Logiciels et Science Ouverte: enjeux et opportunites

Roberto Di Cosmo

November 19th, 2019

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Outline

1. Introduction

2. Archival and reference: use Software Heritage!

3. The road ahead

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

2015 *Software Heritage* at INRIA

2018 *National Committee for Open Science*, France

Source code is *special*

Executable and *human readable* knowledge copyright law

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

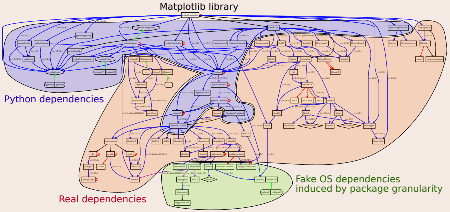
Harold Abelson

Software *evolves* over time

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

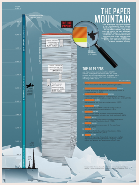
Complexity

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



Software Source code: pillar of Open Science

Software is everywhere in modern research



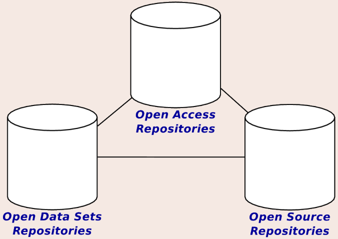
[...] software [...] essential in their fields.

Top 100 papers (Nature, 2014)

Sometimes, if you dont have the software, you dont have the data

Christine Borgman, Paris, 2018

Open Science: three pillars



Nota bene

The links in the picture are essential

Where we stand

A wealth of initiatives!

- Policies: ACM Artifact Review and Badging, …
- Working groups: FORCE11, RDA, SPSO, …
- Metrics: Open Science Monitor (Elsevier!), …
- Journals: IPOL, ReScience, InsightJournal, eLife, ACM DL, …
- 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

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*!)

What is at stake in increasing order of difficulty, cont'd

Sustainability

Organisational schemas, legal tools, ecomonic models, processes and policies to ensure research software can be maintained and sustained over time

Technology transfer and industry collaboration

Approaches, support, methods, processes to establish connections with industry in order to foster uptake and transfer of research software

Outline

1. Introduction
2. Archival and reference: use Software Heritage!
3. The road ahead

Software Heritage in a nutshell

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share the *source code* of *all the software*

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

Reference catalog

Sourceforge
Debian
CPAN
Gitorious
Maven
Inria
Bitbucket
GitHub
BerliOs
CTAN
GitLab
CRAN
GoogleCode
Adullact

find and reference all the source code

Universal archive

damage
disaster
malicious
media
aging
tear
attack
obsolete
dependencies
reference
storage
deletion
dangling
wear
corruption
encryption
format

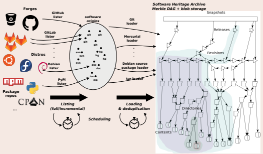
preserve all the source code

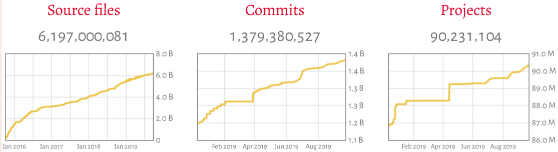
Research infrastructure

enable analysis of all the source code

Highlights

The largest software source code archive *ever*





20 billions intrinsic identifiers for reproducibility

See DIO vs IDO in `bit.ly/swhpidpaper`

Reference archive

See the work done at *swmath.org*

SWH IDs now a standard for Wikidata

See `https://www.wikidata.org/wiki/Property:P6138`

Policy

Now part of the *French National Plan for Open Science*

Archive and reference

Guidelines

https://www.softwareheritage.org/save-and-reference-research-software/

Save code now

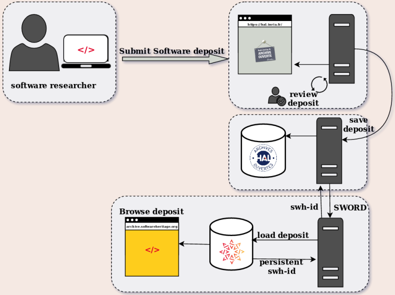
… just a few clicks!

Demo

live…

Describe and cite

How it works, what is special



Generic mechanism:

- SWORD based
- review process
- versioning
- today: deposit .zip or .tar.gz file (*guide*)
- tomorrow:
  - just provide *SWH id*, metadata extraction

Deposit/describe research software in HAL

- author: https://hal.archives-ouvertes.fr/hal-01872189
- moderator: https://hal.archives-ouvertes.fr/hal-01876705

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*

Outline

1. Introduction

2. Archival and reference: use Software Heritage!

3. 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
- join the SPSO GPLO: https://www.ouvrirlascience.fr/logiciels-libres-et-open-source/

Thank you!

