Open Science Best Practices for Source Code Preservation

Stefano Zacchiroli

Software Heritage
Télécom Paris, Polytechnic Institute of Paris

30 March 2023
CERN
Geneva, Switzerland

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Outline

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

About the speaker

- Professor of Computer Science, Télécom Paris, Institut Polytechnique de Paris
  - Research: digital commons, open source software engineereing, computer security, software supply chain.
  - 70+ papers in these fields: upsilon.cc/zack/research/publications
- 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

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

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

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

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 …

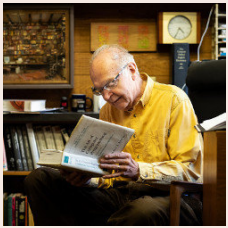
The call is published on Feb 2019

"[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

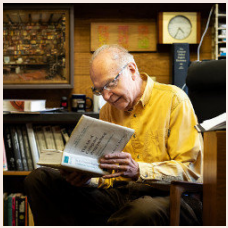


"Telling historical stories is the best way to teach. It's much easier to understand something if you know the threads it is connected to."

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…

Why Open Science?

Open Science (Second National Plan for Open Science, France, 2021)

Unhindered dissemination of results, methods and products from scientific research.
It draws on *the opportunity provided by recent digital progress* to develop *open access* to *publications* and – as much as possible – *data*, *source code* and *research methods*.

Why Open Science?

Open Science (Second National Plan for Open Science, France, 2021)

Unhindered dissemination of results, methods and products from scientific research.
It draws on *the opportunity provided by recent digital progress* to develop *open access* to *publications* and – as much as possible – *data*, *source code* and *research methods*.

Jean-Eric Paquet (EU DGRI, on the objective of Open Science)

"Increase *scientific quality*, the *pace of discovery and technological development*, as well as *societal trust in science*."

Why Open Science?

Open Science (Second National Plan for Open Science, France, 2021)

Unhindered dissemination of results, methods and products from scientific research.
It draws on *the opportunity provided by recent digital progress* to develop *open access* to *publications* and – as much as possible – *data*, *source code* and *research methods*.

Jean-Eric Paquet (EU DGRI, on the objective of Open Science)

"Increase *scientific quality*, the *pace of discovery and technological development*, as well as *societal trust in science*."

Mariya Gabriel (EU Commissionner for Research)

The COVID-19 crisis has also shown that cooperation at international level in research and innovation is more important than ever, including through *open access to data and results. No nation, no country can tackle any of these global challenges alone.*

Why Open Science?

Open Science (Second National Plan for Open Science, France, 2021)

Unhindered dissemination of results, methods and products from scientific research.
It draws on *the opportunity provided by recent digital progress* to develop *open access* to *publications* and – as much as possible – *data*, *source code* and *research methods*.

Jean-Eric Paquet (EU DGRI, on the objective of Open Science)

"Increase *scientific quality*, the *pace of discovery and technological development*, as well as *societal trust in science*."

Mariya Gabriel (EU Commissionner for Research)

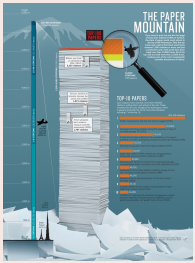
The COVID-19 crisis has also shown that cooperation at international level in research and innovation is more important than ever, including through *open access to data and results. No nation, no country can tackle any of these global challenges alone.*

Yuval Noah Harari (on COVID 19)

"The real antidote [to epidemic] is scientific knowledge *and* global cooperation."

Software is a pillar of Open Science

Software powers modern research

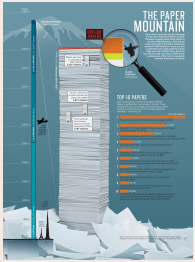


[…] software […] essential in their fields.
Top 100 papers (Nature, 2014)

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

Software is a pillar of Open Science

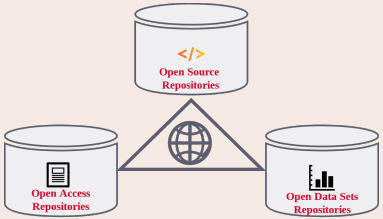
Software powers modern research



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

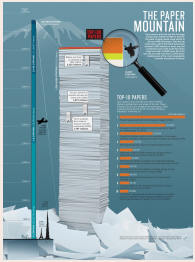
Sometimes, if you dont have the software, you dont have the data
Christine Borgman, Paris, 2018

A key pillar: software (source code)



Software is a pillar of Open Science

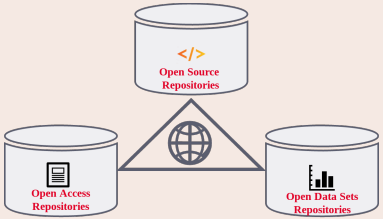
Software powers modern research



[…] 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

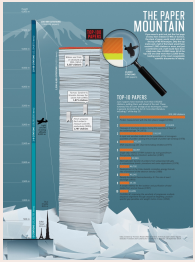
A key pillar: software (source code)



The links in the picture are important

Software is a pillar of Open Science

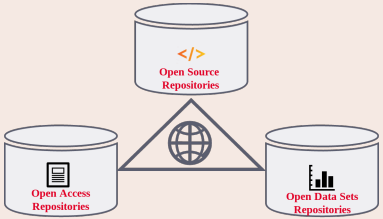
Software powers modern research



[…] 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)



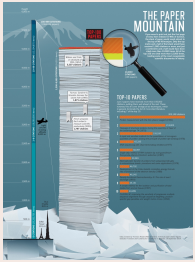
The links in the picture are important

Nota Bene

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

Software is a pillar of Open Science

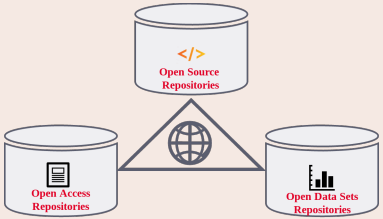
Software powers modern research



[…] 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

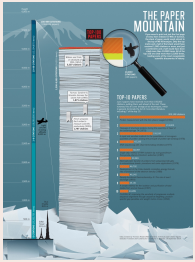
Nota Bene

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

access to the *source code* is essential!

