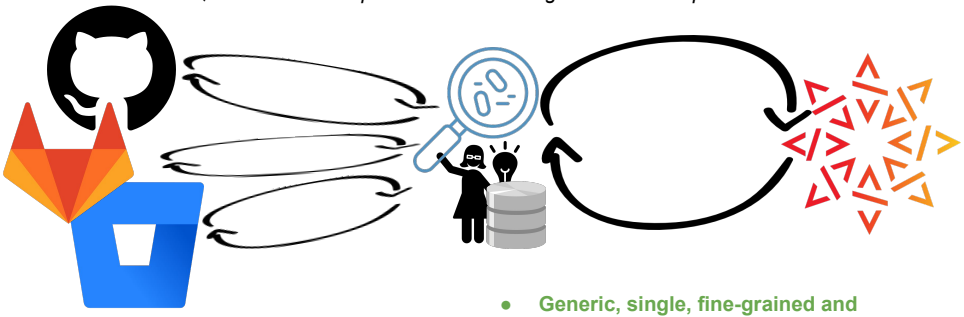
docs.softwareheritage.org/devel/swh-graph/grpc-api.html

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

Mining Android Applications on Software Heritage

RQ: how to build a specific dataset for a given research question?



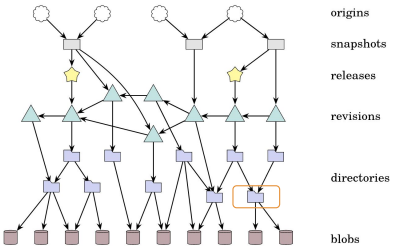
- **Specific and limited API**
- **Hardly reproducible**

- **Generic, single, fine-grained and unlimited API**
- **Growing number of source codes**
- **Easy to update the dataset**

(from the Inria/IRISA DiverSE team)

Using the SWH merkle dag to identify android repositories

Identify android application repositories = Find the AndroidManifest.xml among the sources

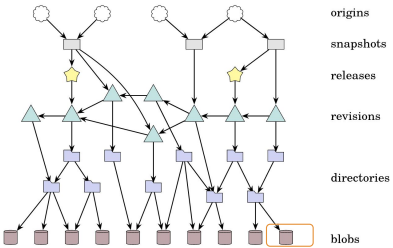


SWH Merkle DAG, Antoine Pietri

1) Iterate over the graph nodes until you find a directory node containing a file named "AndroidManifest.xml".

Using the SWH merkle dag to identify android repositories

Identify android application repositories = Find the AndroidManifest.xml among the sources

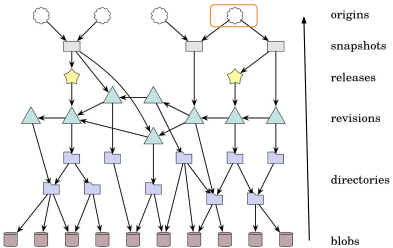


SWH Merkle DAG, Antoine Pietri

2) Extract the SWH identifier of the blob corresponding to the AndroidManifest.xml and download the corresponding file through the SWH Web API

Using the SWH merkle dag to identify android repositories

Identify android application repositories = Find the AndroidManifest.xml among the sources



SWH Merkle DAG, Antoine Pietri

3) Traverse the graph in backward direction to the origin node and get the repository url

Bottomline

Broad variety of sources in *one open dataset*

reduces usual GH bias

Reference simple *standard data format*

VCS and forge details are abstracted away

Simplifies reproducibility packages

no need to create a full copy, *just list the SWHIDs!*

Software Heritage does the heavy lifting for you

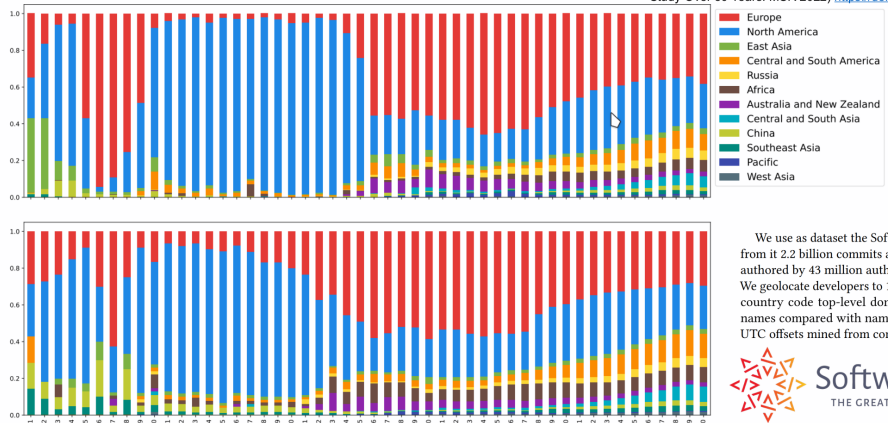
no need to scrape/download repositories all over again

