Preservare il Software Bene Comune per il Futuro dell'Umanità

Stefano Zacchiroli

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

13 Novembre 2022
Biennale Tecnologia
Torino, Italy

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Outline

About the speaker

- Professor of Computer Science, Télécom Paris, Institut Polytechnique de Paris
- Free/Open Source Software activist (20+ years)
- Debian Developer & Former 3x Debian Project Leader
- Former Open Source Initiative (OSI) director
- Software Heritage co-founder & CTO

Outline

Software is everywhere (and a key mediator) in society

Free/Open Source Software (FOSS) is everywhere as well

Definition (Free Software (1986))

A program is free software if the program's users have the four *essential freedoms*:

- Freedom #0, to run the program, for any purpose
- Freedom #1, to study how the program works, and change it
- Freedom #2, to redistribute copies
- Freedom #3, to improve the program, and release improvements

Current estimates: 99% of software products on the market contains at least *some* free software parts (Synopsis 2020).

The Commons and FOSS

Definition (Commons)

The commons is the cultural and natural resources accessible to all members of a society, including natural materials such as air, water, and a habitable earth. These resources are held in common, not owned privately.

Definition (Software Commons)

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 restrictions. Thus *all open source software and all free software are part of the [software] commons.* [...]

Kranich and Schement (2008); Schweik and English (2012).

Software is dual-form knowledge

“The source code for a work means the preferred form of the work for making modifications to it."

GPL Licence

Software is dual-form knowledge

"The source code for a work means the preferred form of the work for making modifications to it." — *GPL Licence*

```
Hello World
```

Software is dual-form knowledge

"The source code for a work means the preferred form of the work for making modifications to it." — *GPL Licence*

Hello World

Program (excerpt of binary)

```
4004e6: 55
4004e7: 48 89 e5
4004ea: bf 84 05 40 00
4004ef: b8 00 00 00 00
4004f4: e8 c7 fe ff ff
4004f9: 90
4004fa: 5d
4004fb: c3
```

Software is dual-form knowledge

"The source code for a work means the preferred form of the work for making modifications to it." — *GPL Licence*

Hello World

Program (excerpt of binary)

```
4004e6: 55
4004e7: 48 89 e5
4004ea: bf 84 05 40 00
4004ef: b8 00 00 00 00
4004f4: e8 c7 fe ff ff
4004f9: 90
4004fa: 5d
4004fb: c3
```

Program (source code)

```
/* Hello World program */

#include<stdio.h>

void main()
{
    printf("Hello World");
}
```

Software *source code* is precious human 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."

Software *source code* is precious human 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
```

Software *source code* is precious human 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;
}
```

Software *source code* is precious human 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."

Outline

But *where* is this commons?

- many disparate development platforms, with a few dominant players (e.g., GitHub)
- a myriad places where distribution may happen
- most of them operated by for-profit companies

Software source code is fragile

Like all digital information, FOSS is fragile

- link rot: projects are created, moved around, removed
- business-driven code loss (e.g., Gitorious, Google Code, Bitbucket)
- data rot: physical media with legacy software decay

Software source code is fragile

Like all digital information, FOSS is fragile

- link rot: projects are created, moved around, removed
- business-driven code loss (e.g., Gitorious, Google Code, Bitbucket)
- data rot: physical media with legacy software decay

If a website disappears you go to the Internet Archive…

where do you go if (a repository on) GitHub or GitLab goes away?

Calling for preservation: UNESCO

Experts call for greater recognition of software source code as heritage for sustainable development

.6 November 2018

UNESCO, Inria, Software Heritage invite

40 international experts meet in Paris …

Calling for preservation: UNESCO

Experts call for greater recognition of software source code as heritage for sustainable development

.6 November 2018

UNESCO, Inria, Software Heritage invite

40 international experts meet in Paris …

The call is published on Feb 2019

Calling for preservation: UNESCO

UNESCO, Inria, Software Heritage invite

40 international experts meet in Paris …

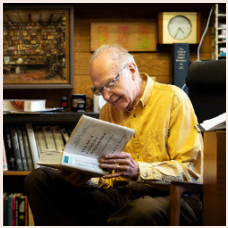
The call is published on Feb 2019

"[We call to] support efforts to gather and preserve the artifacts and narratives of the history of computing, while the earlier creators are still alive"

https://en.unesco.org/foss/paris-call-software-source-code

Calling for preservation: Donald Knuth and Len Shustek

Communications of the ACM, February 2021

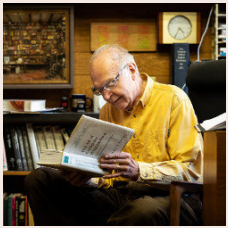


"Telling historical stories is the best way to teach. It's much easier to understand something if you know the threads it is connected to."

