

# Software Heritage

The universal source code archive

Morane Gruenpeter

Software engineer and metadata specialist

Inria, Software Heritage

[morane@softwareheritage.org](mailto:morane@softwareheritage.org)

December 1st, 2020



# Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

- 1 Introduction
- 2 The knowledge is in the source code !
- 3 Data model and SWHID: the source code fingerprint
- 4 The software deposit - a first class research output
- 5 The missing piece- the Metadata
- 6 Development workflow
- 7 Conclusion



# Short Bio: Morane Gruenpeter

Goal: Building the Semantic Web of Free and Open Source Software



1999-2011 Harpist

2012-2015 Licence in Computer Science CNAM

2015-2017 Master STL - M2 R&D UPMC

2017 Internship *Software Heritage* (SWH)

2018-2019 European project EU2020 *CROSSMINER* (on SWH team)

2020-2022 European project *FAIRsFAIR* (on SWH team)

## Goal: Building the Semantic Web of Free and Open Source Software



1999-2011 Harpist

2012-2015 Licence in Computer Science CNAM

2015-2017 Master STL - M2 R&D UPMC

2017 Internship *Software Heritage* (SWH)

2018-2019 European project EU2020 *CROSSMINER* (on SWH team)

2020-2022 European project *FAIRsFAIR* (on SWH team)

## Working groups for Open Science and digital preservation

- the Research Data Alliance's **Software Source Code** Interest Group (SSC IG),
- the FORCE11's **Software Citation** Implementation Working Group (SCI WG),
- the joint RDA & FORCE11 **Software Identification** Working Group (SCID WG)
- WikiData for **Digital Preservation** initiative (WikiDigi).



## 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** 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** all software source code

### Research infrastructure



**enable analysis** of all software source code



**Cultural Heritage**



**Industry**



**Research**



**Education**



**Software Heritage**

**Cultural Heritage**



**Industry**



**Research**



**Education**



## Software Heritage

As of today the archive already contains and keeps safe for you the following amount of objects:

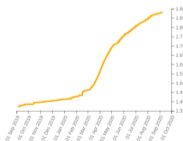
**Source files**

8,846,381,610



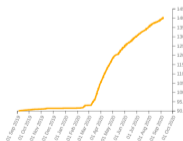
**Commits**

1,880,663,008



**Projects**

140,348,311



**Directories**

7,506,954,410

**Authors**

38,603,337

**Releases**

15,051,940

Raising awareness: landmark agreement, 3/4/2017; grand opening, 7/6/2018



## Sharing the vision



Morane Gruenpeter

## Sponsoring our work



Platinum sponsors



Gold sponsors



Silver sponsors



- 1 Introduction
- 2 The knowledge is in the source code !
- 3 Data model and SWHID: the source code fingerprint
- 4 The software deposit - a first class research output
- 5 The missing piece- the Metadata
- 6 Development workflow
- 7 Conclusion



## Apollo 11 Guidance Computer (~60.000 lines), 1969



"When I first got into it, nobody knew what it was that we were doing. It was like the Wild West." Margaret Hamilton

## The World Wide Web, 1989, at CERN on a NeXT machine

"When somebody has learned how to program a computer ... You're joining a group of people who can do incredible things. They can make the computer do anything they can imagine."



From An Insight, An Idea with Tim Berners-Lee (2013)

# 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.”*

GPL Licence



# 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.”*

GPL Licence

Hello World

# 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.”*

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
```



# 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."*

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");
}
```

# 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

# 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*

# 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.”*

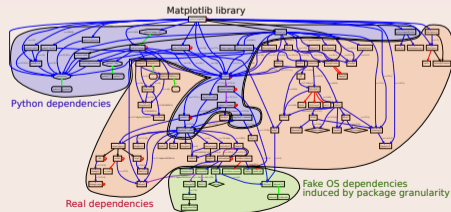
Harold Abelson

*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 human readable and executable knowledge

Full widthHome Development Documentation **Donate**login







## Software Heritage

Archive

---

Features

-  Search
-  Downloads
-  Save code now
-  Help

```
52
53 # THE MASTER IGNITION ROUTINE IS DESIGNED FOR USE BY THE FOLLOWING LEM PROGRAMS: P12, P40, P42, P61, P63.
54 # IT PERFORMS ALL FUNCTIONS IMMEDIATELY ASSOCIATED WITH APS OR DPS IGNITION: IN PARTICULAR, EVERYTHING LYING
55 # BETWEEN THE PRE-IGNITION TIME CHECK -- ARE WE WITHIN 45 SECONDS OF TIG? -- AND TIG + 26 SECONDS, WHEN DPS
56 # PROGRAMS THROTTLE UP.
57 #
58 # VARIATIONS AMONG PROGRAMS ARE ACCOMODATED BY MEANS OF TABLES CONTAINING CONSTANTS (FOR AVEGEXIT, FOR
59 # WAITLIST, FOR PINBALL) AND TCF INSTRUCTIONS. USERS PLACE THE ADRES OF THE APPROPRIATE TABLE
60 # (OF P61TABLE FOR P61LM, FOR EXAMPLE) IN ERASABLE REGISTER 'WHICH' (E4). THE IGNITION ROUTINE THEN INDEXES BY
61 # WHICH TO OBTAIN OR EXECUTE THE PROPER TABLE ENTRY. THE IGNITION ROUTINE IS INITIATED BY A TCF BURNBABY,
62 # THROUGH BANKJUMP IF NECESSARY. THERE IS NO RETURN.
63 #
64 # THE MASTER IGNITION ROUTINE WAS CONCEIVED AND EXECUTED, AND (NOTA BENE) IS MAINTAINED BY ADLER AND EYLES.
65 #
66 #           HONI SOIT QUI MAL Y PENSE
67 #
68 #           *****
69 #           TABLES FOR THE IGNITION ROUTINE
70 #           *****
71 #
72 #           NOLI SE TANGERE
73
74 P12TABLE      VN      0674      # (0)
75              TCF      ULLGNOT   # (1)
76              TCF      COMFAIL3  # (2)
77              TCF      GOCUTOFF   # (3)
78              TCF      TASKOVER   # (4)
79              TCF      P12SPOT    # (5)
80              DEC      0          # (6)      NO ULLAGE
81              EBANK=  WHICH
82              2CADR  SERVEXIT    # (7)
83
84              TCF      DISPCHNG   # (11)
85              TCF      WAITABIT   # (12)
86              TCF      P12IGN     # (13)
87
88 P40TABLE      VN      0640      # (0)
```

Permalinks

# Version Control Systems timeline

## Version control system (VCS)

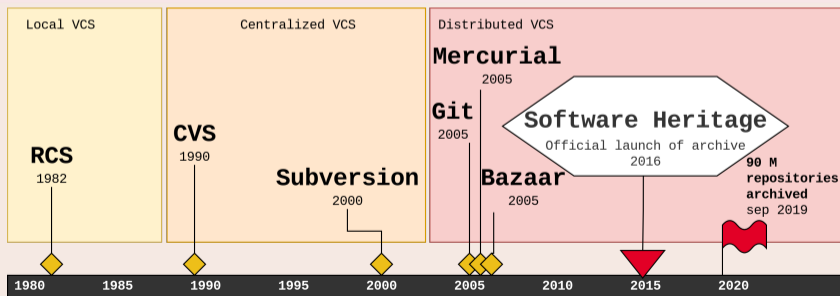
- records changes made to a (set of) *source code file (s)*
- allows to operate on versions: diff/merge/fork/recover etc.
- **essential** tool for software development

# Version Control Systems timeline

## 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



## Requirements for the D in DVCS

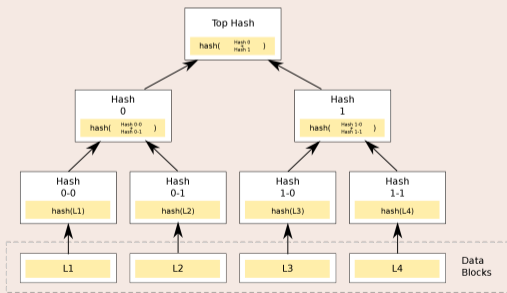
- **intrinsic** unique identifiers... (here: *cryptographic signature*, aka "hash")
- ... that work for **tree structures** (software directories)



## Requirements for the D in DVCS

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

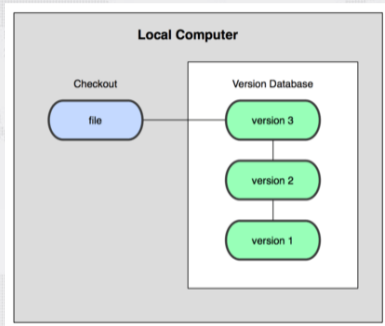
## Merkle tree to the rescue (R. C. Merkle, Crypto 1979)



