Archiving, referencing and attributing research software
towards software as a first class citizen

Roberto Di Cosmo
Seminaire Codes Sources, Paris

March 12, 2020
1. Software Source Code: a (forgotten) pillar of Science
2. Software Heritage
3. Archive and reference all the source code
4. Describe and cite research source code
5. The road ahead
Software source code: a precious part of our heritage

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 CODE580
TC BANKCALL  # PLEASE CRANK THE SILLY THING AROUND
CADR GOPERF1
TCF GOTOPOOH
TCF P63SPOT3  # PROCEED SEE IF HE’S LYING

P63SPOT4
TC BANKCALL # ENTER INITIALIZE LANDING RADAR
CADR SETPOS1
TC POSTJUMP  # OFF TO SEE THE WIZARD ...
```

Quake III source code (excerpt)

```
float Q_rsqrtn(float number)
{
  long i;
  float x2, y;
  const float threehalves = 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 * (threehalves - (x2 * y * y)); // 1st iteration
  // y = y * (threehalves - (x2 * y * y)); // 2nd iteration, this can be removed
  return y;
}
```

Len Shustek, Computer History Museum

“Source code provides a view into the mind of the designer.”
Source code is a *special* and endangered heritage

Software *evolves* over time

- projects may last decades
- the *development history* is key to its *understanding*

Complexity

- *millions* of lines of code
- large *web of dependencies*
  - easy to break, difficult to maintain
- sophisticated *developer communities*

Precious, endangered *Executable* and *human readable* knowledge

key people are passing away …

no organised effort to catalog and archive it (more later)
Software is everywhere in modern research

[...] software [...] essential in their fields.

*Top 100 papers (Nature, 2014)*

Sometimes, if you don’t have the software, you don’t have the data

*Christine Borgman, Paris, 2018*

**Nota bene**

The links in the picture are *essential*
The state of the art (in CS!) is far from ideal

ICSE (Zannier, Melrik, Maurer, 2006)
- complete absence of replication studies

ACM TOSEM 2001 to 2006
- 60% of all papers have tools: only 20% installable

Collberg’s 2015 study
- [http://reproducibility.cs.arizona.edu/](http://reproducibility.cs.arizona.edu/)
- 601 mainstream papers: 508 with tools, only 40% installable

Main reasons
- source code (or the right version of it) cannot be found
Where we stand

A wealth of initiatives!
- Policies: ACM Artifact Review and Badging, …
- Working groups: FORCE11, RDA, SPSO, …
- Metrics: Open Science Monitor (Elsevier!), …
- Repositories: FigShare, Zenodo, …

but …

Lack of recognition
- not (yet) a first class citizen
  - in the EOSC plan
  - in the scholarly works

Lack of proper guidance on how to
- archive and reference software
- choose a license
- cite a software project
A plurality of needs

**Researcher**
- archive and reference sw used in articles
- get credit for the software they develop
- verify/reproduce/improve results

**Laboratory/team**
- track software contributions
- produce up-to-date report/web page

**University/Research Organization**
- central view of research software assets
- tech transfer
- impact metrics
What is at stake in increasing order of difficulty

Archival
Research software artifacts must be properly archived
make it sure we can retrieve them (reproducibility)

Identification
Research software artifacts must be properly referenced
make it sure we can identify them (reproducibility)

Metadata
Research software artifacts must be properly described
make it easy to discover them (visibility)

Citation
Research software artifacts must be properly cited (not the same as referenced!)
to give credit to authors (evaluation!)
1. Software Source Code: a (forgotten) pillar of Science
2. Software Heritage
3. Archive and reference *all* the source code
4. Describe and cite *research* source code
5. The road ahead
Collect, preserve and share the source code of all the software

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

Reference catalog
find and reference all the source code

Universal archive
preserve all the source code

Research infrastructure
enable analysis of all the source code
An international, non profit initiative built for the long term

Sharing the vision

United Nations Educational, Scientific and Cultural Organization

And many more ...

www.softwareheritage.org/support/testimonials

Donors, members, sponsors

Platinum sponsors

Microsoft

INTEL

Société Générale

HUAWEI

Gold sponsor

Université de Paris

Silver sponsors

openInventionNetwork

Creative Commons

Open Source Initiative

Free Software Foundation

openInventionNetwork

Bronze sponsors

DANS

FLOSSID

Nokia Bell Labs

Université de Québec a Montréal
The largest software archive, a shared infrastructure

Source files: 6,197,000,081
Commits: 1,379,380,527
Projects: 90,231,104

Software Heritage
Global development history permanently archived in a unique git-like Merkle DAG

- ~400 TB (uncompressed) blobs, ~20 B nodes, ~280 B edges
Reference platform for *Big Code*

- **unique observatory** of all software development
- **big data, machine learning** paradise: classification, trends, coding patterns, code completion…

First datasets are available!

- full graph of software development (~20Bn nodes, ~200Bn edges) see Pietri, Spinellis, Zacchirolì, MSR 2019
  [https://dx.doi.org/10.1109/MSR.2019.00030](https://dx.doi.org/10.1109/MSR.2019.00030)
UNESCO, Inria, Software Heritage invite 40 international experts meet in Paris …

Their call is published on Feb 2019

It’s an important *policy tool*, already referenced and used … yes, you can sign it!

Outline

1. Software Source Code: a (forgotten) pillar of Science
2. Software Heritage
3. Archive and reference *all* the source code
4. Describe and cite *research* source code
5. The road ahead
Archive and reference

Software Heritage: a revolutionary infrastructure

- universal archive of all source code
  - we archive all software: both research and non research
  - we proactively collect software in a systematic way

- intrinsic identifiers for reproducibility
  - identify software artefacts without any third party
  - cryptographically strong, compatible with git hashes

Full guidelines available!  https://www.softwareheritage.org/save-and-reference-research-software/

Save code now … in just a few clicks

Demo

My 2012 Parmap paper before and after; other links: Apollo 11 (and blog), Quake III Arena
The SWH-ID schema