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

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

Appendix

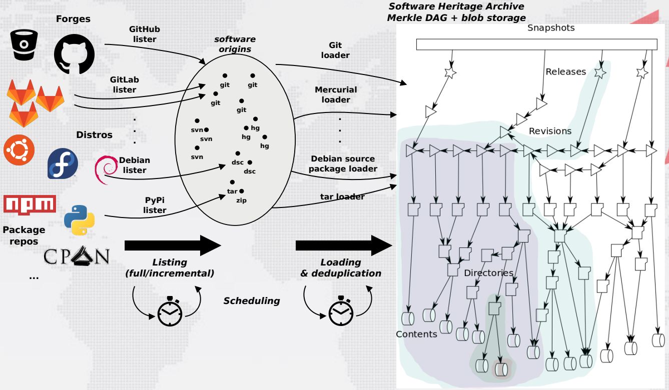
Outline

4 Under the hood

5 Identifiers are not easy

6 Looking for the right identifiers

Automation, and storage



- full development history permanently archived!

Outline

4 Under the hood

5 Identifiers are not easy

6 Looking for the right identifiers

URL decay disrupts the *web of reference*

Web links *are not* permanent (even *permalinks*)

there is no general guarantee that a URL… which at one time points to a given object continues to do so

T. Berners-Lee et al. Uniform Resource Locators. RFC 1738.

URLs used in articles *decay*!

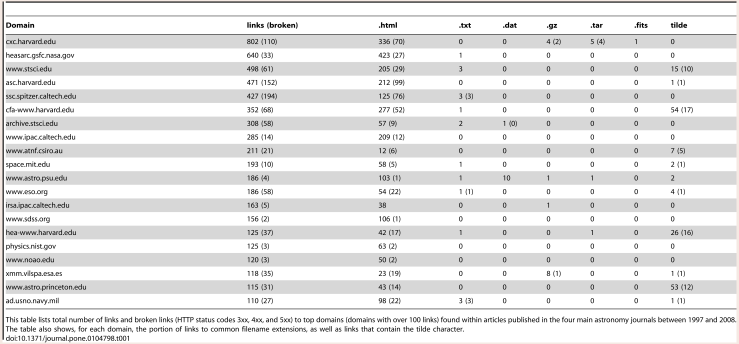
Analysis of *IEEE Computer* (Computer), and the *Communications of the ACM* (CACM): 1995-1999

- the *half-life* of a referenced URL *is approximately 4 years* from its publication date
  D. Spinellis. The Decay and Failures of URL References.
  Communications of the ACM, 46(1):71-77, January 2003.

Similar findings in Lawrence, S. et al. *Persistence of Web References in Scientific Research*, IEEE Computer, 34(2), pp. 26–31, 2001.

Scholar roster of broken links

An example from Astronomy



Domain	links (broken)	.html	.txt	.dat	.gz	.tar	.fits	tilde
cxc.harvard.edu	802 (110)	336 (70)	0	0	4 (2)	5 (4)	1	0
heasarc.gsfc.nasa.gov	640 (33)	423 (27)	1	0	0	0	0	0
www.stsci.edu	498 (61)	205 (29)	3	0	0	0	0	15 (10)
asc.harvard.edu	471 (152)	212 (99)	0	0	0	0	0	1 (1)
ssc.spitzer.caltech.edu	427 (194)	125 (76)	3 (3)	0	0	0	0	0
cfa-www.harvard.edu	352 (68)	277 (52)	1	0	0	0	0	54 (17)
archive.stsci.edu	308 (58)	57 (9)	2	1 (0)	0	0	0	0
www.ipac.caltech.edu	285 (14)	209 (12)	0	0	0	0	0	0
www.atnf.csiro.au	211 (21)	12 (6)	0	0	0	0	0	7 (5)
space.mit.edu	193 (10)	58 (5)	1	0	0	0	0	2 (1)
www.astro.psu.edu	186 (4)	103 (1)	1	10	1	1	0	2
www.eso.org	186 (58)	54 (22)	1 (1)	0	0	0	0	4 (1)
irsa.ipac.caltech.edu	163 (5)	38	0	0	1	0	0	0
www.sdss.org	156 (2)	106 (1)	0	0	0	0	0	0
hea-www.harvard.edu	125 (37)	42 (17)	1	0	0	1	0	26 (16)
physics.nist.gov	125 (3)	63 (2)	0	0	0	0	0	0
www.noao.edu	120 (3)	50 (2)	0	0	0	0	0	0
xmm.vilspa.esa.es	118 (35)	23 (19)	0	0	8 (1)	0	0	1 (1)
www.astro.princeton.edu	115 (31)	43 (14)	0	0	0	0	0	53 (12)
ad.usno.navy.mil	110 (27)	98 (22)	3 (3)	0	0	0	0	1 (1)

