Archiving And Referencing All Software Source Code

Roberto Di Cosmo

February 19th, 2020
Short Bio: Roberto Di Cosmo

Computer Science professor in Paris, now working at INRIA

- 30 years of research (Theor. CS, Programming, Software Engineering, Erdos #: 3)
- 20 years of Free and Open Source Software
- 10 years building and directing structures for the common good

1999 *DemoLinux* – first live GNU/Linux distro

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

2008 *Mancoosi project* [www.mancoosi.org](http://www.mancoosi.org)

2010 *IRILL* [www.irill.org](http://www.irill.org)

2015 *Software Heritage* at INRIA

2018 *National Committee for Open Science*, France
The knowledge is in the *source code*

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

**Hello World**

<table>
<thead>
<tr>
<th>Program (excerpt of binary)</th>
<th>Program (source code)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4004e6: 55</td>
<td>/* Hello World program */</td>
</tr>
<tr>
<td>4004e7: 48 89 e5</td>
<td><code>#include&lt;stdio.h&gt;</code></td>
</tr>
<tr>
<td>4004ea: bf 84 05 40 00</td>
<td><code>void main()</code></td>
</tr>
<tr>
<td>4004ef: b8 00 00 00 00</td>
<td><code>{</code></td>
</tr>
<tr>
<td>4004f4: e8 c7 fe ff ff</td>
<td><code>printf(&quot;Hello World&quot;);</code></td>
</tr>
<tr>
<td>4004f9: 90</td>
<td><code>}</code></td>
</tr>
<tr>
<td>4004fa: 5d</td>
<td></td>
</tr>
<tr>
<td>4004fb: c3</td>
<td></td>
</tr>
</tbody>
</table>
Source code is special

Executable and human readable knowledge
“Programs must be written for people to read, and only incidentally for machines to execute.”
Harold Abelson

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

.python3-matplotlib
.python3-dateutil
.python3-six
(>= 1.4)
.python3:any
.python-matplotlib-data
(>= ... 0.5.1)
Matplotlib library
Python dependencies
Real dependencies
Fake OS dependencies induced by package granularity

Matplotlib library
Python dependencies
Real dependencies
Fake OS dependencies induced by package granularity
Outline

1. Introduction
2. Academia’s evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?
Software is everywhere in modern science

[...] the vast majority describe [...] or software that have become 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*
… a forgotten pillar of Open Science

Lack of recognition

not (yet) a first class citizen
- in the EOSC plan
- in the scholarly world

Lack of consensus on how to

- *archive* software
- *choose* a license
- *cite* a software project
Pressure to make the source code available is raising

**Why**

Necessary to...

- *reproduce* and *verify*,
- *modify* and *evolve*, building new experiments from old ones

**When and where**

- debate started end of first 2000 decade (bio, statistics, medicine…)
- growing in Computer Science since the ESEC/FSE 2011 Artifact Evaluation Award

**A wealth of initiatives…**

- Policies: ACM Artifact Review and Badging, AEC, …
- Working groups: FORCE11, RDA, SPSO, …
- Repositories: FigShare, Zenodo, …
- Common infrastructures: Software Heritage
<table>
<thead>
<tr>
<th>What is at stake</th>
<th>in increasing order of difficulty</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Archival</strong></td>
<td>Research software artifacts must be properly <strong>archived</strong> make it sure we can <strong>retrieve</strong> them (<em>reproducibility</em>)</td>
</tr>
<tr>
<td><strong>Identification</strong></td>
<td>Research software artifacts must be properly <strong>referenced</strong> make it sure we can <strong>identify</strong> them (<em>reproducibility</em>)</td>
</tr>
<tr>
<td><strong>Metadata</strong></td>
<td>Research software artifacts must be properly <strong>described</strong> make it easy to <strong>discover</strong> them (<em>visibility</em>)</td>
</tr>
<tr>
<td><strong>Citation</strong></td>
<td>Research software artifacts must be properly <strong>cited</strong> (<em>not the same as referenced!</em>) to give <strong>credit</strong> to authors (<em>evaluation!</em></td>
</tr>
</tbody>
</table>