Let's Not Dumb Down the History of Computer Science
Donald E. Knuth, Len Shustek
https://doi.org/10.1145/3442377

Calling for preservation: Donald Knuth and Len Shustek

Communications of the ACM, February 2021



"Telling historical stories is the best way to teach. It's much easier to understand something if you know the threads it is connected to."

Let's Not Dumb Down the History of Computer Science
Donald E. Knuth, Len Shustek
https://doi.org/10.1145/3442377

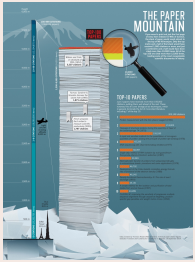
A unique opportunity

most of the creators are still here: we can talk to them!

but the clock is ticking…

Source code history for Open Science

Software powers modern research

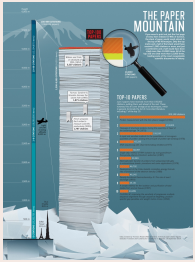


[…] software […] essential in their fields.
Top 100 papers (Nature, 2014)

Sometimes, if you dont have the software, you dont have the data
Christine Borgman, Paris, 2018

Source code history for Open Science

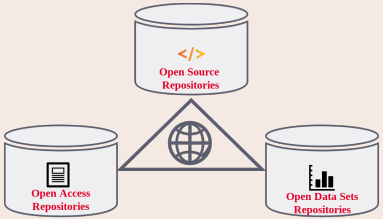
Software powers modern research



[…] software […] essential in their fields.
Top 100 papers (Nature, 2014)

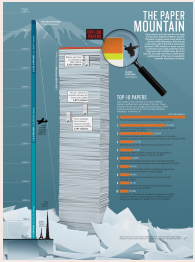
Sometimes, if you dont have the software, you dont have the data
Christine Borgman, Paris, 2018

A key pillar: software (source code)



Source code history for Open Science

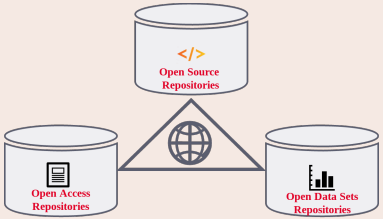
Software powers modern research



[…] software […] essential in their fields.
Top 100 papers (Nature, 2014)

Sometimes, if you dont have the software, you dont have the data
Christine Borgman, Paris, 2018

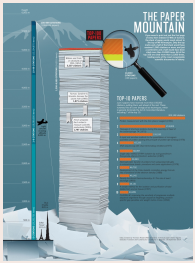
A key pillar: software (source code)



The links in the picture are important

Source code history for Open Science

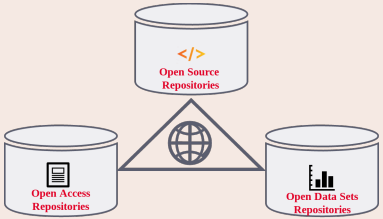
Software powers modern research



[…] software […] essential in their fields.
Top 100 papers (Nature, 2014)

Sometimes, if you dont have the software, you dont have the data
Christine Borgman, Paris, 2018

A key pillar: software (source code)



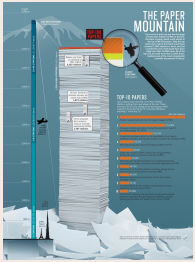
The links in the picture are important

Nota Bene

software may be a *tool*, a *research outcome* and a *research object*

Source code history for Open Science

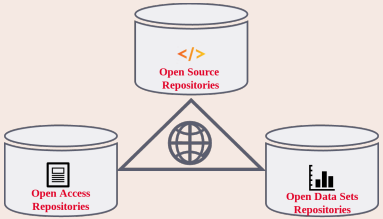
Software powers modern research



[…] software […] essential in their fields.
Top 100 papers (Nature, 2014)

Sometimes, if you dont have the software, you dont have the data
Christine Borgman, Paris, 2018

A key pillar: software (source code)



The links in the picture are important

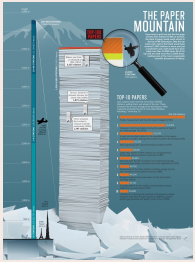
Nota Bene

software may be a *tool*, a *research outcome* and a *research object*

access to the *source code* is essential!