This table lists total number of links and broken links (HTTP status codes 3xx, 4xx, and 5xx) to top domains (domains with over 100 links) found within articles published in the four main astronomy journals between 1997 and 2008. The table also shows, for each domain, the portion of links to common filename extensions, as well as links that contain the tilde character.
doi:10.1371/journal.pone.0104798.t001

How Do Astronomers Share Data?

Pepe, Goodman, Muench, Crosas, Erdmann

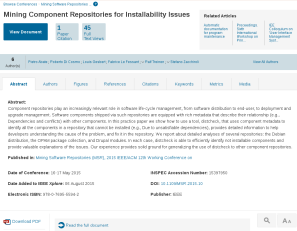
PLOS August 28, 2014

dx.doi.org/10.1371/journal.pone.0104798

DOI limitations

Example: doi:10.1109/MSR.2015.10

- to find what 10.1109/MSR.2015.10 is, go to a *resolver* (e.g. doi.org)
- this returns http://ieeexplore.ieee.org/document/7180064/
- at this URL we find ...



Browse Conferences > Mining Software Repositories ...

Mining Component Repositories for Installability Issues

View Document | 1 Paper Citation | 45 Full Text Views

Related Articles

Automatic documentation for program maintenance | Proceedings. Sixth International Workshop on Prin... | IEE Colloquium on 'User Interface Management Syst...

6 Author(s) | Pietro Abate ; Roberto Di Cosmo ; Louis Gesbert ; Fabrice Le Fessant ; Ralf Treinen ; Stefano Zacchiroli | View All Authors

Abstract | Authors | Figures | References | Citations | Keywords | Metrics | Media

Abstract:
Component repositories play an increasingly relevant role in software life-cycle management, from software distribution to end-user, to deployment and upgrade management. Software components shipped via such repositories are equipped with rich metadata that describe their relationship (e.g., Dependencies and conflicts) with other components. In this practice paper we show how to use a tool, distcheck, that uses component metadata to identify all the components in a repository that cannot be installed (e.g., Due to unsatisfiable dependencies), provides detailed information to help developers understanding the cause of the problem, and fix it in the repository. We report about detailed analyses of several repositories: the Debian distribution, the OPAM package collection, and Drupal modules. In each case, distcheck is able to efficiently identify not installable components and provide valuable explanations of the issues. Our experience provides solid ground for generalizing the use of distcheck to other component repositories.

Published in: Mining Software Repositories (MSR), 2015 IEEE/ACM 12th Working Conference on

Date of Conference: 16-17 May 2015

INSPEC Accession Number: 15397950

Date Added to IEEE *Xplore*: 06 August 2015

DOI: 10.1109/MSR.2015.10

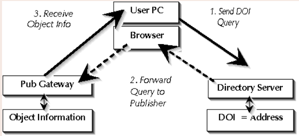
Electronic ISBN: 978-0-7695-5594-2

Publisher: IEEE

Download PDF

Read the full document

Architecture of the DOI infrastructure



- DOI resolution *can change*
- content at URL *can change*
- no *intrinsic* way of noticing
- persistence based on *good will* of *multiple parties*

Outline

4. Under the hood

5. Identifiers are not easy

6. Looking for the right identifiers

Systems of identifiers

A *system of identifiers* is

- a set of labels (the identifiers)
- mechanisms to perform :

Generation (minting)	create a new label
Assignment	associate label to object
Retrieval	get object from a label

- optionally, mechanisms to perform:

Verification	check label and object
Reverse Lookup	get label from an object
Description	get metadata of an object

Mechanisms offered in some systems of identifiers

Mech. / System	Handle	DOI	Ark	PURL
Generation	Yes	Yes	Yes	Yes
Assignment	Yes	Yes	Yes	Yes
Retrieval	Yes	Yes	Yes	Yes
Verification	N.A.	N.A.	N.A.	N.A.
Reverse Lookup	N.A.	N.A.	N.A.	N.A.
Description	Yes	Yes	Yes	N.A.

