From Software Heritage to Code Commons

A vision for transparent and responsible AI in code-based model training

Roberto Di Cosmo
Director, Software Heritage
Inria and Université Paris Cité

April 2025

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Outline

Short Bio: Roberto Di Cosmo

Computer Science professor in Paris, now working at INRIA

- *35+ years* of research (Theor. CS, Programming, Software Engineering, Erdos #: 3)
- *25+ years* of Free and Open Source Software
- *15+ years* building and directing structures for the common good

1999 *DemoLinux* – first live GNU/Linux distro

2007 *Free Software Thematic Group*
150 members 40 projects 200Me

2008 *Mancoosi project* www.mancoosi.org

2010 *IRILL* www.irill.org

2015 *Software Heritage* at INRIA

2018 *National Committee for Open Science*, France

2021 *EOSC Task Force on Infrastructures for Software*, European Union

Software *Source Code* is Precious Knowledge

Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

"Programs must be written for people to read, and only incidentally for machines to execute."

Apollo 11 source code (excerpt)

```
P63SPOT3        CA      BIT6            # IS THE LR ANTENNA IN POSITION 1 YET
                EXTEND
                RAND    CHAN33
                EXTEND
                BZF     P63SPOT4        # BRANCH IF ANTENNA ALREADY IN POSITION 1

                CAF     CODE500         # ASTRONAUT:    PLEASE CRANK THE
                TC      BANKCALL        #               SILLY THING AROUND
                CADR    GOPERF1
                TCF     GOTOPOOH        # TERMINATE
                TCF     P63SPOT3        # PROCEED       SEE IF HE'S LYING

P63SPOT4        TC      BANKCALL        # ENTER         INITIALIZE LANDING RADAR
                CADR    SETPOS1

                TC      POSTJUMP        # OFF TO SEE THE WIZARD ...
                CADR    BURNBABY
```

Quake III source code (excerpt)

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y = number;
    i = * ( long * ) &y; // evil floating point bit level hacking
    i = 0x5f3759df - ( i >> 1 ); // what the fuck?
    y = * ( float * ) &i;
    y = y * ( threehalfs - ( x2 * y * y ) ); // 1st iteration
// y = y * ( threehalfs - ( x2 * y * y ) ); // 2nd iteration, this
can be removed

    return y;
}
```

Len Shustek, Computer History Museum 2006

"Source code provides a view into the mind of the designer."

Unveiled in 2016

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

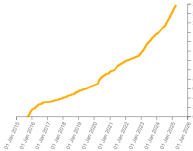
A *universal* software archive, as a shared infrastructure

One infrastructure
open and shared

The largest archive ever built

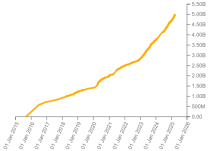
Source files

23,828,196,855



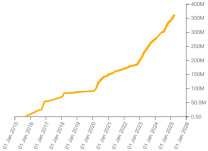
Commits

5,006,891,310



Projects

362,987,832



Directories

18,809,563,175

Authors

90,221,599

Releases

106,890,095

Diamond sponsors

Platinum sponsors

Gold sponsors

Silver sponsors

Bronze sponsors

An international, non profit initiative built for the long term

Sharing the vision

United Nations Educational, Scientific and Cultural Organization

Donors, members, sponsors

INVENTEURS DU MONDE NUMÉRIQUE

Diamond sponsors

Platinum sponsors

Gold sponsors

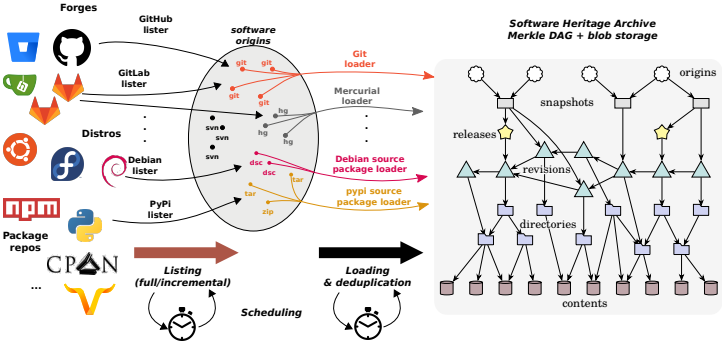
Silver sponsors

Bronze sponsors

And many more ...

www.softwareheritage.org/support/testimonials

The archive under the hood



Global development history permanently archived in a uniform data model

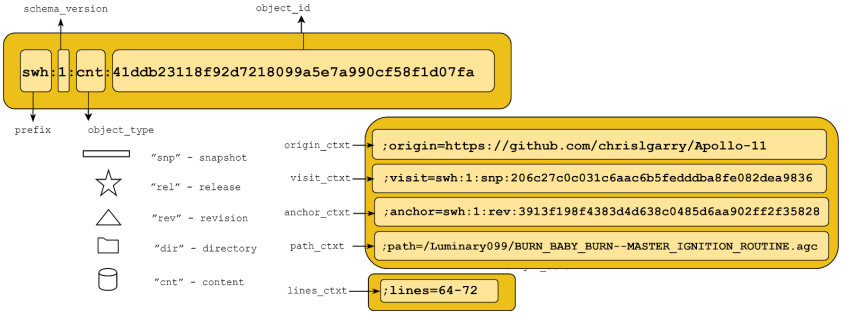
- over 23 billion unique source files from over 360 million software projects
- ~2PB (compressed) blobs, ~50 B nodes, ~800 B edges

The Software Hash persistent identifier (SWHID)

Software Hash Identifiers (SWHID)

see swhid.org

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



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

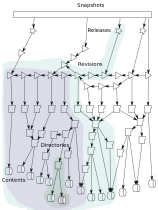
Full fledged *source code references* for traceability, integrity and reproducibility

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

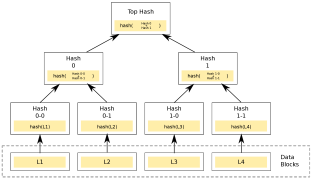
A revolutionary infrastructure

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

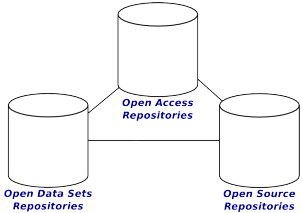
Software Graph



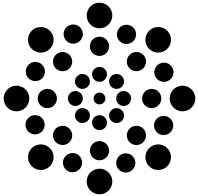
Software Blockchain



Open Science pillar



Big Code



One infrastructure, *shared*: more efficient, less waste …
… addressing a broad spectrum of needs!

Outline

A walkthrough

- Browse + Reference [DIS 18670] (Apollo 11 [excerpt], your work may be already there !)
- Trigger archival, use the updateswh browser extension, configure the webhooks
- Cite with biblatex-software (CTAN, Overleaf ACMART template)
- Describe with Codemeta (use codemeta generator)
- Curated deposit in SWH via HAL, see for example: LinBox, SLALOM, Givaro, NS2DDV, SumGra, Coq proof, ...
- Extracting all the software products for Inria, for CNRS, for CNES, for LIRMM or for Rémi Gribonval using HalTools
- Example with Parmap: devel on Github, archive in SWH, curated deposit in HAL
- Example research articles:
  - compare Fig. 1 and conclusions in the 2012 version and the updated version
  - SWHID in a replication experiment

The full graph in the AWS Open Data collection

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

Registry of Open Data on 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;
```