Source code history for Open Science

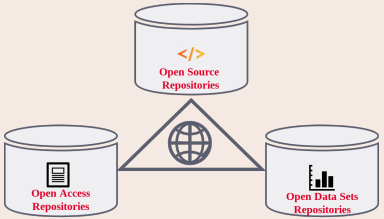
Software powers modern research



[…] software […] essential in their fields.
Top 100 papers (Nature, 2014)

Sometimes, if you dont have the software, you dont have the data
Christine Borgman, Paris, 2018

A key pillar: software (source code)



The links in the picture are important

Nota Bene

software may be a *tool*, a *research outcome* and a *research object*

access to the *source code* is essential!

Preserving the history of source code is important for *reproducibility*

Takeaways

Software Commons

- Good news: thanks to FOSS developers we are growing by the day a novel kind of digital commons rich in valuable human knowledge: the software commons.

Risk of losing it

- However: there is a risk of "ruining" it (specifically: losing access to it forever).
- As per other commons (including traditional material commons, like natural resources), we need to apply individual care, public policies, and proactive initiatives to mitigate this risk.

Outline

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

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

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

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

The largest public source code archive, principled

bit.ly/swhpaper

archive.softwareheritage.org

The largest public source code archive, principled

bit.ly/swhpaper

archive.softwareheritage.org

Technology

- transparency and FOSS
- replicas all the way down

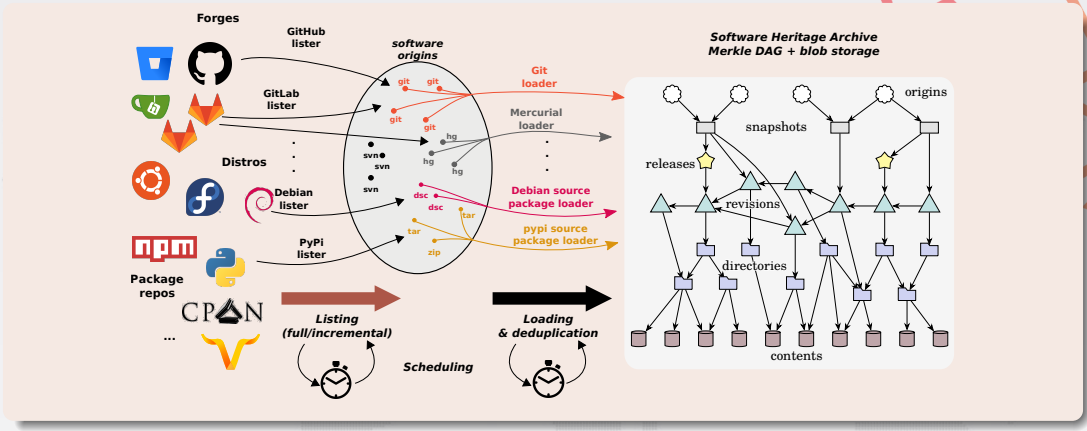
Content (billions!)