## Combination of

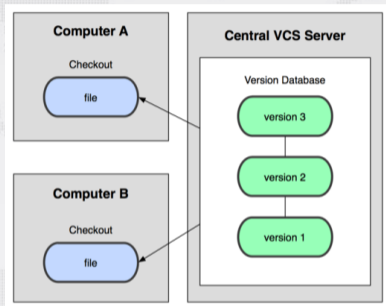
- tree
- hash function

# Version Control Systems explained



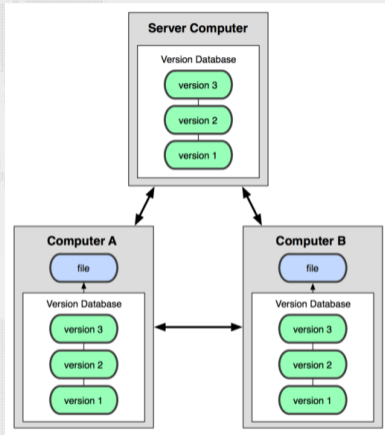
```
co -r1.2 file.c
```

# Version Control Systems explained



```
cvs co -r Rel-1A ProgABC
```

# Version Control Systems explained



```
git checkout df3b1b08f756569eff0919e37d8af1f403515b31
```

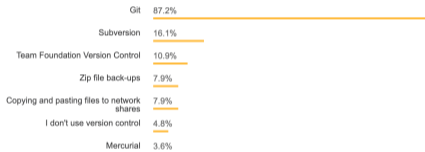
## Stack Overflow

[Survey 2018]

### Version Control

All Respondents

Professional Developers



74,298 responses; select all that apply

Git is the dominant choice for version control for developers today, with almost 90% of developers checking in their code via Git.

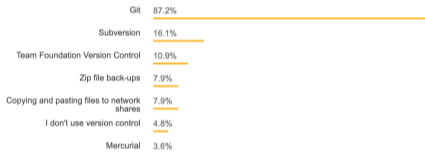
## Stack Overflow

[Survey 2018]

### Version Control

All Respondents

Professional Developers



74,298 responses; select all that apply

Git is the dominant choice for version control for developers today, with almost 90% of developers checking in their code via Git.

## In numbers

GitHub [Octoverse 2017] [Blog 2018]

- 100.000.000+ repositories
- 40.000.000+ developers worldwide

Bitbucket [Blog 2019]

- 28.000.000+ repositories
- 10.000.000+ developers worldwide

GitLab [Blog 2019]

- 1.000.000 MRs March 19'

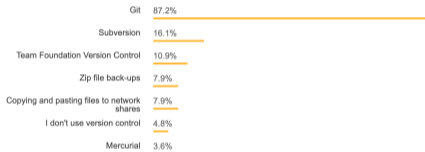
## Stack Overflow

[Survey 2018]

### Version Control

All Respondents

Professional Developers



74,298 responses; select all that apply

Git is the dominant choice for version control for developers today, with almost 90% of developers checking in their code via Git.

## In numbers

GitHub [Octoverse 2017] [Blog 2018]

- 100.000.000+ repositories
- 40.000.000+ developers worldwide

Bitbucket [Blog 2019]

- 28.000.000+ repositories
- 10.000.000+ developers worldwide

GitLab [Blog 2019]

- 1.000.000 MRs March 19'

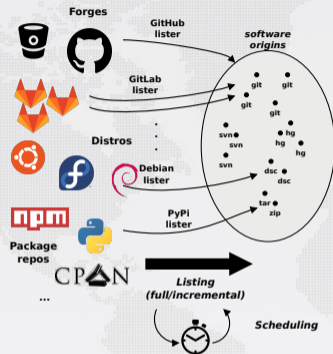
Let's use it!

- 1 Introduction
- 2 The knowledge is in the source code !
- 3 Data model and SWHID: the source code fingerprint**
- 4 The software deposit - a first class research output
- 5 The missing piece- the Metadata
- 6 Development workflow
- 7 Conclusion

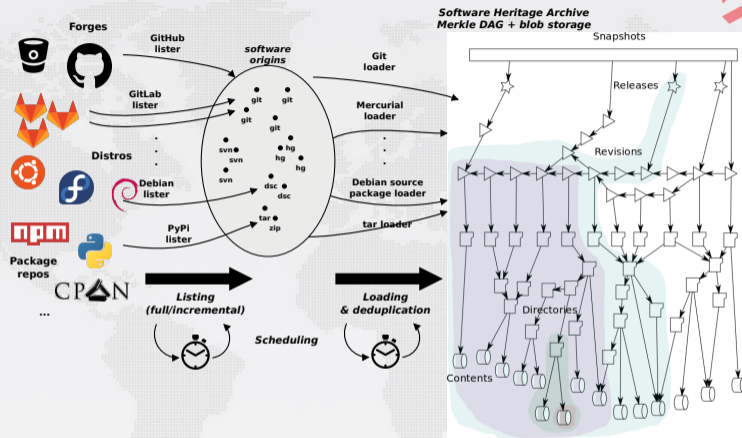




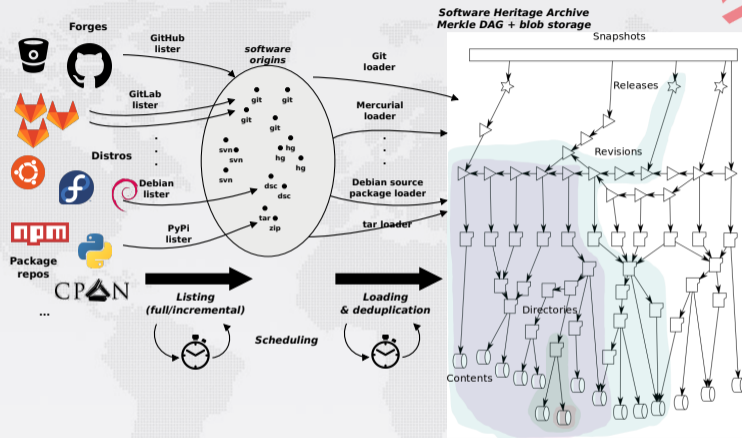
# Under the hood: Automation, and storage



# Under the hood: Automation, and storage



# Under the hood: Automation, and storage

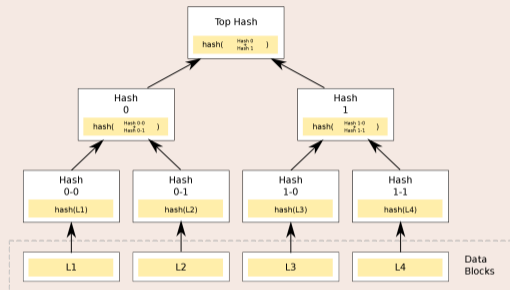


Global development history permanently archived in a uniform data model

- over 6 billion unique source files from over 90 million software projects
- ~400 TB (uncompressed) blobs, ~20 B nodes, ~280 B edges