Let’s focus on the first two!
1. Introduction
2. Academia’s evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?
Software Heritage in a nutshell

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
**All the software source code**

<table>
<thead>
<tr>
<th>Source files</th>
<th>Commits</th>
<th>Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>6,197,000,081</td>
<td>1,379,380,527</td>
<td>90,231,104</td>
</tr>
</tbody>
</table>

The largest software source code archive *ever*

**Uniform and intrinsic** identifiers for reproducibility

Tracking over 20 billion *software artifacts*, and counting...  
/bit.ly/swhpipaper

**Adoption highlights**

- reference archive for *swmath.org*, HAL, etalab
- part of the french National Plan for Open Science
1. Introduction
2. Academia’s evolving practice
3. Archiving and referencing *all* the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?
Modern software development

Version control system (VCS)

- records changes made to a (set of) source code file(s)
- allows to operate on versions: diff/merge/fork/recover etc.
- essential tool for software development

Three decades of evolution
Intrinsic identifiers for modern software development

Requirements for the D in DVCS

- **intrinsic** unique identifiers… (here: *cryptographic signature*, aka "hash")
- … that work for tree structures (software directories)

Merkle tree to the rescue (R. C. Merkle, Crypto 1979)

Combination of
- tree
- hash function
The SWH-ID schema: syntax and semantics

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

- `schema_version`: 1
- `object_id`: cnt
- `prefix`: swh
- `object_type`: 41ddb23118f92d7218099a5e7a990cf58f1d07fa
The SWH-ID schema: syntax and semantics

swh:1:cnt:41dbb23118f92d7218099a5e7a990cf58f1d07fa

prefix object_type

schema_version object_id

"snp" - snapshot
"rel" - release
"rev" - revision
"dir" - directory
"cnt" - content
The SWH-ID schema: syntax and semantics

```
swh:1:cnt:41dcb23118f92d7218099a5e7a990cf58f1d07fa
```

**prefix**

- `swh:`
- `1:`
- `cnt:`

**object_type**

- "snip" - snapshot
- "rel" - release
- "rev" - revision
- "dir" - directory
- "cnt" - content

**object_id**

- `41dcb23118f92d7218099a5e7a990cf58f1d07fa`

**lines_ctx**

- `;lines=64-72`

**origin_ctx**

- `;origin=https://github.com/chrislgarry/Apollo-11`
Walkthrough the Parmap article

6. Conclusions

Parmap is a minimalistic library allowing to exploit multi-core architecture for OCaml programs. It has been designed with the goal of providing parallel map and reduce to OCaml programmers in a fairly natural way, such that the "minimal disruption" principle stated by Cohe in his skeleton manifesto paper is enforced. In fact, in order to use Parmap, it is sufficient to substitute the calls to List functions with calls to the equivalent Parmap functions. The clean and efficient implementation of Parmap is such that nearly optimal speedups are achieved on state-of-the-art multi-core architectures when suitable grain computations are parallelized. The full source code of the Parmap library is available under the LGPL licence from [https://gitorious.org/parmap](https://gitorious.org/parmap), and is now also incorporated in the GODI installation system for OCaml libraries.


