Software Heritage

10 years of archiving, sharing, and studying source code at the ultimate scale

Stefano Zacchiroli

Software Heritage
Télécom Paris, Polytechnic Institute of Paris

23 Jan 2025
10 Ans du Séminaire Codes Sources, Sorbonne Université
Paris, France

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Outline

About the speaker

- Professor of Computer Science, Télécom Paris, Polytechnic Institute of Paris
- Free/Open Source Software activist (25+ years)
- Debian Developer & Former 3x Debian Project Leader
- Former Open Source Initiative (OSI) director
- Software Heritage co-founder & CSO

Previously, on Séminaire Codes Sources

Flashback to 16 Dec 2015, on the 1st year of this seminar, talk *"Large-scale source code archival, publishing, and indexing with Debsources"* by yours truly (slides):

Debsources in a nutshell

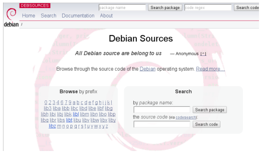
1. an infrastructure to publish Debian source code on the Web
2. a notable instance indexing *all* Debian source code to date: `http://sources.debian.net`

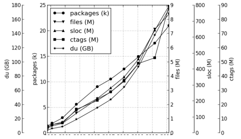
For developers:

- browse/search source code
- syntax highlighting
- pinpoint code lines, annotate

For data miners:

- Debian evolution over time
- 20+ years of FOSS history
- live change monitoring

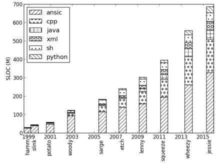

DEBSOURCES
debian
Home Search Documentation About
Search package
Search code
Debian Sources
All Debian source are belong to us — Anonymous
Browse through the source code of the Debian operating system. Read more...
Browse by prefix
Search
by package name:
Search package
the source code (via codesearch):
Search code

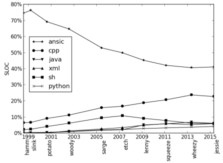


Stefano Zacchiroli (UPD / IRILL) Debsources Séminaire Codes Sources 8 / 45

Highlight #2: programming languages

top-5 most popular programming languages in Debian over time





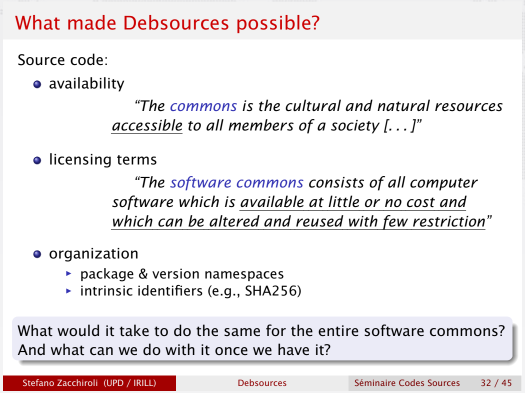
Recent trends (post-*etch*, 2007):

- C still leads, steady (absolute) growth
- C stops losing (relative) ground to C++
- decrease of Perl/Shell popularity
- Python rises (more maintainable glue code?)
- Lisp halves its popularity
- Java no longer under-represented

Stefano Zacchiroli (UPD / IRILL) Debsources Séminaire Codes Sources 28 / 45

Previously, on Séminaire Codes Sources (cont.)

Flashback to 16 Dec 2015, on the 1st year of this seminar, talk *"Large-scale source code archival, publishing, and indexing with Debsources"* by yours truly (slides):



What made Debsources possible?

Source code:

- availability

  "The commons is the cultural and natural resources accessible to all members of a society [. . .]"

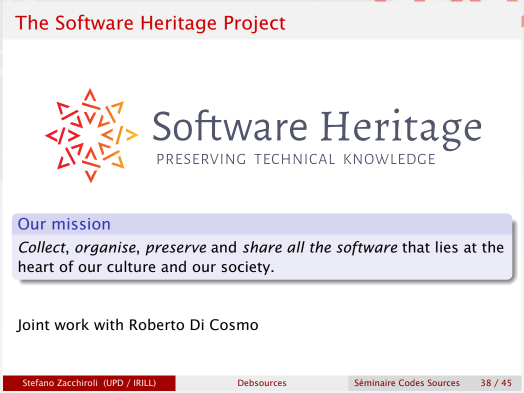
- licensing terms

  "The software commons consists of all computer software which is available at little or no cost and which can be altered and reused with few restriction"

- organization
  - package & version namespaces
  - intrinsic identifiers (e.g., SHA256)

What would it take to do the same for the entire software commons?
And what can we do with it once we have it?