- intrinsic identifiers
- facts and provenance

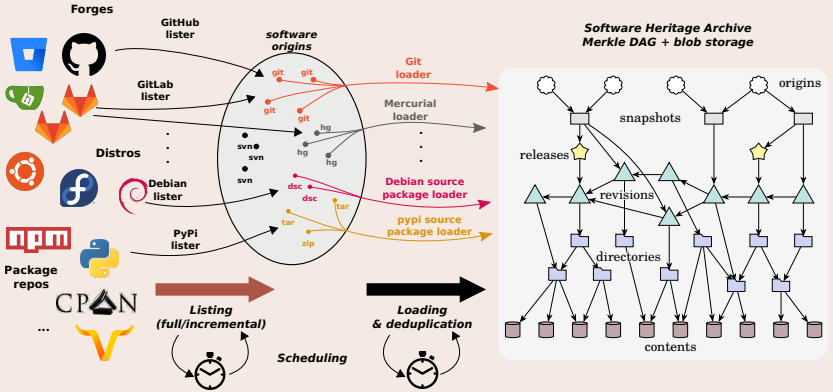
Organization

- non-profit
- multi-stakeholder

A peek under the hood: a global view on the software commons

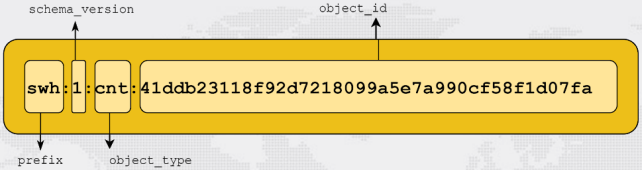


A peek under the hood: a global view on the software commons

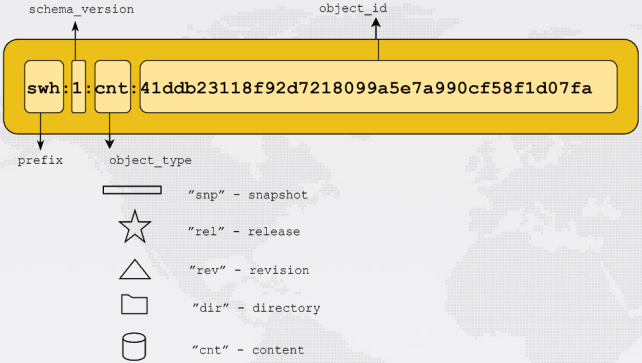


A global graph linking together fully deduplicated source code artifact (files, commits, directories, releases, etc.) to the places that distribute them (e.g., Git repositories), providing a unified view on the entire *Software Commons*.
(Size: ~30 B nodes, ~300 B edges, ~1 PiB blobs)

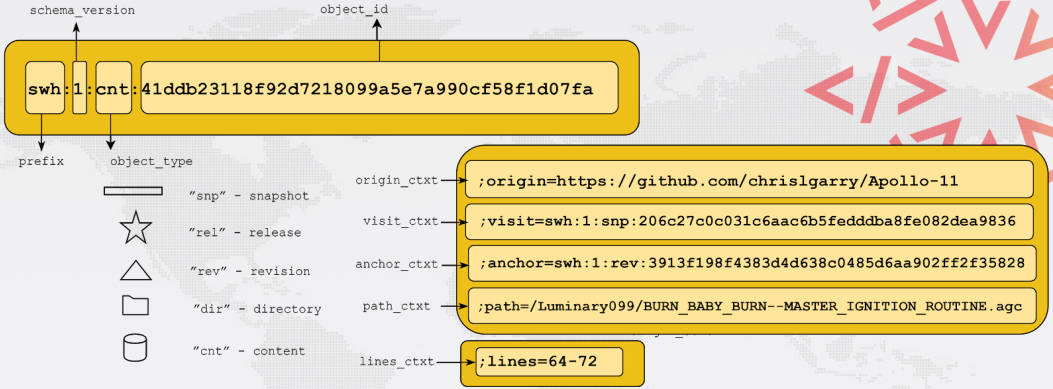
Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



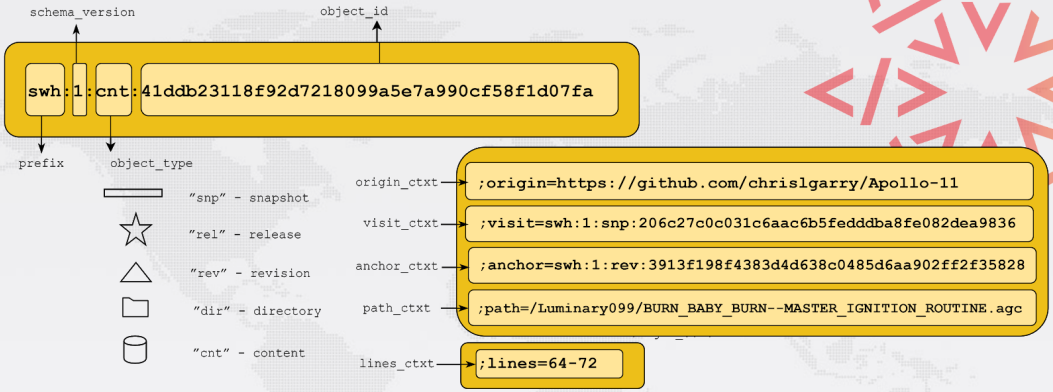
Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