Software is a pillar of Open Science

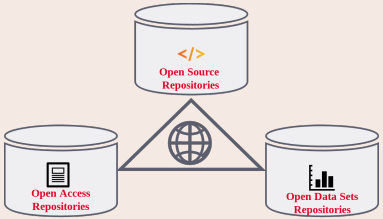
Software powers modern research



[…] 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*

Outline

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

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

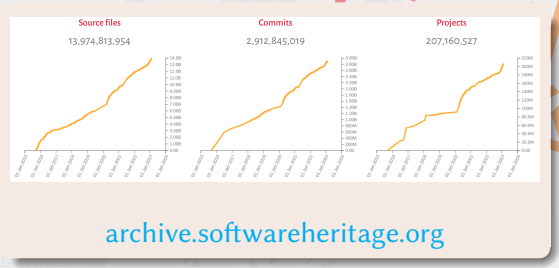
preserve and share all software source code

Research infrastructure

enable analysis of all software source code

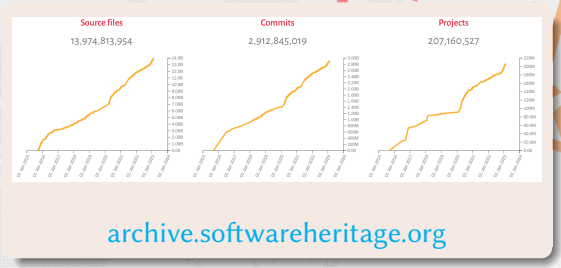
The largest public source code archive, principled

bit.ly/swhpaper



The largest public source code archive, principled

bit.ly/swhpaper



Technology

- transparency and FOSS
- replicas all the way down

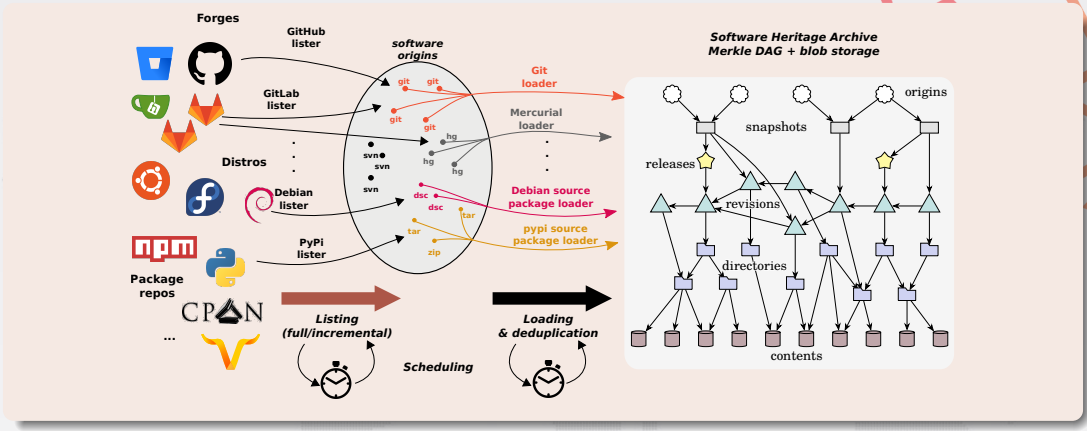
Content (billions!)

- intrinsic identifiers
- facts and provenance

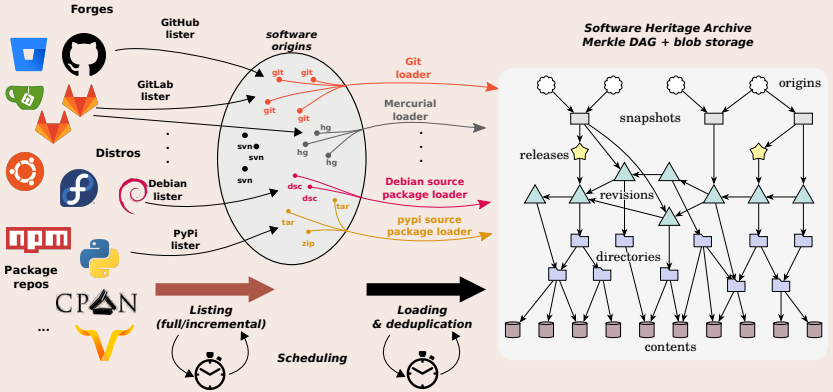
Organization

- non-profit
- multi-stakeholder

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

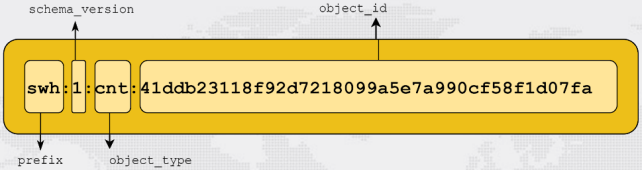


A peek under the hood: a global view on the software commons

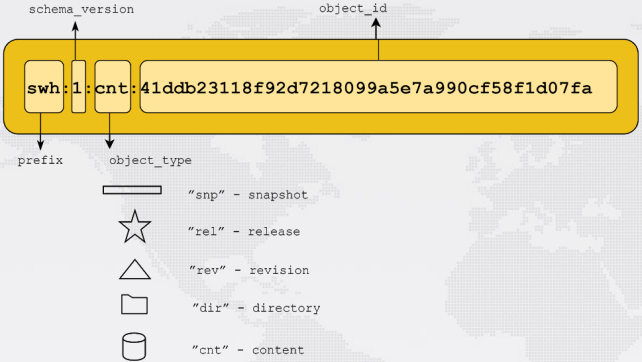


A global graph linking together fully deduplicated source code artifact (files, commits, directories, releases, etc.) to the places that distribute them (e.g., Git repositories), providing a unified view on the entire *Software Commons*.
(Size: ~30 B nodes, ~300 B edges, ~1 PiB blobs)

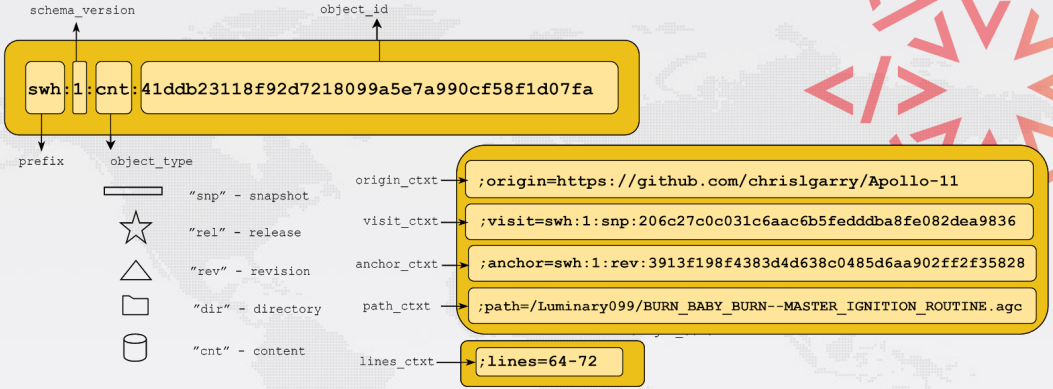
Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