Only 8 years later!
Figure 1 in [Danelutto and Di Cosmo, 2012]

Parmap’s implementation of the distribution, fork, and recollection phase

```
1  let simplestmapper scores compute opid al combine =
2     (* init task parameters *)
3     let ln = Array.length al in
4     let chunksize = ln/ncores in
5     (* create descriptors to mmap *)
6     let fdarr=Array.init ncores (fun _ -> tempfd()) in
7     (* spawn children *)
8     for i = 0 to ncores-1 do
9         match Unix.fork () with
10            0 -> (* children code: compute on the chunk *)
11                (let lom=chunksize in
12                 let hi=if i=ncores-1 then ln-1 else (i+1)*chunksize-1 in
13                 let v = compute al le hi opid in
14                 marshal fdarr.(i).v) ;
15                exit 0;
16            _ -> failwith "Fork error"
17     done;
18     pid -> ();
19     done:
20     (* wait for all children *)
21     for i = 0 to ncores-1 do ignore (Unix.wait () done;
22     (* read in all data *)
23     let res = ref [] in
24      (* accumulate the results in the right order *)
25     for i = 0 to ncores-1 do
26         res := (marshal fdarr.(scores-1-i-1));!
27     done;
28      (* combine results *)
29     combine !res;
```

http://doi.org/10.5281/zenodo.3630224
Zoom on the trust model for identifiers

Trust model for usual DOIs

Trust model for DOIs with checksums

Trust model for SWH-IDs
Outline

1. Introduction
2. Academia’s evolving practice
3. Archiving and referencing all the source code: Software Heritage
4. Zoom on the SWH-ID
5. Practical guidelines for archiving and referencing
6. What about metadata and citation?
Prepare your software source code

Prepare your public repository with:

- README, LICENSE, AUTHORS & codemeta.json files

What’s a good README

extracted from Eric Steven Raymond and Make a README

*MUST* include:

- Name and a description of the software.

*SHOULD* include:

- how to run and use the source code
- build environment, installation, requirements

*CAN* include:

- project website or documentation pointer and recent news
- visuals
Submit save request on SWH

Save code now on https://archive.softwareheritage.org/save/

- git, svn or mercurial
- intrinsic metadata files
- complete history
Reference software artifacts in your articles

Choose the granularity level for the reference:

**file (with code fragment)**

```
swh:1:cnt:c60366bc03936eede6509b23307321faf1035e23;lines=473-537
... and add ;origin=https://github.com/sagemath/sage/
```

James McCaffrey’s **algorithm** in sageMath

**directory**

```
swh:1:dir:c6f07c2173a458d098de45d4c459a8f1916d900f
... and add ;origin=https://github.com/id-Software/Quake-III-Arena/
```

source code of **Quake-III Arena** from id-Software
Reference software artifacts in your articles

**specific release**

```text
swh:1:rel:22ece559cc7cc2364ecd5e5593d63ae8bd229f9f
... and add ;origin=https://github.com/darktable-org/darktable/
```

**release** 2.3.0 of Darktable, dated 24 December 2016

**full snapshot (including all branches and all releases)**

```text
swh:1:snp:c7c108084bc0bf3d81436bf980b46e98bd338453
... and add ;origin=https://github.com/darktable-org/darktable/
```

**a snapshot** of the entire Darktable repository (4 May 2017, GitHub)
1. Introduction

2. Academia’s evolving practice

3. Archiving and referencing *all* the source code: Software Heritage

4. Zoom on the SWH-ID

5. Practical guidelines for archiving and referencing

6. What about metadata and citation?
It’s more complex than it seems!

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
- **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 metadata and citation in the scholarly world

Refined ontology for contributors

- Design, Architecture,
- Coding, Testing, Debugging,
- Documentation, Maintenance, Support,
- Management

Reference is distinct from citation

- **Reference** is for *reproducibility* and now we can get it right!
- **Citation** is for *credit* and the jury is still out…
  They must not be conflated

Keep the human in the loop

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

Humans *are needed* to get *quality information*

Experiments are ongoing on *moderated* software deposit … (IDCC 2020)

*Curated Archiving of Research Software Artifacts: lessons learned from the French open archive (HAL) https://hal.archives-ouvertes.fr/hal-02475835v1*
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

Where can you participate?
- Software Source Code Interest group - RDA-SSC IG
- Software Source code Identification Working Group - RDA-Force11-SCID WG
- Software Citation Implementation Working Group - Force11-SCIWG
References