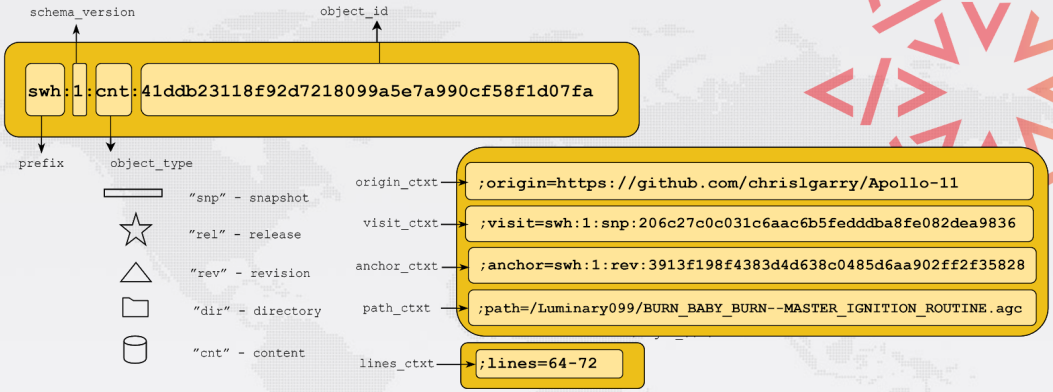
Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



An emerging standard

- in Linux Foundation's SPDX 2.2
- IANA registered, WikiData property P6138

Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



An emerging standard

- in Linux Foundation's SPDX 2.2
- IANA registered, WikiData property P6138

Examples:

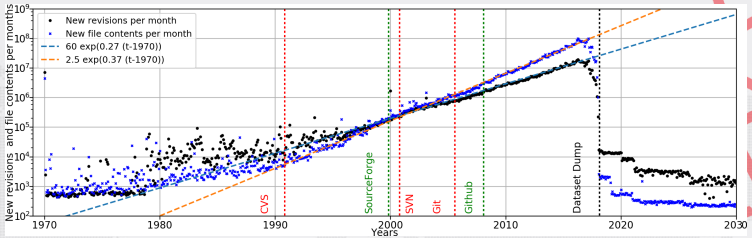
- Apollo 11 AGC excerpt
- Quake III rsqrt

Demo time!

- Browse the archive
- Trigger archival of your preferred software in a breeze
- Get and use SWHIDs (full specification available online)
- The Apollo 11 AGC source code example
- Cite software with the biblatex-software style from CTAN
- Example use in a research article: compare Fig. 1 and conclusions
  - in the 2012 version
  - in the updated version using SWHIDs and Software Heritage
- Example in a journal: an article from IPOL
- Curated deposit in SWH via HAL, see for example: LinBox, SLALOM, Givaro, NS2DDV, SumGra, Coq proof, …
- Rescue landmark legacy software, see the SWHAP process with UNESCO

Outline

Software provenance and evolution



Key findings

- The amount of original commits in public code doubles every ~30 months and has been doing so for 20+ years; original source code files double every ~22 months
- It is possible to trace the provenance of source code artifacts at this scale in a compact relational model via the notion of isochrone graphs.

Rousseau, Di Cosmo, Zacchiroli

Software Provenance Tracking at the Scale of Public Source Code

In Empirical Software Engineering, 2020

Diversity, equity, and inclusion

Idea

Archived commit metadata contains public information that can be mined to study long-term trends of diversity, equity, and inclusion (DEI) traits of the global population of public code contributors.

Key findings on the gender gap

- Male authors contributed 92% of public code commits up to 2019.
- The ratio of female authors (and their contributions) has grown stably for 15 years reaching for the first time 10% of yearly contributions in 2019.
- The COVID-19 pandemic has reversed the trend.

Diversity, equity, and inclusion (cont.)

Key findings on the geographic gap

- The early decades of public code were 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.
- The trend of increased female contributions is almost worlwide, with the notable exception of specific regions of Asia were it is either slower or flat.