Our challenges in the PID landscape

Typical properties of systems of identifiers

uniqueness, non ambiguity, persistence, abstraction (opacity)

Key needed properties from our use cases

gratis	identifiers are free (billions of objects)
integrity	the associated object cannot be changed (sw dev, *reproducibility*)
no middle man	no central authority is needed (sw dev, *reproducibility*)

we could not find systems with both integrity and no middle man !

An important distinction: DIOs vs. IDOs

*The term “Digital Object Identifier” is construed as “digital identifier of an object,"
rather than “identifier of a digital object”* *Norman Paskin. 2010*

DIO (Digital Identifier of an Object)

digital identifiers for (potentially) non digital objects

- epistemic complexity (manifestations, versions, locations, etc.)
- need an authority to ensure persistence and uniqueness

IDO (Identifier of a Digital Object)

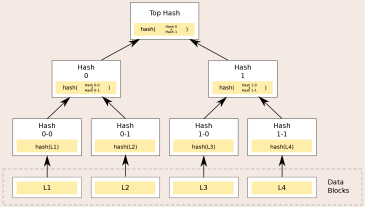
digital identifiers (only) for digital objects

- can provide both integrity and no middle man
- broadly used in modern software development (git, etc.)

for the core Software Heritage archive, IDOs are enough

IDOs in Software Development: the origins

Merkle tree (R. C. Merkle, Crypto 1979)



Combination of

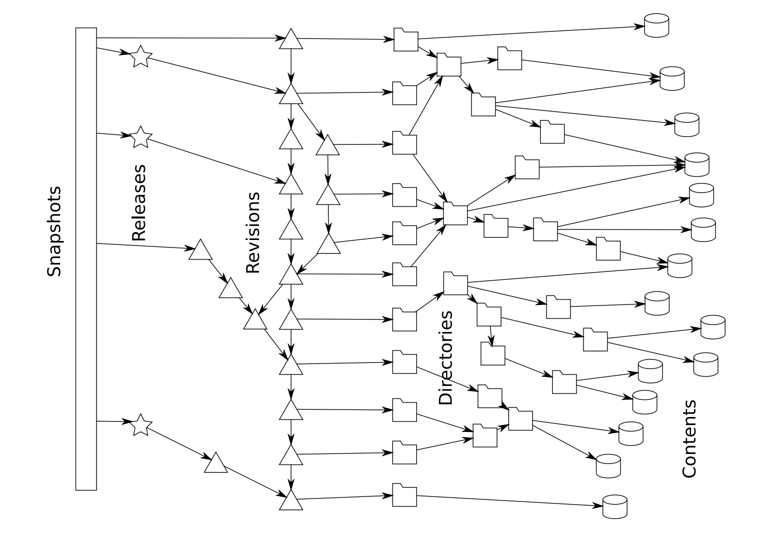
- tree
- hash function

Classical cryptographic construction

fast, parallel signature of large data structures, built-in deduplication

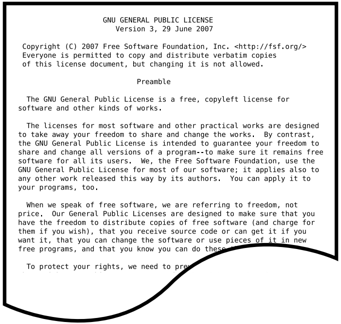
- satisfies all three criteria: gratis, integrity, no middle man!
- widely used in industry (e.g., Git, nix, blockchains, IPFS, . . .)