Total cost: approximately .5 euros

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 (50 B nodes, 700 B edges) in 300 GiB RAM

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

Java, gRPC and Rust APIs available

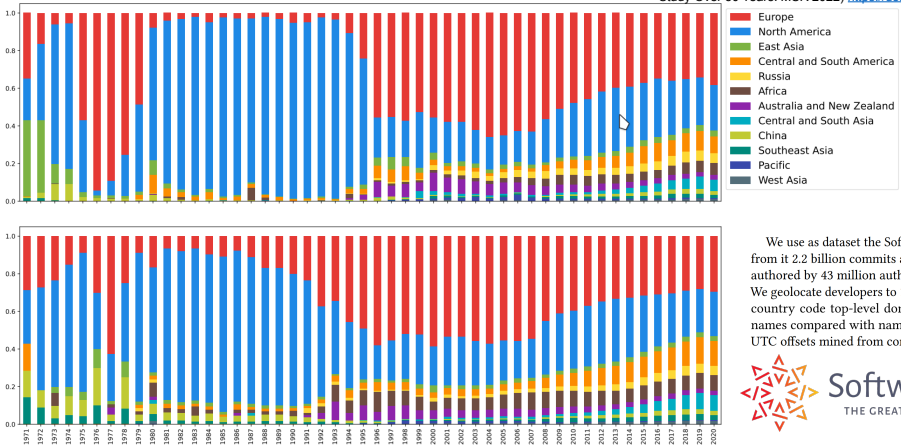
docs.softwareheritage.org/devel/swh-graph/grpc-api.html

Outline

Because software is naturally international !

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

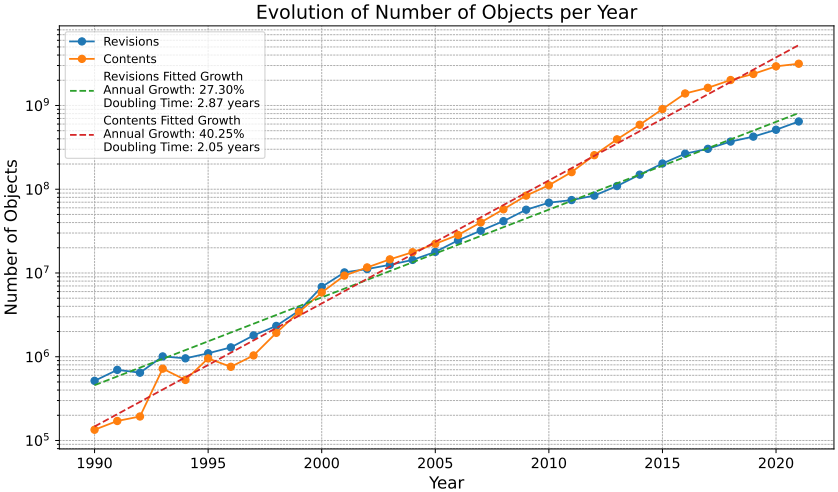


We use as dataset the Software Heritage archive [3] and analyze from it 2.2 billion commits archived from 160 million projects and authored by 43 million authors during the 1971–2021 time period. We geolocate developers to 12 world regions, using as signals email country code top-level domains (ccTLDs)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