```
swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa
```

- `schema_version`
- `object_id`
- `prefix`
- `object_type`
The SWH-ID schema

```
swh:1:cnt:41dbb23118f92d7218099a5e7a990cf58f1d07fa
```

- `schema_version`: The version of the schema.
- `object_id`: The unique identifier of the object.
- `prefix`: The prefix defining the type of the object.
- `object_type`: The type of the object, with the following options:
  - "snp" - snapshot
  - "rel" - release
  - "rev" - revision
  - "dir" - directory
  - "cnt" - content
The SWH-ID schema

```
swh:1:cnt:41dbb23118f92d7218099a5e7a990cf58f1d07fa
```

- **prefix**: swh
- **object_type**: cnt
- **object_id**: 41dbb23118f92d7218099a5e7a990cf58f1d07fa

- **lines_ctx**: ;lines=64-72
- **origin_ctx**: ;origin=https://github.com/chrislgarry/Apollo-11

```
"snp" - snapshot
"rel" - release
"rev" - revision
"dir" - directory
"cnt" - content
```
A worked example
A worked example

Contents

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies of
this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom
to share and change all versions of a program--to make sure it remains
free software for all its users. We, the Free Software Foundation, use
the GNU General Public License for most of our software; it applies also
to any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to make restrictions that prevent
proprietariness from degrading the freedom of others.

sha1: 8624bcdae55baeef...
sha256: 8ceb4b9ee5aded...
sha1_git: 94a9ed024d385...
length: 35147
A worked example

Directories