Example: (Open) Source Code comes from all over the world

MSR '22, May 23–24, 2022, Pittsburgh, PA, USA

Davide Rossi and Stefano Zacchiroli

Geographic Diversity in Public Code Contributions: An Exploratory Large-Scale Study Over 50 Years. MSR 2022) https://doi.org/10.1145/3524842.3528471

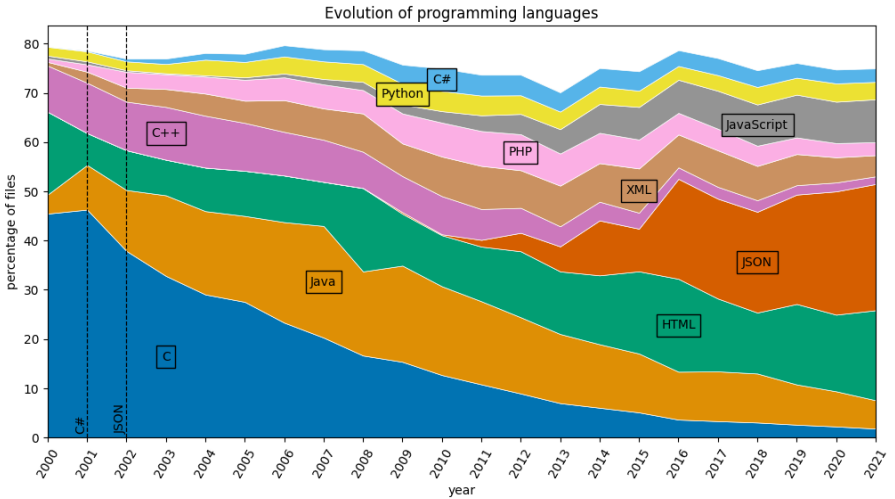


We use as dataset the Software Heritage archive [3] and analyze from it 2.2 billion commits archived from 160 million projects and authored by 43 million authors during the 1971–2021 time period. We geolocate developers to 12 world regions, using as signals email country code top-level domains (ccTLDs) and author (first/last) names compared with name distributions around the world, and UTC offsets mined from commit metadata.

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

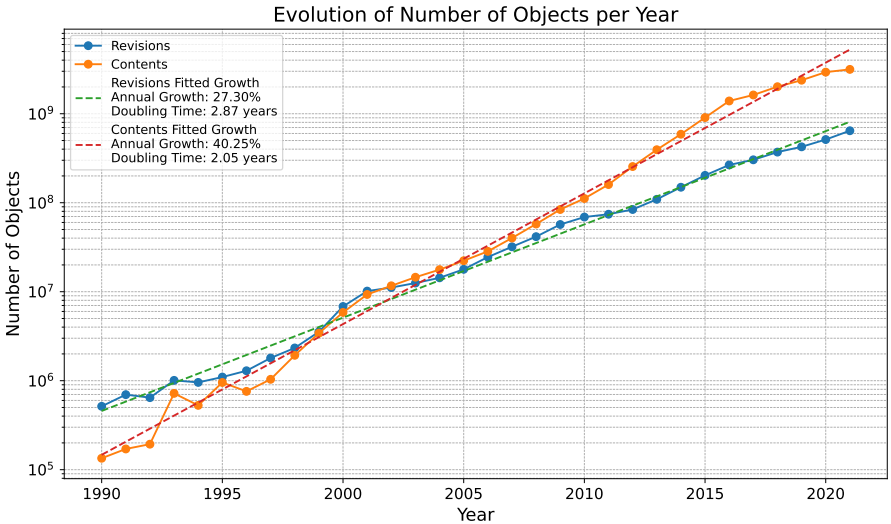
Figure 3: Ratio of commits (above) and active authors (below) by world zone over the 1971–2020 period.