# Much more than an archive!

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



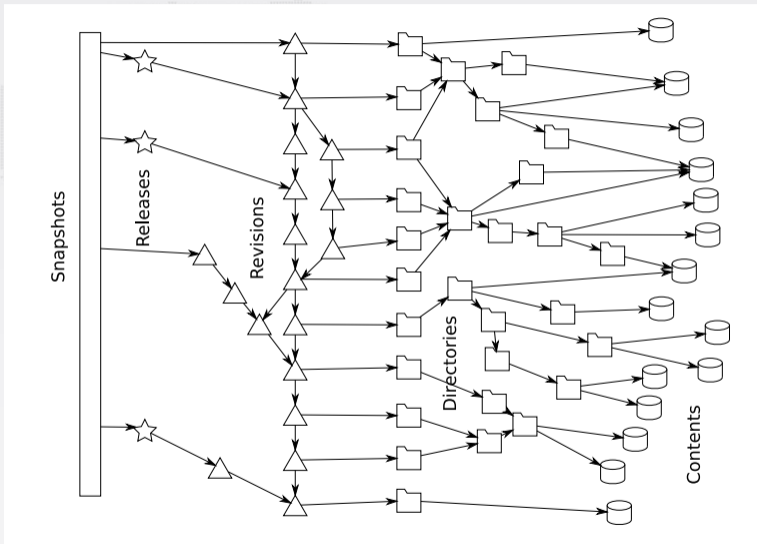
Combination of

- tree
- hash function

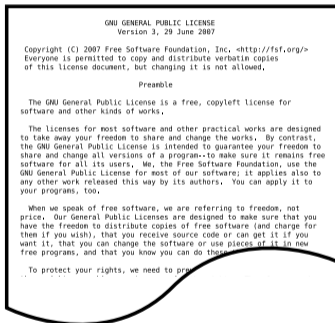
## Classical cryptographic construction

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

# The archive in pictures

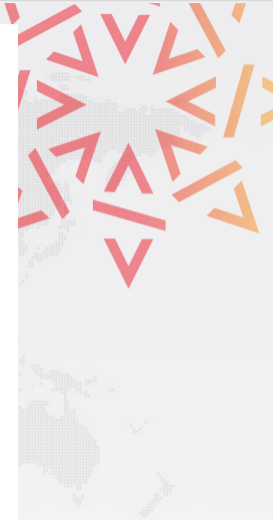
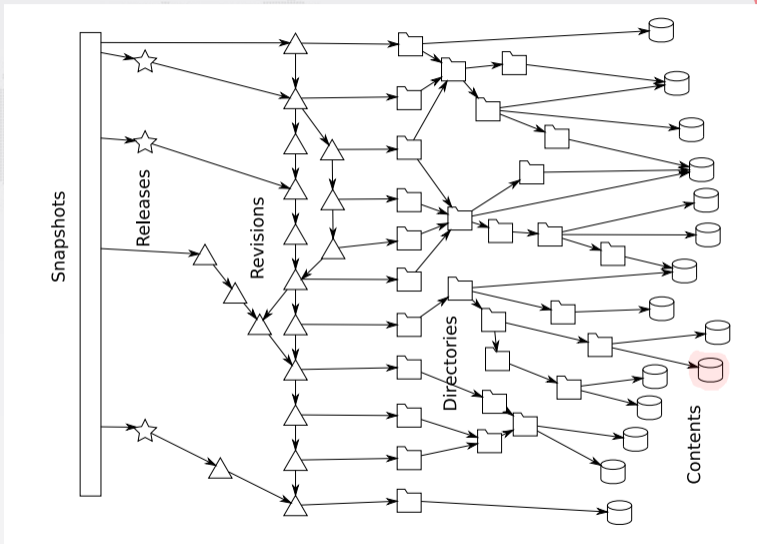


## Contents



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

# The archive in pictures





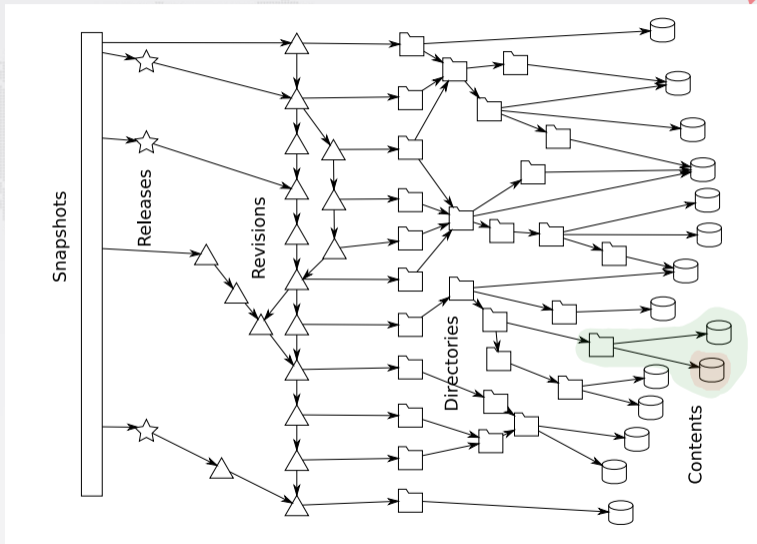
## Directories

```
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 224bb4c1f4c67fca1d160bffdd2d06094e7e1abf3 sql
040000 tree 8631c9cd77bbe993168107ab5baf51f40c6300be swh
040000 tree 8fb905b56ba8ed692f1209b2773b474c6c1d66c1 utils
```


id: 515f00d44e92c65322aaa9bf3fa097c00ddb9c7d



# The archive in pictures



## Revisions

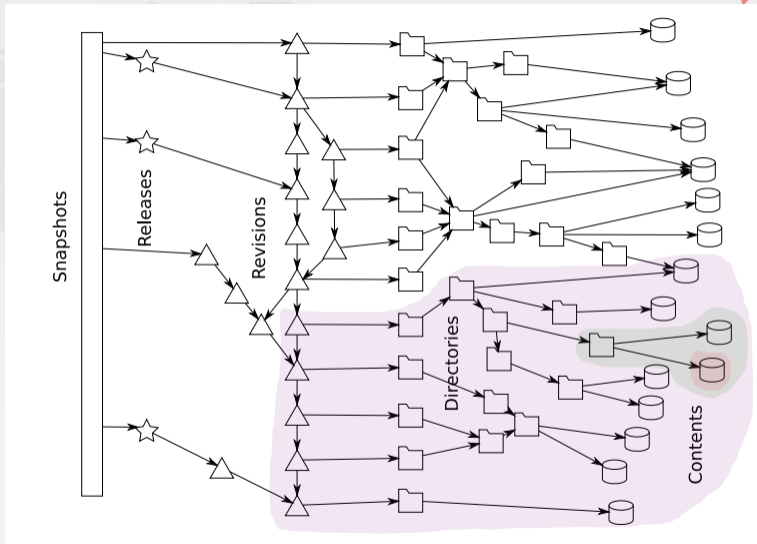
Details	Changes	Files
SHA: 963634dca6ba5dc37e3ee426ba091092c267f9f6		
Author: <a href="mailto:nicolas@dandrimont.eu">Nicolas Dandrimont &lt;nicolas@dandrimont.eu&gt;</a> (Thu Sep 1 14:26:13 2016)		
Committer: <a href="mailto:nicolas@dandrimont.eu">Nicolas Dandrimont &lt;nicolas@dandrimont.eu&gt;</a> (Thu Sep 1 14:26:13 2016)		
Subject: provenance.tasks: add the revision -> origin cache task		
Parent: <a href="#">fc3a8b59ca1df424d860f2c29ab07fee4dc35d10</a> : test...storage: properly pipeline origin and cont...		
provenance.tasks: add the revision -> origin cache task		
<a href="#">sw/wh/storage/provenance/tasks.py</a>  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

# The archive in pictures



## 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

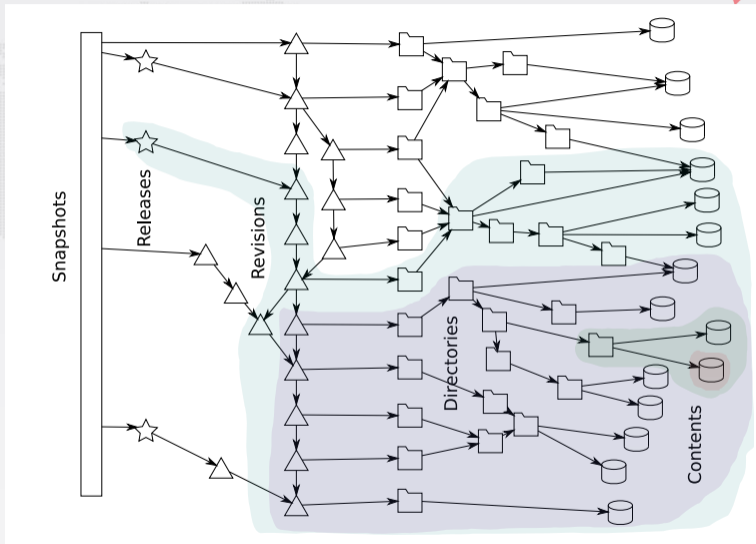
Release swh.storage v0.0.51

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

iQIzBAABCAAdBQJXvZTNFhxuaWNvbGFzQGRhbmRyaW1vbnQuZlUACgkQ7AWLMo2+  
neqorw//aq6SOB5DijzEa+kWN3rXgVS+1K1vEVh1wNKAw8eKJ7aX2kEILDtt7uf  
ahpZ6pz3q8nqs6aC1+YrxBfcih3L2YtrdZeWXWqr8xWNMaEoYDb8qaphwh8AD5t2  
ICBlit2ujXuCrDt93eKKPwvzZXg+h80sMwy35Dr6jW7Z7K4Mu/PgGlyIHPY55yo  
IGEndWno7VFH1Vm6t1n5qB7I5mXRaqA+becqddubTZ2xjj+jpUqC8cyqN3hm/fL  
qsJ2mu8kyz3t8tG/H1/pV+I5OwBlNpO5STH0tujoEvgPK/dHSP79QuHDHZFkCao  
klj6kAWyU80Mxb+nKV/jeLbrR3+yWBFJ3Qp5a1/V8o0ThE1dALcNmPcEaKCoKtMt  
d/gMRax111/g0EDfnsW67G6sDwKPKPHngfVLQ3nV3GaQQTnu1RpMz006H9/tAwzC  
Gg/K1PdHT4hz0jI46wYPZyje0U2VXGFu6vVU9vFQ4ZR/Wjn+0zZdcRdrJJSUOMn  
RpTTfUshXUeXHGOpkgXhSYTnvp1gdPc76U5TsK0aGe84AZm1Ik0mGrwXCvFpQYo  
nhhibB5HBNMoqyF6yTSOpUbyK70tpYRRUGKwDeRK0wKSxkWKUZGtKzy6jYqJjo29  
gulwZQif5qWQC80ontAL2+HvFfaVyckMejUhg62cP/+EHlvUk=  
=kOxP  
-----END PGP SIGNATURE-----