Jean-François Abramatic, Roberto Di Cosmo, Stefano Zacchiroli
*Building the Universal Archive of Source Code*, CACM, October 2018 (10.1145/3183558)

Roberto Di Cosmo, Morane Gruenpeter, Stefano Zacchiroli
*Referencing Source Code Artifacts: a Separate Concern in Software Citation*, CiSE 2020 (10.1109/MCSE.2019.2963148) (hal-02446202)

Pierre Alliez, Roberto Di Cosmo, Benjamin Guedj, Alain Girault, Mohand-Said Hacid, Arnaud Legrand and Nicolas Rougier
*Attributing and referencing (research) software: Best practices and outlook from Inria*, CiSE 2020 (10.1109/MCSE.2019.2949413) (hal-02135891)
Worked example Merkle tree
A worked example

[Diagram showing relationships between snapshots, releases, revisions, directories, and contents]
A worked example
A worked example

Directories

```
100644 blob c5baade4c44766042186ef858c0fd63d587ebf09 .gitignore
100644 blob 2d6a34af6f52cf3cf6b0c27bd0648fbd255e77f AUTHORS
100644 blob 94a9ed824d3859793618152ea559a168bbcb5e2 LICENSE
100644 blob d9b2665a435a43f8a79a84e0887751dfb9957cb Manifest.in
100644 blob 524175c2bad0b35bb975f79284c27a5d965eaf2eb4 Makefile
100644 blob 5c7e3a5b5bddd038682ba7793f440492ed96788b3 Makefile.local
100644 blob 8617980829c2d4e680804f09aa749b85b3e67b README.db_testing
100644 blob 76b29f94cc8f15e8869c414d38d7b7ce08ec514e README.dev
040000 tree e1e10ecef948af0b93adb0372af89f12e92618a bin
040000 tree 83e56d0beaf7793c7a45a345c80fc8b8af503013 debian
040000 tree a34c9c4ba213f9cedc67f98163348d27955577af5 docs
100644 blob f2a6d32c6135aa7287bb76167b01df2aae4f1539 requirements.txt
100755 blob eee147c36caflbbc2d820da8dc026cb56b68180c setup.py
040000 tree 224bb4c1f4c67fca1d606bdf2d06094e71e1abf3 sql
040000 tree 8631c9cd77be993168170ab5baf51f40c6300be swh
040000 tree 8fb905b56ba8ed692f1209b2773b474c6cd66c1 utils
```

id: 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d
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 2020)</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 2020)</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>provenance.tasks: add the revision -&gt; origin cache task</td>
<td></td>
<td></td>
</tr>
<tr>
<td>swh/storage/provenance/tasks.py</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
515f00d44e92c65322aaa9bf3fa097c00ddb9c7d
tree 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d
cf3a8b59ca1df424d860f2c29ab07fee4dc35d10
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

object cbcf916b1e134f593e7567570a1761b156e6eb1d
  type commit
  tag v0.0.51
  tagger Nicolas Dandrimont <nicolas@dandrimont.eu> 1472042163 +0200