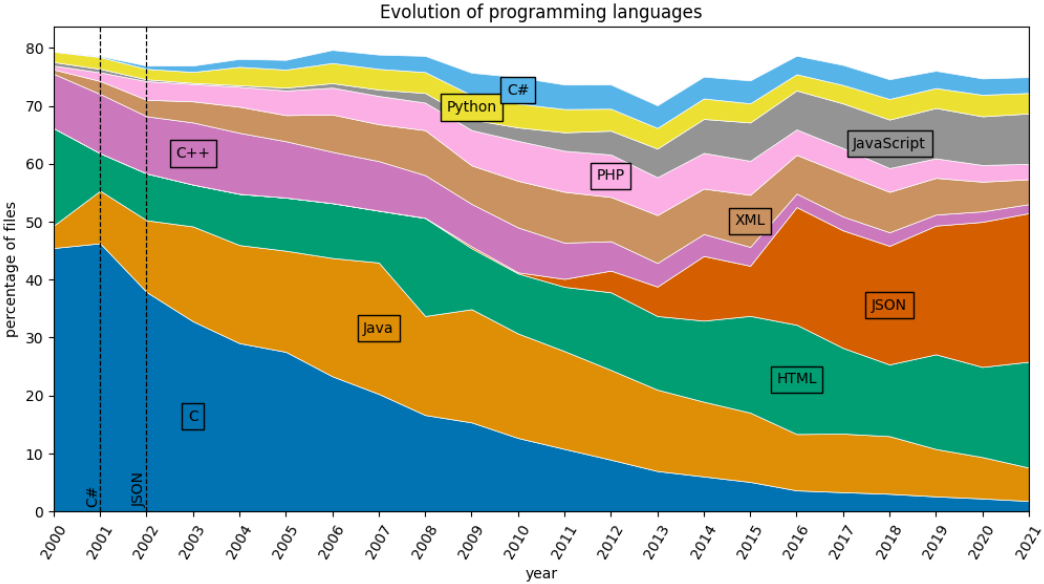
Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

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

30 years of growth of public source code



Programming language evolution over 50 years



Analysis of UChile contributions to Open Source

122271

Contributions

6379

Software Projects

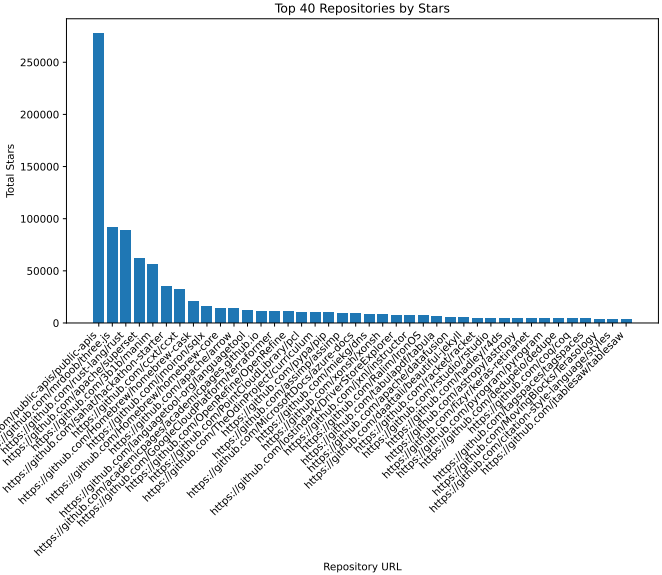
1420

Contributors

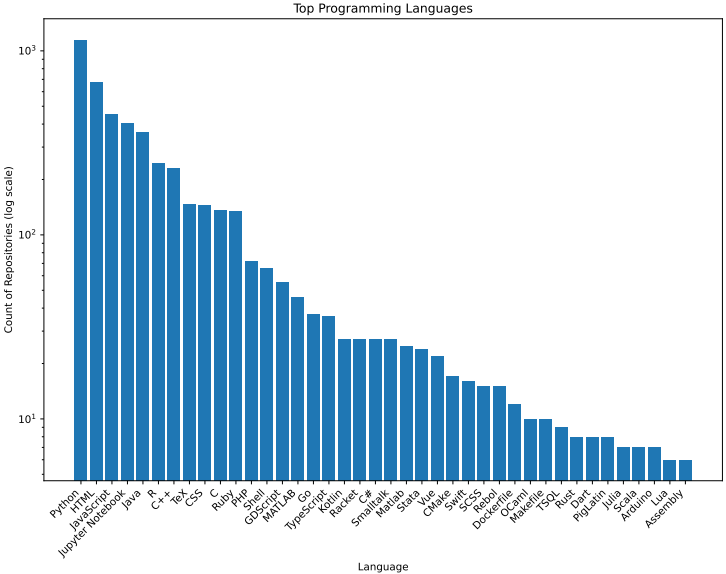
2002-08-24
2024-11-20

Time Span

Analysis of UChile contributions to Open Source, cont’d



Analysis of UChile contributions to Open Source, cont'd



Outline

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

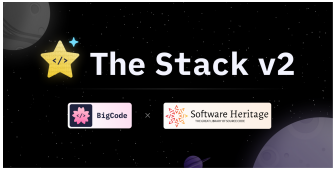
Software Heritage Statement on Large Language Models for Code