id: 85083a5cc14a441c89dea73f5bdf67c3f9c6afdb

# The archive in pictures



## Snapshots

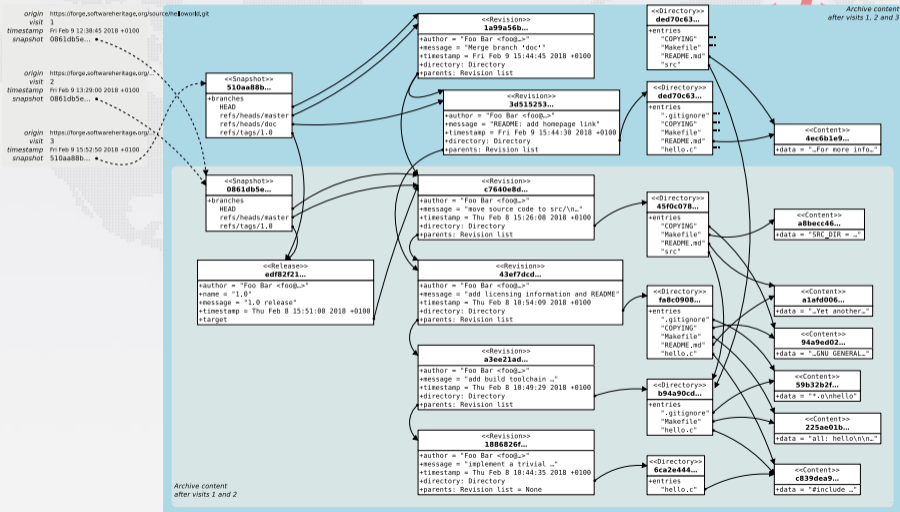
```
commit 08ffeb25770109525eb3ce21691466c53a1d9158 refs/heads/atime
commit ba5443a24e3f9fe323a46c292cec4fcbe61c67eb refs/heads/directory-listing-arrays
commit d69e0dbf892383ff6589b27fbc1c05d27238d9c5 refs/heads/foo
commit cf7ff9eea0eb22f8946908f5a8019f67de468e08 refs/heads/master
commit 7eca197fc66d2024047e54b1ed9e8b44361a0fc2 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 586fba4e580b4f5fab05f599367643cbb1a9c7f refs/tags/v0.0.15
tag 8cd8b885f4098bf363177742bd289f660e5be51c refs/tags/v0.0.16
tag a542444ee3f0fbcd35efb202fee035c809abc7d6 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 8cddb0e8da4d731c5d262789e460a16ac3c72aba4 refs/tags/v0.0.23
...
```

git show-refs

id: b464cad1b66fff266a37b46ea6e7a04b545e904b

# Under the hood: identifying billions of objects

<https://bit.ly/2w00myV>



## Typical properties of systems of identifiers

uniqueness, non ambiguity, persistence, abstraction (opacity)



## 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*)

# 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

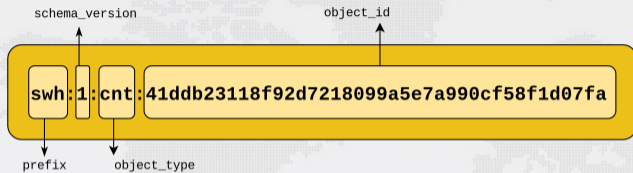
**gratis** identifiers are free (billions of objects)

**integrity** the associated object cannot be changed (sw dev, *reproducibility*)

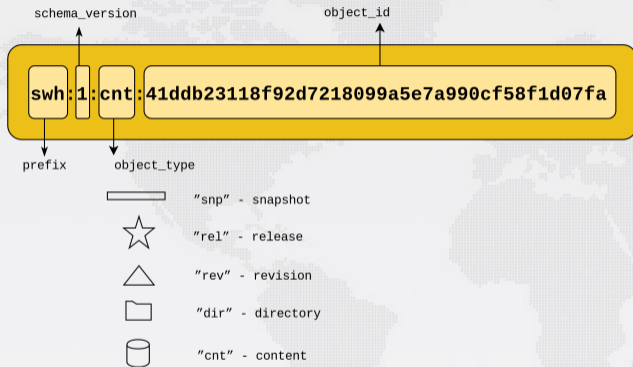
**no middle man** no central authority is needed (sw dev, *reproducibility*)

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

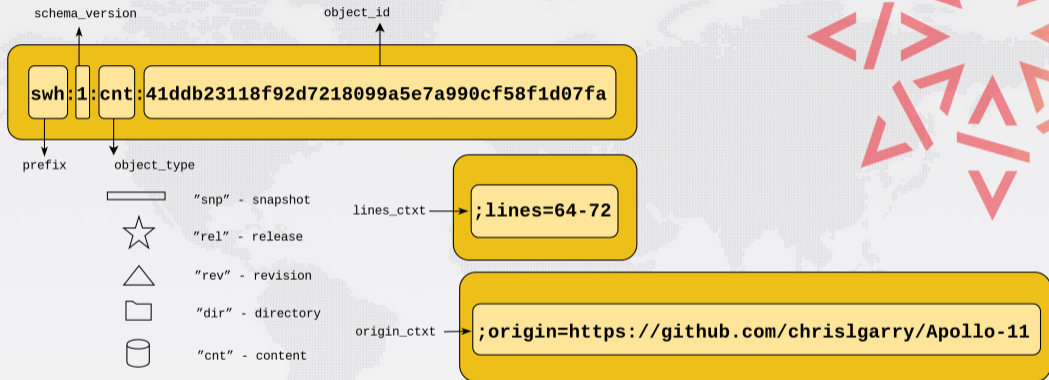
# The SWH-ID schema



# The SWH-ID schema



# The SWH-ID schema



Let's look at some famous excerpts of source code

## Let's look at some famous excerpts of source code

### Apollo 11 source code (excerpt)

```
P63SP0T3      CA      BIT6          # IS THE LR ANTENNA IN POSITION 1 YET
              EXTEND
              RAND     CHAN33
              EXTEND
              BZF      P63SP0T4      # BRANCH IF ANTENNA ALREADY IN POSITION 1

              CAF      CODE500      # ASTRONAUT: PLEASE CRANK THE
              TC       BANKCALL     # SILLY THING AROUND
              CADR     GOPERF1
              TCF      GOTOP00H     # TERMINATE
              TCF      P63SP0T3     # PROCEED SEE IF HE'S LYING

P63SP0T4      TC       BANKCALL     # ENTER INITIALIZE LANDING RADAR
              CADR     SETPOS1

              TC       POSTJUMP     # OFF TO SEE THE WIZARD ...
              CADR     BURNBABY
```

## Let's look at some famous excerpts of source code

### Apollo 11 source code (excerpt)

```
P63SP0T3      CA      BIT6          # IS THE LR ANTENNA IN POSITION 1 YET
              EXTEND
              RAND      CHAN33
              EXTEND
              BZF      P63SP0T4      # BRANCH IF ANTENNA ALREADY IN POSITION 1

              CAF      CODE500      # ASTRONAUT: PLEASE CRANK THE
              TC       BANKCALL     # SILENT THING AROUND
              CADR     GOPERF1
              TCF      GOTOP00H     # TERMINATE
              TCF      P63SP0T3     # PROCEED SEE IF HE'S LYING

P63SP0T4      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;
}
```



Let's look at some famous excerpts of source code

## Apollo 11 source code (excerpt)

```
P63SP0T3      CA      BIT6          # IS THE LR ANTENNA IN POSITION 1 YET
              EXTEND
              RAND      CHAN33
              EXTEND
              BZF       P63SP0T4      # BRANCH IF ANTENNA ALREADY IN POSITION 1

              CAF       CODE500      # ASTRONAUT: PLEASE CRANK THE
              TC        BANKCALL     # SILLY THING AROUND
              CADR      GOPERF1
              TCF       GOTOP00H     # TERMINATE
              TCF       P63SP0T3     # PROCEED SEE IF HE'S LYING

P63SP0T4      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;
}
```

It works!