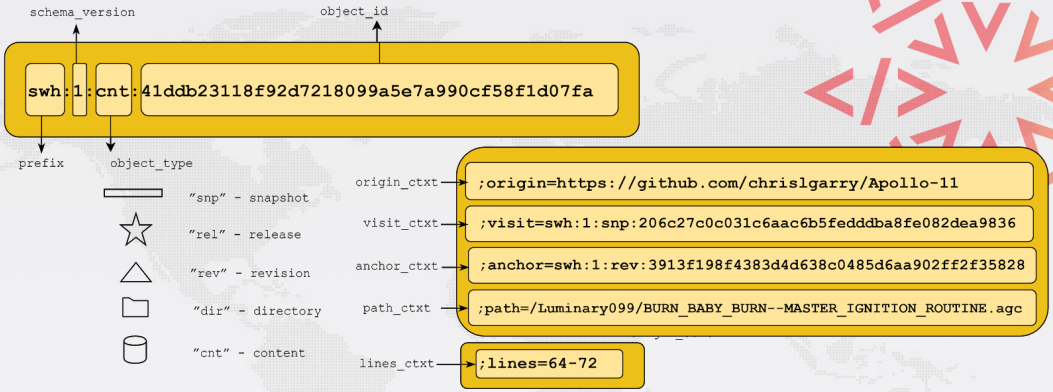
Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



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



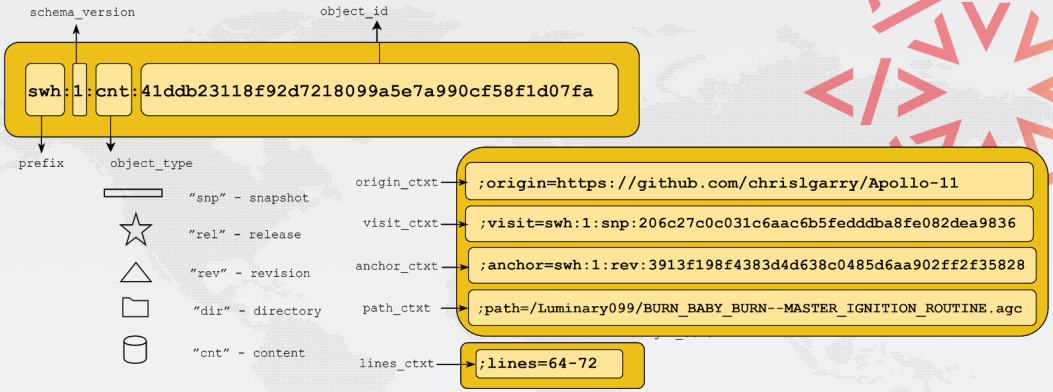
Software Heritage *intrinsic* Identifiers (SWHID) (full spec)



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

Outline

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

Source code in Open Science: a plurality of needs to address

Researchers

- archive and reference software used in articles
- find useful software
- get credit for developed software
- verify, reproduce, improve results

Source code in Open Science: a plurality of needs to address

Researchers

- archive and reference software used in articles
- find useful software
- get credit for developed software
- verify, reproduce, improve results

Laboratories/teams

- track software contributions
- produce reports
- maintain web page

Source code in Open Science: a plurality of needs to address

Researchers

- archive and reference software used in articles
- find useful software
- get credit for developed software
- verify, reproduce, improve results

Laboratories/teams

- track software contributions
- produce reports
- maintain web page

Research Organizations and/or Funders

know its software assets

- technology transfer
- impact metrics
- funding strategy
- career evaluation

Archive, Reference, Describe, Cite and Credit

Archive

Research software artifacts must be properly archived

make sure we can *retrieve* them (*reproducibility*)

Archive, Reference, Describe, Cite and Credit

Archive

Research software artifacts must be properly archived

make sure we can *retrieve* them (*reproducibility*)

Reference

Research software artifacts must be properly referenced

make sure we can *identify* them (*reproducibility*)

Archive, Reference, Describe, Cite and Credit

Archive

Research software artifacts must be properly archived

make sure we can *retrieve* them (*reproducibility*)

Reference

Research software artifacts must be properly referenced

make sure we can *identify* them (*reproducibility*)

Describe

Research software artifacts must be properly described

make it easy to *discover* and *reuse* them (*visibility*)

Archive, Reference, Describe, Cite and Credit

Archive

Research software artifacts must be properly archived

make sure we can *retrieve* them (*reproducibility*)

Reference

Research software artifacts must be properly referenced

make sure we can *identify* them (*reproducibility*)

Describe

Research software artifacts must be properly described

make it easy to *discover* and *reuse* them (*visibility*)

Cite/Credit

Research software artifacts must be properly cited *(not the same as referenced!)*

to give *credit* to authors (*evaluation*!)

Outline

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

Some old popular approaches

A - Since the ~~1970's~~ 1990's

.zip or .tar file on:

- ~~ftp server~~ (e.g. gnu)
- web page (example)
- document archive (+ DOI sample)

Some old popular approaches

A - Since the ~~1970's~~ 1990's

.zip or .tar file on:

- ~~ftp server~~ (e.g. gnu)
- web page (example)
- document archive (+ DOI sample)

B - Since the 2000's

Rely on *software forges*

- institutional/project (e.g. example)
- free commercial ones: BitBucket, GitHub, GitLab, … (e.g. parmap)

Some old popular approaches

A - Since the ~~1970's~~ 1990's

.zip or .tar file on:

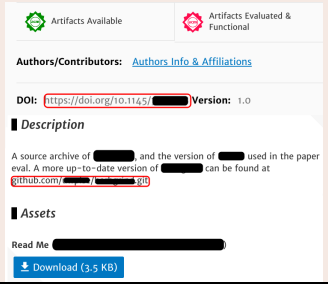
- ~~ftp server~~ (e.g. gnu)
- web page (example)
- document archive (+ DOI sample)

B - Since the 2000's

Rely on *software forges*

- institutional/project (e.g. example)
- free commercial ones: BitBucket, GitHub, GitLab, … (e.g. parmap)

C: a mix of the two



Some old popular approaches

A - Since the ~~1970's~~ 1990's

.zip or .tar file on:

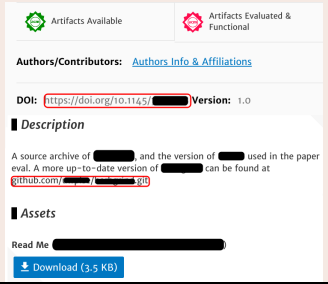
- ~~ftp server~~ (e.g. gnu)
- web page (example)
- document archive (+ DOI sample)

B - Since the 2000's

Rely on *software forges*

- institutional/project (e.g. example)
- free commercial ones: BitBucket, GitHub, GitLab, … (e.g. parmap)

C: a mix of the two



Can get no satisfaction…

A *Poor user experience*

B *No preservation guarantee*

C Can do *so much* better

Forges are *not* archives!