IDOs in Software Heritage: a worked example



IDOs in Software Heritage: a worked example

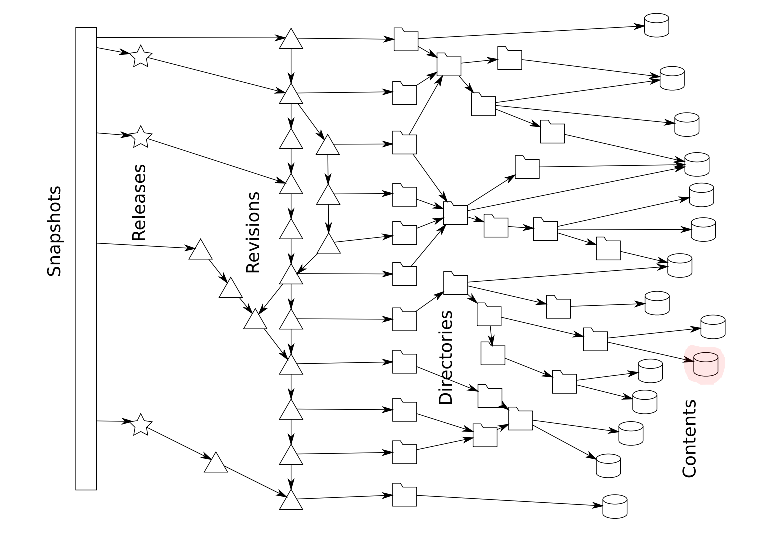
Contents



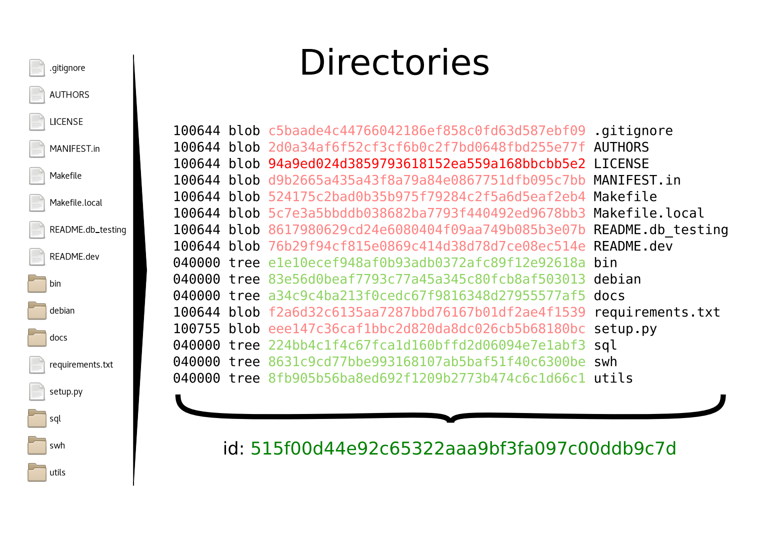
```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 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

  To protect your rights, we need to pre
```

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

IDOs in Software Heritage: a worked example



IDOs in Software Heritage: a worked example



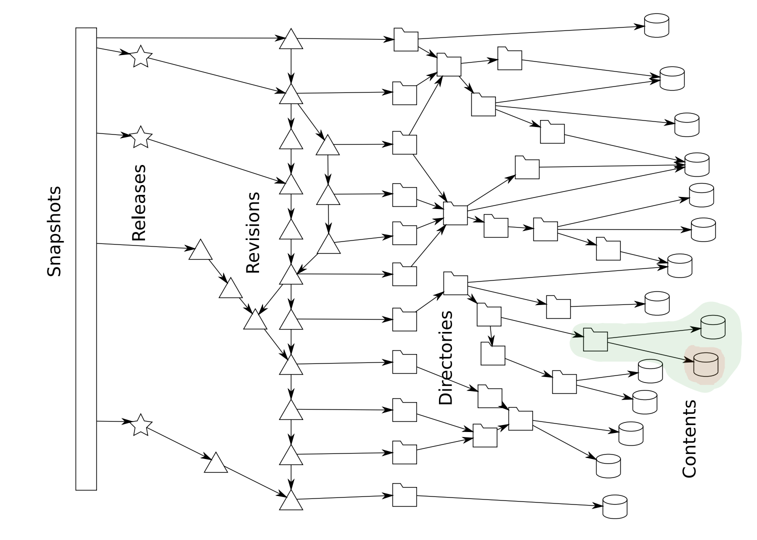
.gitignore
AUTHORS
LICENSE
MANIFEST.in
Makefile
Makefile.local
README.db_testing
README.dev
bin
debian
docs
requirements.txt
setup.py
sql
swh
utils

Directories

```
100644 blob c5baade4c44766042186ef858c0fd63d587ebf09 .gitignore
100644 blob 2d0a34af6f52cf3cf6b0c2f7bd0648fbd255e77f AUTHORS
100644 blob 94a9ed024d3859793618152ea559a168bbcbb5e2 LICENSE
100644 blob d9b2665a435a43f8a79a84e0867751dfb095c7bb MANIFEST.in
100644 blob 524175c2bad0b35b975f79284c2f5a6d5eaf2eb4 Makefile
100644 blob 5c7e3a5bbddb038682ba7793f440492ed9678bb3 Makefile.local
100644 blob 8617980629cd24e6080404f09aa749b085b3e07b README.db_testing
100644 blob 76b29f94cf815e0869c414d38d78d7ce08ec514e README.dev
040000 tree e1e10ecef948af0b93adb0372afc89f12e92618a bin
040000 tree 83e56d0beaf7793c77a45a345c80fcb8af503013 debian
040000 tree a34c9c4ba213f0cedc67f9816348d27955577af5 docs
100644 blob f2a6d32c6135aa7287bbd76167b01df2ae4f1539 requirements.txt
100755 blob eee147c36caf1bbc2d820da8dc026cb5b68180bc setup.py
040000 tree 224bb4c1f4c67fca1d160bffd2d06094e7e1abf3 sql
040000 tree 8631c9cd77bbe993168107ab5baf51f40c6300be swh
040000 tree 8fb905b56ba8ed692f1209b2773b474c6c1d66c1 utils
```