we have *intrinsic* identifiers for all 20+ billion objects in the archive

- 1 Introduction
- 2 The knowledge is in the source code !
- 3 Data model and SWHID: the source code fingerprint
- 4 The software deposit - a first class research output
- 5 The missing piece- the Metadata
- 6 Development workflow
- 7 Conclusion



# Software is a *forgotten* pillar of Open Science

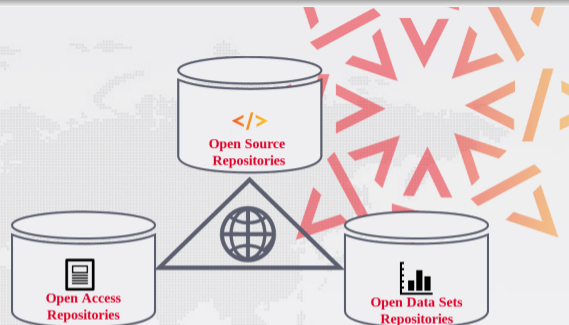
## Lack of recognition

not (yet) a first class citizen

- in the EOSC plan
- in the scholarly world

*Sometimes, if you don't have the software, you don't have the data*

*Christine Borgman, Paris, 2018*



# Software is a *forgotten* pillar of Open Science

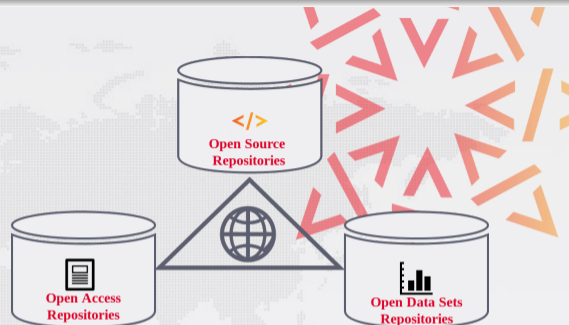
## Lack of recognition

not (yet) a first class citizen

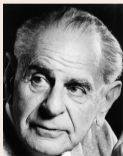
- in the EOSC plan
- in the scholarly world

*Sometimes, if you don't have the software, you don't have the data*

*Christine Borgman, Paris, 2018*



## Reproducibility is the key



*non-reproducible single occurrences are of no significance to science*

*Karl Popper, The Logic of Scientific Discovery, 1934*

## Archival

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

## 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*)

## 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*)

## 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

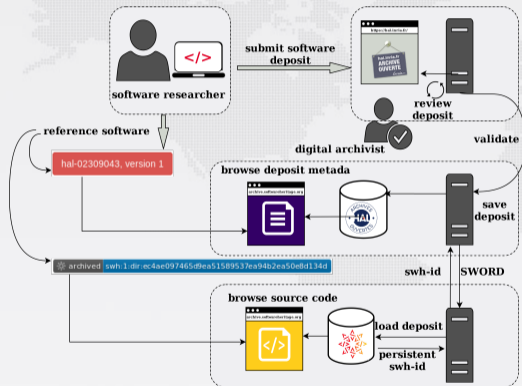
Research software artifacts must be properly **described**  
make it easy to *discover* them (*visibility*)

## Citation

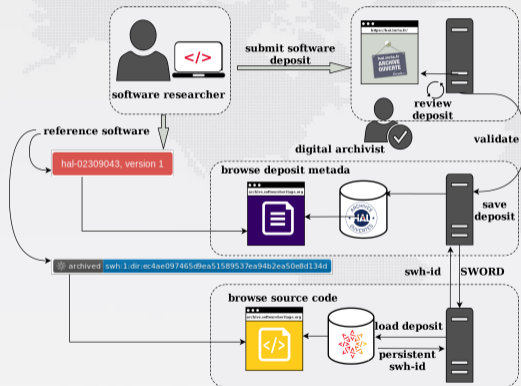
Research software artifacts must be properly **cited** (*not the same as referenced!*)  
to give *credit* to authors (*evaluation!*)



# The research software (deposit) use case



# The research software (deposit) use case



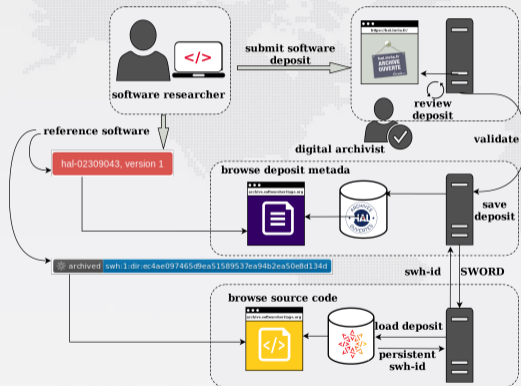
Deposit software in HAL

poster

Generic mechanism:

- SWORD based
- review process
- versioning

# The research software (deposit) use case



## Deposit software in HAL

poster

### Generic mechanism:

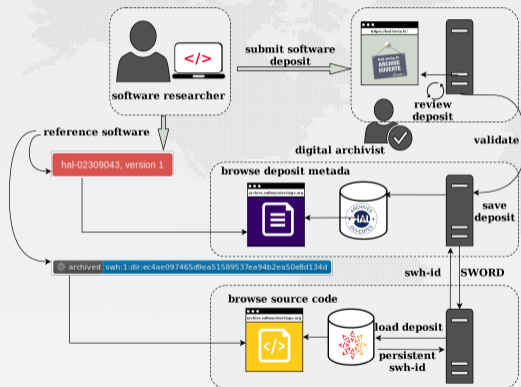
- SWORD based
- review process
- versioning

### How to do it:

*(guide)*

- deposit .zip or .tar.gz file with metadata

# The research software (deposit) use case



## Deposit software in HAL

poster

### Generic mechanism:

- SWORD based
- review process
- versioning

### How to do it:

(guide)

- deposit .zip or .tar.gz file with metadata

### Timeline:

- *March 2018*: test phase on **HAL-Inria**
- *September 2018*: open to all **HAL**
- *December 2019*:
  - 80 complete source code deposits
  - 98 software records

Document type \*

Title \*

Domain \*

- Book sections
- Directions of work or proceedings
- Patents
- Other publications
- Documents**
  - Preprints, Working Papers, ...
  - Reports
  - Notes de synthèse
- Academic works**
  - Theses
  - Habilitation à diriger des recherches
  - Master thesis
  - Lectures
- Research data**
  - Photos
  - Videos
  - Audios
  - Maps
  - Software

CCSD HAL - Epistémologie, Méthéphysique, Sociologie

HAL-Inria Les publications, logiciels, ... des scientifiques Inria

Accueil Accueil Chercher tout HAL Recherche Services Documentation OpenAccess/Archives Publications Inria Messagerie

Document

Depositer votre logiciel

Compléter les métadonnées de document

ATTENTION la liste complète des métadonnées est

Type de document \*

Nom \*

Description

Mots-clés

Identifiant

Date de publication/évolution

Classification

Financement

Projet(s) financé(s)

Langage de programmation

Code Propriété

Détails MSS

- Chercheur
- Université (pr)
- Mathématiques (math)
- Informatique (cs)
- Physique (physics)
- Sciences cognitives
- Sciences de l'environnement
- Sciences de la vie (biology)
- Sciences de l'homme et de la société
- Sciences de l'ingénierie (physics)
- Mathématiques (math)
- Sciences et Technologie (misc)

Software License \*

One or more licenses under which the software is published (can use autocomplete)

AG

Adobe Systems Incorporated Source Code License Agreement

Affero General Public License v1.0

CeCILL Free Software License Agreement v1.0

CeCILL Free Software License Agreement v1.1

CeCILL Free Software License Agreement v2.0

CeCILL Free Software License Agreement v2.1

Complete the author(s) data

CCSD **preprod** HAL - Episciences.org Sciencesconf.org Support fr en Morane Gruenpeter

**Inria** inventeurs du monde numérique **50 ANS** 1967-2017 Imaginons notre futur **hal.inria.fr** ARCHIVE OUVERTE **HAL - Inria** Archive ouverte / Open archive **ARCHIVES HAL OUVERTES**

Accueil Déposer Consulter tout HAL - Publications Inria Recherche Services - Documentation - OpenAccess@Inria Mon espace - Privilèges -

hal-01588935, version 1

## The assignment problem

Morane Gruenpeter <sup>1</sup> [Détails](#)

<sup>1</sup> Initiative pour la Recherche et l'Innovation sur le Logiciel Libre - IRILL


A java implementation for The Assignment Problem a distributed system as a set of processors that can perform tasks (or processes) in parallel. We therefore consider a set of m processors, each equipped with a certain amount of random access memory (RAM). We associate a cost to pay to perform this task on this processor, and each pair of tasks is associated with a communication cost. The Assignment problem works on minimizing the cost and maximizing the tasks performed.