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

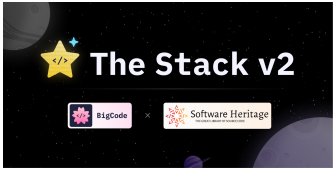
Software Heritage Statement on Large Language Models for Code

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

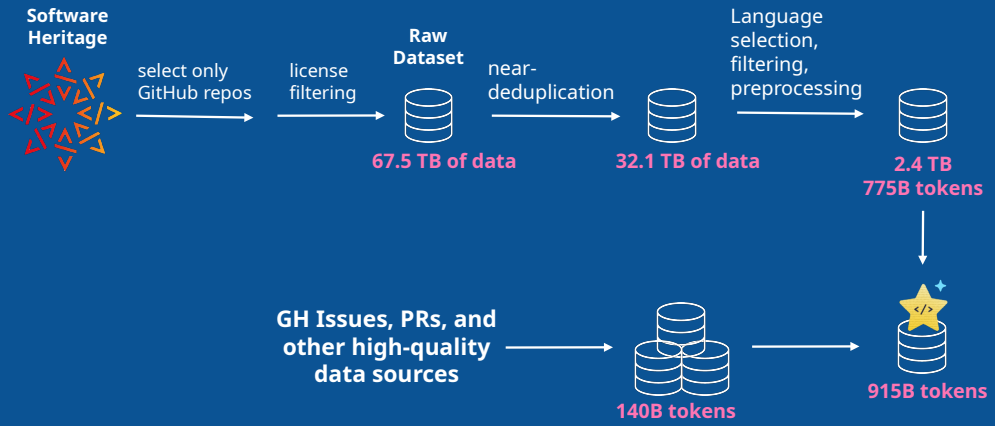
Yes, it's possible!



But it's hard…

The Stack v2

Data collection pipeline fully open and transparent built by BigCode



This slide is courtesy of Leandro Von Werra and Harm de Vries

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

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

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)

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)

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**

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

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!

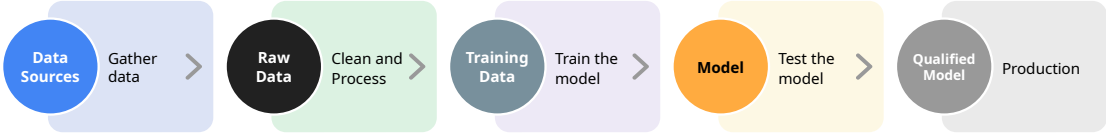
Opt out is complex: who is *the real right owner*? (similar issues to license compliance)

+

- **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!

GENERATIVE AI FOR CODE : OPEN ISSUES

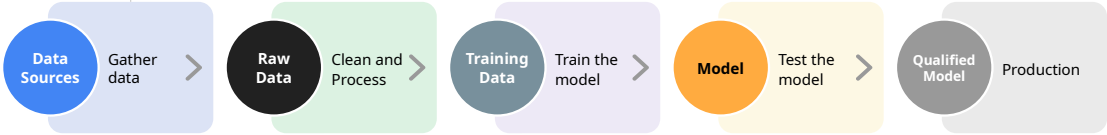


Issues

GENERATIVE AI FOR CODE : OPEN ISSUES

Gousios et al. GHTorrent
: GitHub's data from a firehose, MSR 2012

Collect source code, issues, PR, discussions, etc. **is very expensive**. **Redoing** it over and over again **is an anti-ecological waste**.

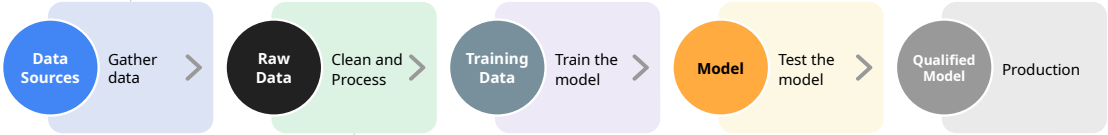


Issues

GENERATIVE AI FOR CODE : OPEN ISSUES

Gousios et al. GHTorrent
: GitHub's data from a firehose, MSR 2012

Collect source code, issues, PR, discussions, etc. **is very expensive**. **Redoing** it over and over again **is an anti-ecological waste**.



Building a quality training set is a **very complex task,** redoing it over and over again behind closed doors is a waste of energy and human resources

Gunasekar et al. « Textbooks Are All You Need » 2023
https://arxiv.org/abs/2306.11644

Issues

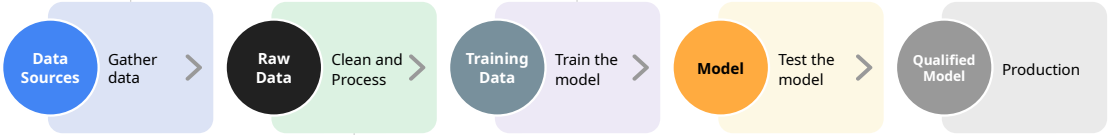
GENERATIVE AI FOR CODE : OPEN ISSUES

Gousios et al. GHTorrent
: GitHub's data from a firehose, MSR 2012

Collect source code, issues, PR, discussions, etc. **is very expensive**. **Redoing** it over and over again **is an anti-ecological waste**.