```
100644 blob c5baade4c44766042186ef858c0fd63d587efb09 .gitignore
100644 blob 2d6a34af6f52cf3cf6b6c2f7bd0648fbd255e7f AUTHORS
100644 blob 94a9ed824d3859793618152ea559a1688bbcb5e2 LICENSE
100644 blob d9b26e5a435a43f8a79a84e0867751dfb95c7bb MANIFEST.in
100644 blob 524175c2bad0b35b975f79284c2f5a6d5eaf2eb4 Makefile
100644 blob 5c7e3a35b4dbb038682ba7793f440492ed9678bb3 Makefile.local
100644 blob 8617980629c2d4e6080404f09aa749b685b3e67b README.db_testing
100644 blob 76b29f94c815e0869c414d38d78d7ce08ec514e README.dev
040000 tree ele10ecef948af0b93adb0872afec89f12e92618a bin
040000 tree 83e56d8beaf7793c77a45a345c80fcb8af503013 debian
040000 tree a34c9c4ba213f0c6dc67f9816348d2795557af5 docs
100644 blob f2a6d32c6135aa7287bd76167b01df2ae4f1539 requirements.txt
100755 blob eee147c36caflbc2d820da8dc026cb5b68180c setup.py
040000 tree 224bb4c1f4c67fca1d160bffd2d0694e7e1abf3 sql
040000 tree 8631c9cd77bb093168107ab5ba51f40c6300e swl
040000 tree 8fb905b56ba8ed692f1209b2773b474c61d66c1 utils
```

id: 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d
A worked example
Revisions

<table>
<thead>
<tr>
<th>Details</th>
<th>Changes</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHA: 963634dca6ba5dc37e3ee426ba091092c267f9f6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Author: Nicolas Dandrimont <a href="mailto:nicolas@dandrimont.eu">nicolas@dandrimont.eu</a> (Thu Sep 1 14:26:13 2018)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Committer: Nicolas Dandrimont <a href="mailto:nicolas@dandrimont.eu">nicolas@dandrimont.eu</a> (Thu Sep 1 14:26:13 2018)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subject: provenance.tasks: add the revision -&gt; origin cache task</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parent: fc3a8b59ca1df424d860f2c29ab07fee4dc35d10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>: test...storage: properly pipeline origin and cont...provenance.tasks: add the revision -&gt; origin cache task</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

swf/storage/provenance/tasks.py

515f00d44e92c65322aaa9bf3fa097c00dab9c7d
parent: fc3a8b59ca1df424d860f2c29ab07fee4dc35d10
author Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200
committer Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200

provenance.tasks: add the revision -> origin cache task

id: 963634dca6ba5dc37e3ee426ba091092c267f9f6
A worked example
A worked example

**Releases**

- Add new metadata column to origin_visit
- Update swh-add-directory script for updated API

--- BEGIN PGP SIGNATURE ---

id: 85083a5cc14a441c89dea73f5bdf67c3f9c6afdb

--- END PGP SIGNATURE ---
A worked example
A worked example

Snapshots

commit 08fbeb2577010952e3b3eb2c21e91466c53a1d9158 refs/heads/atime
commits b5a443a24e9f9f3e3b46c292cc4f4c6e61c67eb refs/heads/directory.listing-arrays
commits 0f96d0b4f92338f1f5869b027fbc1b5f67238269c5 refs/heads/TODO
commits c7f7e9b9aebe2b22b4698f05a8631f67e60e808 refs/heads/master
commits 7ceca19f7e6d2d284047e54b1ed9e8b4a4361a8fc2 refs/heads/tmp-directory-addr
commits 6422a2853f85e585b5a560427b53e4e4f302252e62e refs/heads/tmp/generic-releases
tag 29019e3b203e5178c7f6896559779074697c6f7f2555 refs/tags/v0.1

tag 72a2193da384e35996d68b76b70bb072aee2cd refs/tags/v0.2.10

tag 3599e7a6e8eb6b786b337667695f28bb83bf4f5c5c refs/tags/v0.11

tag 33738427a4483b35a4777780d99676f4d5556 refs/tags/v0.3.12

tag 86f74652753b327cf590311c2b9a03c6f3b4b3d refs/tags/v0.13
	tag 5a6325fe686d4b5c8f442867592a1c2f2b3d refs/tags/v0.14
	tag 586f2ba4580b4f75fb08559307643cb3a1a9c7f refs/tags/v0.15
	tag 8ed88bf5f4898b363177742b289f6605eb51c refs/tags/v0.16
	tag a542444e33f9beb3ce8f32b7f3e9335899c2bdc refs/tags/v0.17
	tag 22827f5590d1222e5655546e2e06f299f3d9 refs/tags/v0.18
	tag 66979a4ca85d497fc0d24aad09dce82536e4f7c refs/tags/v0.19
	tag 32f5a59fca32ca3baad65e15a51a3d23ec75a67 refs/tags/v0.20
	tag 3174c3d31cc46c7f49279801e096b1237e0d2f2c7 refs/tags/v0.21
	tag 215ea0dab11ee82ee0726e6b4e6873a87908 refs/tags/v0.21
	tag 3fb188c275a2d82521124257a1d5dfce85ffaf1df refs/tags/v0.22