Type de document : [Logiciel](#)

Domaine : [Informatique \[cs\]](#)

Liste complète des métadonnées [Voir](#)

**BROWSE**

 Software Heritage - Identifiant : 2d7bce631fc791080311eb835c47428e586a6ea4 [Browse](#)

**MÉTADONNÉES**

Keywords :  OR

softwareLicence [GNU](#)

programmingLanguage [Java](#)

codeRepository <https://github.com/moranegg/AffectationRO>

**CITATION**

Morane Gruenpeter. The assignment problem. 2018.  
([hal-01588935](#))  
([swh:1:rev:594617d1cd9d9d6bc0cfbd531bbaa1ed19627e9b](#))

**PARTAGER**

[📄](#) [G](#) [t](#) [+](#) 0

**MÉTRIQUES**

## Credit & Attribution

- a metadata record
- all authors & contributors



# Reference vs. citation

## Credit & Attribution

- a metadata record
- all authors & contributors

## Reuse & Reproducibility

- a specific artifact
- with complementary information (docs)





# Reference vs. citation

## Credit & Attribution

- a metadata record
- all authors & contributors

## Reuse & Reproducibility

- a specific artifact
- with complementary information (docs)

## Archive & Index

- metadata record (HAL)
  - artifact itself (SWH)
- connect the dots...

# Reference vs. citation

## Credit & Attribution

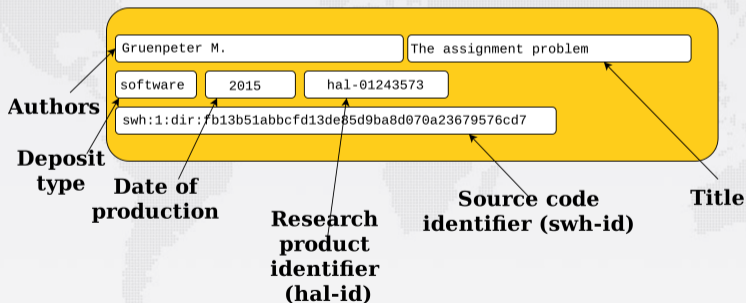
- a metadata record
- all authors & contributors

## Reuse & Reproducibility

- a specific artifact
- with complementary information (docs)

## Archive & Index

- metadata record (HAL)
  - artifact itself (SWH)
- connect the dots...



# Reference vs. citation

## Credit & Attribution

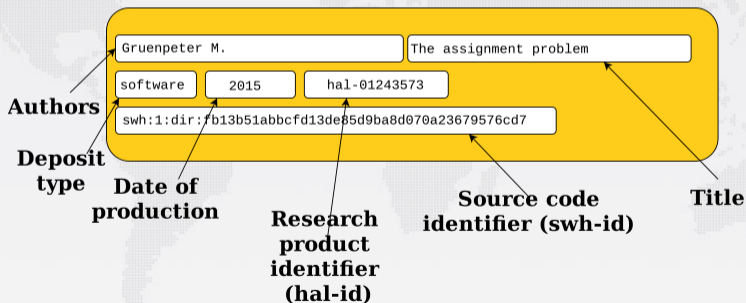
- a metadata record
- all authors & contributors

## Reuse & Reproducibility

- a specific artifact
- with complementary information (docs)

## Archive & Index

- metadata record (HAL)
  - artifact itself (SWH)
- connect the dots...



Prepare your public repository with:

- README, LICENSE, AUTHORS & codemeta.json files

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.

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

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**

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

- git, svn or mercurial
- intrinsic metadata files
- complete history

Home **Archive** Development Documentation Donate

Software Heritage Archive

Archive Access

- Browse
- Web API

Features

- Search
- Vault
- Save code now**

Miscellaneous

- Help

## Save code now

**Beta version**

- **Origin type** the type of version control system the software origin is using. Currently, the only supported type is **git**, for origins using git. Soon, the following origin types will also be available to save into the archive:
  - **svn** for origins using Subversion
  - **hg** for origins using Mercurial
  - **darcs** for origins using Darcs
- **Origin url** the url of the remote repository for the software origin. In order to avoid saving access from Software Heritage, you should provide the direct link of a given by the provider hosting the software origin.

Origin type:  Origin url:



Choose the granularity level for the reference:



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

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

## specific release

```
swh:1:rel:22ece559cc7cc2364edc5e5593d63ae8bd229f9f
```

... and add *;origin=https://github.com/darktable-org/darktable/*

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

## specific release

```
swh:1:rel:22ece559cc7cc2364edc5e5593d63ae8bd229f9f
```

... 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)

```
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 The knowledge is in the source code !
- 3 Data model and SWHID: the source code fingerprint
- 4 The software deposit - a first class research output
- 5 The missing piece- the Metadata
- 6 Development workflow
- 7 Conclusion



*“Ontologies are agreements, made in a social context, to accomplish some objectives. It’s important to understand those objectives, and be guided by them.”*

*T. Gruber, The Pragmatics of Ontology, 2003*

*“Ontologies are agreements, made in a social context, to accomplish some objectives. It’s important to understand those objectives, and be guided by them.”*

*T. Gruber, The Pragmatics of Ontology, 2003*

## *Software Ontology*

- What is software ?
- With what terms should we describe a *software artifact*?
- What about *software source code* ?



What is software ?



## What is software ?

### Software as a concept

- software project / entity

## What is software ?

### Software as a concept

- software project / entity
- the creators and the community around it

## What is software ?

### Software as a concept

- software project / entity
- the creators and the community around it

### Software artifact

- the binaries for different environments

## What is software ?

### Software as a concept

- software project / entity
- the creators and the community around it

### Software artifact

- the binaries for different environments
- the **software source code** for each version



*Ceci n'est pas une pipe.*

What about *software source code*?

Software metadata objectives

manage, share, discover, archive *software source code*

## Software metadata objectives

manage, share, discover, archive *software source code*

## Use cases

- **semantic search**: find software by author, version, keywords
- browse *source code* with context information
- cite and be cited



## Software metadata objectives

manage, share, discover, archive *software source code*

## Use cases

- **semantic search**: find software by author, version, keywords
- browse *source code* with context information
- cite and be cited

## LOV- Linked open vocabularies

“Vocabularies provide the *semantic glue* enabling data to become *meaningful data*. ”

# Where is the metadata available ?

## Catalogs and registries

- [libraries.io](https://libraries.io)
- [OpenHub](https://openhub.com)
- [OntoSoft](https://ontosoft.com)

## Publisher's repositories

- [GitHub](https://github.com)
- [Bitbucket](https://bitbucket.com)
- [SourceForge](https://sourceforge.com)

# Where is the metadata available ?

## Catalogs and registries

- libraries.io
- OpenHub
- OntoSoft

## Publisher's repositories

- GitHub
- Bitbucket
- SourceForge

## advantages and drawbacks

	registries	repositories
accuracy	- not created by author	+ added by authors/maintainers
completeness	+ very detailed	- not a priority
longevity	- depends on registry	- depends on publisher

# Where is the metadata available ?

in the *software source code* itself

- package management file
- CITATION file
- .About file
- codemeta.json file

# Where is the metadata available ?

## in the *software source code* itself

- package management file
- CITATION file
- .About file
- codemeta.json file

## advantages and drawbacks

	metadata file
accuracy	+ created by author and evolves with code
completeness	- depends on the authors knowledge of metadata
longevity	+ not dependent on platform (repository or registry )

# Where is the metadata available ?

## in the *software source code* itself

- package management file
- CITATION file
- .About file
- codemeta.json file

## advantages and drawbacks

	metadata file
accuracy	+ created by author and evolves with code
completeness	- depends on the authors knowledge of metadata
longevity	+ not dependent on platform (repository or registry )

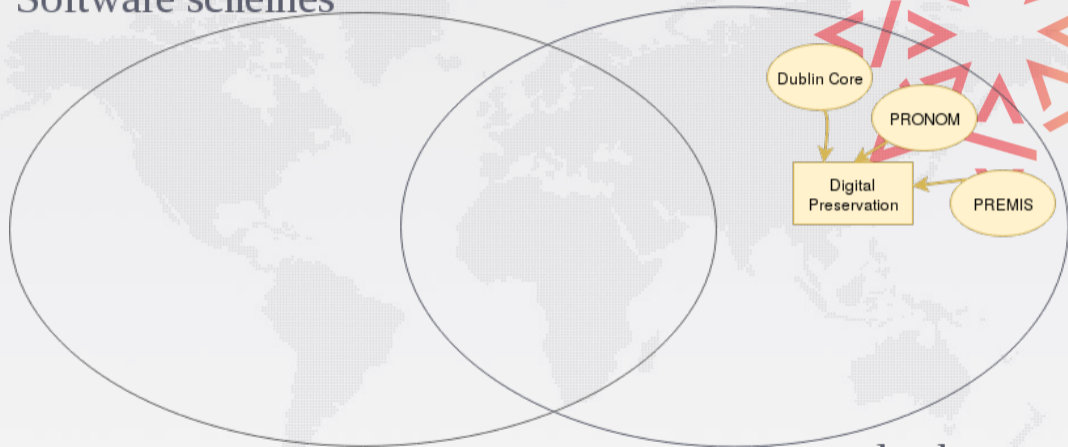
\*

**Bottomline:** to insure the archival of metadata, keep it **in** the data

## Software Citation Principles ( FORCE11's 2015 conference and WG)

- **Importance** : first class citizen in the scholarly ecosystem
- **Credit and attribution** : authors, maintainer
- **Unique identification**: points to a unique, specific software version (DOI, Git SHA1 hash, etc..)
- **Persistence** : identification beyond the lifespan of the software (swh-id)
- **Accessibility**: url, publisher
- **Specificity** : version, environment

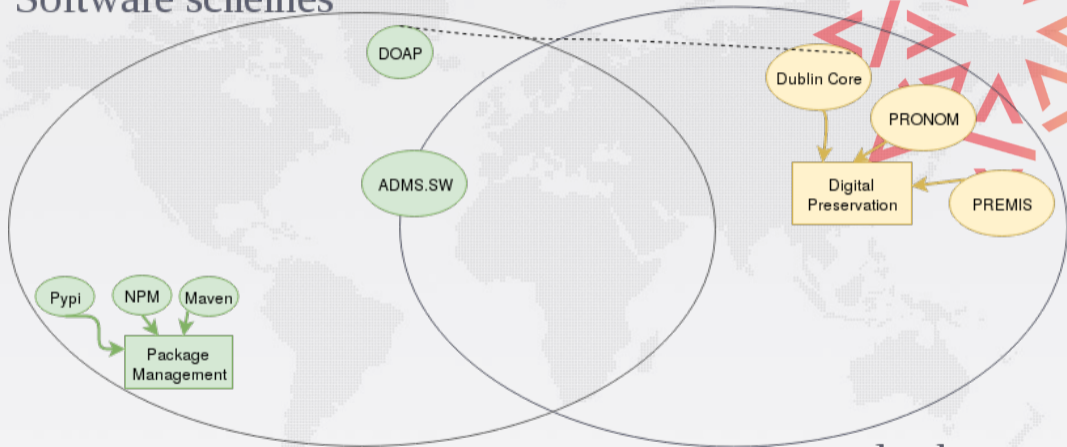
Software schemes



General schemes

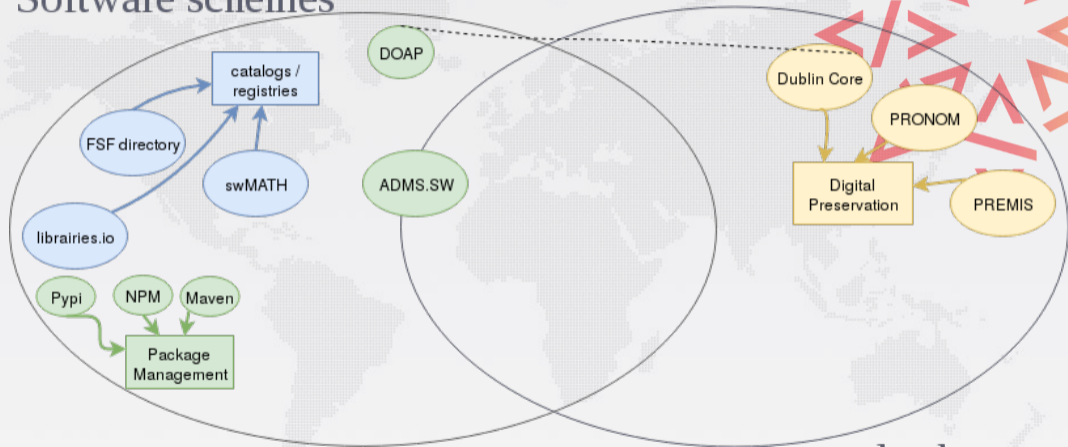


## Software schemes



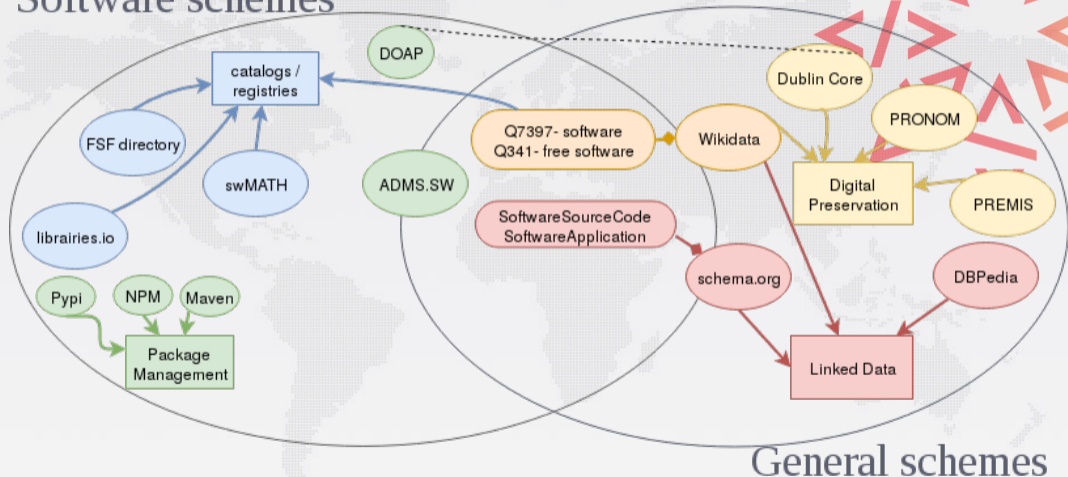
## General schemes

## Software schemes

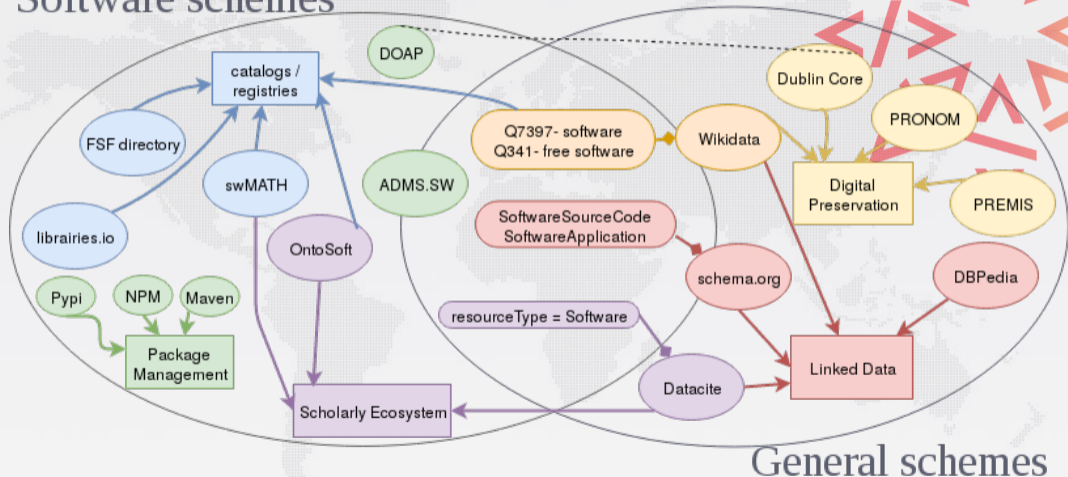


## General schemes

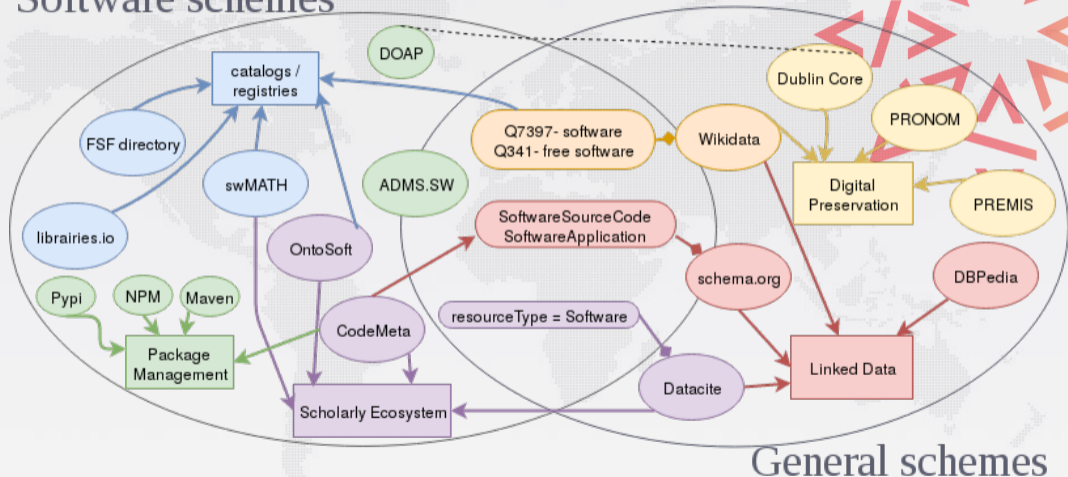
## Software schemes



## Software schemes



## Software schemes



## identify

- identifier
- title
- authors
- version
- type
- origin  
source

## identify

- identifier
- title
- authors
- version
- type
- origin  
source

## execute

- link to a  
compiled  
version
- repository
- compiler
- environment
- examples

## identify

- identifier
- title
- authors
- version
- type
- origin source

## execute

- link to a compiled version
- repository
- compiler
- environment
- examples

## classify

- description
- keywords
- in/out data
- references
- algorithms
- docs url



## identify

- identifier
- title
- authors
- version
- type
- origin source

## execute

- link to a compiled version