Lefeuvre et al. Fingerprinting and Building Large Reproducible Datasets REP'23

No precise identification and **lack of availability** of training data are huge obstacles to **transparency** and **reproducibility**.



Building a quality training set is a **very complex task,** redoing it over and over again behind closed doors is a waste of energy and human resources

Gunasekar et al. « Textbooks Are All You Need » 2023
https://arxiv.org/abs/2306.11644

Issues

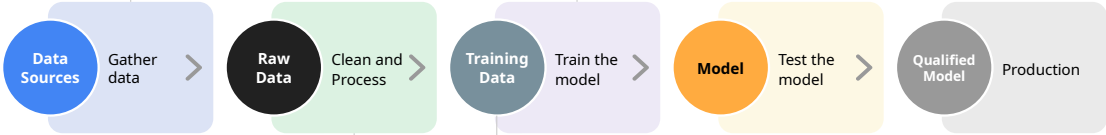
GENERATIVE AI FOR CODE : OPEN ISSUES

Gousios et al. GHTorrent
:GitHub's data from a firehose, MSR 2012

Collect source code, issues, PR, discussions, etc. **is very expensive**. **Redoing** it over and over again **is an anti-ecological waste**.

Lefeuvre et al. Fingerprinting and Building Large Reproducible Datasets REP'23

No precise identification and **lack of availability** of training data are huge obstacles to **transparency** and **reproducibility**.



Building a quality training set is a **very complex task,** redoing it over and over again behind closed doors is a waste of energy and human resources

Gunasekar et al. « Textbooks Are All You Need » 2023
https://arxiv.org/abs/2306.11644

Extracting qualified subsets for training is **difficult** and time consuming.

Ledivarec et al.
HyperDiff: Computing Source Code Diffs at Scale
ASE 2023

Issues

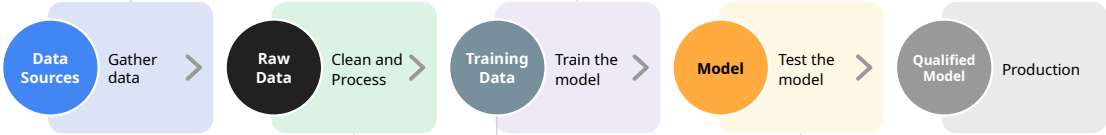
GENERATIVE AI FOR CODE : OPEN ISSUES

Gousios et al. GHTorrent
:GitHub's data from a firehose, MSR 2012

Collect source code, issues, PR, discussions, etc. **is very expensive**. **Redoing** it over and over again **is an anti-ecological waste**.

Lefeuvre et al. Fingerprinting and Building Large Reproducible Datasets REP'23

No precise identification and **lack of availability** of training data are huge obstacles to **transparency** and **reproducibility**.



Building a quality training set is a **very complex task,** redoing it over and over again behind closed doors is a waste of energy and human resources

Gunasekar et al. « Textbooks Are All You Need » 2023
https://arxiv.org/abs/2306.11644

Extracting qualified subsets for training is **difficult** and time consuming.

Ledivarec et al.
HyperDiff: Computing Source Code Diffs at Scale
ASE 2023

Extracting **quality subsets** should allow to **specialize** LLMs to perform **quality programming and software engineering tasks**.

Fan et al. Large language models for software engineering: Survey and open problems
FoSE 2023

Issues

GENERATIVE AI FOR CODE : OPEN ISSUES

Gousios et al. GHTorrent
:GitHub's data from a firehose, MSR 2012

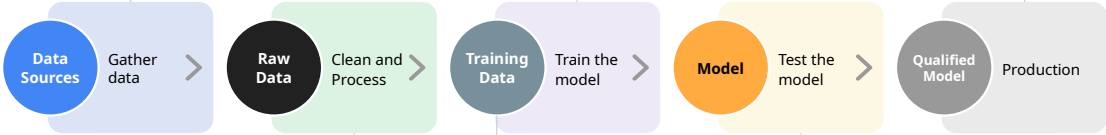
Collect source code, issues, PR, discussions, etc. **is very expensive**. **Redoing** it over and over again **is an anti-ecological waste**.

Lefeuvre et al. Fingerprinting and Building Large Reproducible Datasets REP'23

No precise identification and **lack of availability** of training data are huge obstacles to **transparency** and **reproducibility**.

Sallam et al.
ChatGPT utility in healthcare education, research, and practice: systematic review on the promising perspectives and valid concerns
2023

Lack of **traceability** of generative AI outputs make it **irrespective of authors**



Building a quality training set is a **very complex task,** redoing it over and over again behind closed doors is a waste of energy and human resources

Gunasekar et al. « Textbooks Are All You Need »
2023
https://arxiv.org/abs/2306.11644

Extracting qualified subsets for training is **difficult** and time consuming.