Example: Programming language evolution in (Open) Source



A. Desmazières, R. Di Cosmo, V. Lorentz
50 years of programming language evolution through the Software Heritage Looking Glass
International Conference on Mining Softwre Repositories (MSR) 2025.
https://hal.science/hal-04924849/

Example: (Open) Source Code grows at an exponential rate



Software Heritage

Get the data and the code

Example: Institutional contribution analysis

Question:

can we leverage the Software Heritage graph to map contributions from a research institution in the galaxy of software development?

Answer: let's try

- Extract: projects and contributions using email domain matching in the Software Heritage graph.
- Deduplicate: Cluster forks/copies using commit metadata, select a canonical repository.
- Enrich: Add GitHub metadata (topics, README, etc.) for context and visibility.
- Analyze: Compute key metrics on impact, contributor domains, and research relevance.

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

From the **Software Sky** to **Institutional Insight:** Anatomy of an Observational Pipeline

Observation parameters (config file)

Stage 0: The Software Sky

The process begins with a complete map of the software universe, built from the Software Heritage archive, containing all observable software 'stars'. This stage is configured by observation parameters and a spectral filter definition (email domain selection).

Phase 1: Target Selection

Identify the stars associated with a specific institution by selecting contributors using institutional identifiers, which isolates the relevant region of the software sky for focused observation.

Quality Gate: Signal Strength Check

Before proceeding, the pipeline verifies that the selected region contains enough observable signals (software contributions) to produce statistically meaningful results.

Egeait to Nous Kaba

Phase 2: Gravitational Cleaning (Deforking)

The process renvents gravitational artifects such as mitrers and fures from the data to retain only the physically meaningful celestial objects (original software projects).

Phase 4: Spectral Enrichment

Each observed object is euriched with spectral data—such as activits, domains, and licenses—collected from multiple observation instruments to provide a deeper understanding.

Activity

Domains

Licenses

Adeptive Observation Scheduling

This fecture ensures the automatic relation of observation tokens to bypass iassument (DPQ limits, ensuring continuous data collection.

Observation Cache

A oerWixeen observation log is maintained, preventing redandant measurements and speeding up subsequent analyzes.

Phase 5: Astrophysical Analysis

The entiched observations are analysed to compute metrics, classily objects into scientific domains, and generate a synthetic, comprehensive view of the observed system.

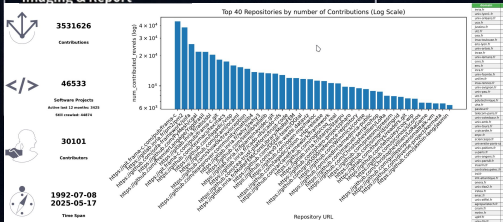
Phase 6: High-Resolution Imaging & Report

3531626
Contributions

46533
Software Projects
Active last 12 months: 3025
Still-created: 16874

30101
Contributors

1992-07-08
2025-05-17
Time Span

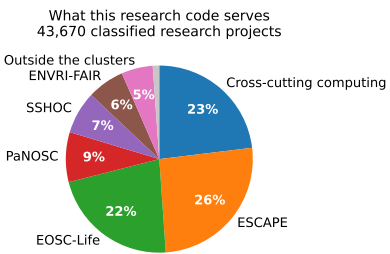


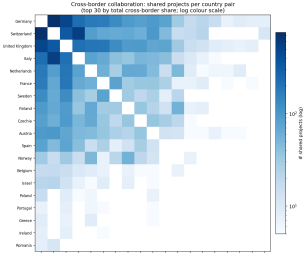
43,670 research code projects from the institutions of the five EOSC Science Clusters (ESFRI), **24,457** contributors, **20** countries.

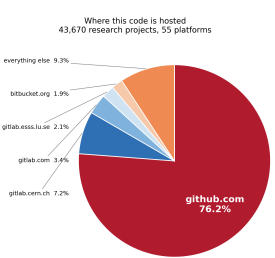
76% of them on **github.com**, 7.2% on CERN's own forge, 2.1% on ESS's.

5.4%, **2,356 projects**, no longer found at their origin, preserved in Software Heritage.