Release swh.storage v0.0.51
  - Add new metadata column to origin_visit
  - Update swh-add-directory script for updated API
    ---- BEGIN PG SIGNATURE -----
    IQIzBAAABCAdLBQJxTZNFIhxxaWNvbGFQfQRHbmvRyaWIvbQZUXICgqQ7AULMo2+
    neqprwtae6509555jze4KWNjYXOGV5=+1K1v3Evl5WNKww8KJ7xX2sEILD7Tuf
    ahp76j2q3i8n16ac1+yR8fFch3L7yYtrcZeWXXWQrKw4wNMAIoF1Tb8qahw8AAd5t2
    ICBI2jXUcGR993eKFWzw2ZXY=+b89pmW3y3Dr6jW777K4MuJPQgyhyHP75yo
    IGE9Enwno7V1y1V5d13n5Q75mXRaqA+Becq6dub7Z2xjy+jiUqC9cyhV3hirr/FL
    q627mukyzyH86kLy+1jv+isDwBlnPrb55THOjyi5jVXpK/zh3SFqRuiB3ZFKcaco
    klj6kAwV7UB0Meb=n+kV7eJbrR3+iYWBF3jQpa51/V0qOTN761dALcMN6eAKcKOMkt
    d9JMax11i+j9yEDHqszW7Q6jDwKPK PhhVLQ3o3V3GauQ7Trn1qpcMc006h9FAw2C
    GgK1pDHF4xOj454wP7Zyy0U2VXGdUsxUW9rFQ2ZkWjww=ZcZmcdk9drj3UOMm
    RpTTU8SUXaXHIGqUpX7YTvpy1pDcP7c6USKQD4eG4AvZkm3kl0GmXwvXCVQPhyO
    n755h8BNHMvOyEfe75T9jUb1YK7t0pFTRUGXWDwKQwXKXOZGK7x6faj0z9
    gJj6Qf72QWQ85domQAL+HvPakVyc3MepUhhs2ePhzElhVux=
    =kOXp
    ---- END PG SIGNATURE -----

id: 85083a5ccc14a441c89dea73f5bdf67c3f9c6afdb
A worked example
Snapshots

commit 08ffeb2577010952eb3ce21691466c53a1d9158 refs/heads/atime
commit ba5443a2e43f4e32e3b46c292ce4f4e61c6e2eb refs/heads/directory-listing-arrays
commit d69e6a9f82938f5e569b27fb1e05672738d9c5 refs/heads/faq
commit c7f7e9eae0eb22b28469b975a8e13e67f62e468e08 refs/heads/master
commit 7ca197f7c6dd26b26a047e54b1ed9e8b44361a8f2c refs/heads/tmp/tmp-addr
commit 0422a2b5f15d65085a65d042b7534ee4f025252e02e refs/heads/tmp/generic-releases
commit 29104b3c1378c7f8696597790f6d6f97c727f755 refs/tags/v0.1

tag 72a21991a38ae5399680b867fb9b0ee572ee2cd refs/tags/v0.1.0

tag 35996eac6be687b856376765a23b2b5f4f4fa5c5 refs/tags/v0.1.1

tag 333784274aa483ba59ea7777808e667f0c6554 refs/tags/v0.1.2

tag 65f74652753b27c5f9c0311c2bfaf036c3f4b34d5 refs/tags/v0.1.3

tag 5a6325fe86dbb4b5861e442687d92a11e32f3b3d refs/tags/v0.1.4

tag 58efba4d5b8b4d5f4a8959307643cb1a9c7f refs/tags/v0.1.5

tag 8cd8bb85f4908b363177742bd289f6e9e0b5e51c refs/tags/v0.1.6

tag a542a446e347f0be3c5eb2a6e538b90a7c7d6 refs/tags/v0.1.7

tag 22ba2f5f59d1d222e56555462e1e9c2e60f2993d9 refs/tags/v0.1.8

tag 696979a4ca5d4547fc8d2a4a0d0c8e283563e47c refs/tags/v0.1.9

tag 32bf5a59f2ca2f3badae5ff15a55ead382ec275a67 refs/tags/v0.1.10

tag 3147fc3d3d4dc6fe64927f801e96b1237ebdf2f7c refs/tags/v0.1.11

tag 215ea5dabab11ee82de072e76eb4b678a387980 refs/tags/v0.1.12

tag 3fb18c2872a5d8252124257a1a5d4c8f5fa1df refs/tags/v0.1.13

tag 8c0bee8d0a4d73f5c526789e4e0a16ac3c72aba4 refs/tags/v0.1.14

id: b464cad1b66ff266a37b46ea6e7a04b545e904b