Stefano Zacchiroli (UPD / IRILL) Debsources Séminaire Codes Sources 32 / 45



The Software Heritage Project

Software Heritage
PRESERVING TECHNICAL KNOWLEDGE

Our mission

Collect, *organise*, *preserve* and *share all the software* that lies at the heart of our culture and our society.

Joint work with Roberto Di Cosmo

Stefano Zacchiroli (UPD / IRILL) Debsources Séminaire Codes Sources 38 / 45

Outline

Software Heritage in a nutshell

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

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

Reference catalog

find and reference all software source code

Universal archive

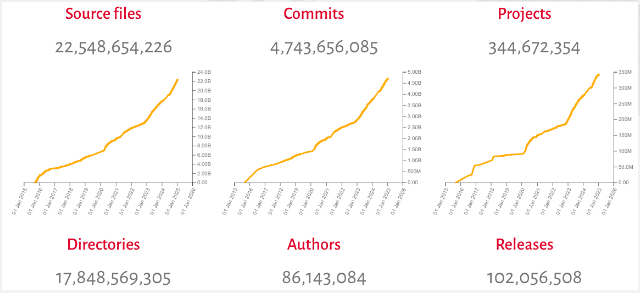
preserve and share all software source code

Research infrastructure

enable analysis of all software source code

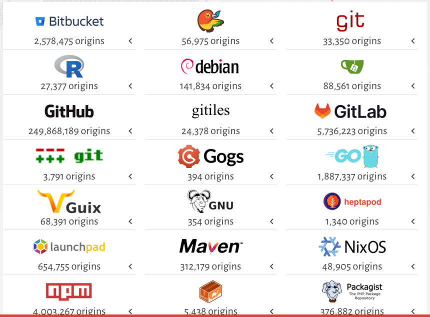
A *universal* software archive, as a shared infrastructure

- One infrastructure, open and shared
- The largest archive ever built



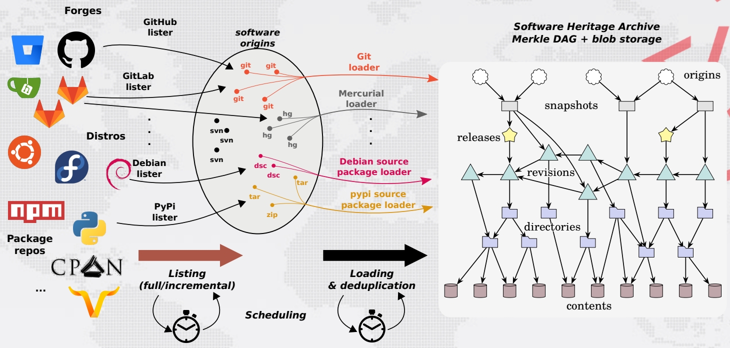
Source files	Commits	Projects
22,548,654,226	4,743,656,085	344,672,354

Directories	Authors	Releases
17,848,569,305	86,143,084	102,056,508



Bitbucket 2,578,475 origins	56,975 origins	git 33,350 origins
R 27,377 origins	debian 141,834 origins	88,561 origins
GitHub 249,868,189 origins	gitiles 24,378 origins	GitLab 5,736,223 origins
git 3,791 origins	Gogs 394 origins	GO 1,887,337 origins
Guix 68,391 origins	GNU 354 origins	heptapod 1,340 origins
launchpad 654,755 origins	Maven 312,179 origins	NixOS 48,905 origins
npm 4,003,267 origins	5,438 origins	Packagist 376,882 origins