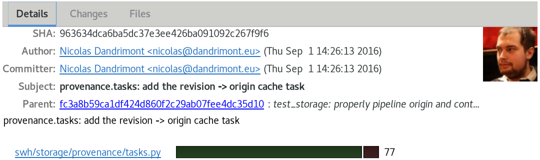
id: 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d

IDOs in Software Heritage: a worked example



IDOs in Software Heritage: a worked example

Revisions



SHA: 963634dca6ba5dc37e3ee426ba091092c267f9f6

Author: Nicolas Dandrimont <nicolas@dandrimont.eu> (Thu Sep 1 14:26:13 2016)

Committer: Nicolas Dandrimont <nicolas@dandrimont.eu> (Thu Sep 1 14:26:13 2016)

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

Parent: fc3a8b59ca1df424d860f2c29ab07fee4dc35d10 : *test_storage: properly pipeline origin and cont...*

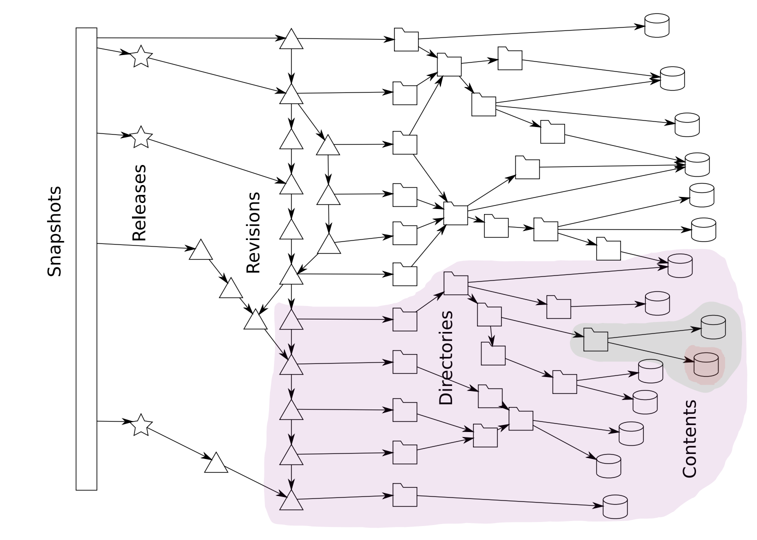
provenance.tasks: add the revision -> origin cache task