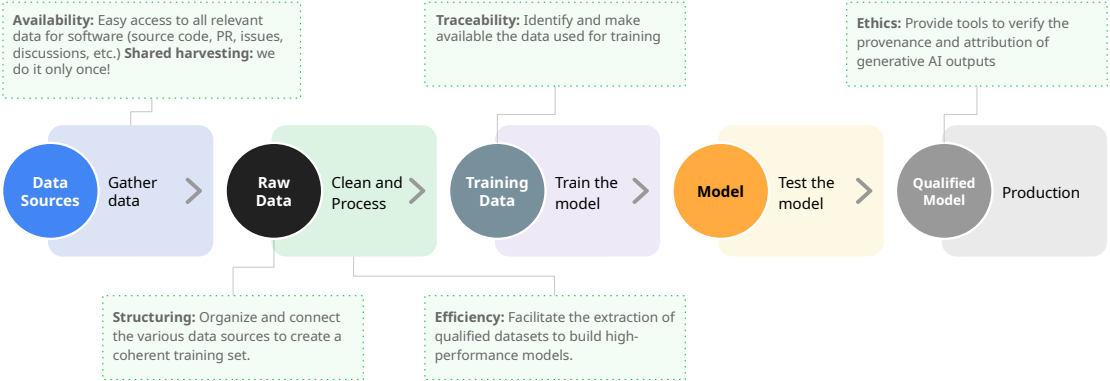
Ledivarec et al.
HyperDiff: Computing Source Code Diffs at Scale
ASE 2023

Extracting **quality subsets** should allow to **specialize** LLMs to perform **quality programming and software engineering tasks**.

Fan et al. Large language models for software engineering: Survey and open problems
FoSE 2023

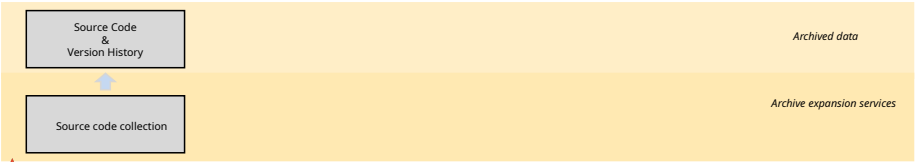
Issues

A STEP FORWARD: CodeCommons



Solutions

CodeCommons: bird’s eye view (technical focus)

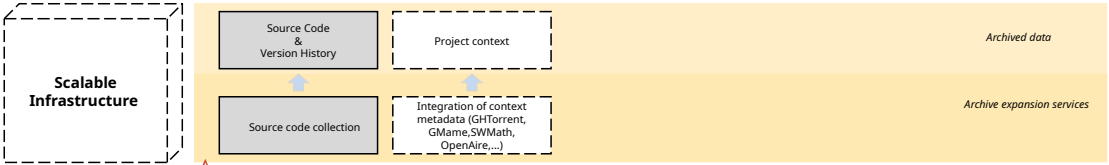


Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

CodeCommons: bird’s eye view (technical focus)

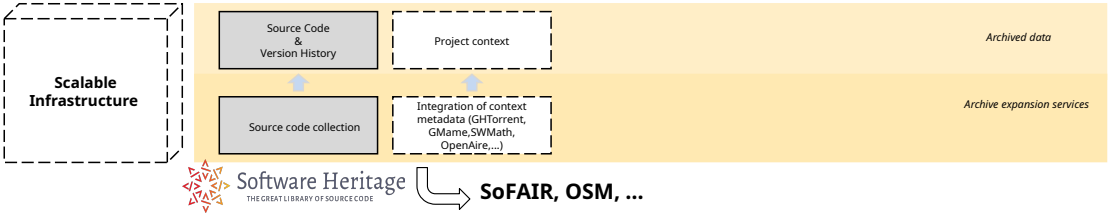
Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

CodeCommons: bird’s eye view (technical focus)

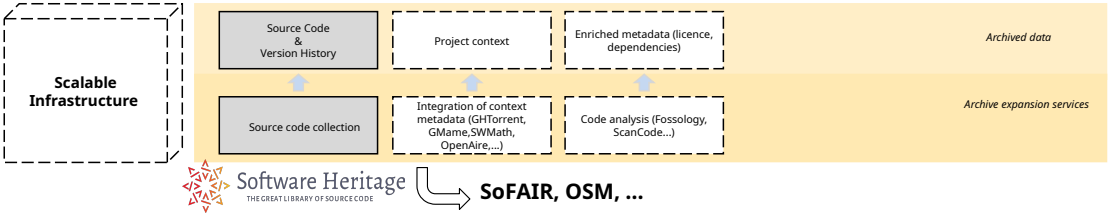


Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

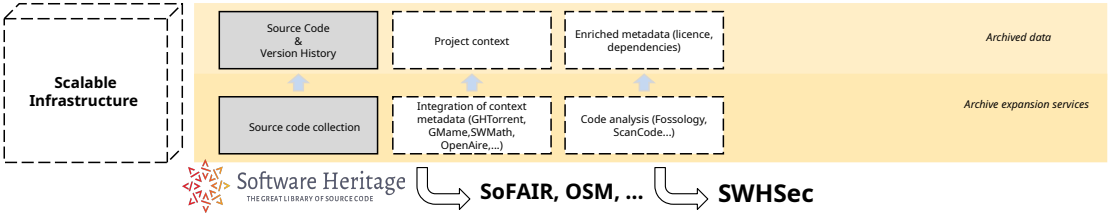
CodeCommons: bird’s eye view (technical focus)