The archive under the hood



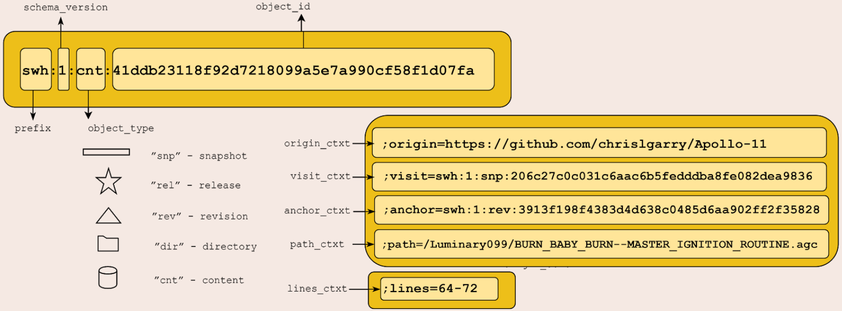
Global development history permanently archived in a uniform data model

- over 20 billion unique source files from over 300 million software projects
- ~2PB (compressed) blobs, ~50 B nodes, ~700 B edges

The Software Hash persistent identifier (SWHID)

Software Hash Identifiers (SWHID) see swhid.org

50+B intrinsic, decentralised, cryptographically strong identifiers, SWHIDs


schema_version
object_id
swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa
prefix
object_type
"snp" - snapshot
"rel" - release
"rev" - revision
"dir" - directory
"cnt" - content
origin_ctxt
visit_ctxt
anchor_ctxt
path_ctxt
lines_ctxt
;origin=https://github.com/chrislgarry/Apollo-11
;visit=swh:1:snp:206c27c0c031c6aac6b5fedddba8fe082dea9836
;anchor=swh:1:rev:3913f198f4383d4d638c0485d6aa902ff2f35828
;path=/Luminary099/BURN_BABY_BURN--MASTER_IGNITION_ROUTINE.agc
;lines=64-72

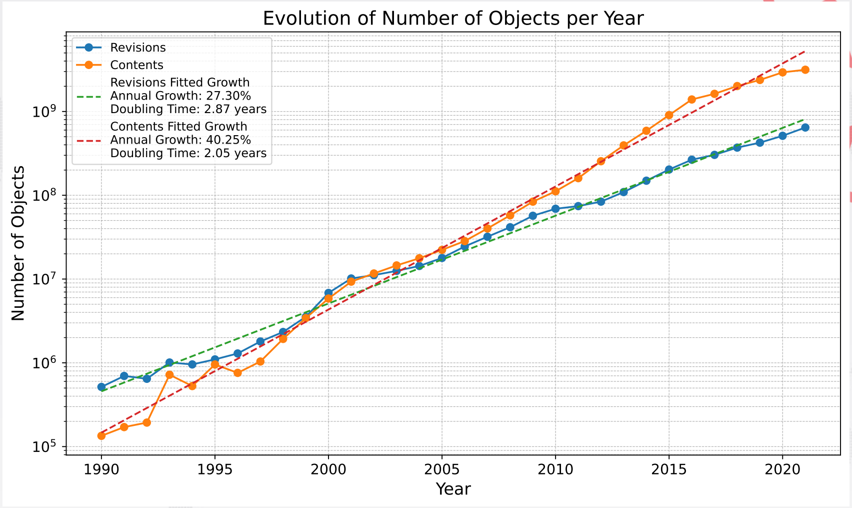
In SPDX 2.2; IANA "swh:"; WikiData P6138; ISO standardization ongoing DIS 18670

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

Examples: Apollo 11 AGC, Quake III rsqrt; Guidelines available: HOWTO and ICMS 2020