References

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

Software Heritage 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: https://doi.org/10.5281/zenodo.2583978
- Chosen as subject for the MSR 2020 Mining Challenge

Formats

- Local use: PostgreSQL dumps, or Apache Parquet files (~1 TiB each)
- Live usage: Amazon Athena (SQL-queriable), Azure Data Lake

Outline

Academia & policy: growing adoption (selection)

HAL software curated deposit workflow

Curated Archiving of Research Software Artifacts
International Journal of Digical Curation, 2020

Reference archive for swmath.org

See *code* links, e.g.
SemiPar package

Academia & policy: growing adoption (selection)

HAL software curated deposit workflow

Curated Archiving of Research Software Artifacts
International Journal of Digical Curation, 2020

Reference archive for swmath.org

swMATH
an information service for mathematical software

See *code* links, e.g. SemiPar package

IPOL (image processing)

- archive (deposit)
- reference
- BibLaTeX

eLife (life sciences)

- archive (save code now)
- reference

JTCAM (mechanics)

- instructions for authors
- biblatex-software in journal LaTeX class

Academia & policy: growing adoption (selection)

HAL software curated deposit workflow

Curated Archiving of Research Software Artifacts
International Journal of Digical Curation, 2020

Reference archive for swmath.org

swMATH
an information service for mathematical software

See *code* links, e.g. SemiPar package

IPOL (image processing)

- archive (deposit)
- reference
- BibLaTeX

eLife (life sciences)

- archive (save code now)
- reference

JTCAM (mechanics)

- instructions for authors
- biblatex-software in journal LaTeX class

Policy: France

NATIONAL PLAN FOR OPEN SCIENCE

National Plan for Open Science and Research Infrastructures

Policy: Europe

EOSC SIRS report

- SWHIDs
- archive

Guidelines

- summary
- ICMS 2020

An international, non profit initiative built for the long term

Sharing the vision

United Nations Educational, Scientific and Cultural Organization

www.softwareheritage.org/support/testimonials

Donors, members, sponsors

Diamond sponsor

Platinum sponsors

Gold sponsors

Silver sponsors

Bronze sponsors

www.softwareheritage.org/support/sponsors

You can help!

Foster adoption and best practices

- archive and reference relevant source code (save code now, and deposit)
- use Software Heritage in research articles, journals, and books
- rescue and preserve landmark legacy source code with SWHAP and Software Stories

Engage with Software Heritage as an individual

- join the ambassador program, help raise awareness
- contribute to technical and scientific development

Engage with Software Heritage as an organization

- become a member/sponsor
- build a Software Heritage mirror

Thank you!

Resources

homepage	www.softwareheritage.org
archive	archive.softwareheritage.org
development	forge.softwareheritage.org
blog	www.softwareheritage.org/blog
newsletter	www.softwareheritage.org/newsletter

Selected references

(complete bibliography: www.softwareheritage.org/publications)

Roberto Di Cosmo, Stefano Zacchiroli
Software Heritage: Why and How to Preserve Software Source Code
iPRES 2017: Intl. Conf. on Digital Preservation

J.F. Abramatic, R. Di Cosmo, S. Zacchiroli,
Building the Universal Archive of Source Code
Communications of the ACM, October 2018

Appendix

Archiving goals

Targets: VCS repositories & source code releases (e.g., tarballs, packages)

We DO archive

- file content (= blobs)
- revisions (= commits), with full metadata
- releases (= tags), ditto
- where (origin) & when (visit) we found any of the above

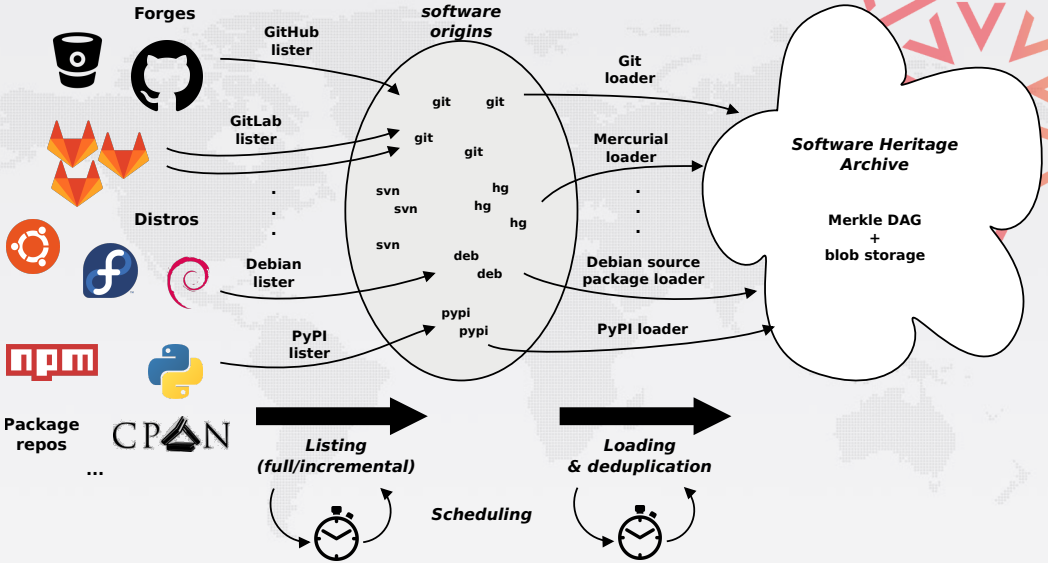
… in a VCS-/archive-agnostic canonical data model