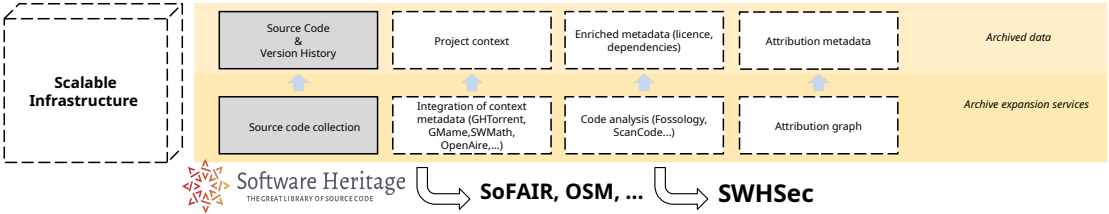
CodeCommons: bird’s eye view (technical focus)



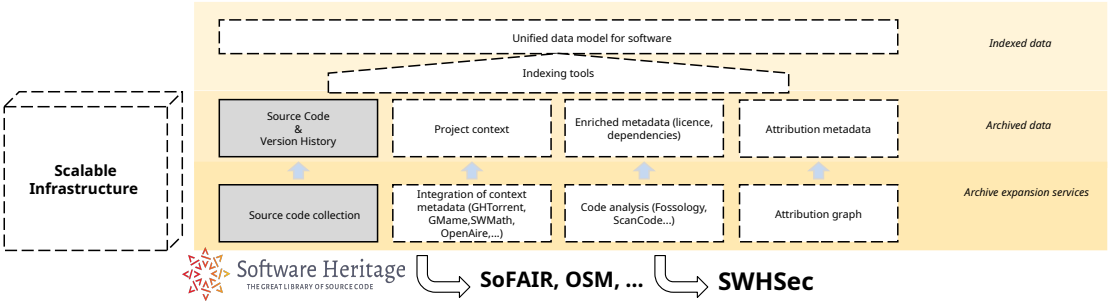
CodeCommons: bird’s eye view (technical focus)



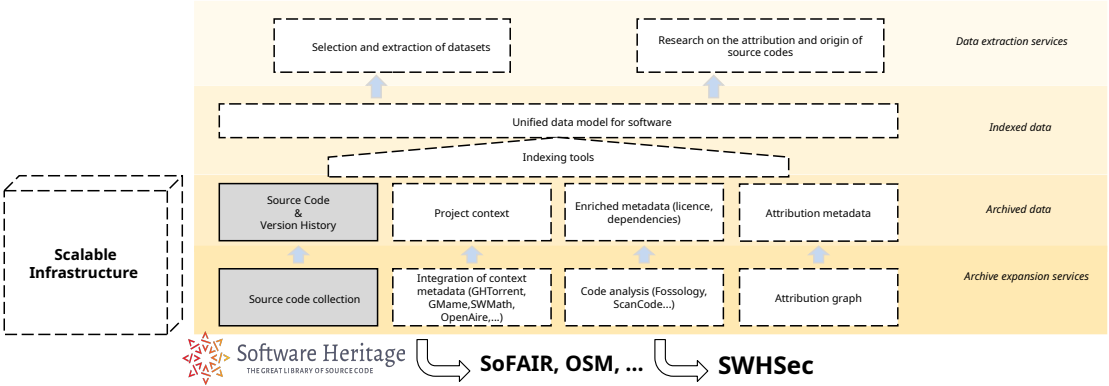
CodeCommons: bird’s eye view (technical focus)



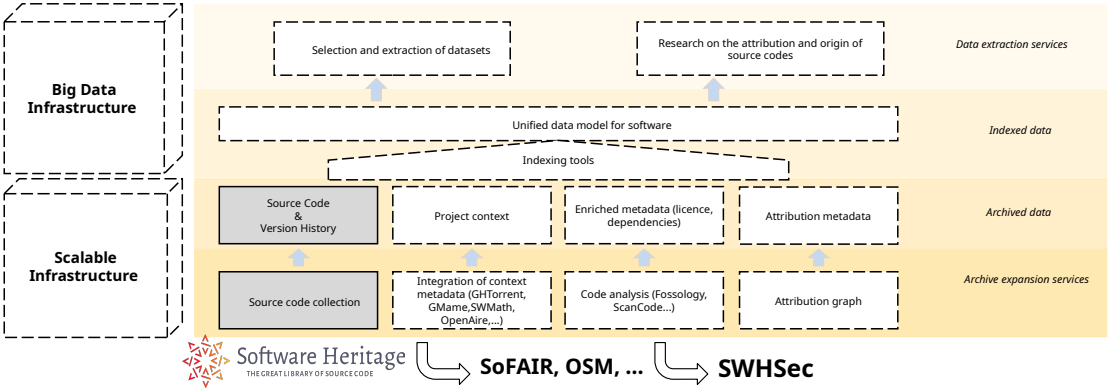
CodeCommons: bird’s eye view (technical focus)