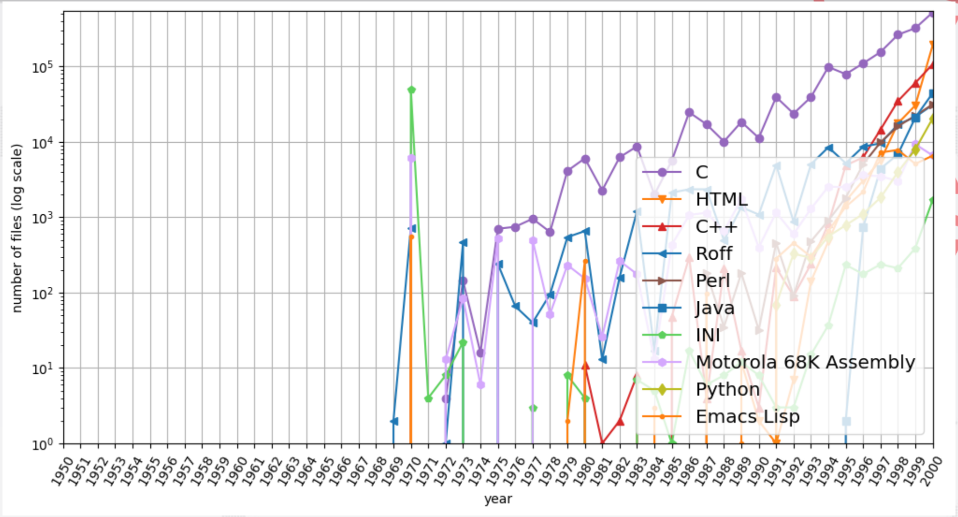
Outline

Public code growth



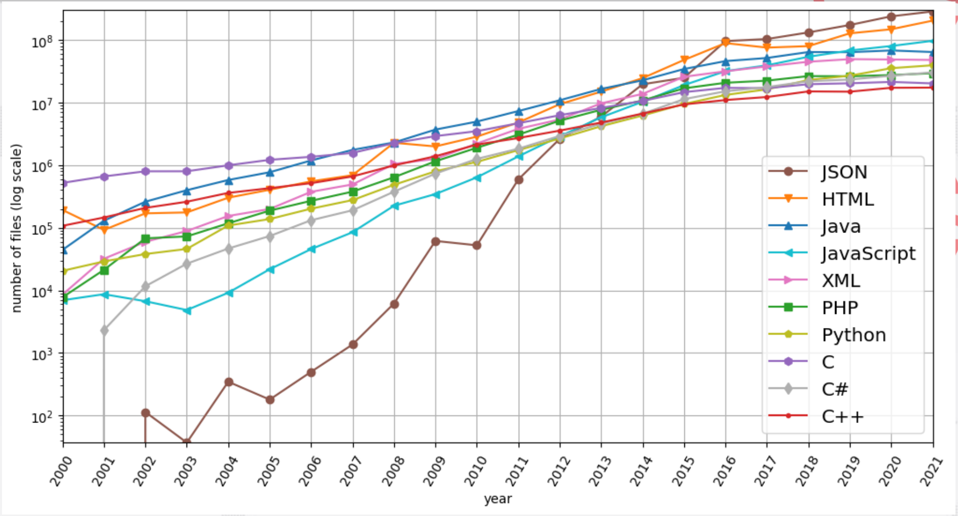
Rousseau, Di Cosmo, Zacchiroli. *Software provenance tracking at the scale of public source code.* Empir. Softw. Eng. 25(4): 2930-2959 (2020)

Programming language evolution (1950–2000)



Desmazières, Di Cosmo, Lorentz. *50 Years of Programming Language Evolution through the Software Heritage looking glass.* MSR 2025, to appear.

Programming language evolution (2000–2021)



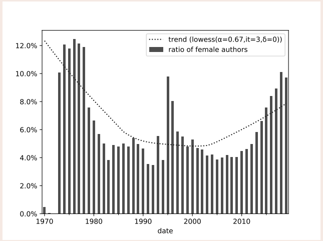
Desmazières, Di Cosmo, Lorentz. *50 Years of Programming Language Evolution through the Software Heritage looking glass.* MSR 2025, to appear.

Diversity, equity, and inclusion

Archived commit metadata contains public information that can be mined to study DEI traits of the global population of public code authors.

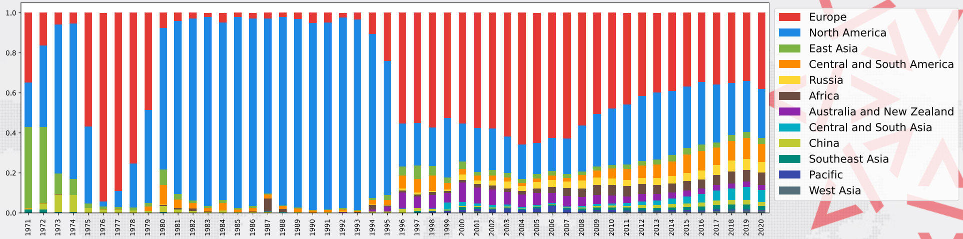
Gender gap — key findings

- 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

Diversity, equity, and inclusion (cont.)