	tag 8c0bee80a4d73f5a262789e4e6ba16ac5c72aba4 refs/tags/v0.23

...
Zoom on the trust model for identifiers

Trust model for usual DOIs

- DOI
- DOI resolver without checksums
- sw URL
- integrity verifier
- sw & checksum

Trust model for DOIs with checksums

- DOI
- DOI resolver with checksums
- sw URL & checksum
- integrity verifier

Trust model for SWH-IDs

- SWH
- sw archive
- integrity verifier
1. Software Source Code: a (forgotten) pillar of Science
2. Software Heritage
3. Archive and reference *all* the source code
4. Describe and cite *research* source code
5. The road ahead
Context

Many articles/guidelines
- reproducibility
- archival
- credit and evaluation

Most common limitations
- software is ‘just data’
- citation = reference = DOIs
- citation produced by automated tools

A few remarkable exceptions
- ASCL (since 1999): metadata only, carefully curated
- geodynamics.org: source, documentation, metadata
- swmath.org: software catalog via articles

Software Citation WG at Inria (since 10/2018)
- leverage a 50 year experience, make recommendations
- read more https://hal.archives-ouvertes.fr/hal-02135891
Why it is not simple

Software is complex

- **Structure**: monolithic/composite; self-contained/external dependencies
- **Lifetime**: one-shot/long term
- **Community**: one man/one team/distributed community
- **Authorship**: complex set of roles *(more later)*
- **Authority**: institutions/organizations/communities/single person

Various granularities

- **Exact status of the source code** for reproducibility, e.g.
  
  “you can find at `swh:1:cnt:cdf19c4487c43c76f3612557d4dc61f9131790a4;lines=146-187` the core algorithm used in this article”

- **(Major) release** “This functionality is available in OCaml version 4”

- **Project** “Inria has created OCaml and Scikit-Learn”.
## Proposals for the scholarly world

### Refined ontology for contributors
- Design, Architecture,
- Coding, Testing, Debugging,
- Documentation, Maintenance, Support,
- Management

see also CRediT, Geodynamics

### Reference is distinct from citation
- **Reference** is for *reproducibility*
- **Citation** is for *credit*

They must not be conflated

Beware of the numbers game:

… do we really want an *s-index*?

### Keep the human in the loop

When *credit* is at stake, automation/crowdsourcing is not enough!

Humans *are needed* to get *quality information*
First steps with HAL / Software Heritage

How it works, what is special

**Generic mechanism:**
- SWORD based
- review process
- versioning

**Today:** deposit .zip or .tar.gz file ([guide](https://hal.archives-ouvertes.fr/hal-01872189))

**Tomorrow:** just provide the SWH id

Deposit/describe research software in HAL

- **author:** [https://hal.archives-ouvertes.fr/hal-01872189](https://hal.archives-ouvertes.fr/hal-01872189)
- **moderator:** [https://hal.archives-ouvertes.fr/hal-01876705](https://hal.archives-ouvertes.fr/hal-01876705)

Examples

LinBox, SLALOM, Givaro, NS2DDV, SumGra, Coq proof, …
The swmath.org approach

Article based citation

See for example:

- SemiPar on swmath.org
Outline

1. Software Source Code: a (forgotten) pillar of Science
2. Software Heritage
3. Archive and reference all the source code
4. Describe and cite research source code
5. The road ahead
Conclusion

Research software

- pillar of open science
- finally in the limelight

Doing it right is not easy

- simplistic approaches, "just data", ...
- soon part of research evaluation

You can help make a change

- leverage Software Heritage in conferences and journals for archival and reference
- join the conversation on software citation and software evaluation criteria
- tackle the scientific problems: big code, classification, infrastructure, etc.

Thank you!

Jean-François Abramatic, Roberto Di Cosmo, Stefano Zacchioli
Building the Universal Archive of Source Code
Communications of the ACM, October 2018

Roberto Di Cosmo, Morane Gruenpeter, Stefano Zacchioli
Identifiers for Digital Objects: the Case of Software Source Code Preservation