2015: the first big bad news

Google Code and Gitorious.org shutdown: ~1M endangered repositories

- broken links in the web of knowledge (my papers too)

Forges are *not* archives!

2015: the first big bad news

Google Code and Gitorious.org shutdown: ~1M endangered repositories

- broken links in the web of knowledge (my papers too)

Big bad news keep coming in

- summer 2019: BitBucket announces Mercurial VCS sunset
- july 2020: BitBucket erases *250.000+* repositories (including research software)
- summer 2022: GitLab.com considers erasing all projects that are inactive for a year

Forges are *not* archives!

2015: the first big bad news

Google Code and Gitorious.org shutdown: ~1M endangered repositories

- broken links in the web of knowledge (my papers too)

Big bad news keep coming in

- summer 2019: BitBucket announces Mercurial VCS sunset
- july 2020: BitBucket erases *250.000+* repositories (including research software)
- summer 2022: GitLab.com considers erasing all projects that are inactive for a year

In Academia too!

- 2021: Inria's old gforge is unplugged… breaks the Opam build chain for OCaml

Forges are *not* archives!

2015: the first big bad news

Google Code and Gitorious.org shutdown: ~1M endangered repositories

- broken links in the web of knowledge (my papers too)

Big bad news keep coming in

- summer 2019: BitBucket announces Mercurial VCS sunset
- july 2020: BitBucket erases *250.000+* repositories (including research software)
- summer 2022: GitLab.com considers erasing all projects that are inactive for a year

In Academia too!

- 2021: Inria's old gforge is unplugged… breaks the Opam build chain for OCaml

We need a universal archive of software source code:

Forges are *not* archives!

2015: the first big bad news

Google Code and Gitorious.org shutdown: ~1M endangered repositories

- broken links in the web of knowledge (my papers too)

Big bad news keep coming in

- summer 2019: BitBucket announces Mercurial VCS sunset
- july 2020: BitBucket erases *250.000+* repositories (including research software)
- summer 2022: GitLab.com considers erasing all projects that are inactive for a year

In Academia too!

- 2021: Inria's old gforge is unplugged… breaks the Opam build chain for OCaml

We need a universal archive of software source code: now we have one!

Software Heritage is *radically different*

A quick tour as a user

- designed for source code: Browse (e.g. Apollo 11 excerpt, see also Apollo 11 blog post) like on a developer platform, not a document archive!

Software Heritage is *radically different*

A quick tour as a user

- designed for source code: Browse (e.g. Apollo 11 excerpt, see also Apollo 11 blog post) like on a developer platform, not a document archive!
- reference source code: all granularities, using SWHIDs (full specification available online)
  - compare Fig. 1 and conclusions in the 2012 version and the updated version
  - SWHID in a replication experiment
  - guidelines and a full article
  - SWHIDs *guarantee integrity* like in *blockchains*
    demo if time left:
    1. download a version of a project for a given SWHID
    2. compute locally the SWHID with swh-identify
    3. check that the computed id match the given one

Software Heritage is *radically different* (cont.)

Getting software archived

- automated harvesting: over 200 million software origins, your researchers' work may already be there (actually, here)!

Software Heritage is *radically different* (cont.)

Getting software archived

- automated harvesting: over 200 million software origins, your researchers' work may already be there (actually, here)!
- universal archive: *all* source code from *all* platforms (BitBucket, GitHub, GitLab, your own forge, etc.)
  - trigger archival of *any code* in one click with the updateswh browser extension
  - use webhooks to automatically archive *your code* (a GitHub action is available too)
  - journals, libraries, open access portals may *deposit sourcecode and metadata*
    - Example article from IPOL
    - Example article from eLife

A look at some adoption indicators

From Melissa Harrison's OSEC 2022 talk

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

- 6205 "software" references identified
- Top 8 listed, then long tail of 1055 other sites – 932 are unique "source"

A look at some adoption indicators

From Melissa Harrison's OSEC 2022 talk

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

- 6205 "software" references identified
- Top 8 listed, then long tail of 1055 other sites – 932 are unique "source"

Upcoming on replicabilitystamp.org

A look at some adoption indicators

From Melissa Harrison's OSEC 2022 talk

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

- 6205 "software" references identified
- Top 8 listed, then long tail of 1055 other sites – 932 are unique "source"

Upcoming on replicabilitystamp.org

HAL+SWH in the Open Science software booklet

Funding agencies recommendations ANR 2023 guidelines (p. 17)

Enfin, conformément au 2ème Plan national pour la science ouverte, L'ANR recommande que les logiciels développés durant le projet soient mis à disposition sous une licence libre[30] et que les codes sources soient stockés dans l'archive Software Heritage[31] en indiquant la référence au financement ANR.

Outline

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

Describe, cite, credit

Software metadata: codemeta.json

- example from Parmap, created using the Codemeta generator

Describe, cite, credit

Software metadata: codemeta.json

- example from Parmap, created using the Codemeta generator

Integration with the HAL national french open access archive

- Curated deposit: metadata quality due to moderation
  - all pieces of the puzzle together: one researcher does all the steps (Parmap)

Describe, cite, credit

Software metadata: codemeta.json

- example from Parmap, created using the Codemeta generator

Integration with the HAL national french open access archive

- Curated deposit: metadata quality due to moderation
  - all pieces of the puzzle together: one researcher does all the steps (Parmap)
- export of citation information for biblatex-software
- examples: LinBox, SLALOM, Givaro, NS2DDV, SumGra, Coq proof, …
- generation of reports, cv, web pages: for Inria, for CNRS, for CNES, for LIRMM or for Rémi Gribonval using HalTools

Bottomline

Software Heritage + a *curated* metadata repository allows to address all needs …

- *researcher, engineer*: archival, reference, credit, CV etc. *with a little effort from them*
- *labs, organizations*: track and report software production in a simple way
- *technology transfer offices*: view the software production
- *national level*: a *curated* catalog of the software production

Outline

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

French National plan for Open Science, 2021-2024

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

Liberté
Égalité
Fraternité

SECOND FRENCH PLAN FOR OPEN SCIENCE

Generalising open science in France 2021-2024

1

Launch on 6 July 2021 by Frédérique Vidal, Minister for Higher Education, Research and Innovation