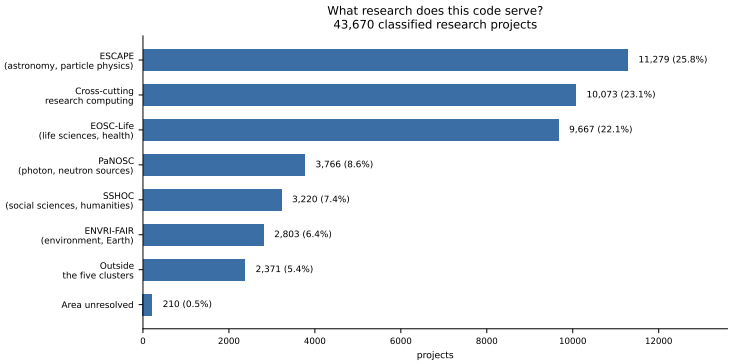
the full report



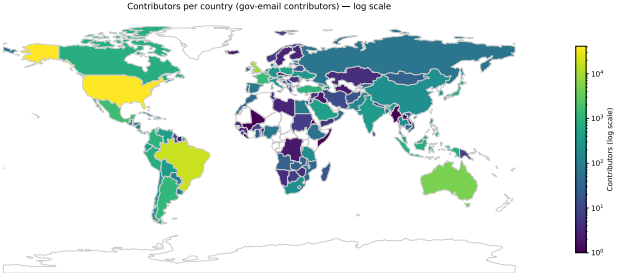


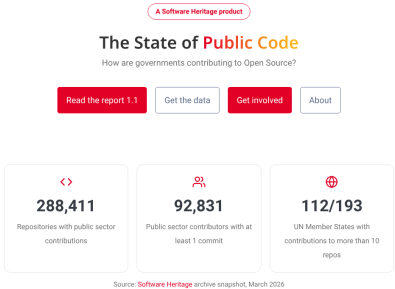


Left: what the code serves; 1.1% carry no cluster label. Middle: 102 country pairs, 6,430 shared projects. Right: where the projects are hosted. Selection: 196 institutional email domains, at least 25 contributions, every project classified as research.



Software Heritage is the only infrastructure capable of producing these metrics





- Public sector contributors, top three countries: US 41,253, Brazil 17,725, UK 9,252

stateofpubliccode.org

With UN ODET and UNESCO • gov-domain contributions: a floor, not a ceiling

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

Programme

France 2030 PTCC — 5 years (2024–2028) — 8 research teams

Co-leaders: Olivier Barais (Inria, DiverSE) · Stefano Zacchiroli (Télécom Paris, ACES)

Partners — academic

- APR — LIP6, Sorbonne Université *(prog. languages, static analysis MOPSA, verification)*
- ACES — Télécom Paris *(cybersecurity, software supply chain)*
- CEA List *(static analysis: Frama-C, Binsec)*

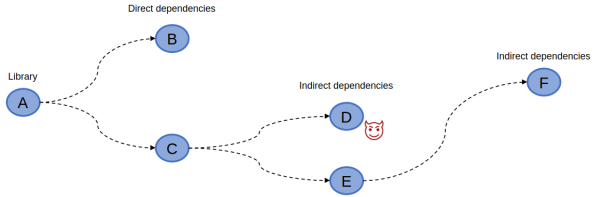
Partners — Inria

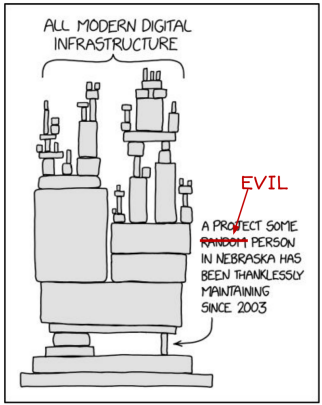
- DiverSE — supply chain security
- RMOD — reverse engineering (Moose)
- Software Heritage — archive, infrastructure
- Spirals — self-protective systems, web security
- Whisper — Linux kernel, Coccinelle

Software supply chain attacks

Reusing OSS via dependencies

- Software dependencies: a popular way of reusing open source software.
- Software product A uses functionalities implemented in OSS product B . . . and so on.



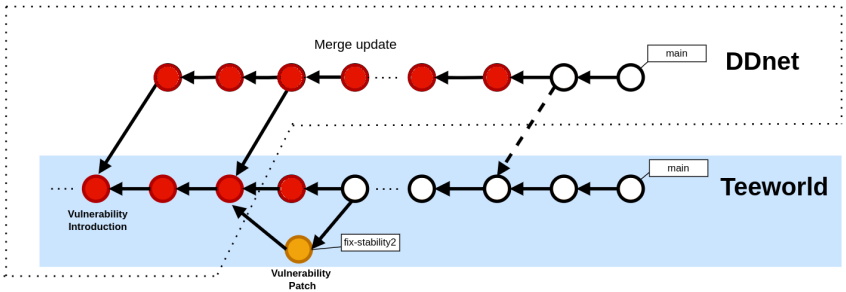


based on xkcd.com/2347

Attacking the software supply chain

- Attacking undermaintained "leaf" packages (e.g., D) → efficient attack strategy
- Many documented attacks: event-stream (2018), node-ipc (2022), XZ utils (2024), . . .