Geographic gap — key findings

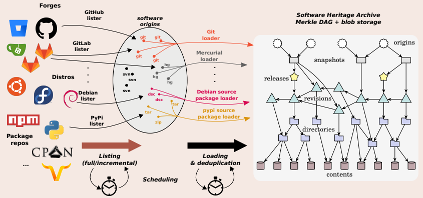
- Early decades of public code dominated by contributions from North America, followed by a period of alternating dominance between North America and Europe.
- Since then geographic diversity has increased constantly, with raising importance of contributions from Central and South America.
- Geo *and* 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

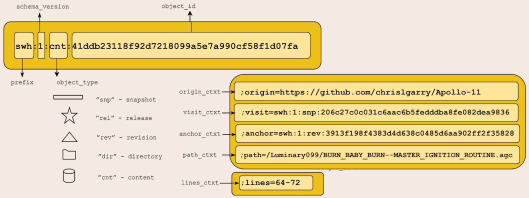
Outline

Addressing key needs for open science (ARDC)

Archive (20B+ files, 320M+ projects)



Reference (50 billion SWHIDs)



schema_version
object_id
swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa
prefix
object_type
"snp" - snapshot
"rel" - release
"rev" - revision
"dir" - directory
"cnt" - content
origin_ctxt
visit_ctxt
anchor_ctxt
path_ctxt
lines_ctxt
;origin=https://github.com/chrislgarry/Apollo-11
;visit=swh:1:snp:206c27c0c031c6aac6b5fedddba8fe082dea9836
;anchor=swh:1:rev:3913f198f4383d4d638c0485d6aa902ff2f35828
;path=/Luminary099/BURN_BABY_BURN--MASTER_IGNITION_ROUTINE.agc
;lines=64-72

Describe

- *Intrinsic metadata* from source code
- Contributed the Codemeta generator

Cite/Credit

- Contributed biblatex-software style
- Software Citation from the archive!

A few adoption indicators

Policy

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR, DE LA RECHERCHE ET DE L'INNOVATION

Second French Plan for Open Science

GENERALISING OPEN SCIENCE IN FRANCE 2021-2024

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR, DE LA RECHERCHE ET DE L'INNOVATION

French national strategy on research infrastructures

Edition 2021

European Commission

Scholarly Infrastructures for Research Software

Report from the EOSC Executive Board Working Group (WG) Architecture Task

SCHOLARLY ECOSYSTEM
Aggregators
Publishers
Scholarly repositories
INDUSTRY
PUBLIC ADMINISTRATION
CULTURAL HERITAGE
Universal Software Archive
Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

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

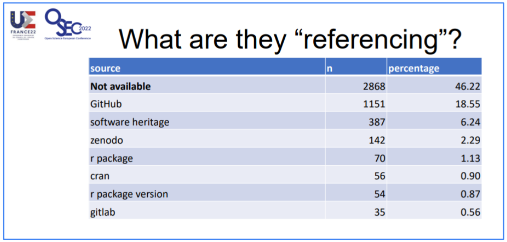
Projects

eosc | FAIRCORE4EOSC Core Components Supporting a FAIR EOSC

The CodeMeta Project

eosc | FAIR-IMPACT Expanding FAIR solutions across EOSC

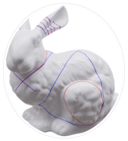
Users and collaborations


FRANCE22 OSEC 2022 Open Science European Conference

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

Outline

Tech preview: swh-scanner

Problem

- Detect unknown files in an open source project
- Detect proprietary files leaked on a public forge

Solution

`swh-scanner` : open source and open data source code scanner for compliance workflows, backed by the largest public archive of public source code.