- Multiplying the **levers for change** in order to **generalise open science practices**
- Structuring the **policy for opening up or sharing research data**
- New commitments to the **opening of source code** produced by research
- **European and international inclusion** in the context of the French Presidency of the European Union
- **Disciplinary and thematic variations**: open science policies must be adapted to disciplinary specificities

2

French National plan for Open Science, 2021-2024

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

Liberté
Égalité
Fraternité

SECOND FRENCH PLAN FOR OPEN SCIENCE

Generalising open science in France 2021-2024

1

Launch on 6 July 2021 by Frédérique Vidal, Minister for Higher Education, Research and Innovation

- Multiplying the **levers for change** in order to **generalise open science practices**
- Structuring the **policy for opening up or sharing research data**
- New commitments to the **opening of source code** produced by research
- **European and international inclusion** in the context of the French Presidency of the European Union
- **Disciplinary and thematic variations**: open science policies must be adapted to disciplinary specificities

2

Path Three :

Opening up and promoting source code produced by research

7 **Recognize and support the dissemination under an open source license of software** produced by publicly funded research programmes

8 **Highlight the production of source code** from higher education, research and innovation

9 Define and promote an **open source software policy**

« The opening of software source code is a major challenge for the **reproducibility** of scientific results. »

« Distribution of software products under **open source licence** will be preferred. »

3

Define and promote an open source software policy

- Produce a **National Charter for Open Source Software** coming from higher education, research and innovation
- Develop the **link between data and software** through a network of **Chief Data Officers** in the various universities and research performing organisations.
- Develop the **economic models of open source software** and make them known within commercialization services
- **Support Software Heritage** and recommend it for the archiving and referencing of source code

Recognise source code as a contribution to research

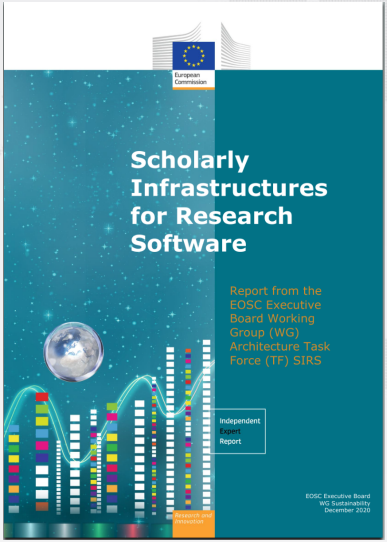
- Create an **open source research software prize**
- **Provide greater recognition** for software production in the career of researchers, research support staff

Build an ecosystem that connects code, data and publications

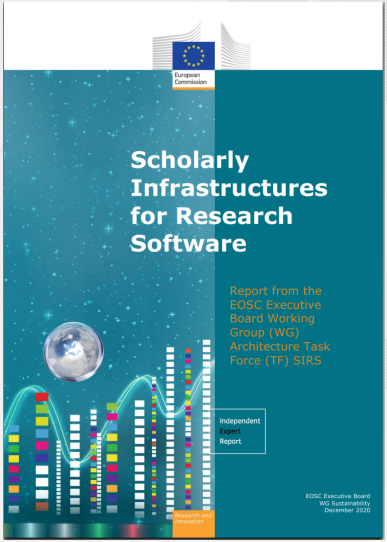
- Develop **proper coordination** between software forges, open publication archives, data repositories and the scientific publishing sector.

4

The EOSC SIRS report: Software Source Code and Open Science, 2020



The EOSC SIRS report: Software Source Code and Open Science, 2020



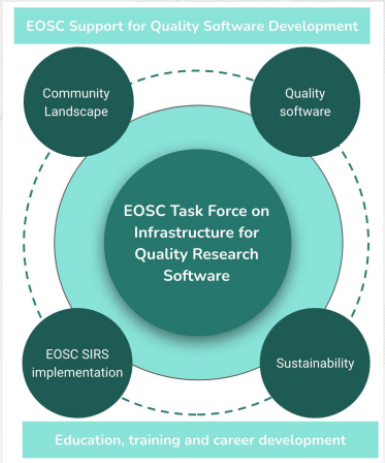
Important *policy tool* in Open Science (Dec 2020)

- 9 infrastructures
  - 3 archives
  - 3 open access publishers
  - 3 aggregators
- recommendations
  - archive in Software Heritage, use SWHID
  - open non profit
  - default to open source for research software

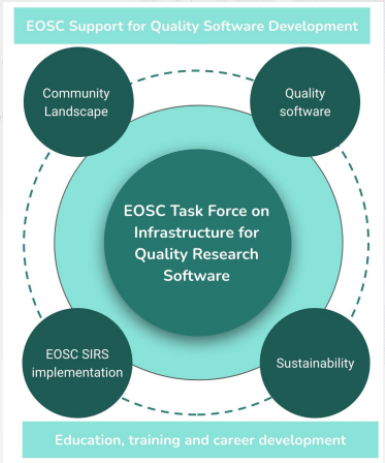
"all research software should be made available under an Open Source license by default, and all deviations from this default practice should be properly motivated"

See https://doi.org/10.2777/28598

Software in the EOSC



Software in the EOSC



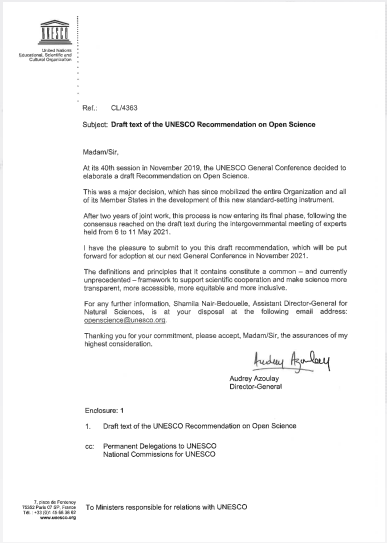
Ongoing action in the EOSC

Task force on infrastructures for quality research software

- Foster the development and deployment of tools and services that allow researchers to properly archive, reference, describe with proper metadata, share and reuse research software.
- Improve the quality of research software, both from the technical and organizational point of view …
- Increase recognition to software developers and maintainers of research software …

See the charter of the task force.

The UNESCO recommendations for Open Science (2021)



United Nations
Educational, Scientific and
Cultural Organization

Ref.: CL/4363

Subject: **Draft text of the UNESCO Recommendation on Open Science**

Madam/Sir,

At its 40th session in November 2019, the UNESCO General Conference decided to elaborate a draft Recommendation on Open Science.

This was a major decision, which has since mobilized the entire Organization and all of its Member States in the development of this new standard-setting instrument.

After two years of joint work, this process is now entering its final phase, following the consensus reached on the draft text during the intergovernmental meeting of experts held from 6 to 11 May 2021.