- 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.



An universal knowledge base about public code vulnerabilities

Vision

- Software Heritage is the perfect (and only) place where to build an universal knowledge base that maps known vulnerabilities to public code artifacts.
- SWH can provide an open data API mapping SWHIDs to CVEs, that knows about *all public commits* and can be leveraged to increase open source security.

EU Cyber Resilience Act (CRA)

- Key helper to abide to CRA obligations, coming into effect ~Q3 2026.
- SWH funding member of the Open Regulatory Compliance Working Group.

Roadmap

- Context: SWHSec project (PTCC-funded, 2023-2027).
- Current status: working prototype that processes OSV.dev data and use it to "color" the entire SWH commit graph (~6 billion commits) with vulnerability information.


Syful Islam, Stefano Zacchiroli.
On the Informativeness of Security Commit Messages: A Large-scale Replication Study.
To appear in EASE 2026, ACM, Glasgow, UK.


Syful Islam, Stefano Zacchiroli.
On the Use of Commit Messages for Corrective Software Maintenance: A Systematic Mapping Study.
To appear in EASE 2026, ACM, Glasgow, UK.


Romain Robbes, Théo Matricon, Thomas Degueule, André Hora, Stefano Zacchiroli.
Promises, Perils, and (Timely) Heuristics for Mining Coding Agent Activity.
To appear in MSR 2026, ACM, Rio de Janeiro, Brazil.


Luís Soeiro, Thomas Robert, Stefano Zacchiroli.
Finding Software Supply Chain Attack Paths with Logical Attack Graphs.
FPS 2025, Brest, France. Springer, 2026.


Solal Rapaport, Laurent Pautet, Samuel Tardieu, Stefano Zacchiroli.
Altered Histories in Version Control System Repositories: Evidence from the Trenches.
ASE 2025, IEEE, Seoul, South Korea. Pages 2183–2194.

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

Software Heritage and Generative AI, first contacts

October 19, 2023

Software Heritage Statement on Large Language Models for Code

Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

Software Heritage and Generative AI, first contacts

October 19, 2023

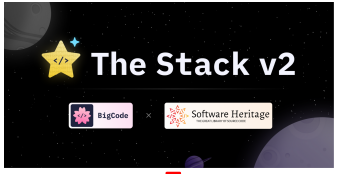
Software Heritage Statement on Large Language Models for Code

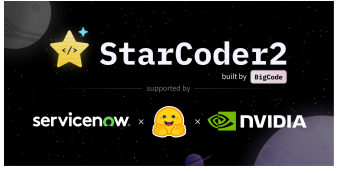
Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

February 2024

Yes, it's possible!





Software Heritage and Generative AI, first contacts

October 19, 2023

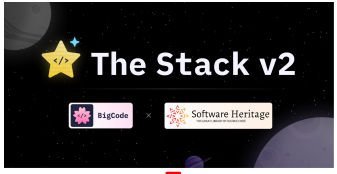
Software Heritage Statement on Large Language Models for Code

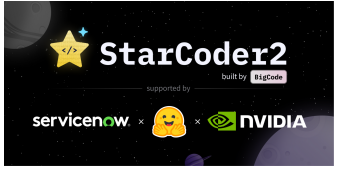
Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

February 2024

Yes, it's possible!





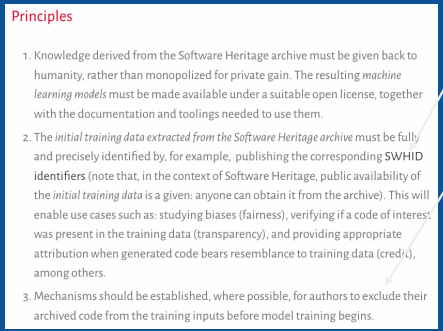
But it's hard…

Lessons learned

Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

Lessons learned



Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

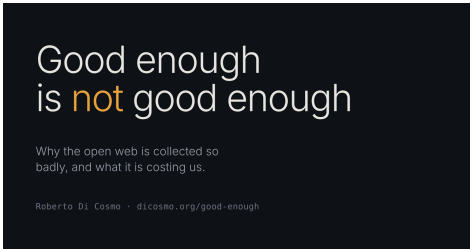
Transparency is easy: SWHID (undergoing ISO standardisation) and Software Heritage
N.B. : may be mandated by regulations!