Design

- Software Heritage archive as source of truth about public code
- Merkle DAG model and SWHIDs for maximum scanning efficiency
- Output: source tree partition into known/unknown + provenance information

Package: pypi.org/project/swh.scanner (GPL 3+)
Ref.: Serafini, Zacchiroli. *Efficient Prior Publication Identification for Open Source Code.* OpenSym 2022: 12:1-12:8

Outline

Looking for founding principles at Software Heritage

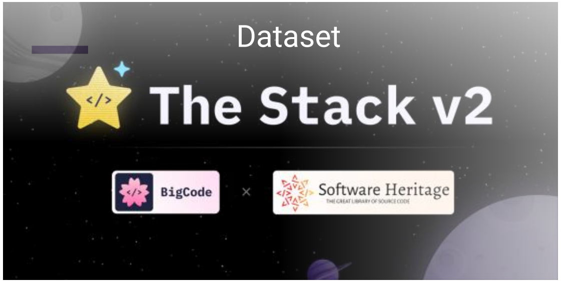
October 19, 2023

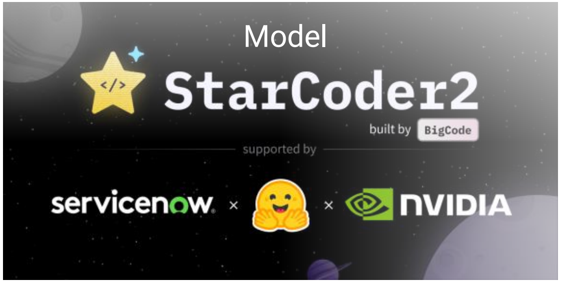
Software Heritage Statement on Large Language Models for Code

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.

Findings from BigCode: The Stack v2 and StarCoder2



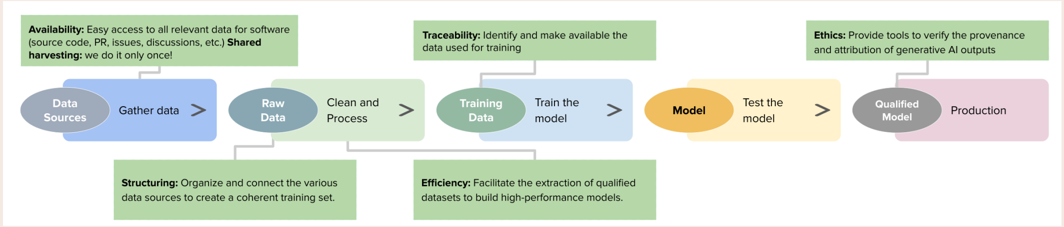


Released February 28th 2024

Yes one can build the best open LLM for code available while fully adhering to the Software Heritage principles for responsible LLMs, …
and even more: the full training pipeline is made public too!

Roadmap: CodeCommons project (BPI-funded, 2025-2027)

Addressing the current limitations in LLMs for code


Availability: Easy access to all relevant data for software (source code, PR, issues, discussions, etc.) **Shared harvesting:** we do it only once!
Traceability: Identify and make available the data used for training
Ethics: Provide tools to verify the provenance and attribution of generative AI outputs
Data Sources
Gather data
Raw Data
Clean and Process
Training Data
Train the model
Model
Test the model
Qualified Model
Production
Structuring: Organize and connect the various data sources to create a coherent training set.
Efficiency: Facilitate the extraction of qualified datasets to build high-performance models.

Vision: create the world reference for LLMs for code

- Massive, transparent and up to date, with qualified information
- Traceability of the contents, code attribution

Outline

Come in, we're open! https://softwareheritage.org

Software Heritage is

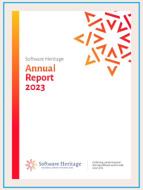
- The largest public archive of software source code and its history.
- A revolutionary infrastructure for industry, research, culture, society.

Software Heritage enables

- Software archival, reference, integrity, traceability, global knowledge base.
- "Big code" research on the entire software commons.

Join us

Annual report 2023



Outline

Graph dataset

Use case: large scale analyses of the most comprehensive corpus on the development history of free/open source software.

Antoine Pietri, Diomidis Spinellis, Stefano Zacchiroli
The Software Heritage Graph Dataset: Public software development under one roof
MSR 2019: 16th Intl. Conf. on Mining Software Repositories. IEEE
preprint: `http://deb.li/swhmsr19`

Dataset

- Relational representation of the full graph as a set of tables
- Available as open data: docs.softwareheritage.org/devel/swh-dataset/graph
- Chosen as subject for the MSR 2020 Mining Challenge

Formats

- Local use: set of Apache ORC files (10+ TiB in total)
- Live usage: Amazon Athena (SQL-queriable), Azure Data Lake