I have the pleasure to submit to you this draft recommendation, which will be put forward for adoption at our next General Conference in November 2021.

The definitions and principles that it contains constitute a common – and currently unprecedented – framework to support scientific cooperation and make science more transparent, more accessible, more equitable and more inclusive.

For any further information, Shamila Nair-Bedouelle, Assistant Director-General for Natural Sciences, is at your disposal at the following email address: openscience@unesco.org.

Thanking you for your commitment, please accept, Madam/Sir, the assurances of my highest consideration.

Audrey Azoulay
Director-General

Enclosure: 1

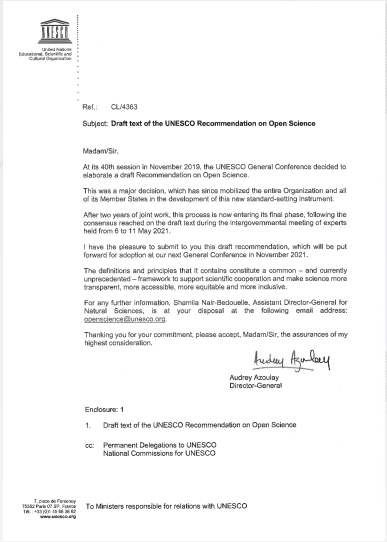
1. Draft text of the UNESCO Recommendation on Open Science

cc: Permanent Delegations to UNESCO
National Commissions for UNESCO

7, place de Fontenoy
75352 Paris 07 SP, France
Tél. : +33 (0)1 45 68 38 62
www.unesco.org

To Ministers responsible for relations with UNESCO

The UNESCO recommendations for Open Science (2021)



United Nations
Educational, Scientific and
Cultural Organization

Ref.: CL/4363

Subject: **Draft text of the UNESCO Recommendation on Open Science**

Madam/Sir,

At its 40th session in November 2019, the UNESCO General Conference decided to elaborate a draft Recommendation on Open Science.

This was a major decision, which has since mobilized the entire Organization and all of its Member States in the development of this new standard-setting instrument.

After two years of joint work, this process is now entering its final phase, following the consensus reached on the draft text during the intergovernmental meeting of experts held from 6 to 11 May 2021.

I have the pleasure to submit to you this draft recommendation, which will be put forward for adoption at our next General Conference in November 2021.

The definitions and principles that it contains constitute a common – and currently unprecedented – framework to support scientific cooperation and make science more transparent, more accessible, more equitable and more inclusive.

For any further information, Shamila Nair-Bedouelle, Assistant Director-General for Natural Sciences, is at your disposal at the following email address: openscience@unesco.org.

Thanking you for your commitment, please accept, Madam/Sir, the assurances of my highest consideration.

Audrey Azoulay
Director-General

Enclosure: 1

1. Draft text of the UNESCO Recommendation on Open Science

cc: Permanent Delegations to UNESCO
National Commissions for UNESCO

7, place de Fontenoy
75352 Paris 07 SP, France
Tél. : +33 (0)1 45 68 38 02
www.unesco.org

To Ministers responsible for relations with UNESCO

Selection from the recommendations

- Open Source for Open Science

  "The source code must be included in the software release and made available on openly accessible repositories and the chosen license must allow modifications, derivative works and sharing under equal or compatible open terms and conditions"

- Infrastructures

  "Open science infrastructures should be organized and financed upon an essentially not-for-profit and long-term vision, which enhance open science practices and guarantee permanent and unrestricted access to all, to the largest extent possible."

Outline

1. Introduction
2. The software commons: an endangered knowledge
3. Avoiding the tragedy of an unattended software commons
4. Source code in open science
5. Archive and reference
6. Describe, cite, credit
7. Policy framework
8. Call to actions

Call to action: best practices for ARDC are available… today!

Archiving and referencing

For all source code used in research (*yes, even small scripts!*)

- ensure it is archived in Software Heritage (see save code now)
- get the proper SWHID for your software (see detailed HOWTO)
- add it to research articles for reproducibility (see detailed HOWTO)

Call to action: best practices for ARDC are available… today!

Archiving and referencing

For all source code used in research (*yes, even small scripts!*)

- ensure it is archived in Software Heritage (see save code now)
- get the proper SWHID for your software (see detailed HOWTO)
- add it to research articles for reproducibility (see detailed HOWTO)

Describing and Citing/Crediting

For software you want to put forward (*mention in your CV, reports, etc., get citations and credit for it*), do the following extra steps:

- add codemeta.json with description (see the codemeta generator)
- reference in the HAL portal (french partners, see online HAL documentation)
- cite software using the biblatex-software package (in CTAN and TeXLive)

Call to action: policy making

A working agenda

- avoid proprietarisation: set the default to open
  - *publicly funded research software should be open source*, exceptions must be justified
- avoid balkanisation
  - build on common, shared, open, non profit infrastructures, like Software Heritage
- support mutualised common infrastructures
  - acknowledge the predominant human component of digital infrastructures
    - recurrent funding of their cost
    - proper evaluation of their service
- establish intelligent incentives
  - count quality software contributions in careers, avoid purely numerical indicators, keep the human in the loop

Appendix

Outline

9 Under the hood

10 HAL and Software Heritage

11 Software Heritage vs. other approaches

12 FAIR software

13 All the source code

14 More on the data model

Under the hood

Deposit *HAL, eLife, IPOL, …*

- SWORD deposit of software artifacts (2017-…) and metadata (2021)
- dashboards (global, per user)

Under the hood

Deposit

HAL, eLife, IPOL, …

- SWORD deposit of software artifacts (2017-…) and metadata (2021)
- dashboards (global, per user)

Save code now

~190.000 requests as of December 2022!

- from slow (2018) to fast version (2021)
- updateswh browser extension (August 2022)
- webhooks for popular forges (end of 2022, *to announce*)

Under the hood

Deposit

HAL, eLife, IPOL, …

- SWORD deposit of software artifacts (2017-…) and metadata (2021)
- dashboards (global, per user)

Save code now

~190.000 requests as of December 2022!

- from slow (2018) to fast version (2021)
- updateswh browser extension (August 2022)
- webhooks for popular forges (end of 2022, *to announce*)

User authentication and authorization

keycloak (*connection with EOSC AAI will be important soon*)

Under the hood

Deposit

HAL, eLife, IPOL, …

- SWORD deposit of software artifacts (2017-…) and metadata (2021)
- dashboards (global, per user)

Save code now

~190.000 requests as of December 2022!