swh/storage/provenance/tasks.py 77

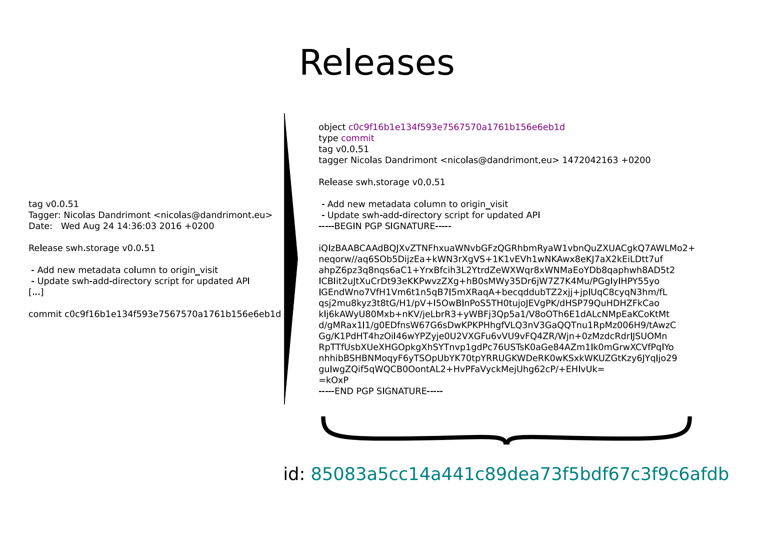
```
tree 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d
parent fc3a8b59ca1df424d860f2c29ab07fee4dc35d10
author Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200
committer Nicolas Dandrimont <nicolas@dandrimont.eu> 1472732773 +0200

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

id: 963634dca6ba5dc37e3ee426ba091092c267f9f6

IDOs in Software Heritage: a worked example



IDOs in Software Heritage: a worked example



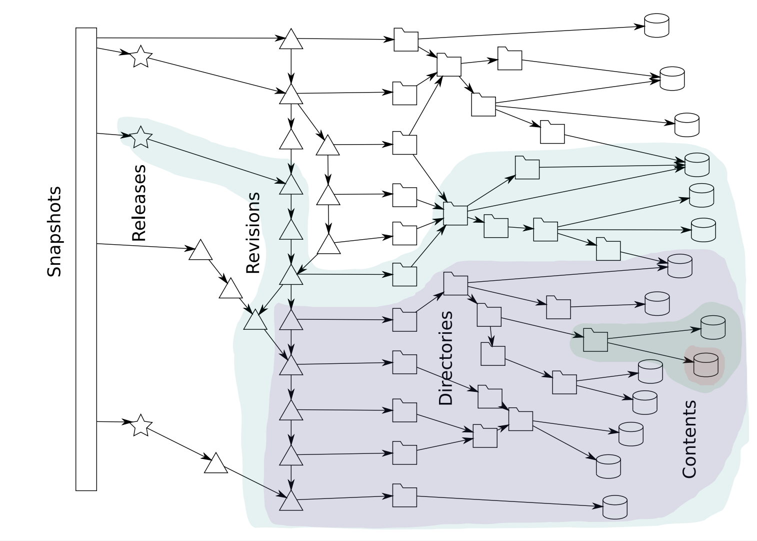
Releases

```
tag v0.0.51
Tagger: Nicolas Dandrimont <nicolas@dandrimont.eu>
Date:   Wed Aug 24 14:36:03 2016 +0200

Release swh.storage v0.0.51

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

commit c0c9f16b1e134f593e7567570a1761b156e6eb1d
```

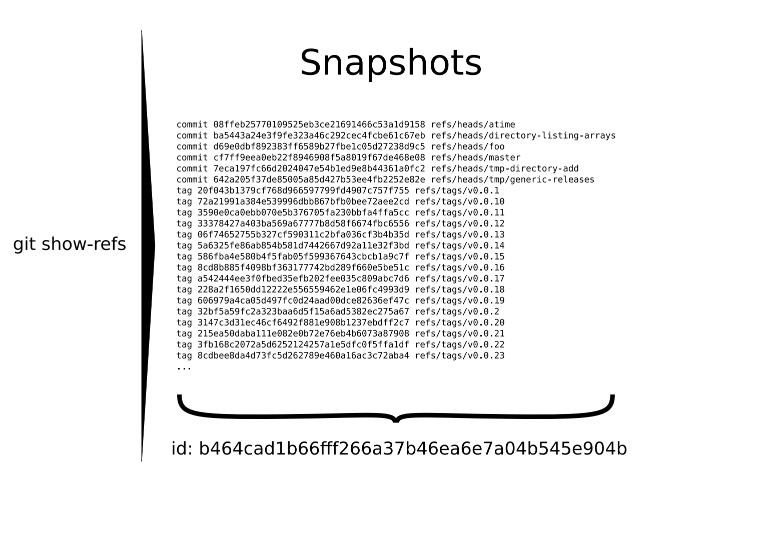
```
object c0c9f16b1e134f593e7567570a1761b156e6eb1d
type commit
tag v0.0.51
tagger Nicolas Dandrimont <nicolas@dandrimont.eu> 1472042163 +0200