We DON'T archive (yet)

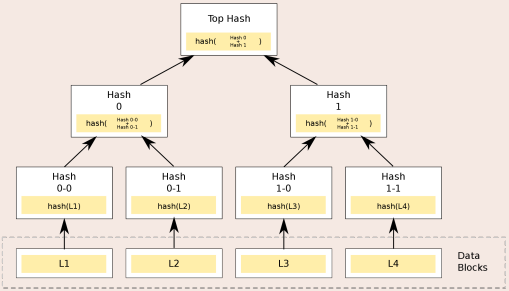
- homepages, wikis
- BTS/issues/code reviews/etc.
- mailing lists

Long term vision: play our part in a *"semantic wikipedia of software"*

Data flow


Forges
GitHub lister
GitLab lister
Distros
Debian lister
PyPI lister
Package repos
CPAN
...
software origins
git
git
git
git
svn
svn
svn
hg
hg
hg
deb
deb
pypi
pypi
Git loader
Mercurial loader
Debian source package loader
PyPI loader
Software Heritage Archive
Merkle DAG + blob storage
Listing (full/incremental)
Scheduling
Loading & deduplication

Merkle tree (R. C. Merkle, CRYPTO 1987)

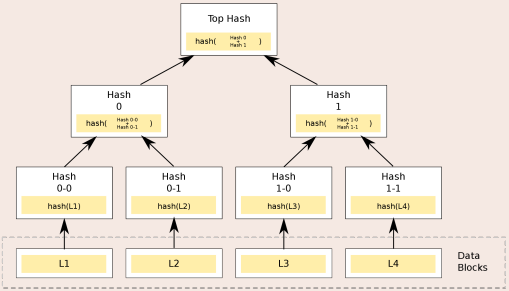


Combination of

- tree
- hash function

Merkle trees

Merkle tree (R. C. Merkle, CRYPTO 1987)