Graph dataset — example

```
SELECT COUNT(*) AS c, word FROM (
  SELECT LOWER(REGEXP_EXTRACT(FROM_UTF8(
  message), '^\w+')) AS word FROM revision)
WHERE word != ''
GROUP BY word ORDER BY COUNT(*) DESC LIMIT 5;
```

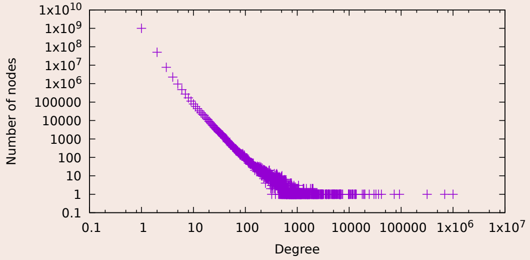
Count	Word
71 338 310	update
64 980 346	merge
56 854 372	add
44 971 954	added
33 222 056	fix

Graph dataset — example

Fork arity

i.e., how often is a commit based upon?

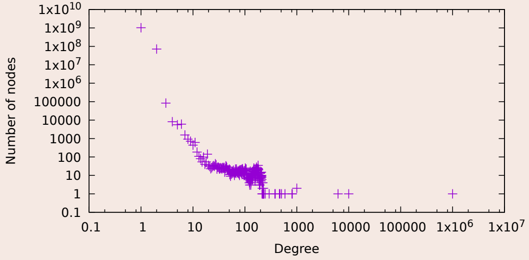
```
SELECT fork_deg, count(*) FROM (
  SELECT id, count(*) AS fork_deg
  FROM revision_history GROUP BY id) t
GROUP BY fork_deg ORDER BY fork_deg;
```



Merge arity

i.e., how large are merges?

```
SELECT merge_deg, COUNT(*) FROM (
  SELECT parent_id, COUNT(*) AS merge_deg
  FROM revision_history GROUP BY parent_id) t
GROUP BY merge_deg ORDER BY merge_deg;
```



License dataset

Stefano Zacchiroli
A Large-scale Dataset of (Open Source) License Text Variants
MSR 2022 (best dataset paper) + Empir. Soft. Eng. 28(6): 147 (2023)
preprint: https://arxiv.org/abs/2308.11258

Dataset

- 6.9 million unique full texts of FOSS license variants
- Detected using filename patterns across the entire SWH archive
  - LICENSE, COPYRIGHT, NOTICE, etc.
- Metadata: file lengths measures, detected MIME type, detected SPDX license (via ScanCode), example origin repository, oldest public commit of origin, ground truth

Use cases

- Empirical studies on FOSS licensing, including phylogenetics
- Training of automated license classifiers
- NLP analyses of legal texts

The Software Heritage Filesystem (SwhFS)

The Software Heritage Filesystem (SwhFS) is a user-space POSIX filesystem that enables browsing parts of the Software Heritage archive as if it were locally available.

- Code: forge.softwareheritage.org/source/swh-fuse
- Documentation: docs.softwareheritage.org/devel/swh-fuse

Thibault Allançon, Antoine Pietri, Stefano Zacchiroli
The Software Heritage Filesystem (SwhFS): Integrating Source Code Archival with Development
ICSE 2021 (Tool track): The 43rd Intl. Conference on Software Engineering
https://arxiv.org/abs/2102.06390

The Software Heritage Filesystem (SwhFS) — example

```
$ mkdir swhfs
$ swh fs mount swhfs/  # mount the archive
$ cd swhfs/

$ cat archive/swh:1:cnt:c839dea9e8e6f0528b468214348fee8669b305b2
#include <stdio.h>

int main(void) {
    printf("Hello, World!\n");
}

$ cd archive/swh:1:dir:1fee702c7e6d14395bbf5ac3598e73bcbf97b030
$ ls | wc -l
127
$ grep -i antenna THE_LUNAR_LANDING.s | cut -f 5
# IS THE LR ANTENNA IN POSITION 1 YET
# BRANCH IF ANTENNA ALREADY IN POSITION 1
```

The Software Heritage Filesystem (SwhFS) — example (cont.)

```
$ cd archive/swh:1:rev:9d76c0b163675505d1a901e5fe5249a2c55609bc

$ ls -F
history/  meta.json@  parent@  parents/  root@

$ jq '.author.name, .date, .message' meta.json
"Michal Golebiowski-Owczarek"
"2020-03-02T23:02:42+01:00"
"Data:Event:Manipulation: Prevent collisions with Object.prototype ..."

$ find root/src/ -type f -name '*.js' | xargs cat | wc -l
10136
```

Graph compression

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

Research question

Is it possible to efficiently perform software development history analyses at the scale of Software Heritage archive on a single, relatively cheap machine?