- from slow (2018) to fast version (2021)
- updateswh browser extension (August 2022)
- webhooks for popular forges (end of 2022, *to announce*)

User authentication and authorization

keycloak (*connection with EOSC AAI will be important soon*)

Metadata access and software citation will be next

GraphQL API (thanks Jayesh), and citation generation widget

Outline

9 Under the hood

10 HAL and Software Heritage

11 Software Heritage vs. other approaches

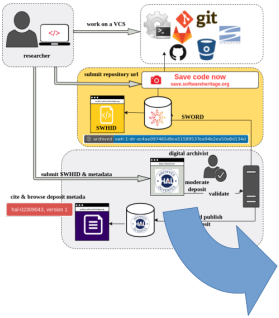
12 FAIR software

13 All the source code

14 More on the data model

Overview of the Software Heritage / HAL synergy

https://hal.archives-ouvertes.fr/hal-02130801



swh:1:dir:393b611a1424f032e83569bf6762502371cfcf65

Outline

9 Under the hood

10 HAL and Software Heritage

11 Software Heritage vs. other approaches

12 FAIR software

13 All the source code

14 More on the data model

Zenodo

- compare the Parmap library on Zenodo and on Software Heritage
- connector with InvenioRDM under development as part of FAIRCORE4EOSC

Outline

9 Under the hood

10 HAL and Software Heritage

11 Software Heritage vs. other approaches

12 FAIR software

13 All the source code

14 More on the data model

What about FAIR? (Findable, Accessible, Interoperable, Reusable)

FAIR data principles *for data*

in a nutshell: metadata, metadata, metadata all over the place (makes sense for data)

What about FAIR? (Findable, Accessible, Interoperable, Reusable)

FAIR data principles *for data*

in a nutshell: metadata, metadata, metadata all over the place (makes sense for data)

But software is *not data* ...

- any decent source code comes with reasonable metadata already: good principles have been there for decades
- the terms *interoperability* and *reusability* have precise technical meaning for software, and *differ significantly* from what is intended by the I and R of FAIR;
  - see the entries for software interoperability and software reusability
  - it is *very difficult* to achieve these properties even for commercial software developed by multi billion dollars corporations

What about FAIR? (Findable, Accessible, Interoperable, Reusable)

FAIR data principles *for data*

in a nutshell: metadata, metadata, metadata all over the place (makes sense for data)

But software is *not data* ...

- any decent source code comes with reasonable metadata already: good principles have been there for decades
- the terms *interoperability* and *reusability* have precise technical meaning for software, and *differ significantly* from what is intended by the I and R of FAIR;
  - see the entries for software interoperability and software reusability
  - it is *very difficult* to achieve these properties even for commercial software developed by multi billion dollars corporations
- let's focus on more actionable issues: ARDC a good starting point

Outline

9 Under the hood

10 HAL and Software Heritage

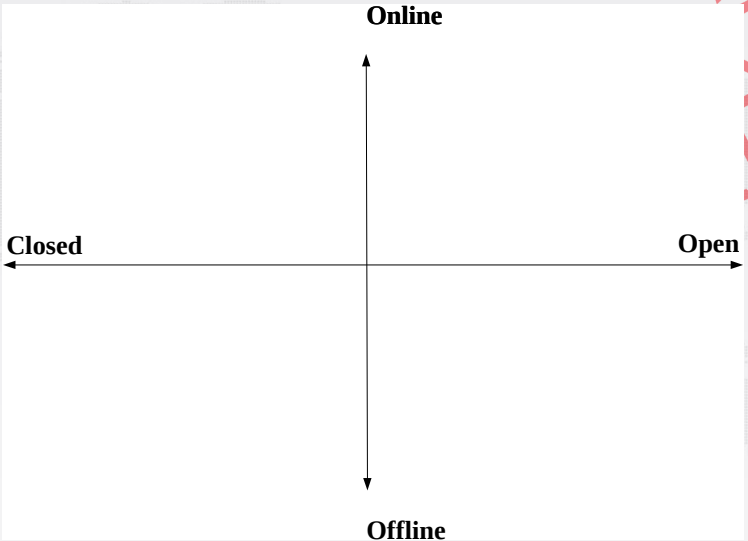
11 Software Heritage vs. other approaches

12 FAIR software

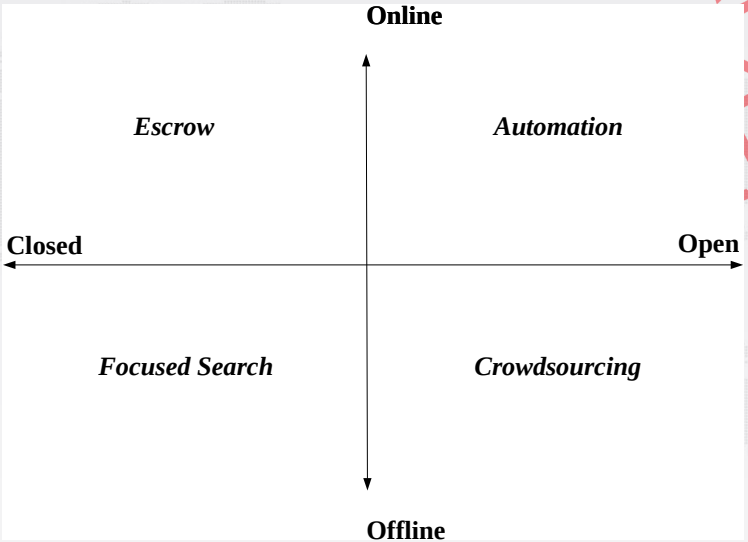
13 All the source code

14 More on the data model

All the source code!