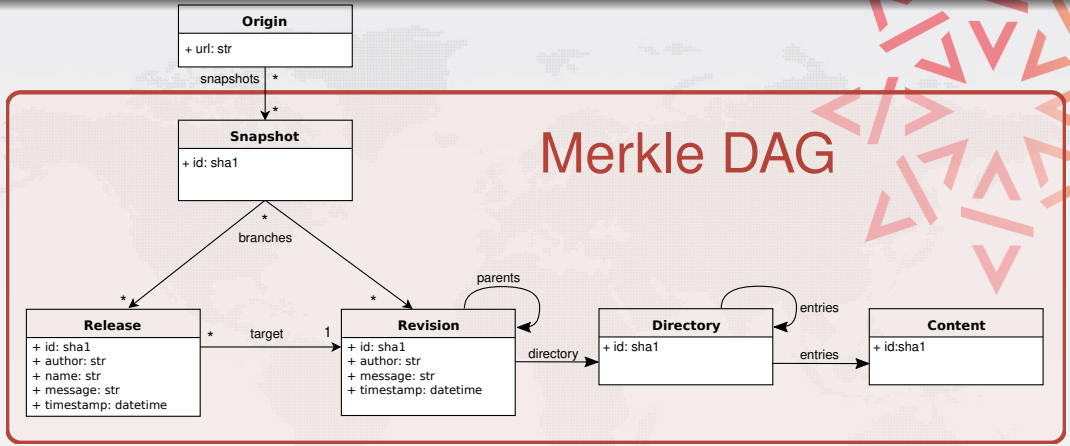
Combination of

- tree
- hash function

Classical cryptographic construction

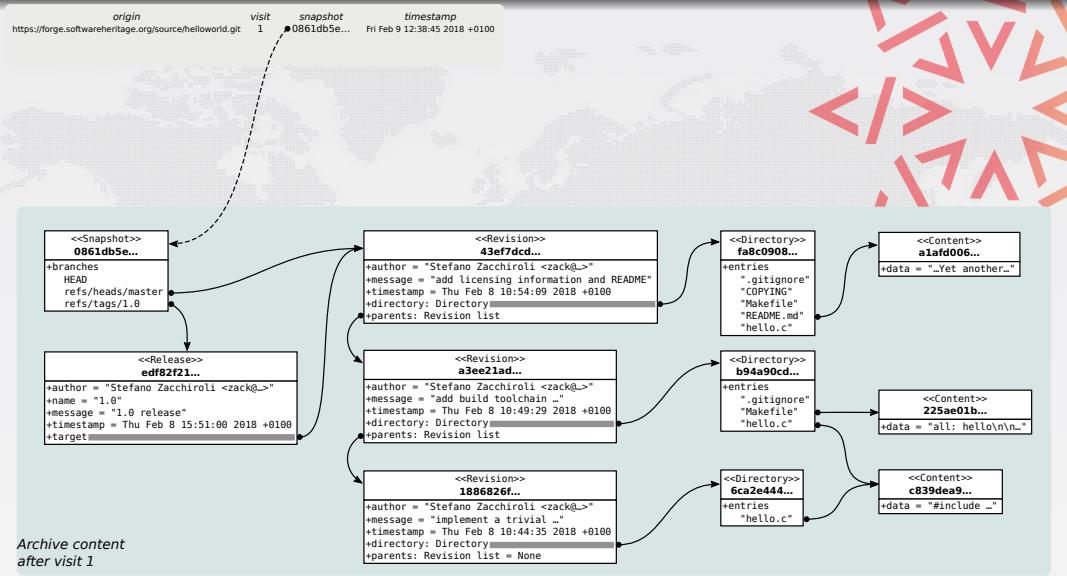
- fast, parallel signature of large data structures
- widely used (e.g., Git, blockchains, IPFS, …)
- built-in deduplication

Data model



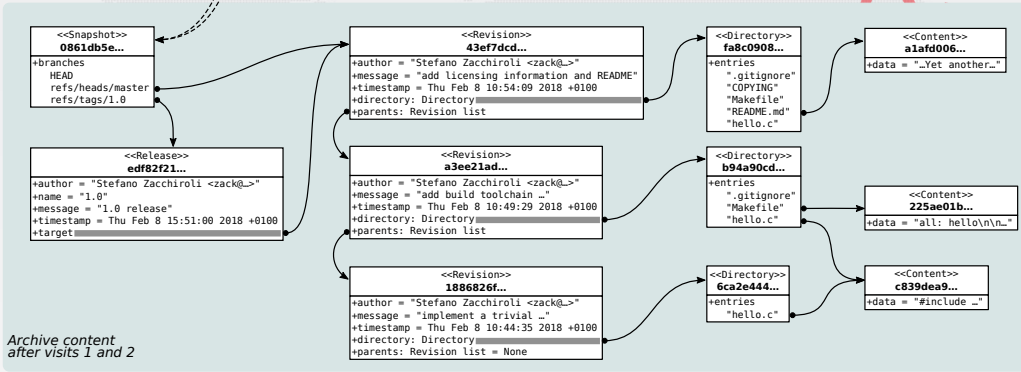
A global graph linking together fully deduplicated source code artifact (files, commits, directories, releases, etc.) to the places that distribute them (e.g., Git repositories), providing a unified view on the entire *Software Commons*.

The archive: a (giant) Merkle DAG



The archive: a (giant) Merkle DAG

origin	visit	snapshot	timestamp
https://forge.softwareheritage.org/source/helloworld.git	1	0861db5e...	Fri Feb 9 12:38:45 2018 +0100
https://forge.softwareheritage.org/source/helloworld.git	2	0861db5e...	Fri Feb 9 13:29:00 2018 +0100



The archive: a (giant) Merkle DAG

origin	visit	snapshot	timestamp
https://forge.softwareheritage.org/source/helloworld.git	1	0861db5e…	Fri Feb 9 12:38:45 2018 +0100
https://forge.softwareheritage.org/source/helloworld.git	2	0861db5e…	Fri Feb 9 13:29:00 2018 +0100
https://forge.softwareheritage.org/source/helloworld.git	3	510aa88b…	Fri Feb 9 15:52:50 2018 +0100