Idea

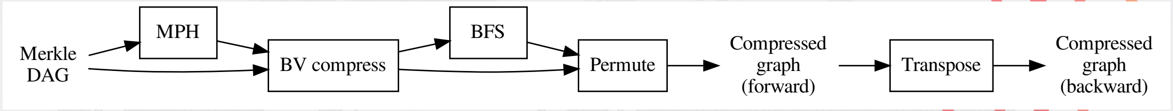
Apply state-of-the-art graph compression techniques from the field of Web graph / social network analysis.

Results

The entire archive graph (35 B nodes, 500 B edges) can be loaded in 300 GiB and then traversed at the cost of tens of ns per edge (= a few hours for a full single-thread visit).

Java and gRPC APIs available: docs.softwareheritage.org/devel/swh-graph/grpc-api.html

Graph compression pipeline



- MPH: minimal perfect hash, mapping Merkle IDs to 0..N-1 integers
- BV compress: Boldi-Vigna compression (based on MPH order)
- BFS: breadth-first visit to renumber
- Permute: update BV compression according to BFS order

(Re)establishing locality

- Key for good compression is a node ordering that ensures locality and similarity
- Which is very much *not* the case with Merkle IDs, …but is the case *again* after BFS reordering

Is my FOSS code (known to be) vulnerable?

"The Common Vulnerabilities and Exposures (CVE) system provides a reference-method for publicly known information-security vulnerabilities and exposures."

- e.g., CVE-2014-0160 (AKA: Heartbleed), CVE-2021-44228 (AKA: Log4Shell)

Tooling

- A number of state-of-the-art tools in FOSS compliance can:
  1. Scan your local code base,
  2. Compare it with a knowledge base that knows about CVEs,
  3. Emit warnings like: "you include/depend on code affected by the following CVEs: . . . ".
- CVE $\leftrightarrow$ code matching heuristics are based for the most part on package metadata (in particular: version numbers)

Example — osv.dev

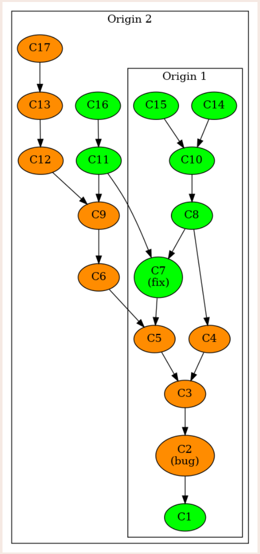
- OSV.dev: *"A distributed vulnerability database for Open Source. An open, precise, and distributed approach to producing and consuming vulnerability information for open source."*
- Service (operated by Google) and data format (standardized by OpenSSF) that:
  - Crawls vulnerability information from many places (GitHub, distros, package manager repos, ...).
  - Provides efficient APIs to query the information and integrate it into compliance toolchains.

Is my code affected by a known CVE?

By commit hash:

```
$ curl -X POST -d \
  '{"commit": "6879efc2c1596d11a6a6ad296f80063b558d5e0f"}' \
  "https://api.osv.dev/v1/query"
{"vulns":[{"id":"OSV-2020-484","summary":"Heap-buffer-overflow in AAT...
```

Reasoning on the global commit graph



- State-of-the-art tech (e.g., OSV.dev by Google) to query whether a given commit is known to be vulnerable do not know about *all* code out there.
- They crawl project repos related to known vulnerabilities, but not their *forks* (across multiple forges!).

Real-world example (one out of many we have identified):

- Linux kernel vulnerability: GSD-2022-1004193
- Example of vulnerable commit: b13baccc3850ca8b8cccbf8ed9912dbaa0fdf7f3
- Fix commit: a92d44b412e75dd66543843165e46637457f22cc
- False negative commit in fork (Raspberry PI Linux): c7c7a1a18af4c3bb7749d33e3df3acdf0a95bbb5

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 (~5 billion commits) with vulnerability information.