- repository
- compiler
- environment
- examples

## classify

- description
- keywords
- in/out data
- references
- algorithms
- docs url

## administrate

- contact
- authorship
- funders
- license
- editor (publisher)
- dates
- status

# Much 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

# Much 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”.*

## supported intrinsic metadata files

- CodeMeta's `codemeta.json`,
- Maven's `pom.xml`,
- NPM's `package.json`,
- Python's `PKG-INFO`,
- Ruby's `.gemspec`

## Check the code

- *blog post*
- *tutorial in docs*

<https://forge.softwareheritage.org/source/swh-indexer.git>

09 May 2020, 21:49 UTC

<> Code Branches (25) Releases (322) Visits

★ Branch: HEAD f409912 / swh / indexer / metadata\_dictionary / codemeta.py

Raw File python Save again Show metadata

Tip revision: 10f8af474codf7511c4b40ea480646ae73596303 authored by Antoine R. Dumont (@ardumont) on 05 May 2020, 07:28 UTC  
Fix blackened strings with unneeded concatenation

codemeta.py

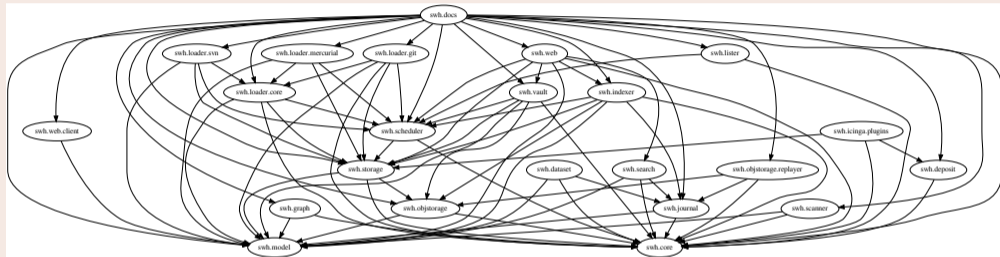
```
1 # Copyright (C) 2018-2019 The Software Heritage developers
2 # See the AUTHORS file at the top-level directory of this distribution
3 # License: GNU General Public License version 3, or any later version
4 # See top-level LICENSE file for more information
5
6 import json
7
8 from swh.indexer.codemeta import CODEMETA_TERMS
9 from swh.indexer.codemeta import expand
10 from .base import SingleFileMapping
11
12
13 class CodemetaMapping(SingleFileMapping):
14     """
15     dedicated class for CodeMeta (codemeta.json) mapping and translation
16     """
17
18     name = "codemeta"
19     filename = b"codemeta.json"
20     string_fields = None
21
22     @classmethod
23     def supported_terms(cls):
24         return [term for term in CODEMETA_TERMS if not term.startswith("@")]
25
26     def translate(self, content):
27         try:
28             return self.normalize_translation(expand(json.loads(content.decode())))
```

Permalinks

- 1 Introduction
- 2 The knowledge is in the source code !
- 3 Data model and SWHID: the source code fingerprint
- 4 The software deposit - a first class research output
- 5 The missing piece- the Metadata
- 6 Development workflow
- 7 Conclusion



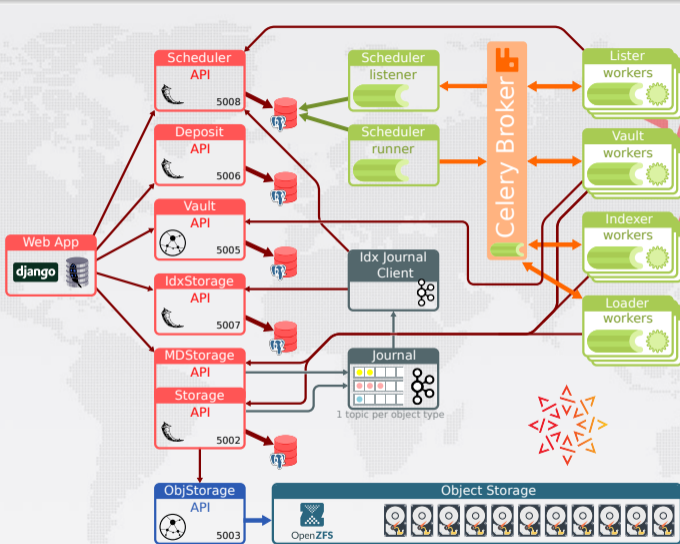
## Using a bit of code



Actually it's not so big:

- ~20ksloc of python3
- ~80 python dependencies
- a bunch of js
- ... keep it as simple as possible, but no simpler... (almost)

# The big picture



*More details in our docs*



## Development documentation

<https://docs.softwareheritage.org/devel/>

- in particular, Developer setup: <https://docs.softwareheritage.org/devel/developer-setup.html>
- i.e.: virtualenv + pip + tox

## "Software Development" pages on the public wiki

[https://wiki.softwareheritage.org/wiki/Category:Software\\_development](https://wiki.softwareheritage.org/wiki/Category:Software_development)

## Internship page on the public wiki

<https://wiki.softwareheritage.org/wiki/Internships>

## Phabricator

<https://forge.softwareheritage.org/>

- all development activities happen here

## The classics

- VCS: Git, with repo browsing using Diffusion

<https://forge.softwareheritage.org/diffusion/>

- Tasks and Bugs: Maniphest

<https://forge.softwareheritage.org/maniphest/>

- one project tag for each software product, e.g., Git Loader:  
<https://forge.softwareheritage.org/project/view/17/>
- we use task priorities, assignees, and tags
- visibility: all dev tasks are public

- 1 Introduction
- 2 The knowledge is in the source code !
- 3 Data model and SWHID: the source code fingerprint
- 4 The software deposit - a first class research output
- 5 The missing piece- the Metadata
- 6 Development workflow
- 7 Conclusion



Use a forge for your academic and personal projects

Github, Gitlab or Bitbucket are the best way to create your **source code cv**

# Research Software Engineer tips

Use a forge for your academic and personal projects

Github, Gitlab or Bitbucket are the best way to create your **source code cv**

Put in your projects metadata files and document your code

**README, LICENSE, AUTHORS** and **codemeta.json** to describe your project

# Research Software Engineer tips

Use a forge for your academic and personal projects

Github, Gitlab or Bitbucket are the best way to create your **source code cv**

Put in your projects metadata files and document your code

**README, LICENSE, AUTHORS** and **codemeta.json** to describe your project

Archive your projects on SWH

Use the **Save Code Now** feature

# Research Software Engineer tips

Use a forge for your academic and personal projects

Github, Gitlab or Bitbucket are the best way to create your **source code cv**

Put in your projects metadata files and document your code

**README, LICENSE, AUTHORS** and **codemeta.json** to describe your project

Archive your projects on SWH

Use the **Save Code Now** feature

Contribute to other projects

When you contribute you learn how to **read code**

# Research Software Engineer tips

Use a forge for your academic and personal projects

Github, Gitlab or Bitbucket are the best way to create your **source code cv**

Put in your projects metadata files and document your code

**README, LICENSE, AUTHORS** and **codemeta.json** to describe your project

Archive your projects on SWH

Use the **Save Code Now** feature

Contribute to other projects

When you contribute you learn how to **read code**

Ask

Don't be afraid to ask on an **issue, mailing list** or **irc channel** (or your teachers)



Come in, we're open!



# Software Heritage

Thank you! Any questions?

Join us on <https://forge.softwareheritage.org/>



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*, iPRES 2018: Intl. Conf. on Digital Preservation

contact: [morane@softwareheritage.org](mailto:morane@softwareheritage.org)