Release swh.storage v0.0.51

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

iQIzBAABCAAdBQJXvZTNFhxuaWNvbGFzQGRhbmRyaW1vbnQuZXUACgkQ7AWLMo2+
neqorw//aq6SOb5DijzEa+kWN3rXgVS+1K1vEVh1wNKAwx8eKJ7aX2kEiLDtt7uf
ahpZ6pz3q8nqs6aC1+YrxBfcih3L2YtrdZeWXWqr8xWNMaEoYDb8qaphwh8AD5t2
lCBIit2uJtXuCrDt93eKKPwvzZXg+hB0sMWy35Dr6jW7Z7K4Mu/PGgIyIHPY55yo
lGEndWno7VfH1Vm6t1n5qB7l5mXRaqA+becqddubTZ2xjj+jplUqC8cyqN3hm/fL
qsj2mu8kyz3t8tG/H1/pV+I5OwBlnPoS5TH0tujoJEVgPK/dHSP79QuHDHZFkCao
klj6kAWyU80Mxb+nKV/jeLbrR3+yWBFj3Qp5a1/V8oOTh6E1dALcNMpEaKCoKtMt
d/gMRax1l1/g0EDfnsW67G6sDwKPKPHhgfVLQ3nV3GaQQTnu1RpMz006H9/tAwzC
Gg/K1PdHT4hzOiI46wYPZyje0U2VXGFu6vVU9vFQ4ZR/Wjn+0zMzdcRdrlJSUOMn
RpTTfUsbXUeXHGOpkgXhSYTnvp1gdPc76USTsK0aGe84AZm1lk0mGrwXCVfPqIYo
nhhibBSHBNMoqyF6yTSOpUbYK70tpYRRUGKWDeRK0wKSxkWKUZGtKzy6JYqljo29
gulwgZQif5qWQCB0OontAL2+HvPFaVyckMejUhg62cP/+EHIvUk=
=kOxP
-----END PGP SIGNATURE-----
```

id: 85083a5cc14a441c89dea73f5bdf67c3f9c6afdb

IDOs in Software Heritage: a worked example



IDOs in Software Heritage: a worked example



```
commit 08ffeb25770109525eb3ce21691466c53a1d9158 refs/heads/atime
commit ba5443a24e3f9fe323a46c292cec4fcbe61c67eb refs/heads/directory-listing-arrays
commit d69e0dbf892383ff6589b27fbe1c05d27238d9c5 refs/heads/foo
commit cf7ff9eea0eb22f8946908f5a8019f67de468e08 refs/heads/master
commit 7eca197fc66d2024047e54b1ed9e8b44361a0fc2 refs/heads/tmp-directory-add
commit 642a205f37de85005a85d427b53ee4fb2252e82e refs/heads/tmp/generic-releases
tag 20f043b1379cf768d966597799fd4907c757f755 refs/tags/v0.0.1
tag 72a21991a384e539996dbb867bfb0bee72aee2cd refs/tags/v0.0.10
tag 3590e0ca0ebb070e5b376705fa230bbfa4ffa5cc refs/tags/v0.0.11
tag 33378427a403ba569a67777b8d58f6674fbc6556 refs/tags/v0.0.12
tag 06f74652755b327cf590311c2bfa036cf3b4b35d refs/tags/v0.0.13
tag 5a6325fe86ab854b581d7442667d92a11e32f3bd refs/tags/v0.0.14
tag 586fba4e580b4f5fab05f599367643cbcb1a9c7f refs/tags/v0.0.15
tag 8cd8b885f4098bf363177742bd289f660e5be51c refs/tags/v0.0.16
tag a542444ee3f0fbed35efb202fee035c809abc7d6 refs/tags/v0.0.17
tag 228a2f1650dd12222e556559462e1e06fc4993d9 refs/tags/v0.0.18
tag 606979a4ca05d497fc0d24aad00dce82636ef47c refs/tags/v0.0.19
tag 32bf5a59fc2a323baa6d5f15a6ad5382ec275a67 refs/tags/v0.0.2
tag 3147c3d31ec46cf6492f881e908b1237ebdff2c7 refs/tags/v0.0.20
tag 215ea50daba111e082e0b72e76eb4b6073a87908 refs/tags/v0.0.21
tag 3fb168c2072a5d6252124257a1e5dfc0f5ffa1df refs/tags/v0.0.22
tag 8cdbee8da4d73fc5d262789e460a16ac3c72aba4 refs/tags/v0.0.23
...
```

The Software Heritage IDO schema (see http://bit.ly/swhpids)

swh:1:**cnt**:94a9ed024d3859793618152ea559a168bbcbb5e2 full text of the GPL3 license

swh:1:**dir**:d198bc9d7a6bcf6db04f476d29314f157507d505 Darktable source code

swh:1:**rev**:309cf2674ee7a0749978cf8265ab91a60aea0f7d

a **revision** in the development history of Darktable

swh:1:**rel**:22ece559cc7cc2364edc5e5593d63ae8bd229f9f

release 2.3.0 of Darktable, dated 24 December 2016

swh:1:**snp**:c7c108084bc0bf3d81436bf980b46e98bd338453

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

Current resolvers: archive.softwareheritage.org and n2t.org