Opt out is complex: who is *the real right owner*?
(similar issues to license compliance)

+

- **Building the training set** is complex: e.g. includes **license compliance** alike work **at massive scale**
- Generating **attribution information** on model output **is more complex** than license compliance

We need **a coordinated effort** to ensure fully open models will succeed!

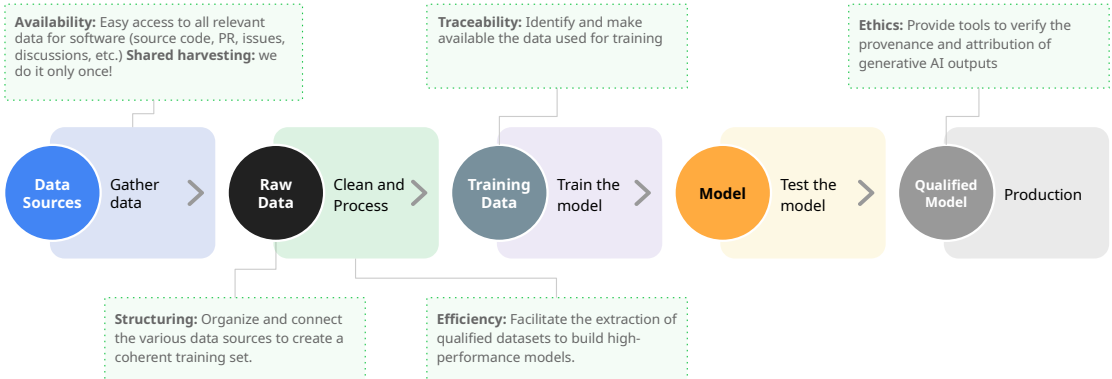


dicosmo.org/good-enough

- Massive multiple scraping — every collector re-fetches every site, over and over, for material that has not changed
- AI bots — Software Heritage itself was hit by waves of automated collection: services unresponsive, engineers blocking addresses instead of building
- Waste of resources — bills explode for everyone, and the open web closes down one emergency measure at a time

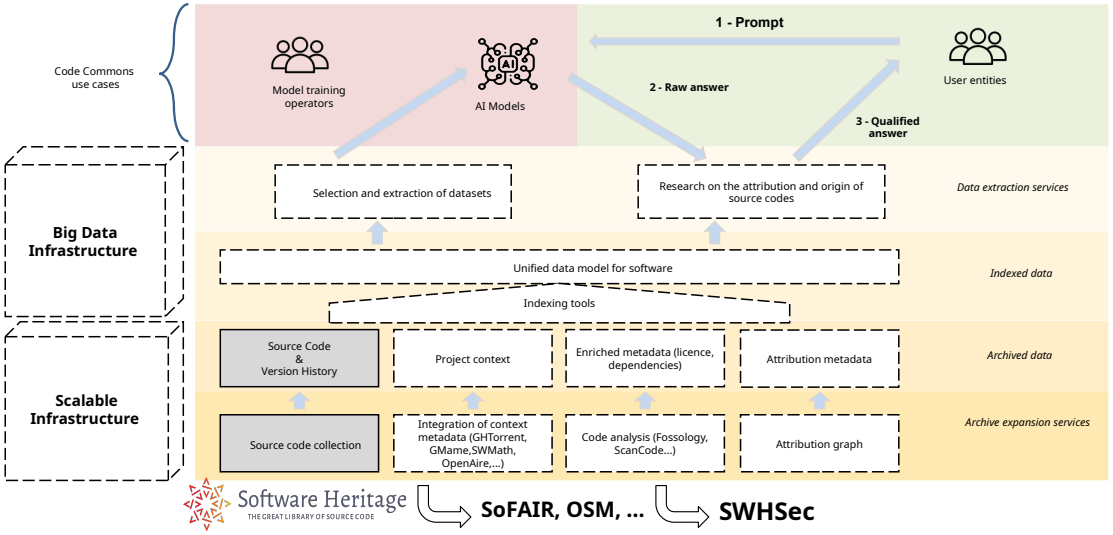
CodeCommons: collect once, share widely — *one logical collection, many copies*

A STEP FORWARD: CodeCommons



Solutions

CodeCommons: bird's eye view (technical focus)



CodeCommons

Open, responsible, and transparent AI: Our shared goal

CodeCommons is an ambitious project to create the world's most comprehensive digital commons for code

Building on the existing foundation of Software Heritage, the largest publicly available source code archive, CodeCommons aims to bring into one place all the **critical** and **qualified** information needed to create **smaller**, **better** datasets for the next generation of AI tools.