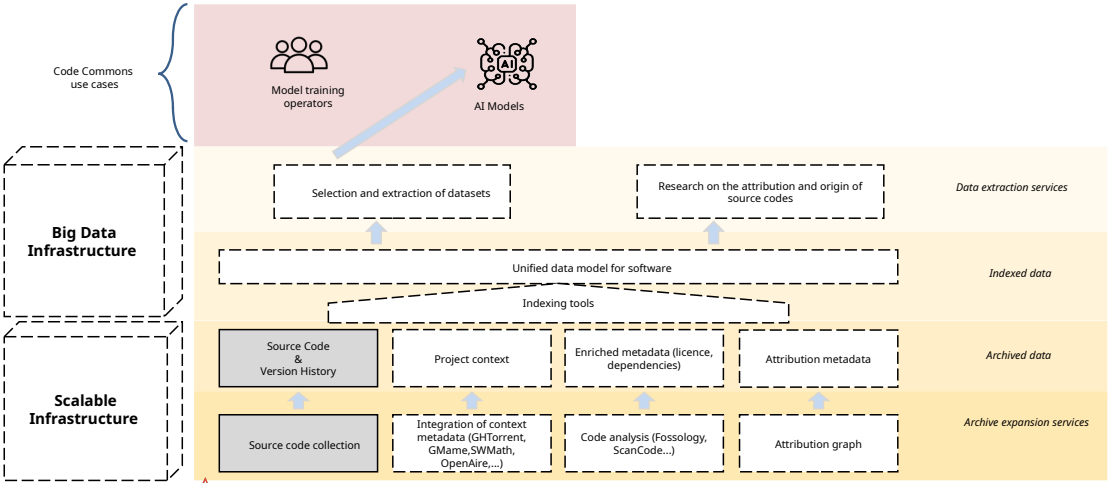
CodeCommons: bird’s eye view (technical focus)



CodeCommons: bird’s eye view (technical focus)



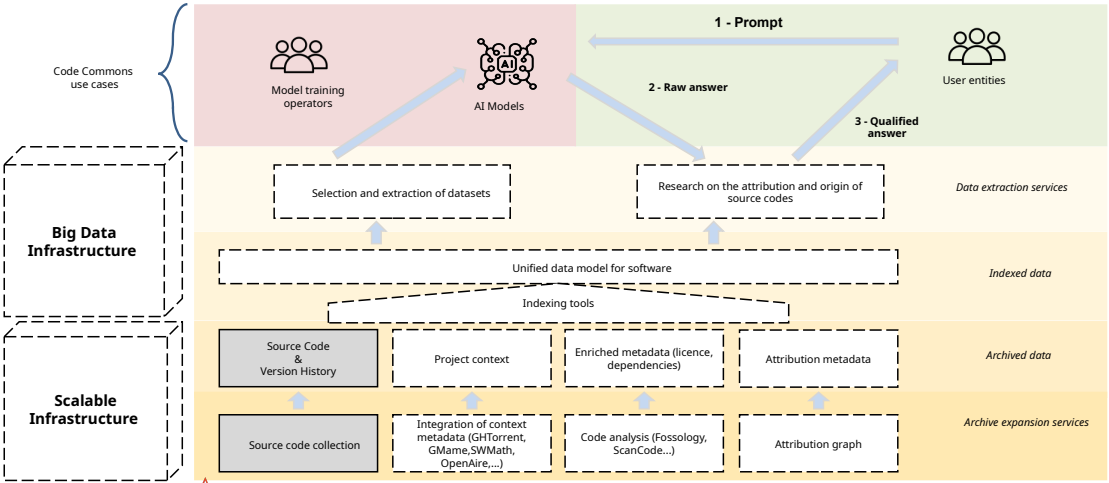
CodeCommons: bird’s eye view (technical focus)



Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

SoFAIR, OSM, ... → **SWHSec**

CodeCommons: bird’s eye view (technical focus)



COMMONS CODE: THE ACTORS

Team	Entity / Referent	Expertise
Funded partners		
Software Heritage	Inria	Universal Archive of Software Source Code
DiverSE	Inria	Software engineering, code, programming, languages, Software variability management Large-scale software evolution Generative AI for software development
Almanac	Inria	Automatic linguistic modeling and analysis and computational humanities
CEDAR	Inria	Analysis and processing of complex, large-scale data
DIASI	cea	Automatic language processing Generative AI
DILS	cea	Engineering, Software and Systems
Software Innovation Lab	TWEAG by Modus Create	Machine learning, Modeling, Natural language processing Distributed computing
Subcontracting (budget < 200k€)		
AboutCode	Philippe Ombredanne	The global benchmark for license detection
Unfunded partners		
Emeritus Inria	Patrick Valduriez	Cutting-edge expertise in big data management
Sant'Anna School of Advanced Studies – Pisa	Paolo Ferragina	Data compression and text algorithms (ACM Paris Kanellakis award 2022)
Università di Pisa	Marco Danelutto	Massively parallel HPC programming expertise
Alma Mater Studiorum Università di Bologna	Maurizio Gabbrielli	Expertise in machine learning and text similarity
Università degli Studi di Torino	Marco Aldinucci	EuroHPC and efficient low-level distributed structure expertise

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

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

Software Heritage is

- vendor neutral, open source
- worldwide, long term

Software Heritage enables

- archival, reference, integrity
- traceability, global knowledge base

Call to action

- support a shared open infrastructure to support your use cases
- develop new applications, tackle new scientific challenges
- collaborate with CodeCommons via the Chile-France binational center for AI!

Join us

Annual report 2024