All the source: strategies



Outline

9 Under the hood

10 HAL and Software Heritage

11 Software Heritage vs. other approaches

12 FAIR software

13 All the source code

14 More on the data model

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

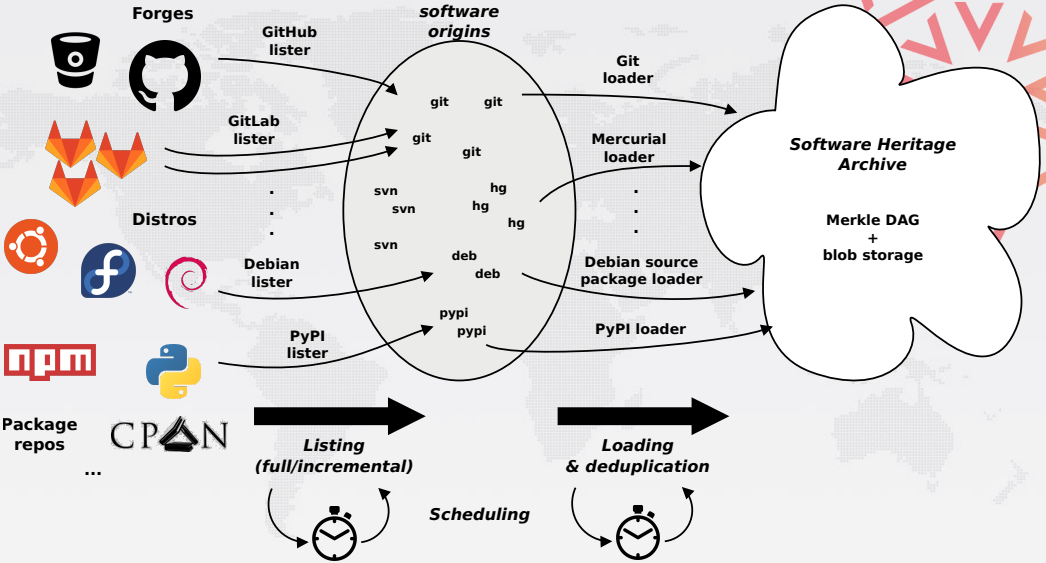
… in a VCS-/archive-agnostic canonical data model

We DON'T archive (yet)

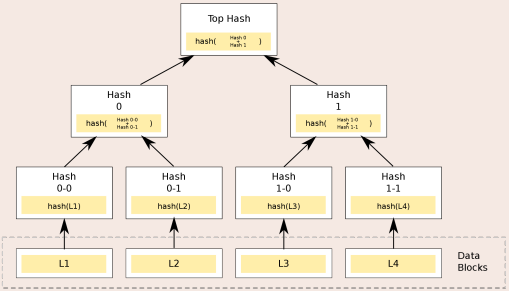
- homepages, wikis
- BTS/issues/code reviews/etc.
- mailing lists

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
...
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)

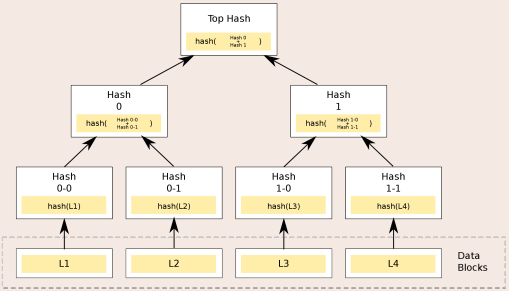


Combination of

- tree
- hash function

Merkle trees

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



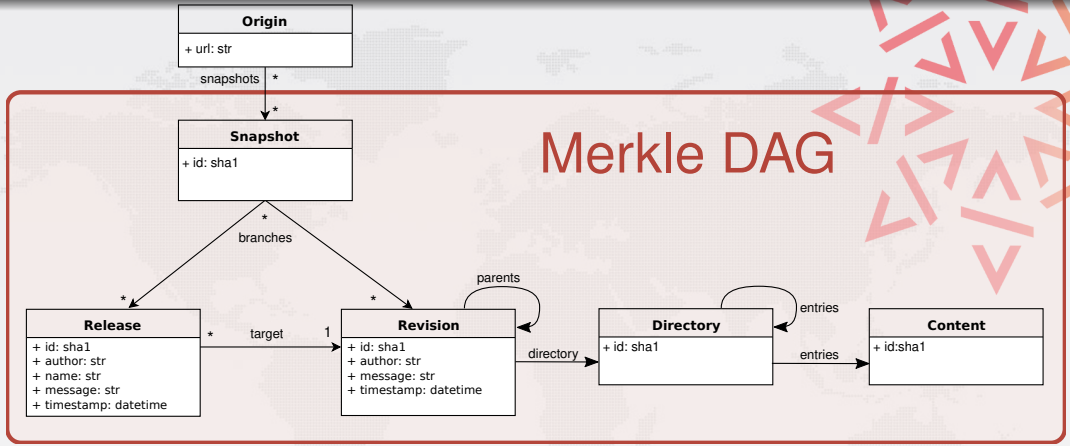
Combination of

- tree
- hash function

Classical cryptographic construction

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

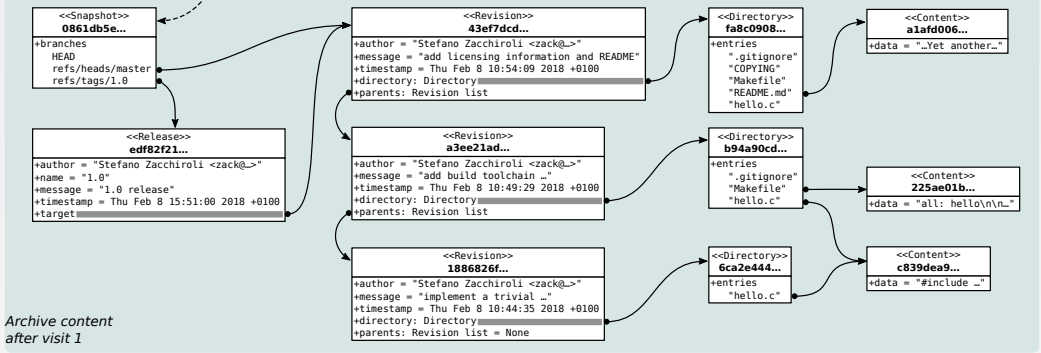
Data model



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

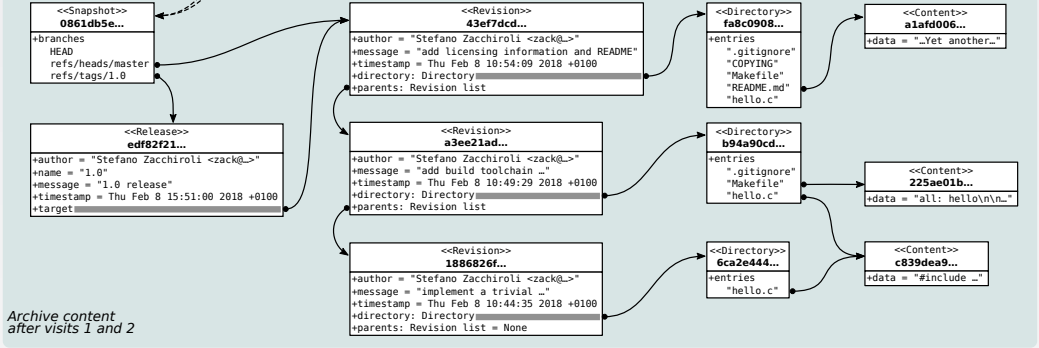
origin	visit	snapshot	timestamp
https://forge.softwareheritage.org/source/helloworld.git	1	0861db5e…	Fri Feb 9 12:38:45 2018 +0100



Archive content after visit 1

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