At its core, the project prioritizes transparency and traceability, enabling model builders and users to **respect creators' rights** while promoting **sovereign** and **sustainable** AI.

Learn more

Meet the teams



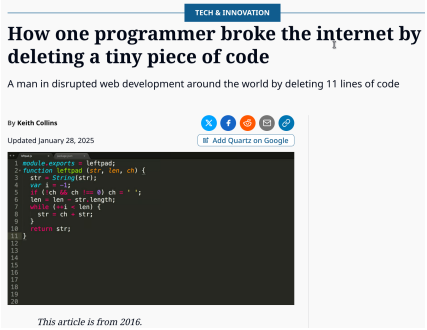


Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

Deep (infra)structural dependencies!

An early warning: left-pad (March 2016)



TECH & INNOVATION

How one programmer broke the internet by deleting a tiny piece of code

A man in disrupted web development around the world by deleting 11 lines of code

By Keith Collins

Updated January 28, 2025

Add Quartz on Google

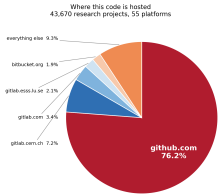
```
module.exports = leftpad;
function leftpad (str, len, ch) {
  str = String(str);
  var i = -1;
  if (!ch && ch !== 0) ch = ' ';
  len = len - str.length;
  while (++i < len) {
    str = ch + str;
  }
  return str;
}
```

This article is from 2016.

See the Wikipedia article:

- Facebook, PayPal, Netflix, Spotify affected
- web broken worldwide ~2.5 hours

A clear and present danger



Used by **European research infrastructure** devs *(Source: Software Heritage)*

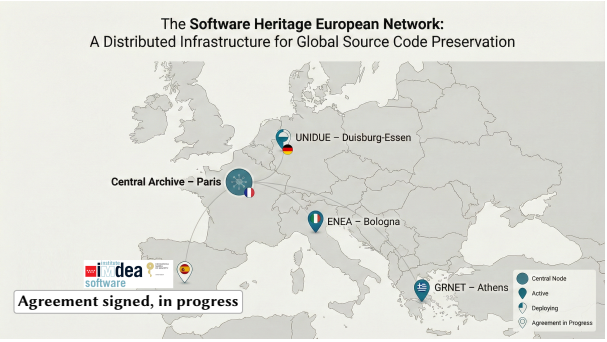
"*If GitHub or PyPI disappeared tomorrow,* research would not just slow down *— it would stop.*"

An ecumenical approach to sovereign fallback of build chains is possible

Add support for fallback in key package managers

npm, PyPI, Maven Central, Cargo, CPAN, RubyGems, Packagist, …

Realize ecumenical resilience



"When a registry fails — the world continues to build."

Four sovereign mirrors today.

Outline

1. Meet Software Heritage
2. From archive to revolutionary infrastructure
3. Software for research: reference archive ACAD_ADOPTION
4. Research on software: datasets from the archive DATASETS
5. Selected highlight: Impact on software studies ESE_STUDIES
6. A Very Large Telescope to explore the software development galaxy
7. Selected highlight: Improving Open Source Security with SWH(Sec) SECURITY
8. Selected highlight: AI and transparent LLMs CODECOMMONS
9. One last thing
10. Conclusion

A growing and active community

Core Team



Ambassadors



All together, 2026 Symposium



Software Heritage

A rally flag for a grand vision

Bring together academia, industry, governments, communities

"to build a reference, global infrastructure for open and better software"

Software Heritage is

- vendor neutral, open source
- a worldwide, long term initiative

Software Heritage enables

- archival, reference, integrity
- qualification, sharing and reuse

Call to action

- leverage for AI, Cyber, SwEng
- get involved research, code
- promote for AEC, Journals, etc.
- fund the long term mission

Annual report 2025



5 years in 5 minutes



Get these slides